

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Інтернет-магазин ювелірних виробів»

Виконав студент групи ІПЗ-21-З
МИРОНЧУК Владислав Олександрович

Керівник роботи:
МОРОЗОВ Андрій Васильович

Рецензент:
ЧИЖМОТРЯ Олена Геннадіївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«14» лютого 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **Мирончук Владислав Олександрович**

Керівник роботи: **Морозов Андрій Васильович**

Тема роботи: **«Інтернет-магазин ювелірних виробів»**,

затверджена Наказом закладу вищої освіти від **14.02.2025р., №61/с**

Вихідні дані для роботи: Процес автоматизації електронної комерції та організації онлайн-продажу ювелірних виробів із застосуванням сучасних інформаційних технологій. Особлива увага приділяється розробці вебзастосунку для інтернет-магазину ювелірної продукції з інтеграцією функцій інтелектуального пошуку, рекомендацій товарів та персоналізації інтерфейсу відповідно до вподобань користувача.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Морозов А.В	20.03.2025	20.03.2025
2	Морозов А.В	20.03.2025	20.03.2025
3	Морозов А.В	20.03.2025	20.03.2025

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з вебзастосунку — сайту інтернет-магазину ювелірних виробів — та пояснювальної записки. Пояснювальна записка до роботи містить 81 сторінок, 29 ілюстрацій та 19 таблиць.

Метою роботи є розробка сучасного, зручного та функціонального вебсайту для продажу ювелірних виробів, який дозволяє користувачам переглядати каталог продукції, ознайомлюватися з детальним описом товарів, фільтрувати та сортувати вироби за категоріями, додавати їх до кошика та здійснювати замовлення. Застосунок орієнтований на широку аудиторію користувачів, які бажають придбати якісні ювелірні прикраси в онлайн-форматі, що підвищує доступність таких товарів через інтернет.

Об'єктом дослідження є процес онлайн-продажу ювелірних виробів за допомогою вебтехнологій.

Предметом дослідження є методи розробки адаптивного вебзастосунку для електронної комерції, зокрема інтернет-магазину, а також принципи побудови зручного користувацького інтерфейсу та ефективної системи керування товарами й замовленнями.

У роботі застосовано сучасні методи веброзробки на основі фреймворку Next.js, створення REST API, роботу з базами даних, реалізацію адаптивного дизайну та інтеграцію платіжної системи. Особливу увагу приділено реалізації зручного пошуку ювелірних виробів, фільтрації за параметрами, системі авторизації та реєстрації користувачів, а також модулю керування замовленнями.

КЛЮЧОВІ СЛОВА: ІНТЕРНЕТ-МАГАЗИН, ЮВЕЛІРНІ ВИРОБИ, ВЕБЗАСТОСУНОК, ЕЛЕКТРОННА КОМЕРЦІЯ, ІНТЕРАКТИВНИЙ ІНТЕРФЕЙС, ВЕБ-ТЕХНОЛОГІЇ.

					ІПЗ.КР.Б – 121 – 25 – ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		В.О. Мирончук			Інтернет-магазин ювелірних виробів Пояснювальна записка	Літ.	Арк.
Керівник		А.В. Морозов					79
Рецензент		О.Г. Чижмотря				Житомирська Політехніка, група ІПЗ-21-3	
Зав. кафедри		Т.А. Вакалюк					

ABSTRACT

The bachelor's qualification thesis consists of a web application — an online jewelry store website — and an explanatory report. The explanatory report contains 81 pages, 29 illustrations, and 19 tables.

The aim of the project is to develop a modern, user-friendly, and functional website for selling jewelry products. The platform allows users to browse the product catalog, view detailed descriptions, filter and sort items by category, add them to the cart, and place orders. The application is designed for a broad audience of users who wish to purchase high-quality jewelry online, thereby increasing the accessibility of such products via the internet.

The object of the study is the process of selling jewelry online using web technologies.

The subject of the study is the methods of developing a responsive web application for e-commerce, particularly an online store, as well as the principles of building a user-friendly interface and an efficient product and order management system.

The project uses modern web development methods based on the Next.js framework, including REST API creation, database integration, responsive design implementation, and payment system integration. Special attention is given to the implementation of intuitive product search, parameter-based filtering, user authentication and registration, and the order management module.

KEYWORDS: ONLINE STORE, JEWELRY, WEB APPLICATION, E-COMMERCE, INTERACTIVE INTERFACE, WEB TECHNOLOGIES.

		<i>В.О. Мирончук</i>			<i>ІІЗ.КР.Б - 121 - 25 - ПЗ</i>	<i>Арк.</i>
		<i>А.В. Морозов</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		<i>4</i>

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ	9
1.1. Постановка задачі.....	9
1.2. Аналіз аналогів програмного продукту	10
1.3. Вибір архітектури платформи інтернет-магазину ювелірних виробів.....	16
1.4. Обґрунтування вибору інструментальних засобів.....	18
Висновки до першого розділу.....	22
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ ЮВЕЛІРНИХ ВИРОБІВ.....	23
2.1. Визначення варіантів використання та структури системи.....	23
2.2 Розробка бази даних.....	27
2.3. Проєктування та реалізація алгоритмів роботи	36
2.4 Реалізація функціоналу.....	40
Висновки до другого розділу	44
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДОДАТКУ	46
3.1 Порядок встановлення та налаштування системи	46
3.2 Структура інтерфейсу та порядок взаємодії з інтернет-магазином ювелірних виробів.....	47
3.3 Порядок встановлення та налаштування системи	62
Висновки до третього розділу.....	64
ВИСНОВКИ.....	65
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	66
ДОДАТКИ.....	70

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних

ПЗ – програмне забезпечення

ІТ – інформаційні технології

СУБД – система управління базами даних

UI – користувацький інтерфейс (англ. User Interface)

UX – користувацький досвід (англ. User Experience)

HTML – мова розмітки гіпертексту (англ. HyperText Markup Language)

CSS – каскадні таблиці стилів (англ. Cascading Style Sheets)

JS – мова програмування JavaScript

API – інтерфейс прикладного програмування (англ. Application Programming Interface)

REST – архітектурний стиль для створення вебсервісів (англ. Representational State Transfer)

HTTPS – захищений протокол передавання гіпертексту

JWT – вебтокен JSON (англ. JSON Web Token)

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У сучасних умовах цифрової трансформації бізнесу онлайн-торгівля займає провідне місце серед способів реалізації товарів і послуг. Одним із динамічно зростаючих напрямів є продаж ювелірних виробів через інтернет, що зумовлено високим попитом на ексклюзивні прикраси, зручністю онлайн-покупок та бажанням користувачів мати доступ до широкого вибору без прив'язки до фізичних магазинів. Проте значна частина компаній, які працюють у сфері ювелірного бізнесу, все ще не мають повноцінно реалізованих цифрових платформ. Це обмежує можливості масштабування та втрати клієнтів. Тому створення сучасного інтернет-магазину з інтуїтивним інтерфейсом, персоналізованими рекомендаціями та надійною інфраструктурою є актуальним завданням, яке відповідає потребам ринку та тенденціям розвитку електронної комерції.

Мета роботи полягає у створенні функціонального, адаптивного та зручного вебсайту для продажу ювелірних виробів, що дозволяє користувачам комфортно переглядати продукцію, знаходити потрібні товари за допомогою пошуку та фільтрів, додавати їх до кошика та здійснювати замовлення з інтегрованою оплатою онлайн.

Для досягнення поставленої мети необхідно вирішити такі завдання:

Провести огляд тенденцій розвитку ринку електронної комерції у сфері ювелірних виробів.

Дослідити існуючі рішення онлайн-магазинів та виявити їхні переваги й недоліки.

Обґрунтувати вибір архітектури вебзастосунку на базі Next.js.

Визначити оптимальні інструменти та фреймворки для реалізації проєкту.

Розробити структуру бази даних для збереження інформації про товари, користувачів і замовлення.

Реалізувати функціонал фільтрації, пошуку, авторизації, кошика, оформлення замовлень і обробки платежів.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				7
Змн.	Арк.	№ докум.	Підпис	Дата		

Здійснити розробку адаптивного вебінтерфейсу для різних типів пристроїв.

Описати сценарії взаємодії користувача з системою та забезпечити зручність навігації.

Провести тестування функціональних модулів для перевірки коректності роботи та відповідності вимогам зручності, безпеки та продуктивності.

Об'єктом дослідження є процес створення вебзастосунку для електронної комерції, зокрема інтернет-магазину ювелірних виробів.

Предметом дослідження є сучасні вебтехнології, інструменти та підходи до розробки функціонального онлайн-магазину, а також принципи формування зручного інтерфейсу, інтеграції платіжних систем і ефективного управління товарами та замовленнями.

		<i>В.О. Мирончук</i>			<i>ІПЗ.КР.Б – 121 – 25 – ПЗ</i>	<i>Арк.</i>
		<i>А.В. Морозов</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		<i>8</i>

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1. Постановка задачі

Сьогодні інтернет-магазини стали невід'ємною частиною комерційної діяльності в різних сферах, зокрема у сфері продажу ювелірних виробів. Цифрова трансформація бізнесу, зростання популярності онлайн-покупок та підвищені очікування клієнтів щодо зручності й якості обслуговування зумовлюють необхідність створення ефективних вебрішень для продажу товарів преміум-сегменту. Наявність сучасного, інтуїтивно зрозумілого вебзастосунку для торгівлі ювелірними виробами дозволяє компаніям розширювати клієнтську базу, оптимізувати процеси обробки замовлень та створити додаткові канали збуту.

Створення інтернет-магазину ювелірної продукції передбачає розробку вебплатформи, яка поєднуватиме адаптивний дизайн, зручний інтерфейс, гнучку систему керування товарами, безпечну оплату та ефективну взаємодію з клієнтами. Такий застосунок повинен забезпечувати повноцінний каталог з детальними описами продукції, можливістю фільтрації за категоріями, матеріалами, ціною та іншими параметрами, додаванням товарів до кошика і подальшим оформленням замовлень. Окрему роль відіграє адміністративна частина, що дозволяє управляти замовленнями, оновлювати асортимент, відслідковувати статистику продажів та працювати з відгуками клієнтів.

Задля підвищення ефективності обробки замовлень і комунікації між адміністрацією магазину та клієнтами доцільним є впровадження телеграм-бота, який інформуватиме про нові замовлення, зміни в статусах доставки та дозволить швидко реагувати на повідомлення користувачів. Важливим аспектом проекту є також інтеграція платіжної системи, що забезпечить безпечні онлайн-транзакції, відповідні сучасним стандартам конфіденційності та зручності.

Таким чином, розробка інтернет-магазину ювелірних виробів охоплює не лише технічні завдання, пов'язані з реалізацією функціоналу та інтерфейсу, а й

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				9
Змн.	Арк.	№ докум.	Підпис	Дата		

потребує врахування особливостей цільової аудиторії, естетичних вимог до дизайну й забезпечення безперебійної роботи в умовах реального комерційного навантаження.

1.2. Аналіз аналогів програмного продукту

Для розробки веб-магазину ювелірних виробів важливо провести детальний аналіз існуючих аналогічних платформ, таких як James Allen [1], Pandora [2], Kuyashii Jewelry [3] та Parts of Four [4]. Це дозволить виявити ключові переваги та недоліки кожного з них, а також визначити функціональні можливості, які слід реалізувати для забезпечення конкурентоспроможності нашого продукту.

James Allen

James Allen — один із провідних онлайн-магазинів ювелірних виробів, що спеціалізується на продажу обручок, класичних кілець та інших прикрас. Платформа вирізняється впровадженням сучасних технологій, зокрема можливістю 3D-перегляду товарів та демонстрацією фотографій з різних ракурсів, що дає змогу покупцям детально розглянути кожен прикрасу до оформлення замовлення. Інтерфейс сайту зручний для користувачів і дозволяє легко знаходити потрібні вироби за допомогою фільтрів та сортування. Клієнти мають змогу отримати професійну консультацію безпосередньо на сайті завдяки інтегрованому онлайн-чату з менеджерами. Крім того, сервіс вирізняється прозорою системою ціноутворення та надає вичерпну інформацію про характеристики дорогоцінного каміння, що є важливим чинником довіри.

Серед недоліків варто зазначити відсутність україномовної версії платформи, що може ускладнити користування для частини аудиторії. Також асортимент товарів має певні обмеження, оскільки основна увага зосереджена саме на кільцях, тоді як інші категорії ювелірних виробів представлені менш широко.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				10
Змн.	Арк.	№ докум.	Підпис	Дата		

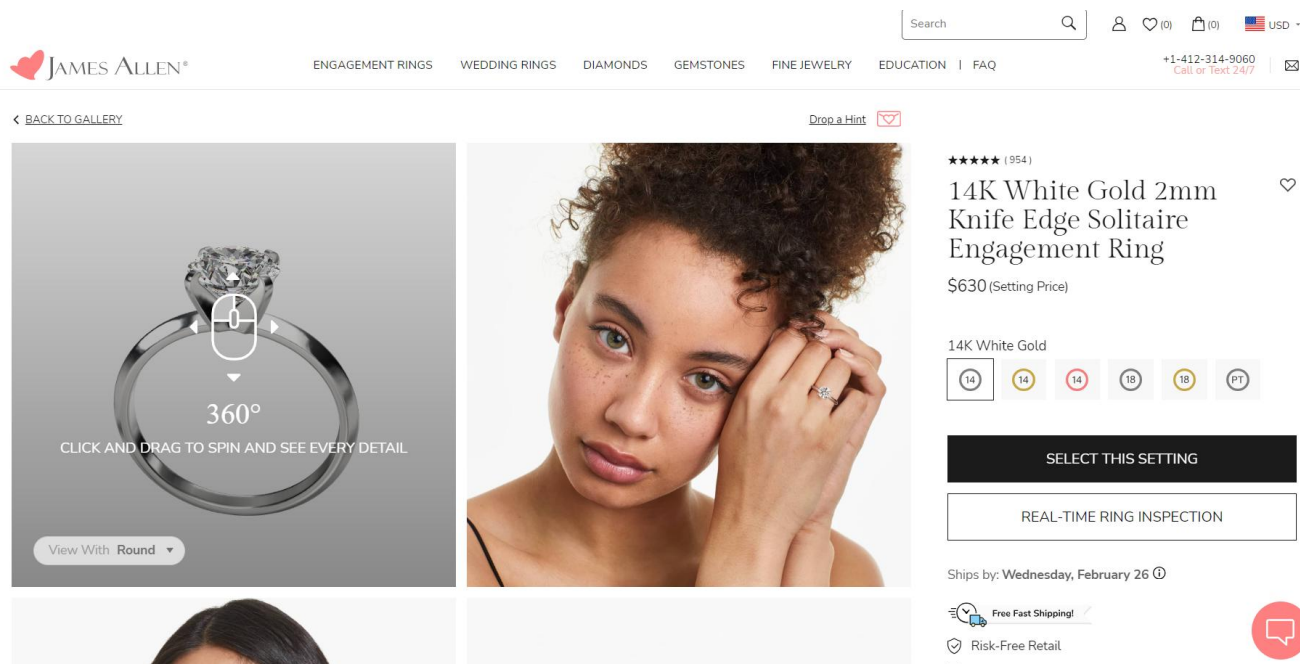


Рис.1.1 — Сторінка товару James Allen

Pandora

Pandora — міжнародний бренд, який реалізує ювелірні вироби через свій офіційний вебсайт і має широку впізнаваність завдяки стильним браслетам та можливості персоналізації прикрас. Користувачі можуть обирати окремі елементи для створення унікальних браслетів, а також замовляти гравірування, що надає виробам індивідуального характеру. Сайт вирізняється привабливим сучасним дизайном і зручним інтерфейсом, який не потребує тривалого часу на освоєння. Важливою перевагою є наявність україномовної версії, що робить користування платформою комфортним для вітчизняної аудиторії.

Разом із тим, асортимент представлений не так широко, як на деяких конкурентних ресурсах, оскільки основний акцент зроблено саме на браслетах та аксесуарах до них. Крім того, на сайті відсутній 3D-перегляд товарів, що дещо обмежує можливості візуального ознайомлення з прикрасами перед купівлею.

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				
Змн.	Арк.	№ докум.	Підпис	Дата		11

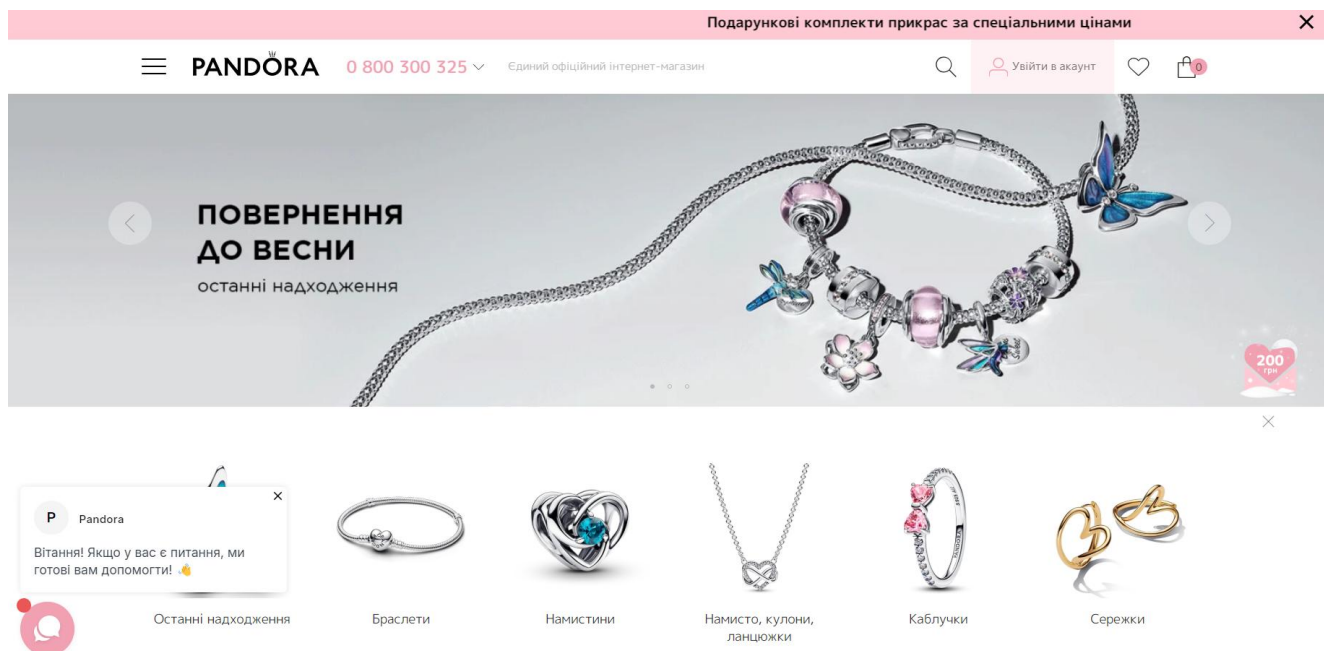


Рис. 1.2 — Головна сторінка Pandora

Kasta

Kuyashii Jewelry — нішевий онлайн-магазин, що спеціалізується на продажу ексклюзивних ювелірних виробів ручної роботи. Платформа орієнтована на клієнтів, які цінують унікальність і шукають авторські прикраси з вираженою індивідуальністю. Кожен товар супроводжується детальним описом і якісними фотографіями, що дозволяє ознайомитися з виробом перед покупкою. Однією з ключових особливостей є можливість замовлення прикрас за індивідуальним дизайном, що робить кожен виріб особливим і неповторним.

Разом із цим, сайт має низку обмежень: на платформі відсутні інструменти фільтрації чи пошуку за характеристиками, що ускладнює навігацію при розширенні асортименту. Крім того, кількість товарів на сайті наразі обмежена, що зменшує вибір для потенційних покупців.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				12
Змн.	Арк.	№ докум.	Підпис	Дата		

SEE MORE: CHAINS

PREVIOUS / NEXT



Barbed Wire Necklace (.925 Sterling Silver)

SOLD OUT

★★★★★
€6,303.00

This product is sold out

If you're looking for something that stands out from the crowd, the Kuyashii Barbed Wire Chain is it. Available in 18" or 20", this chain will be a unique addition to your jewelry collection. Add on one of our larger pendants (Murakami, 360 Cross) to create a personalized touch. Water safe, will never change color of your skin.

Paired best with: Murakami Flower Pendant

PRODUCT SPECIFICS

Length: 18 Inch | 20 Inch

Рис. 1.3 —Сторінка товару Kuyashii Jewelry

Parts of Four

Parts of Four — інноваційний ювелірний бренд, що пропонує прикраси в стилі мінімалізму з виразним сучасним дизайном. Платформа орієнтована переважно на молоду аудиторію, яка віддає перевагу простим, універсальним формам і стилістичній стриманості. Вебсайт вирізняється лаконічним і водночас стильним інтерфейсом, що сприяє швидкому та комфортному оформленню замовлень. Особливу роль у презентації продукції відіграє візуальний контент — численні фотографії та відео дозволяють краще передати естетику кожного виробу.

Водночас сайт має певні недоліки. Зокрема, бракує розгорнутої інформації про технічні характеристики матеріалів, з яких виготовлено прикраси, що може викликати труднощі у прийнятті рішення перед покупкою. Крім того, платформа не має україномовної версії, що знижує її зручність для українських користувачів.

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				
Змн.	Арк.	№ докум.	Підпис	Дата		13

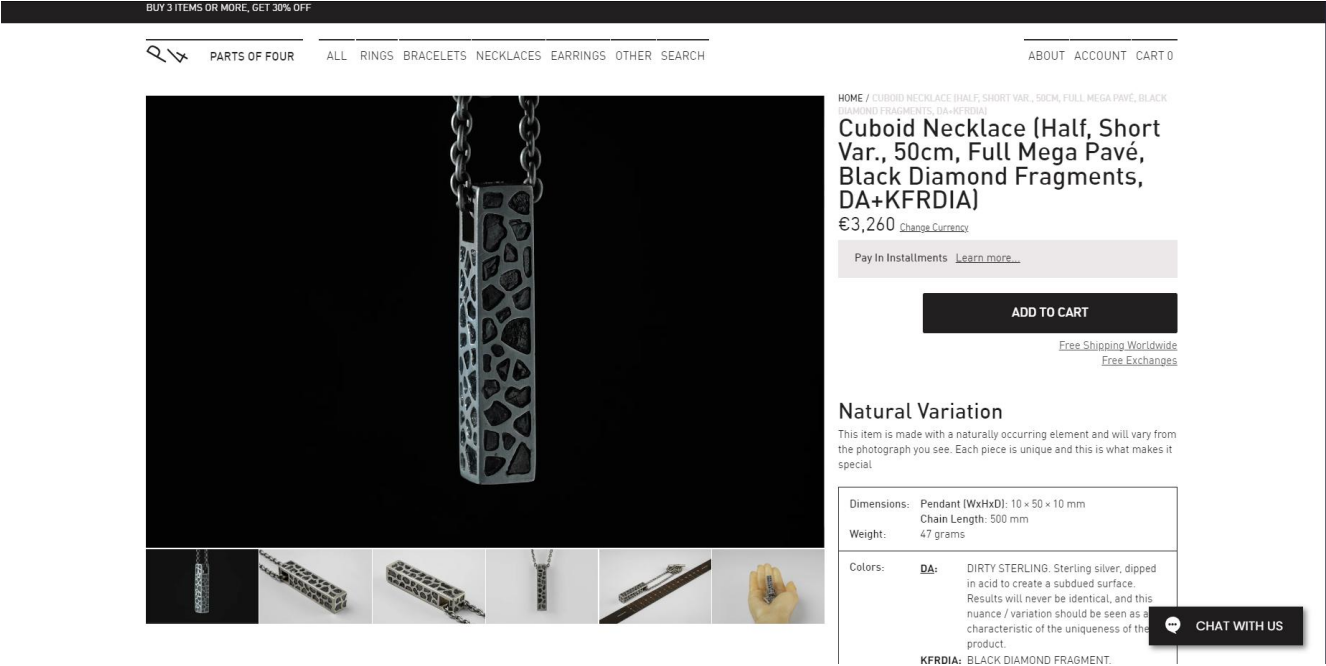


Рис.1.4 —Сторінка товару Parts of Four

Порівняльна характеристика аналогів

Для кращого розуміння переваг та недоліків кожного з розглянутих аналогів було проведено порівняльний аналіз основних характеристик, які є ключовими для веб-магазинів ювелірних виробів.

Таблиця 1.1

Характеристика	James Allen	Pandora	Kuyashii Jewelry	Parts of Four
Зручний інтерфейс	+	+	-	+
Наявність 3D-перегляду	+	-	-	-
Широкий асортимент	+	+	-	-
Україномовний інтерфейс	-	+	-	-
Унікальність товарів	-	-	+	+

Ця таблиця демонструє, як кожен із розглянутих аналогів вирізняється за окремими параметрами. На основі цього аналізу можна зробити висновок, що наш веб-магазин повинен поєднувати переваги всіх платформ, забезпечуючи широкий асортимент, зручний інтерфейс, мультимовність та інноваційні

технології, такі як 3D-перегляд. Це дозволить створити конкурентоспроможний продукт, який задовольнить потреби різних груп користувачів.

Аналіз аналогічних платформ демонструє, що кожен із розглянутих інтернет-магазинів має свої унікальні риси та орієнтований на певну аудиторію. Водночас можна виокремити спільні характеристики, які вважаються визначальними для успішного функціонування подібних вебресурсів. Однією з ключових вимог є висока якість візуального представлення товарів: клієнти очікують побачити детальні фотографії, а іноді й 3D-моделі прикрас, як це реалізовано, наприклад, у James Allen. Це не лише підвищує довіру до товару, а й сприяє прийняттю рішення про покупку.

Не менш важливою є зручність інтерфейсу, який повинен забезпечувати швидкий доступ до інформації, легкий пошук, ефективну фільтрацію та простий процес оформлення замовлення. Вдалу реалізацію таких рішень демонструють Pandora і Parts of Four, де дизайн поєднує естетику з функціональністю. Серед факторів, що покращують користувацький досвід, варто також відзначити можливість персоналізації товарів, участь у програмах лояльності та доступність зручних способів оплати. Такі можливості пропонують, зокрема, Pandora та Kuyashii Jewelry, що дозволяє користувачам отримати унікальний продукт з урахуванням індивідуальних побажань.

Ще одним важливим елементом є підтримка кількох мов, зокрема української. Це значно розширює аудиторію магазину та робить платформу доступною для вітчизняного користувача. Серед усіх розглянутих аналогів лише Pandora має україномовну версію сайту, що дає їй додаткову перевагу на локальному ринку.

З урахуванням виявлених особливостей і переваг конкурентів, доцільно закласти в архітектуру власного інтернет-магазину функціонал, який включатиме якісну візуалізацію товарів із можливістю 3D-огляду, простий та зрозумілий інтерфейс, підтримку української мови й інтеграцію з популярними платіжними системами. Ці складові стануть базою для формування конкурентоспроможної платформи з продажу ювелірних виробів, яка

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				15
Змн.	Арк.	№ докум.	Підпис	Дата		

задовольнятиме потреби клієнтів та сприятиме ефективному розвитку онлайн-бізнесу.

1.3. Вибір архітектури платформи інтернет-магазину ювелірних виробів

Для реалізації інтернет-магазину ювелірних виробів було обрано клієнт-серверну архітектуру на основі фреймворку Next.js [5], що дозволяє об'єднати клієнтську та серверну логіку в межах одного застосунку. Такий підхід забезпечує ефективне розділення функціоналу, спрощує розробку та забезпечує гнучкість масштабування.

Клієнтська частина відповідає за формування користувацького інтерфейсу, який включає головну сторінку, каталог прикрас, детальні сторінки товарів, кошик, сторінки оформлення замовлення та особистий кабінет користувача. Звідси надсилаються HTTP-запити до серверної частини через REST API. Саме клієнтська частина забезпечує взаємодію з відвідувачем сайту, а всі операції, пов'язані з обробкою даних, передаються на сервер.

Серверна логіка також реалізована за допомогою Next.js — через вбудовані API-роути, що обробляють запити типу GET, POST, PUT і DELETE. Вони виконують перевірку автентичності, валідацію вхідних даних, формування замовлень, обрахунок знижок, оновлення профілю користувача, роботу з відгуками тощо. Вся інформація зберігається в базі даних, до якої звертаються API-ендпоінти під час кожної операції.

На схемі нижче зображено загальний принцип роботи клієнт-серверної архітектури, реалізованої в межах одного Next.js-проєкту.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				16
Змн.	Арк.	№ докум.	Підпис	Дата		

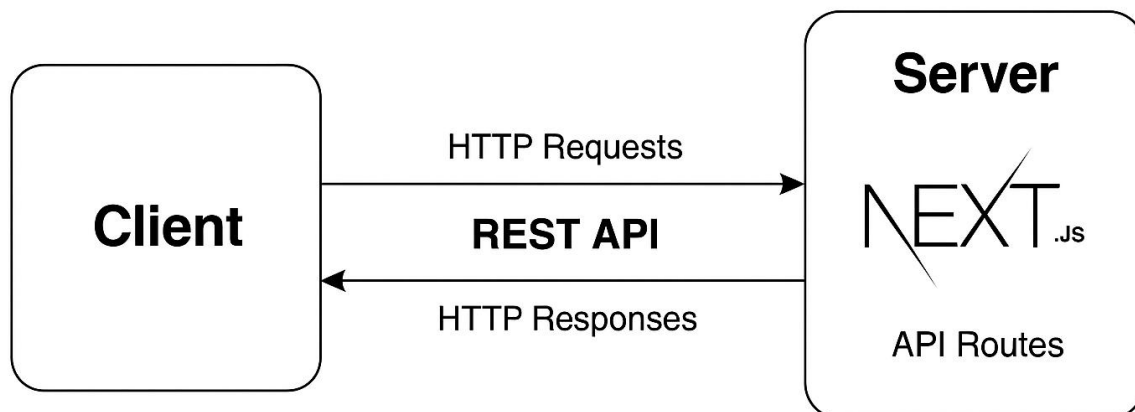


Рис. 1.5 — Взаємодія клієнта з серверною частиною через REST API в архітектурі Next.js

Такий підхід дозволяє значно спростити процес розробки, оскільки як клієнт, так і сервер працюють у єдиному середовищі. Це зменшує кількість помилок, пов'язаних із взаємодією між окремими сервісами, й полегшує розгортання проєкту. REST API також дає змогу легко інтегрувати зовнішні сервіси, як-от платіжні системи чи сервіси доставки, що є важливим для повноцінного функціонування інтернет-магазину.

Клієнт-серверна архітектура в поєднанні з можливостями Next.js дозволяє досягти високої продуктивності, безпеки та масштабованості. Всі критично важливі операції виконуються на сервері, а клієнт забезпечує зручну й швидку взаємодію з користувачем. Така архітектура є оптимальним рішенням для сучасного інтернет-магазину ювелірних виробів, який потребує стабільної роботи, безпечного зберігання даних і гнучкого розширення функціоналу.

1.4. Обґрунтування вибору інструментальних засобів

Для реалізації інтернет-магазину ювелірних виробів було використано сучасний набір інструментальних засобів, які забезпечують високу продуктивність, масштабованість, безпеку та зручність подальшої розробки. Основу проєкту становить фреймворк Next.js, який дозволяє поєднувати клієнтську і серверну логіку в єдиному середовищі, підтримує рендеринг на стороні сервера та має розвинену екосистему для створення складних вебзастосунків.

Взаємодія з користувачем реалізована за допомогою бібліотеки React 19, яка забезпечує динамічне оновлення інтерфейсу без перезавантаження сторінки та дозволяє будувати модульні й повторно використовувані компоненти. Візуальні елементи іконок на сайті реалізовано з використанням бібліотеки lucide-react, яка надає великий набір SVG-іконок, адаптованих під сучасні інтерфейси.

Для керування слайдерами та динамічним візуальним контентом використано бібліотеку Swiper, яка дозволяє реалізувати адаптивні й інтерактивні галереї, зокрема каруселі зображень для товарів.

Система автентифікації реалізована з використанням NextAuth.js [7] у поєднанні з next-auth/prisma-adapter, що дозволяє інтегрувати авторизацію через базу даних із мінімальними зусиллями. Це забезпечує підтримку сесій, реєстрації, входу та управління користувачами на високому рівні безпеки.

Бекенд-частина побудована на основі Prisma ORM [8], що забезпечує зручну роботу з базою даних, дозволяє описувати структуру через схемні файли та генерує безпечні типізовані запити. Клієнтська бібліотека @prisma/client використовується для взаємодії з базою безпосередньо з серверної частини Next.js. Для зберігання паролів застосовується бібліотека bcrypt, яка дозволяє надійно хешувати чутливу інформацію.

Для генерації унікальних ідентифікаторів, які використовуються в замовленнях, товарах та інших об'єктах системи, використано бібліотеку uuid. Зберігання та керування мультимедійним контентом реалізовано через API

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				18
Змн.	Арк.	№ докум.	Підпис	Дата		

Vercel Blob, який спрощує роботу з файлами без необхідності самостійного конфігурування хмарної інфраструктури.

Особливу увагу приділено інтеграції сучасних візуалізаційних технологій: за допомогою Google Model Viewer реалізовано можливість перегляду 3D-моделей ювелірних виробів безпосередньо на сторінці товару, що підвищує залучення користувачів і дозволяє детальніше ознайомитися з продукцією.

Для реалізації системи оплати планується інтеграція сервісу Stripe, що є одним із найнадійніших і найпоширеніших рішень для обробки онлайн-платежів. Stripe дозволяє проводити транзакції з високим рівнем безпеки, підтримує різні способи оплати та легко інтегрується з існуючою архітектурою через REST API.

У рамках розробки інтернет-магазину для зберігання структурованих даних про користувачів, замовлення, товари, відгуки та інші об'єкти було обрано реляційну базу даних PostgreSQL. Це сучасна, масштабована та надійна СУБД з відкритим кодом, яка підтримує складні зв'язки між таблицями, транзакційність, розширені типи даних і високий рівень безпеки. PostgreSQL ефективно інтегрується з ORM Prisma, що забезпечує зручне моделювання та доступ до даних у межах проєкту на Next.js.

Перед остаточним вибором було розглянуто декілька популярних рішень, які активно застосовуються у веброзробці: PostgreSQL, MySQL та MongoDB [15-17]. Для порівняння було проаналізовано низку ключових характеристик, зокрема максимальний обсяг збережених даних (табл. 1.2), підтримку транзакцій (табл. 1.3) та можливості виконання складних запитів (табл. 1.4).

Як показано в таблиці 1.2, усі розглянуті СУБД здатні працювати з великими обсягами інформації. MongoDB забезпечує гнучке масштабування завдяки шардінгу, але її документоорієнтована модель не завжди підходить для строго структурованих зв'язків між сутностями. MySQL також підтримує зберігання значних обсягів даних, однак має обмеження, пов'язані з типами таблиць і файловою системою. PostgreSQL, у свою чергу, надає потужні засоби

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				19
Змн.	Арк.	№ докум.	Підпис	Дата		

для роботи з великими таблицями, включаючи підтримку партиціонування та розподіленого зберігання.

Таблиця 1.2

Максимальний обсяг збережених даних

СУБД	Максимальний обсяг збережених даних	Особливості обмеження
MongoDB	Теоретично необмежений (до 1 ТБ на колекцію у 32-бітній системі)	Залежить від файлової системи, при шардінгу можливо зберігати петабайти даних
PostgreSQL	Обмежено файловою системою (до 64 ТБ на таблицю у 64-бітній системі)	Підтримує партиціонування та розподілене зберігання
MySQL	Теоретично необмежений (на практиці — кілька ТБ на таблицю)	Обмеження пов'язані з файловою системою та типом таблиць (наприклад, InnoDB)

У таблиці 1.3 подано порівняння підтримки транзакцій. PostgreSQL забезпечує повну підтримку транзакцій відповідно до ACID-принципів, включаючи вкладені транзакції та контроль цілісності даних. У MongoDB транзакційна модель з'явилася лише в новіших версіях і має певні обмеження, тоді як у MySQL вона залежить від типу таблиць і не є уніфікованою для всієї системи.

Таблиця 1.3

Підтримка транзакцій

СУБД	Підтримка транзакцій	Особливості реалізації
MongoDB	Обмежена	Повноцінні транзакції з'явилися лише в останніх версіях і не є типовими

Продовження таблиці 1.3

PostgreSQL	Повна (ACID)	Надійна транзакційна модель із підтримкою вкладених транзакцій
MySQL	Часткова	Залежить від типу таблиць (наприклад, лише для InnoDB, але не для MyISAM)

Що стосується підтримки складних запитів (табл. 1.4), PostgreSQL демонструє беззаперечну перевагу, оскільки підтримує всі необхідні SQL-операції, включаючи вкладені підзапити, віконні функції, агрегацію та гнучку роботу з JSON-даними. У MongoDB складні зв'язки реалізуються через специфічні агрегатори, як-от \$lookup, що може ускладнювати логіку запитів і знижувати продуктивність. MySQL також підтримує складні запити, проте менш гнучкий при роботі з нетипізованими структурами даних.

Таблиця 1.4

Підтримка складних запитів

СУБД	Підтримка JOIN-запитів	Підтримка агрегації	Особливості роботи зі складною структурою
MongoDB	Обмежена	Є	JOIN-операції моделюються через \$lookup, що ускладнює запити
PostgreSQL	Повна	Повна	Потужна мова запитів, підтримка підзапитів, віконних функцій
MySQL	Повна	Повна	Підтримує складні SQL-запити, але менш гнучкий при роботі з JSON

Таким чином, з урахуванням усіх порівняльних характеристик PostgreSQL було обґрунтовано обрано як основну СУБД для розробки інтернет-магазину ювелірних виробів, що потребує високої надійності, структурованості даних, транзакційності та масштабованості.

Висновки до першого розділу

У першому розділі було проведено аналіз предметної області електронної комерції, зокрема ринку ювелірних виробів в онлайн-сегменті. Розглянуто актуальність створення інтернет-магазину як ефективного інструменту для просування продукції, підвищення доступності товарів та автоматизації продажів. Наведено приклади існуючих рішень у цій сфері, проаналізовано їхні сильні та слабкі сторони, що дозволило визначити ключові функціональні вимоги до майбутнього вебзастосунку.

Було обґрунтовано вибір архітектури клієнт-серверного типу з реалізацією REST API на основі фреймворку Next.js. Такий підхід забезпечує ефективний розподіл обов'язків між клієнтською та серверною частинами, що сприяє гнучкості, масштабованості й безпеці застосунку. Окрему увагу приділено вибору інструментальних засобів розробки: використано сучасні бібліотеки для інтерфейсу, візуалізації, автентифікації, роботи з базою даних і взаємодії з сервером.

На основі порівняльного аналізу різних систем керування базами даних обґрунтовано вибір PostgreSQL як основної СУБД для проєкту. Вона забезпечує підтримку складних зв'язків, транзакційність, високу продуктивність і гнучкість у моделюванні структури даних, що є критично важливим у контексті інтернет-магазину з великою кількістю взаємопов'язаних об'єктів.

Таким чином, за результатами першого розділу було сформовано технічне й концептуальне підґрунтя для реалізації функціонального, надійного та зручного вебзастосунку для продажу ювелірних виробів в онлайн-середовищі.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				22
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ ЮВЕЛІРНИХ ВИРОБІВ

2.1. Визначення варіантів використання та структури системи

Інтернет-магазин ювелірних виробів реалізовано як багаторівневу систему з розмежованим доступом до функціональності залежно від ролі користувача. Основу платформи становлять модулі, що забезпечують перегляд каталогу продукції, керування кошиком, оформлення замовлень, обробку відгуків, а також адміністрування вмісту та аналітики. Вся інформація зберігається у структурованій базі даних, що гарантує цілісність, узгодженість і доступність даних незалежно від поточного стану взаємодії користувача з сайтом.

Гість — це користувач, який не здійснив реєстрацію або не виконав вхід до облікового запису. Йому надається доступ до відкритої частини магазину, де він може ознайомитися з асортиментом прикрас, скористатися фільтрами для пошуку необхідних позицій, переглянути детальний опис кожного товару та перейти до реєстрації або авторизації, якщо бажає продовжити процес оформлення замовлення чи зберегти обрані позиції.

Після реєстрації та входу користувач переходить у статус зареєстрованого клієнта. У цьому режимі він отримує доступ до розширеного функціоналу: може додавати товари до кошика, формувати замовлення з вибором варіанта оплати та доставки, залишати відгуки про придбану продукцію, переглядати історію попередніх покупок, відстежувати статус замовлень, а також змінювати особисті дані, такі як ім'я, електронна пошта чи адреса доставки. Цей рівень є ключовим для забезпечення персоналізованого досвіду користування платформою.

Адміністратор виступає як внутрішній користувач системи з розширеними правами доступу. Його функціональність включає керування товарним каталогом, у тому числі створення нових позицій, оновлення характеристик прикрас, їх видалення та зміна структури категорій. Крім того, він має змогу створювати та активувати акційні пропозиції, обробляти замовлення з урахуванням їхніх статусів, переглядати деталі кожної транзакції та отримувати

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				23
Змн.	Арк.	№ докум.	Підпис	Дата		

доступ до аналітичних звітів щодо продажів, популярності товарів і поведінки користувачів. Адміністратор також контролює роботу з обліковими записами, зокрема має можливість блокування, редагування чи видалення профілів у разі виявлення порушень.

На рис. 2.1 наведено діаграму варіантів використання інтернет-магазину, яка ілюструє взаємодію кожного типу користувачів із системою. Зображення наочно демонструє, як саме розподіляється функціональність між гістьовим режимом, зареєстрованим профілем і адміністративною панеллю.

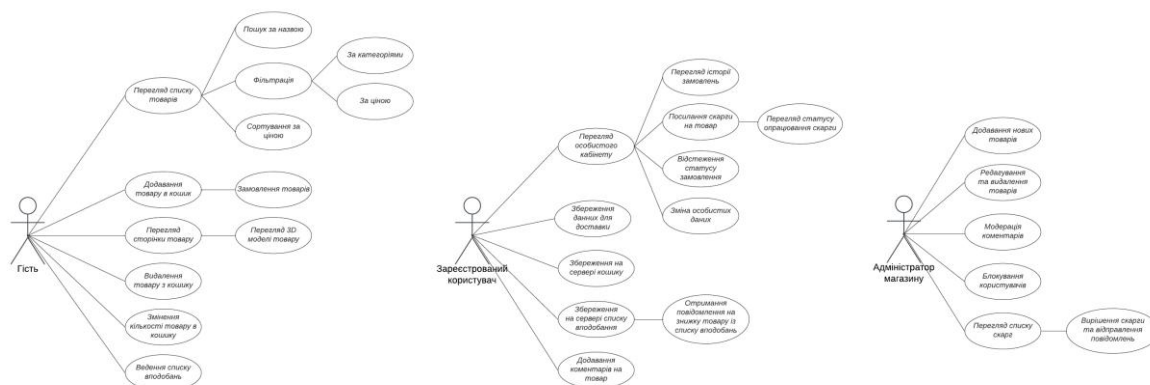


Рис. 2.1. Діаграма варіантів використання інтернет-магазину ювелірних виробів.

Запропонована структура дозволяє реалізувати ефективну взаємодію користувача з системою, гарантує зручність при здійсненні покупок, а також надає адміністрації повний контроль над усіма критично важливими процесами — від формування каталогу до аналізу результатів продажу.

Задля повноцінного функціонування інтернет-магазину ювелірних виробів у системі визначено як функціональні, так і нефункціональні вимоги. Функціональні вимоги формують основу взаємодії користувачів із платформою та відображають її прикладну цінність. Важливою частиною функціоналу є авторизація, що передбачає реєстрацію та вхід користувачів із поділом на три ролі: гість, зареєстрований покупець та адміністратор. Кожен із них отримує відповідний рівень доступу до можливостей сайту залежно від прав.

Ключовим модулем системи є товарний каталог, який дозволяє відвідувачам переглядати продукцію з можливістю застосування фільтрів,

сортування та повнотекстового пошуку. Окрему увагу приділено реалізації 3D-візуалізації прикрас, що підвищує рівень довіри користувачів до товару та дозволяє детально ознайомитися з його зовнішнім виглядом. Модуль кошика забезпечує зручну роботу з обраними товарами: користувач може додавати або видаляти позиції, змінювати кількість та переходити до оформлення замовлення з вибором параметрів доставки й оплати. Особистий кабінет відкриває доступ до управління персональними даними, перегляду історії замовлень, оновлення контактної інформації та роботи зі списком вподобаних товарів.

У системі передбачено можливість комунікації з адміністрацією через модуль подання скарг із відображенням поточного статусу звернення. Крім того, реалізовано функціонал коментування товарів, який сприяє формуванню соціального довіри та дає новим відвідувачам змогу орієнтуватися на думку інших клієнтів. Адміністративна частина містить інструменти для керування товарними позиціями, перевірки й модерації відгуків, управління профілями користувачів та реагування на скарги. Таким чином, забезпечується повний цикл управління контентом і обслуговування замовлень із боку персоналу магазину.

Окрім функціональних вимог, важливу роль відіграють нефункціональні характеристики системи, які визначають її якість, надійність і зручність використання. Зокрема, встановлено вимоги до продуктивності: сторінки платформи повинні завантажуватися не довше ніж за дві секунди навіть при значному навантаженні, а система має стабільно обслуговувати до тисячі одночасних користувачів без втрати швидкодії. Щодо надійності, допускається проведення технічного обслуговування в межах п'яти відсотків від загального часу роботи платформи, що дозволяє гарантувати її високу доступність протягом усього життєвого циклу.

Інтерфейс платформи має бути інтуїтивно зрозумілим, з чіткою логікою навігації, що дозволить користувачам швидко адаптуватися до роботи з нею без потреби в додатковому навчанні. Таким чином, загальна архітектура системи базується не лише на розгалуженій прикладній логіці, але й на ретельно сформульованих нефункціональних вимогах, які забезпечують стабільну,

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				25
Змн.	Арк.	№ докум.	Підпис	Дата		

безпечну та зручну взаємодію між користувачами і функціоналом онлайн-магазину.

Аналіз функціональних вимог дозволив виокремити основні сутності, необхідні для реалізації бізнес-логіки інтернет-магазину ювелірних виробів. До них належать користувачі, товари, категорії, замовлення, кошик, адреси доставки, коментарі, оплати, 3D-моделі та обране. На основі цього сформовано модель даних, реалізовану за допомогою ORM-бібліотеки Prisma, яка відображає структуру об'єктів і зв'язки між ними.

Центральною сутністю є User, який може мати адреси, кошик, замовлення, улюблені товари та коментарі. Товари зв'язані з категоріями, мають зображення, 3D-моделі, коментарі, а також входять до складу замовлень. Кожне Order включає перелік товарів, адресу доставки та інформацію про оплату. Всі зв'язки реалізовано через чітко визначені відношення у базі даних.

На рисунку 2.2 представлено діаграму класів, яка відображає основні сутності, їхні атрибути та взаємозв'язки в межах системи.

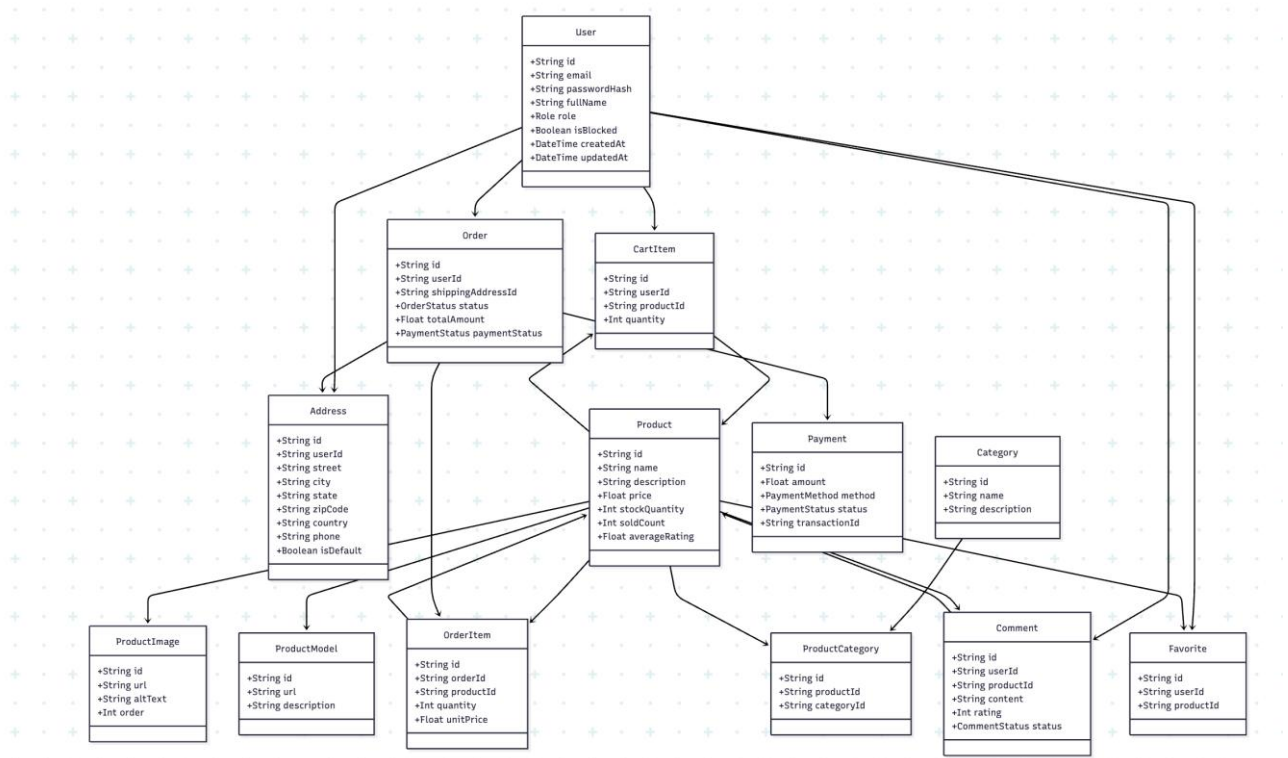


Рис. 2.2. Діаграма класів інтернет-магазину ювелірних виробів.

2.2 Розробка бази даних

Одним із ключових етапів створення інтернет-магазину ювелірних виробів є проєктування реляційної бази даних, яка повинна забезпечувати надійне збереження, цілісність та швидкий доступ до всієї інформації, необхідної для функціонування системи — від користувачів і товарів до замовлень, платежів та відгуків. База даних має враховувати як функціональні, так і нефункціональні вимоги системи, зокрема масштабованість, безпеку, цілісність зв'язків між сутностями та підтримку транзакцій.

У рамках реалізації проєкту була обрана система керування базами даних PostgreSQL, яка поєднує високу продуктивність з можливістю обробки складних типів даних. Для опису структури бази даних застосовано ORM-бібліотеку Prisma, що дозволяє декларативно задавати моделі, зв'язки між ними та автоматично генерувати SQL-структури для створення таблиць і забезпечення зв'язності. Такий підхід значно спрощує супровід схеми та пришвидшує розгортання системи.

У реалізованій моделі бази даних інтернет-магазину ювелірних виробів передбачено повний набір сутностей, кожна з яких виконує свою чітко визначену функцію в межах інформаційної системи. Сутність User відповідає за зберігання облікових записів користувачів, включаючи електронну пошту, пароль, ім'я, роль (звичайний користувач або адміністратор) та стан блокування, а також містить зв'язки з адресами доставки, замовленнями, кошиком, відгуками та списком бажаного. Для зберігання інформації про адреси доставки використовується сутність Address, яка містить такі дані, як вулиця, місто, область, індекс, країна, номер телефону та позначення адреси як основної.

Категорії товарів реалізовані за допомогою сутності Category, яка дозволяє структурувати асортимент за типами прикрас (наприклад, сережки, каблучки, ланцюжки), зберігаючи назву та опис кожної категорії. Основною сутністю, що описує ювелірні вироби, є Product, яка містить назву, опис, ціну, кількість на складі, кількість продажів, середній рейтинг, а також зв'язки з іншими сутностями: зображеннями (ProductImage), 3D-моделями (ProductModel),

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				27
Змн.	Арк.	№ докум.	Підпис	Дата		

категоріями (ProductCategory), товарами в кошику (CartItem), відгуками (Comment) та обраними позиціями (Favorite). Зв'язок між товарами і категоріями реалізовано через проміжну сутність ProductCategory, яка дозволяє одному товару належати до кількох категорій та забезпечує унікальність пари товар-категорія.

Для зберігання фотографій товарів використовується сутність ProductImage, яка містить URL-адресу зображення, альтернативний текст і порядок відображення. Схожим чином, сутність ProductModel відповідає за збереження 3D-моделей прикрас, що дає змогу реалізувати функцію перегляду виробів у форматі 360° з можливістю інтерактивної взаємодії. Дані про додані в кошик товари відображаються в сутності CartItem, яка фіксує, який користувач і коли додав певну кількість конкретного товару.

Замовлення, оформлені користувачами, зберігаються в сутності Order, яка містить інформацію про користувача, адресу доставки, статус замовлення, загальну суму, статус оплати, дату створення та оновлення. Структура замовлення деталізується за допомогою OrderItem — сутності, яка описує перелік товарів у межах одного замовлення, включаючи їх кількість, одиничну та загальну ціну. Оплати відображаються через сутність Payment, яка фіксує суму, спосіб оплати (наприклад, банківська карта, PayPal або переказ), статус транзакції, ідентифікатор операції та дату створення.

Функціональність залишення відгуків реалізовано через сутність Comment, яка дає змогу користувачам оцінювати товари, залишати коментарі, вказувати рейтинг, а також передбачає статус модерації, дату публікації та інформацію про модератора, якщо такий є. Для реалізації списку бажаного передбачена сутність Favorite, яка пов'язує користувача з обраними товарами.

З метою наочного відображення даних, що використовуються в системі, на рисунку 2.3 представлено загальну структуру бази даних інтернет-магазину жіночого одягу. Діаграма ілюструє ключові сутності, взаємозв'язки між ними та принципи організації інформації в межах системи.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				28
Змн.	Арк.	№ докум.	Підпис	Дата		

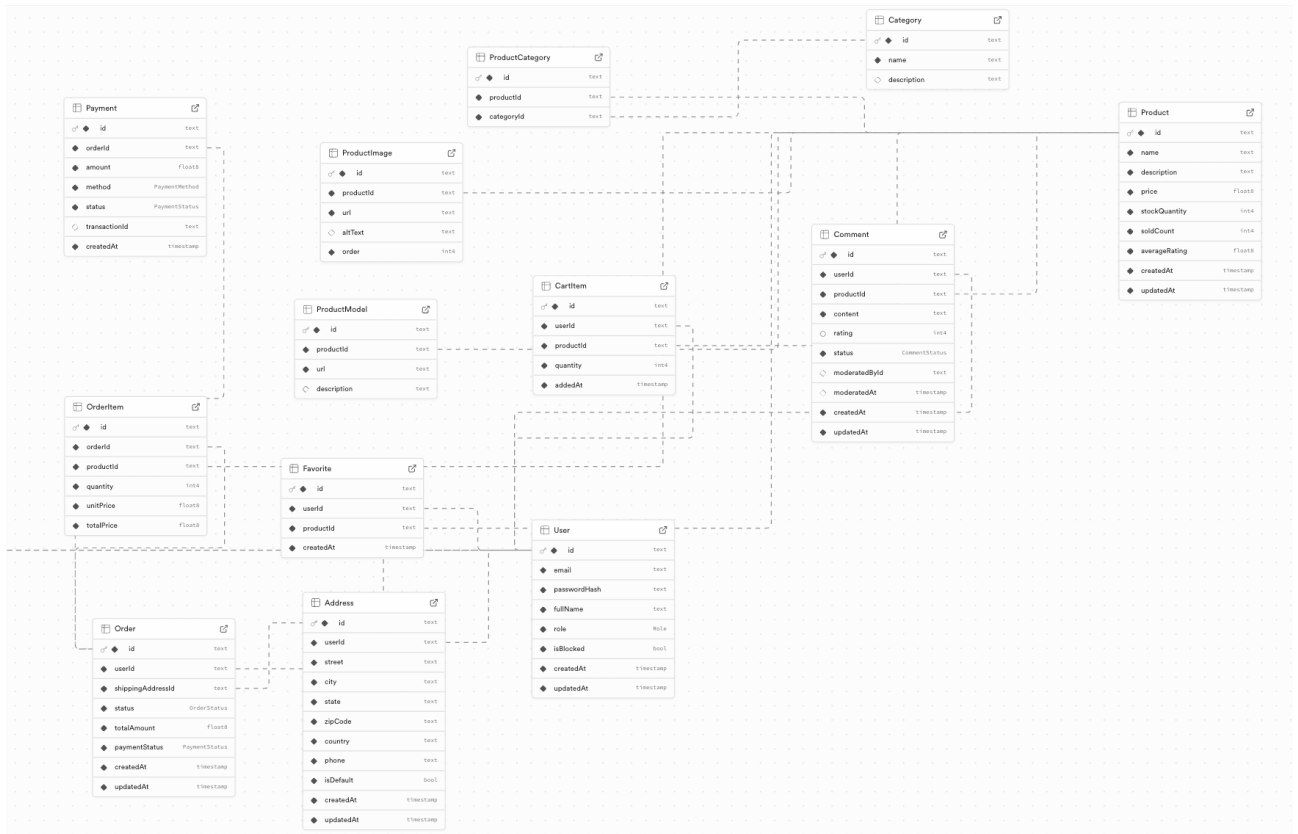


Рис. 2.3 Структура бази даних

Після загального представлення структури, у подальших таблицях подано докладний опис кожної сутності: зазначено назви полів, типи даних, позначення первинних (PK) та зовнішніх (FK) ключів, а також визначено їх функціональне призначення.

Таблиця 2.1

Структура сутності "User"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор користувача
email	String	-	-	Адреса електронної пошти
passwordHash	String	-	-	Хеш пароля
fullName	String	-	-	Повне ім'я користувача
role	Enum	-	-	Роль користувача (USER або ADMIN)
isBlocked	Boolean	-	-	Ознака блокування облікового запису
createdAt	DateTime	-	-	Дата створення облікового запису

Продовження таблиця 2.1

updatedAt	DateTime	-	-	Дата останнього оновлення облікового запису
-----------	----------	---	---	---

Таблиця 2.2

Структура сутності "Address"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор адреси
userId	String	-	+	Посилання на користувача, до якого належить ця адреса
street	String	-	-	Назва вулиці
city	String	-	-	Назва міста
state	String	-	-	Область або регіон
zipCode	String	-	-	Поштовий індекс
country	String	-	-	Назва країни
phone	String	-	-	Номер телефону, пов'язаний з адресою
isDefault	Boolean	-	-	Позначення, чи є ця адреса основною
createdAt	DateTime	-	-	Дата створення запису про адресу
updatedAt	DateTime	-	-	Дата останнього оновлення запису про адресу

Таблиця 2.3

Структура сутності "Category"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор категорії
name	String	-	-	Назва категорії (наприклад, сережки, каблучки)

description	String?	-	-	Опис категорії (необов'язкове поле)
-------------	---------	---	---	-------------------------------------

Таблиця 2.4

Структура сутності "Product"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор товару
name	String	-	-	Назва ювелірного виробу
description	String	-	-	Детальний опис товару
price	Float	-	-	Вартість одиниці товару
stockQuantity	Int	-	-	Кількість одиниць на складі
soldCount	Int	-	-	Кількість проданих одиниць
averageRating	Float	-	-	Середній рейтинг товару за відгуками
createdAt	DateTime	-	-	Дата створення товару
updatedAt	DateTime	-	-	Дата останнього оновлення інформації про товар

Таблиця 2.5

Структура сутності "ProductCategory"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор запису зв'язку
productId	String	-	+	Посилання на товар
categoryId	String	-	+	Посилання на категорію

Таблиця 2.6

Структура сутності "ProductImage"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор зображення
productId	String	-	+	Посилання на товар, до якого належить зображення
url	String	-	-	Посилання на зображення
altText	String?	-	-	Альтернативний текст для зображення (необов'язковий)
order	Int	-	-	Порядковий номер для сортування зображень

Таблиця 2.7

Структура сутності "ProductModel"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор 3D-моделі
productId	String	-	+	Посилання на товар, до якого належить 3D-модель
url	String	-	-	URL-посилання на файл 3D-моделі
description	String?	-	-	Опис або примітки щодо 3D-моделі (необов'язкове поле)

Таблиця 2.8

Структура сутності "CartItem"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор запису у кошику
userId	String	-	+	Посилання на користувача, до якого належить кошик
productId	String	-	+	Посилання на товар, доданий у кошик
quantity	Int	-	-	Кількість одиниць товару у кошику
addedAt	DateTime	-	-	Дата та час додавання товару до кошика

Таблиця 2.9

Структура сутності "Order"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор замовлення
userId	String	-	+	Посилання на користувача, що зробив замовлення
shippingAddressId	String	-	+	Адреса доставки, вказана при оформленні замовлення
status	Enum	-	-	Статус замовлення (PENDING, PROCESSING, SHIPPED, DELIVERED, CANCELLED)
totalAmount	Float	-	-	Загальна сума замовлення
paymentStatus	Enum	-	-	Статус платежу (PENDING, PAID, FAILED, REFUNDED)
createdAt	DateTime	-	-	Дата створення замовлення

Продовження таблиці 2.9

updatedAt	DateTime	-	-	Дата останнього оновлення інформації про замовлення
-----------	----------	---	---	---

Таблиця 2.10

Структура сутності "OrderItem"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор позиції
orderId	String	-	+	Посилання на замовлення, до якого належить ця позиція
productId	String	-	+	Посилання на товар, що був замовлений
quantity	Int	-	-	Кількість одиниць замовленого товару
unitPrice	Float	-	-	Ціна однієї одиниці товару на момент замовлення
totalPrice	Float	-	-	Загальна вартість ($\text{unitPrice} \times \text{quantity}$)

Таблиця 2.11

Структура сутності "Payment"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор платежу
orderId	String	-	+	Посилання на замовлення, до якого належить цей платіж
amount	Float	-	-	Сума платежу
method	Enum	-	-	Метод оплати (CREDIT_CARD, PAYPAL, BANK_TRANSFER)
status	Enum	-	-	Статус платежу (PENDING, PAID, FAILED, REFUNDED)

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				34
Змн.	Арк.	№ докум.	Підпис	Дата		

Продовження таблиця 2.11

transactionId	String?	-	-	Ідентифікатор транзакції (опціональне поле)
createdAt	DateTime	-	-	Дата та час створення платежу

Таблиця 2.12

Структура сутності "Comment"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор відгуку
userId	String	-	+	Посилання на користувача, який залишив відгук
productId	String	-	+	Посилання на товар, до якого належить відгук
content	String	-	-	Текст коментаря
rating	Int?	-	-	Рейтинг товару, виставлений користувачем (опціональне поле)
status	Enum	-	-	Статус модерації (PENDING, APPROVED, REJECTED)
moderatedById	String?	-	-	Ідентифікатор адміністратора, що модерував коментар (необов'язково)
moderatedAt	DateTime?	-	-	Час модерації відгуку (опціональне поле)
createdAt	DateTime	-	-	Дата створення відгуку
updatedAt	DateTime	-	-	Дата останнього оновлення коментаря

Структура сутності "Favorite"

Назва поля	Тип даних	PK	FK	Опис
id	String	+	-	Унікальний ідентифікатор запису у списку бажаного
userId	String	-	+	Посилання на користувача, який додав товар до обраного
productId	String	-	+	Посилання на товар, що був доданий до списку бажаного
createdAt	DateTime	-	-	Дата та час додавання товару до списку бажаного

Таким чином, проєктована база даних інтернет-магазину ювелірних виробів охоплює всі ключові аспекти функціонування системи — від управління користувачами та товарами до обробки замовлень, платежів, відгуків і списків бажаного. Кожна сутність має чітко визначену роль та побудована відповідно до принципів нормалізації, що сприяє підтриманню цілісності даних і гнучкості в подальшому розвитку платформи. Завдяки використанню ORM-бібліотеки Prisma забезпечено зручність у керуванні моделями, автоматичну генерацію SQL-структур і тісну інтеграцію з системою керування базами даних PostgreSQL. Ретельно спроектована схема дозволяє легко масштабувати систему, підключати нові функціональні модулі та забезпечує високу продуктивність при роботі з великими обсягами даних.

2.3. Проєктування та реалізація алгоритмів роботи

Для реалізації вебзастосунку інтернет-магазину ювелірних виробів була обрана клієнт-серверна архітектура з використанням сучасного фреймворку Next.js 15, що забезпечує гнучке поєднання серверного рендерингу (SSR), статичної генерації сторінок (SSG), клієнтського оновлення інтерфейсу та використання React Server Components. Такий підхід дозволяє досягти високої

продуктивності, швидкого завантаження сторінок і зручності масштабування застосунку.

На відміну від підходів, заснованих на Server Actions, у цьому проєкті реалізацію серверної логіки здійснено за допомогою API-роутів, що розміщені в директорії `app/api`. Кожен API-ендпоінт відповідає за виконання окремої операції — створення замовлення, додавання товару до кошика, обробку платежу, модерацію відгуку тощо. Такий підхід забезпечує чітке розділення відповідальності, зручність тестування, а також сумісність із сторонніми клієнтами або мобільними додатками в майбутньому.

Загальний алгоритм роботи системи подано на рисунку 2.4. Він охоплює основні сценарії взаємодії користувача з інтернет-магазином: від авторизації та перегляду товарів до оформлення замовлення та здійснення оплати. Після входу в систему визначається роль користувача — "USER" або "ADMIN". Для звичайного користувача відкривається можливість переглядати каталог, здійснювати пошук і фільтрацію, додавати товари до кошика та оформлювати замовлення з вибором способу доставки й оплати. У разі успішної оплати формується підтвердження та завершується процес покупки. Якщо ж роль користувача — "ADMIN", то він отримує доступ до адміністративної панелі, де реалізовано функції управління товарами, категоріями, замовленнями, модерації відгуків і керування користувачами. Така логіка дозволяє чітко відокремити функціональні сценарії для різних ролей і забезпечити безпечну, послідовну роботу системи.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				37
Змн.	Арк.	№ докум.	Підпис	Дата		

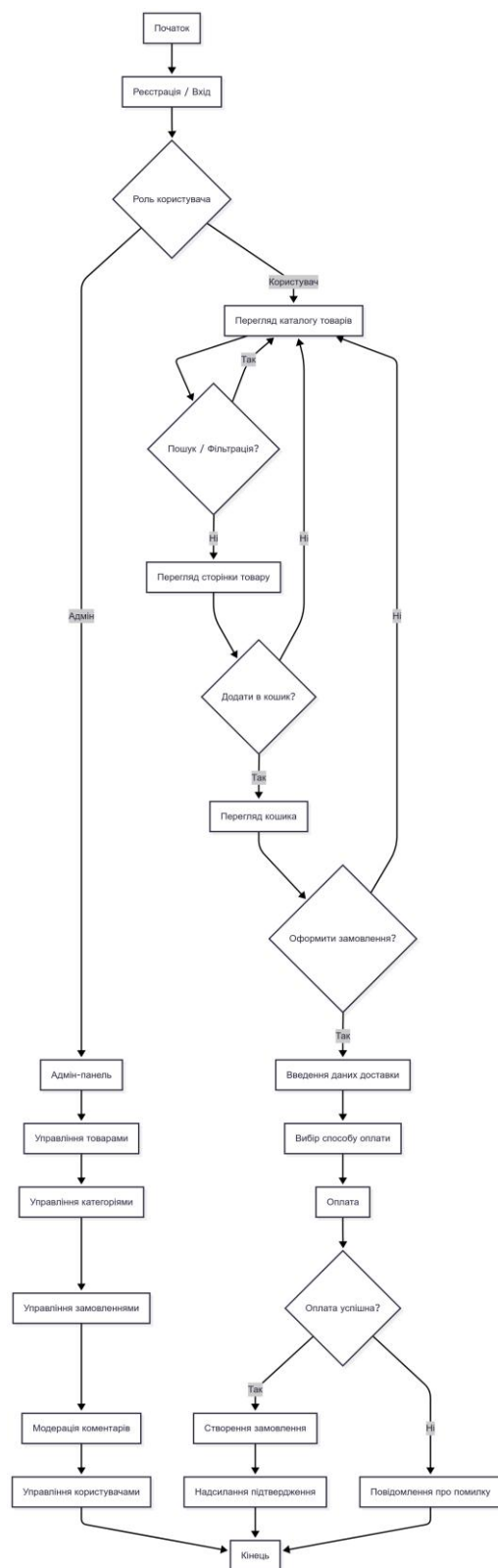


Рис. 2.4 – Блок-схема алгоритму взаємодії користувача з інтернет-магазином ювелірних виробів

Для кращого розуміння внутрішньої логіки процесу купівлі ювелірних виробів в інтернет-магазині була створена діаграма послідовності, що відображає основні дії користувача на шляху до оформлення замовлення. Вона

охоплює такі етапи, як пошук потрібного товару, додавання його до кошика, введення даних доставки, реєстрація та підтвердження електронної пошти (за потреби), проведення оплати й очікування підтвердження. У діаграмі задіяно ключові модулі системи — каталог товарів, кошик, замовлення, обробка даних користувача.

Використання такої схеми дозволяє наочно показати послідовність запитів і відповідей, логіку переходу між етапами та взаємодію між клієнтом і сервером. Це сприяє більш чіткому формуванню вимог до функціоналу, дозволяє уникнути логічних суперечностей під час реалізації, а також полегшує майбутнє тестування та підтримку системи.

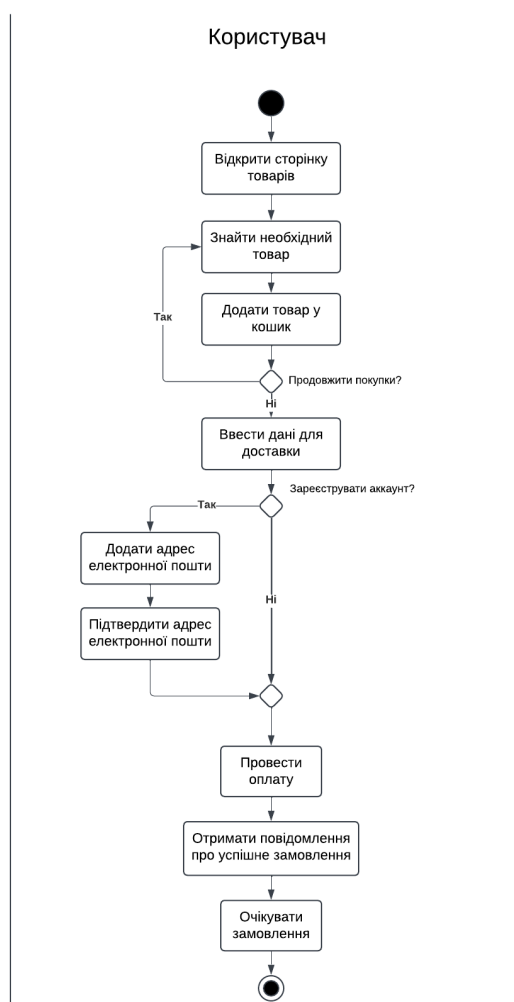


Рис. 2.5 – Діаграма послідовності: оформлення замовлення

На діаграмі послідовності, зображеній на рисунку 2.5, подано типовий сценарій оформлення замовлення в інтернет-магазині ювелірних виробів. Процес починається з перегляду товарів, пошуку потрібної позиції та додавання її до кошика. Після цього користувач переходить до етапу введення даних для доставки. Якщо обліковий запис ще не створено, система пропонує пройти реєстрацію та підтвердити електронну пошту.

Далі користувач переходить до оплати. У разі успішного платежу система генерує замовлення, надсилає підтвердження та переводить користувача в режим очікування доставки. Така діаграма дозволяє чітко відстежити послідовність дій і взаємодію з ключовими модулями системи — товарами, кошиком, реєстрацією, оплатою та оформленням покупки.

2.4 Реалізація функціоналу

Реалізація інтернет-магазину ювелірних виробів здійснена з використанням фреймворку Next.js 15, який поєднує переваги серверного та клієнтського рендерингу, забезпечуючи високу продуктивність і зручну організацію коду. Основна архітектура побудована на розділенні логіки між інтерфейсними компонентами та API-роутами, що обробляють запити до серверної частини. Такий підхід дозволив створити масштабовану систему з чітким поділом відповідальностей і можливістю подальшого розширення функціоналу.

Усі механізми автентифікації користувачів реалізовані за допомогою бібліотеки NextAuth, що забезпечує захищене зберігання сесій, підтримку JWT та можливість гнучкого налаштування провайдерів входу. У проєкті використовується провайдер "Credentials", що дозволяє користувачеві входити в систему за електронною поштою та паролем. Для перевірки облікових даних застосовується порівняння збереженого хешу пароля з введеним значенням за допомогою bcrypt. Крім того, до токена сесії додається роль користувача (USER або ADMIN), що надалі використовується для контролю доступу до різних частин інтерфейсу. Основна логіка автентифікації подана нижче:

```
async authorize(credentials) {
```

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				40
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    if (!credentials?.email || !credentials.password) {
      return null;
    }
    const user = await prisma.user.findUnique({
      where: { email: credentials.email },
    });
    if (!user) return null;

    const isValid = await bcrypt.compare(
      credentials.password,
      user.passwordHash
    );
    if (!isValid) return null;
    return {
      id: user.id,
      email: user.email,
      name: user.fullName,
      role: user.role,
    };
  }
}

```

Повний код конфігурації NextAuth із усіма параметрами, callback-функціями та реєстрацією обробників GET/POST наведено у Додатку А.

Для взаємодії з базою даних у проєкті використовується клієнт Prisma, ініціалізація якого реалізована з використанням патерна Singleton. Такий підхід дозволяє уникнути повторного створення екземпляра клієнта під час гарячого перезапуску у режимі розробки (hot-reloading), що є типовою проблемою при використанні Next.js. Клієнт зберігається в глобальному об'єкті `global.prisma`, і повторно створюється лише у разі його відсутності. Це гарантує стабільність підключення до бази даних протягом життєвого циклу застосунку. Нижче наведено фрагмент реалізації цього механізму:

```

export const prisma =
  global.prisma ||
  new PrismaClient({
    log: process.env.NODE_ENV === "development" ? ["query"] : [],
  });

if (process.env.NODE_ENV === "development") global.prisma = prisma;

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				41
Змн.	Арк.	№ докум.	Підпис	Дата		

Сама структура бази даних задається за допомогою Prisma Schema Language — декларативного синтаксису, що дозволяє наочно й формально описувати всі моделі даних, зв'язки між ними, обмеження, а також перелічувані типи (enum), які забезпечують чітку типізацію полів. У межах схеми було реалізовано повний набір сутностей, необхідних для функціонування інтернет-магазину ювелірних виробів. Зокрема, це користувачі з інформацією про роль та облікові дані, адреси доставки, каталог товарів із підтримкою категорій, зображень та 3D-моделей, позиції у кошику, замовлення з деталями кожного товару, платіжна інформація з методами та статусами, а також модуль коментарів і обране. Окрему увагу приділено службовим полям, які потрібні для роботи механізмів автентифікації (наприклад, зв'язок із сесіями та акаунтами). Завдяки структурованості та формалізованому підходу ця схема є основою всієї логіки взаємодії з базою даних, забезпечує цілісність інформації, її узгодженість і можливість розширення без порушення архітектури. Повний текст схеми наведено у Додатку А.

Уся серверна логіка інтернет-магазину реалізована через API-роути, розміщені в директорії app/api. Кожен з них відповідає за окремий функціональний блок: роботу з товарами, категоріями, кошиком, замовленнями, платежами, відгуками тощо. Роутинг реалізовано відповідно до принципів REST — з використанням методів GET, POST, PATCH, DELETE, а також із валідацією вхідних даних і обробкою помилок. В основі всіх маршрутів — єдиний підхід: запит аналізується, дані зчитуються або записуються через Prisma-клієнт, а відповідь формується у форматі JSON. Це забезпечує прозору, стабільну та розширювану архітектуру.

Типовий приклад — API-роут для обробки товарів. Він дозволяє отримати список товарів із фільтрацією за категоріями, ціновими діапазонами, пагінацією й сортуванням, а також додає нові товари разом із зображеннями та 3D-моделями. Нижче подано фрагмент реалізації обробки запиту GET:

```
export async function GET(request: Request) {
  const { searchParams } = new URL(request.url);
  const category = searchParams.get("category") ?? undefined;
```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				42
Змн.	Арк.	№ докум.	Підпис	Дата		

```

const priceMin = parseFloat(searchParams.get("priceMin") ?? "0");
const priceMax = parseFloat(
  searchParams.get("priceMax") ?? String(Number.MAX_VALUE)
);
const sortBy = (searchParams.get("sortBy") as keyof any) ??
"createdAt";
const order: "asc" | "desc" = (searchParams.get("order") as any)
?? "desc";
const page = Math.max(parseInt(searchParams.get("page") ?? "1",
10), 1);
const perPage =
Math.min(Math.max(parseInt(searchParams.get("perPage") ?? "10", 10), 1),
100);

const where: any = { price: { gte: priceMin, lte: priceMax } };
if (category) {
  where.categories = {
    some: { category: { name: { contains: category, mode:
"insensitive" } } },
  };
}

const [total, products] = await Promise.all([
  prisma.product.count({ where }),
  prisma.product.findMany({
    where,
    include: {
      images: true,
      models3D: true,
      categories: { include: { category: true } },
    },
    orderBy: { [sortBy]: order },
    skip: (page - 1) * perPage,
    take: perPage,
  }),
]);

return NextResponse.json({
  data: products,
  meta: { page, perPage, total, totalPages: Math.ceil(total /
perPage) },
});
}

```

Повна реалізація цього маршруту, включно з обробкою POST для створення товарів, завантаженням файлів через Vercel Blob, перевіркою категорій і збереженням зображень і 3D-моделей, наведена у Додатку А. Усі інші API-роути побудовані за схожим принципом і структурно уніфіковані.

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				43
Змн.	Арк.	№ докум.	Підпис	Дата		

Оплата замовлень у вебзастосунку реалізована за допомогою платіжного провайдера Stripe, що дозволяє здійснювати безпечні транзакції з використанням банківських карток та локалізованих методів оплати. Під час оформлення замовлення клієнтська частина ініціює запит до API-роуту `/api/create-payment-intent`, який створює об'єкт `PaymentIntent` у системі Stripe. Для коректної обробки суми створено допоміжну функцію `convertToSubcurrency`, що конвертує значення у копійки (оскільки Stripe працює з найменшими грошовими одиницями). Нижче наведено приклад реалізації відповідного API-роуту:

```
import { NextRequest, NextResponse } from "next/server";
const stripe = require("stripe")(process.env.STRIPE_SECRET_KEY);

export async function POST(request: NextRequest) {
  try {
    const { amount } = await request.json();

    const paymentIntent = await stripe.paymentIntents.create({
      amount: amount,
      currency: "uah",
      automatic_payment_methods: { enabled: true },
    });

    return NextResponse.json({
      paymentIntent: paymentIntent,
      clientSecret: paymentIntent.client_secret });
  } catch (error) {
    console.error("Internal Error:", error);
    return NextResponse.json(
      { error: `Internal Server Error: ${error}` },
      { status: 500 }
    );
  }
}
```

Висновки до другого розділу

У другому розділі було здійснено комплексне проектування інтернет-магазину ювелірних виробів, починаючи з побудови архітектури системи та закінчуючи реалізацією її ключових функціональних модулів. Було визначено основні ролі користувачів (звичайний користувач та адміністратор), розроблено логіку їх взаємодії з інтерфейсом, а також побудовано діаграми, що відображають послідовність дій при оформленні замовлення. Структура реляційної бази даних створена з урахуванням бізнес-вимог та охоплює всі

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				44
Змн.	Арк.	№ докум.	Підпис	Дата		

сутності, необхідні для повноцінної роботи платформи — товари, категорії, кошик, замовлення, платежі, відгуки, облікові записи користувачів та інші допоміжні об'єкти.

Крім того, у розділі детально описано реалізацію основної серверної логіки за допомогою API-роутів Next.js, що забезпечують обробку запитів на створення, читання та фільтрацію даних. Розглянуто механізми автентифікації через NextAuth, використання Stripe для обробки платежів, а також застосування патерна Singleton для стабільної роботи з клієнтом бази даних Prisma. Такий підхід забезпечив гнучкість системи, простоту масштабування та високу відповідність функціоналу поставленим вимогам.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				45
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДОДАТКУ

3.1 Порядок встановлення та налаштування системи

Діаграма розгортання виступає візуальним відображенням фізичної архітектури системи — розміщення її компонентів у середовищі виконання та способів взаємодії між ними. Це дозволяє оцінити конфігурацію мережевої інфраструктури, розподіл обов'язків між клієнтом, сервером і зовнішніми службами. На рисунку 3.1 наведено діаграму розгортання інтернет-магазину ювелірних виробів, створеного з використанням фреймворку Next.js, серверного API, бази даних PostgreSQL та інтеграцій із зовнішніми сервісами, такими як Stripe, Vercel Blob Storage і поштовий сервіс. Система побудована за класичною клієнт-серверною архітектурою з використанням серверного рендерингу (SSR) та API-роутів на стороні Next.js-застосунку.

Взаємодія розпочинається на стороні клієнта — користувач працює через веббраузер, який звертається до застосунку через HTTP/HTTPS. На рівні серверного застосунку Next.js виконується рендеринг сторінок і обробка API-запитів. React-компоненти динамічно отримують дані з API, які реалізовані як серверні функції у файлах `app/api`. Ці маршрути відповідають за доступ до бази даних PostgreSQL через Prisma, а також за взаємодію з іншими зовнішніми компонентами. Для обробки завантажень зображень та 3D-моделей використовується хмарне сховище Vercel Blob Storage, до якого застосунок надсилає запити на збереження й отримання файлів. Процес оплати реалізовано через Stripe Payment Gateway, до якого застосунок надсилає HTTPS-запити для створення платіжних намірів і підтвердження транзакцій. Окремо передбачена взаємодія з поштовим сервісом, який надсилає листи-підтвердження замовлень. Така архітектура забезпечує модульність, масштабованість і надійність роботи застосунку.

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				46
Змн.	Арк.	№ докум.	Підпис	Дата		

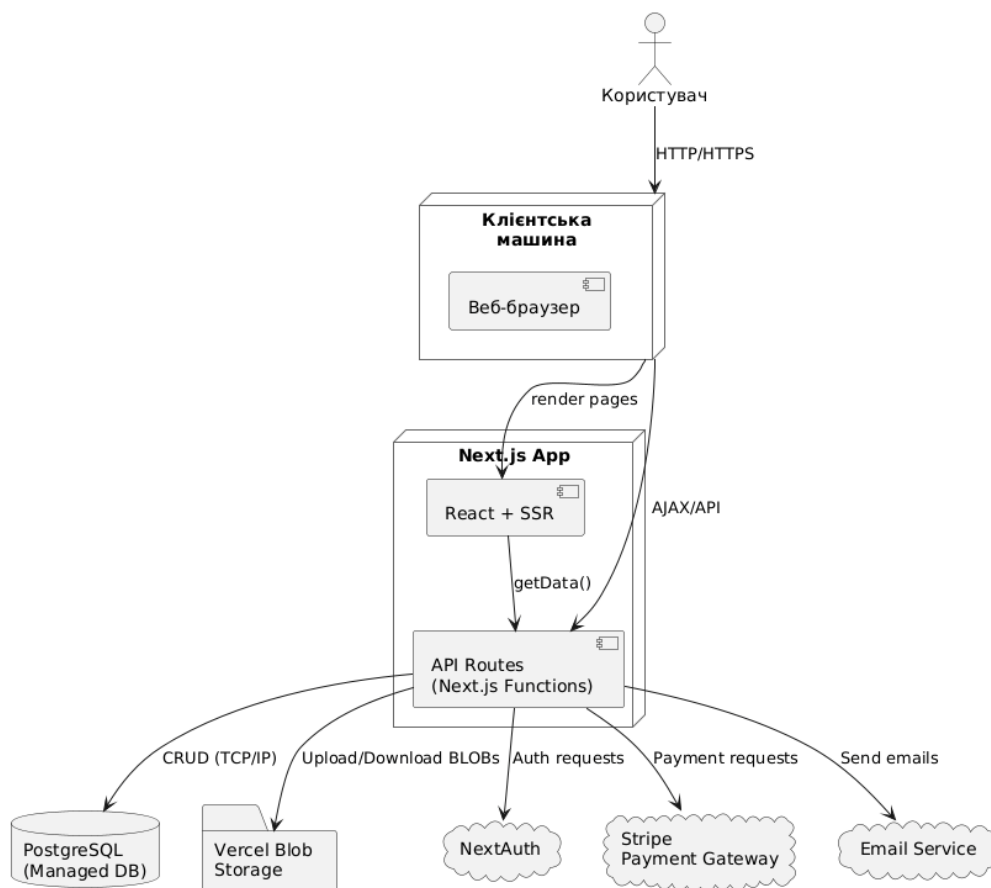


Рис. 3.1 – Діаграма розгортання інтернет-магазину ювелірних виробів

3.2 Структура інтерфейсу та порядок взаємодії з інтернет-магазином ювелірних виробів

Після відкриття сайту користувач потрапляє на головну сторінку (рис. 3.2). Тут відображається каталог товарів, пошуковий рядок, фільтри та меню навігації. У верхній частині інтерфейсу розміщені посилання на розділи сайту, а також кнопки входу, реєстрації, доступ до кошика та обраного. Інтерфейс підтримує адаптивність для мобільних пристроїв. Фільтри дозволяють обрати категорію, задати мінімальну й максимальну ціну та вибрати порядок сортування. Кожен товар подано у вигляді картки з фото, назвою, ціною, рейтингом і кнопкою «Додати до кошика». Перехід на детальну сторінку виробу здійснюється за натисканням на картку.

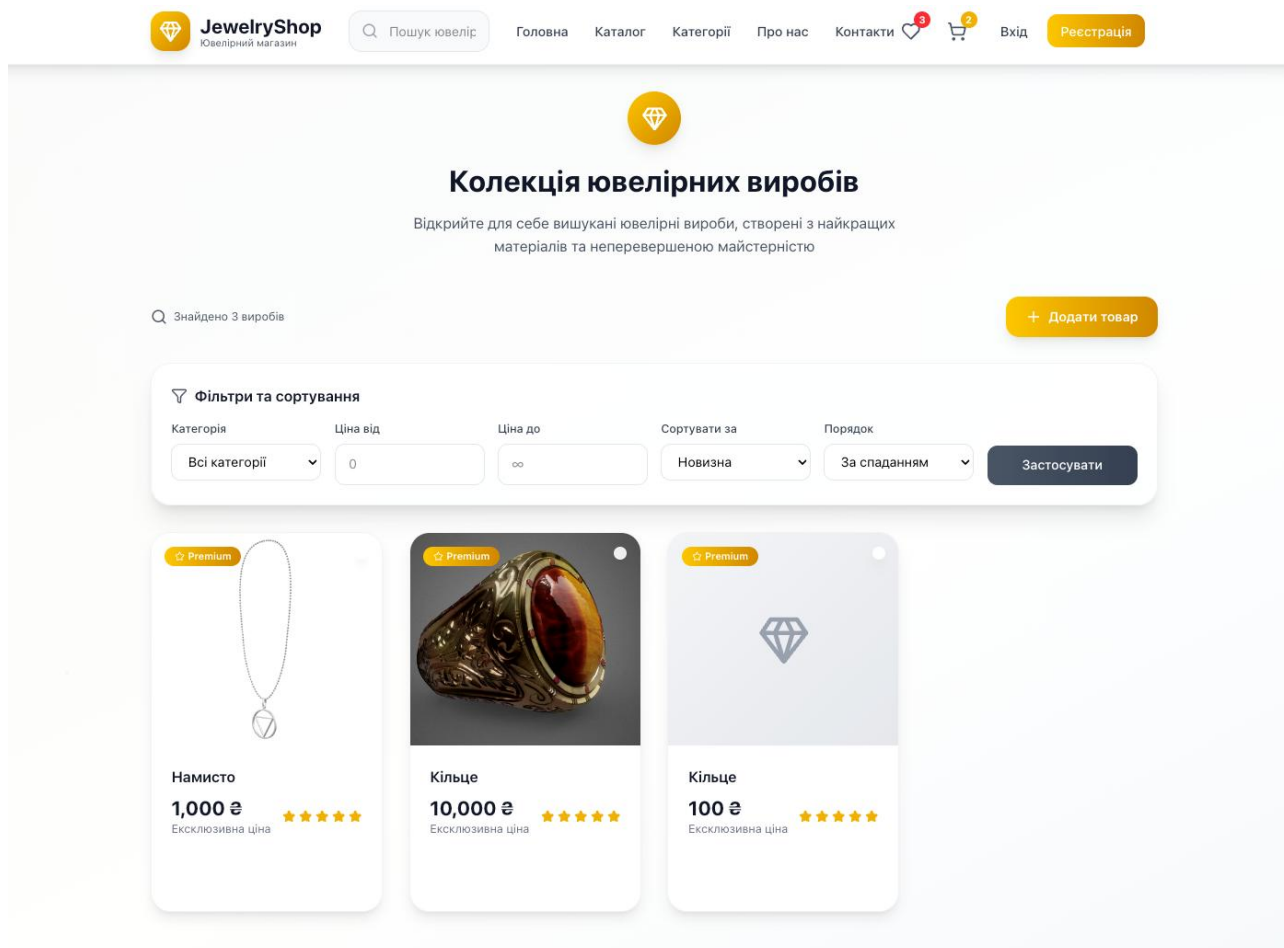


Рис. 3.2 – Головна сторінка інтернет-магазину ювелірних виробів

Неавторизований користувач може вільно переглядати каталог і переходити на сторінку товару, де відображаються детальні зображення, характеристики та інтерактивна 3D-модель виробу (рис. 3.3). Це дозволяє ознайомитися з товаром без потреби входу в систему. Проте, якщо користувач намагається додати товар до кошика або скористатися функцією «Купити в один клік», система перенаправляє його на сторінку авторизації (рис. 3.4). Така логіка забезпечує захист дій, що пов'язані з оформленням замовлень, і дозволяє зберігати історію покупок у межах облікового запису. Після успішного входу користувач повертається до вибраного товару й може завершити оформлення покупки.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				48
Змн.	Арк.	№ докум.	Підпис	Дата		

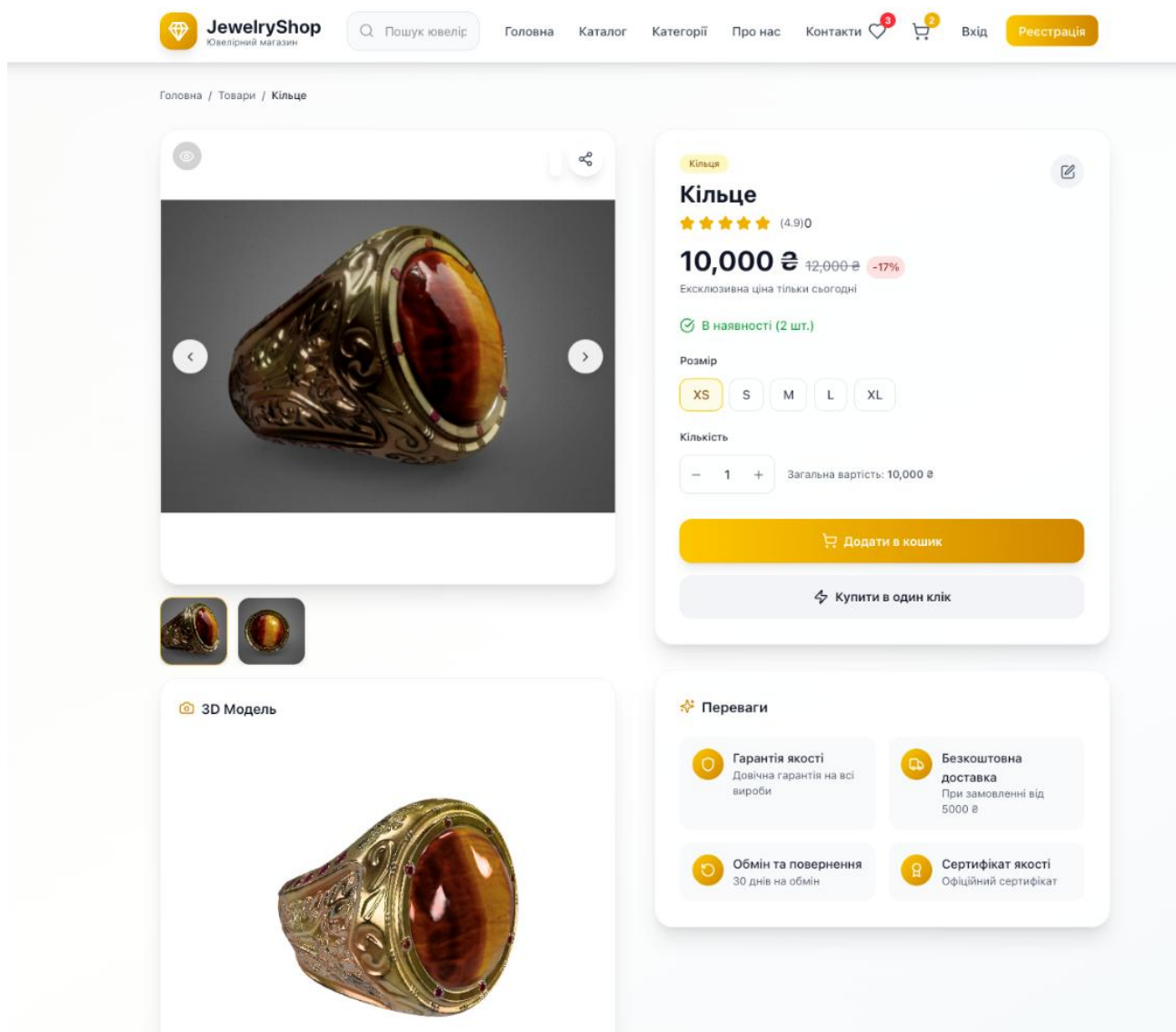


Рис. 3.3 – Сторінка перегляду товару

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				49
Змн.	Арк.	№ докум.	Підпис	Дата		

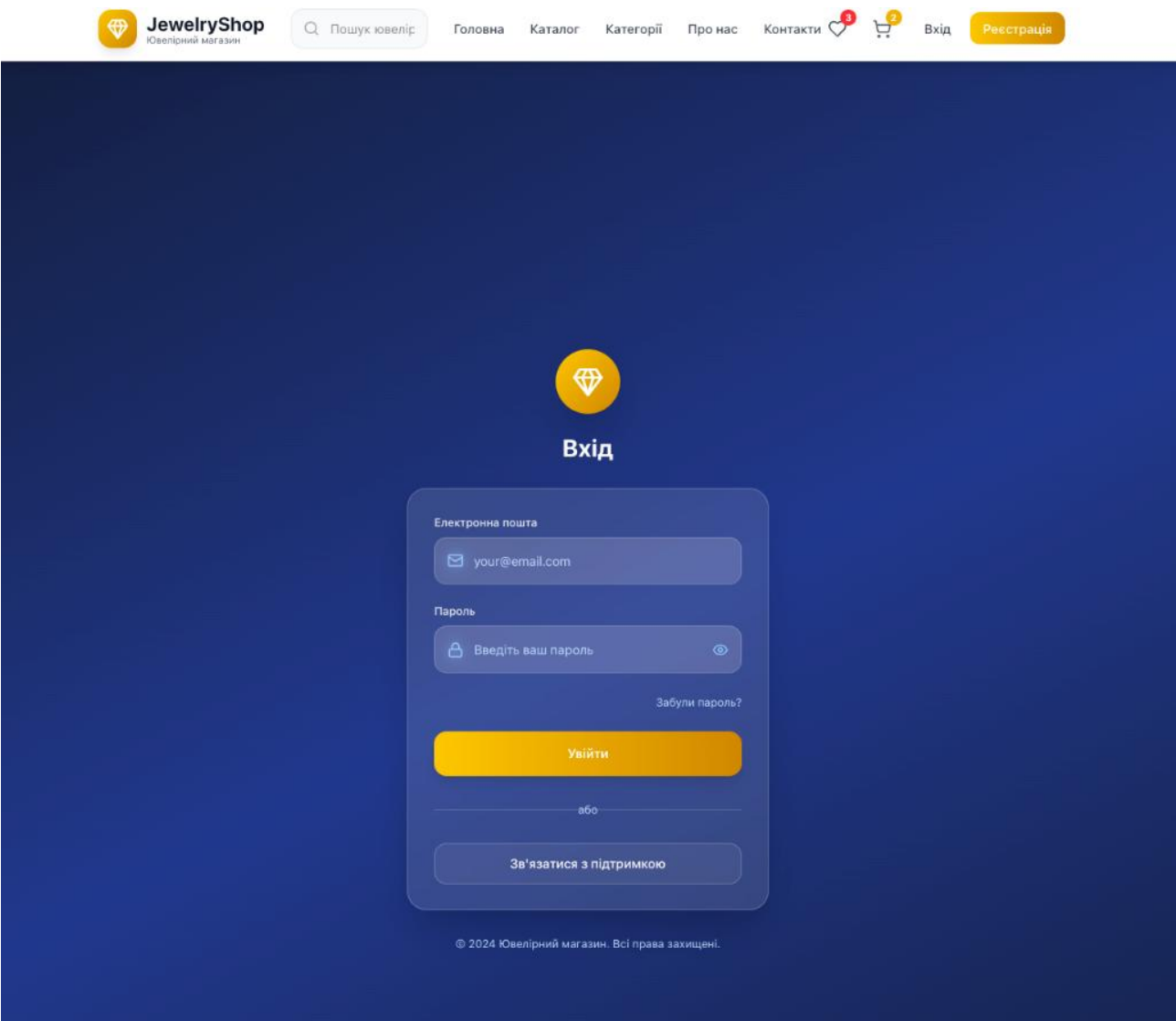


Рис. 3.4 – Сторінка авторизації користувача

Якщо у користувача ще немає облікового запису, система пропонує пройти реєстрацію. Для цього необхідно вказати повне ім’я, електронну пошту та задати надійний пароль із підтвердженням (рис. 3.5). Після успішної реєстрації користувач отримує доступ до особистого кабінету, можливості додавати товари до кошика, оформлювати замовлення та переглядати історію покупок.


JewelryShop
Ювелірний магазин

[Головна](#)
[Каталог](#)
[Категорії](#)
[Про нас](#)
[Контакти](#)

 3

 2

[Вхід](#)
[Реєстрація](#)



Створити акаунт

Приєднайтесь до нашої ювелірної спільноти

Повне ім'я

Електронна пошта

Пароль



Підтвердження пароля



Вимоги до пароля:

- Мінімум 8 символів
- Одна велика літера
- Одна цифра

Зареєструватися

або

Вже маєте акаунт? [Увійти](#)

Рис. 3.5 – Сторінка реєстрації нового користувача

Після авторизації користувач може додавати товари до списку вподобаного. На відповідній сторінці (рис. 3.6) відображаються всі збережені позиції з можливістю перейти на сторінку товару або прибрати його з переліку. Це дозволяє зручно зберігати вибране для подальшої покупки або перегляду.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				51
Змн.	Арк.	№ докум.	Підпис	Дата		

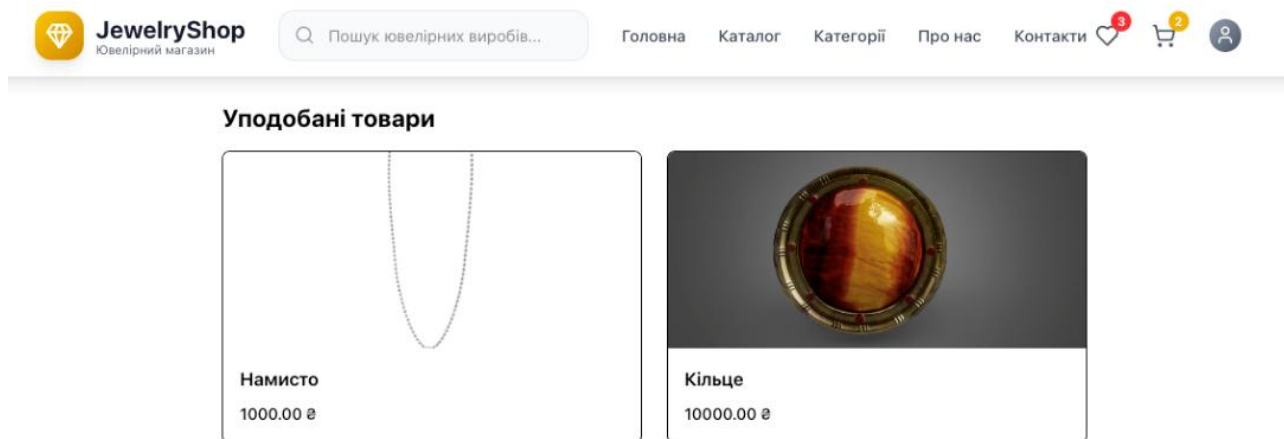


Рис. 3.6 – Список уподобаних товарів авторизованого користувача

Авторизований користувач може додавати товари до кошика для подальшого оформлення замовлення. У кошику відображається список обраних товарів, їх кількість, підсумкова сума та умови доставки (рис. 3.7). Користувач може змінювати кількість одиниць або повністю видалити товар із переліку. Після перевірки замовлення система дозволяє перейти до етапу оформлення з вибором способу доставки й оплати.

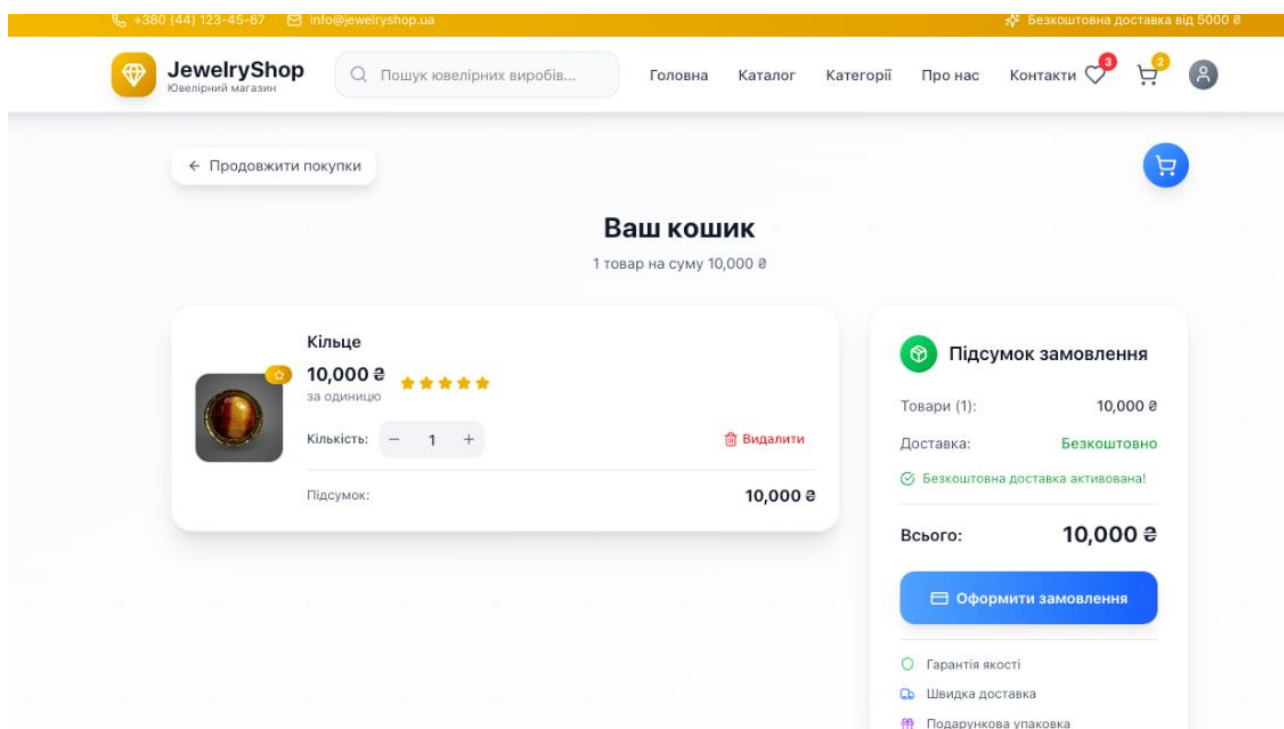


Рис. 3.7 – Сторінка кошика користувача з підсумком замовлення

На сторінці оформлення замовлення користувач бачить загальний список обраних товарів, адресу доставки, тип доставки та спосіб оплати (рис. 3.8). Усі поля заповнюються вручну або обираються збережені дані. Система

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				52
Змн.	Арк.	№ докум.	Підпис	Дата		

автоматично підраховує суму до сплати з урахуванням вартості доставки. Після перевірки всіх даних користувач натискає кнопку «Підтвердити замовлення» для завершення покупки.

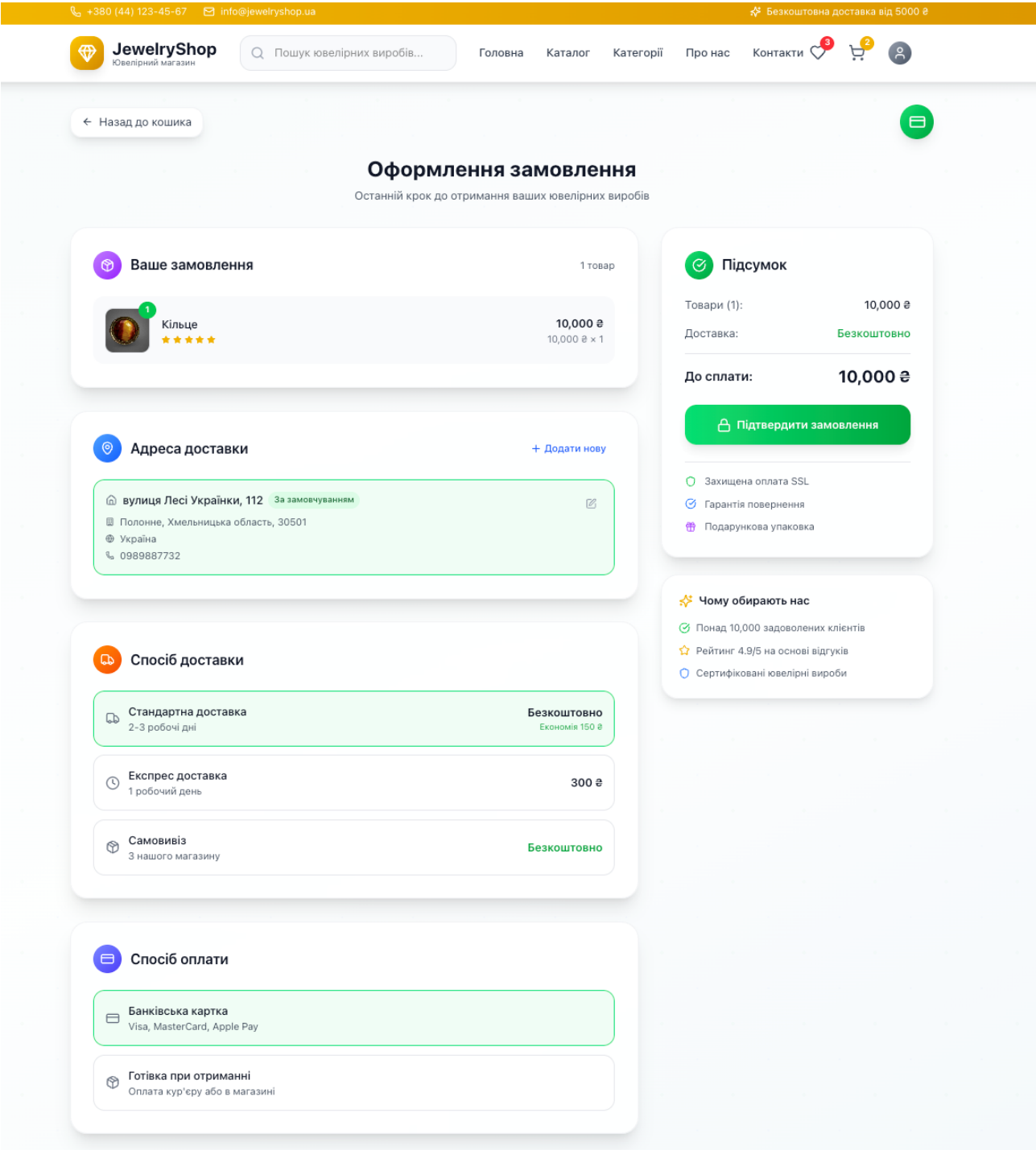


Рис. 3.8 – Сторінка оформлення замовлення з вибором доставки та оплати

Після оформлення замовлення користувача автоматично перенаправляє на сторінку «Мої замовлення» (рис. 3.9). Тут відображається історія всіх покупок, їхній статус, загальна сума та дата оформлення. Користувач може переглянути

деталі кожного замовлення, включно з товарами, адресою доставки та поточним станом обробки.

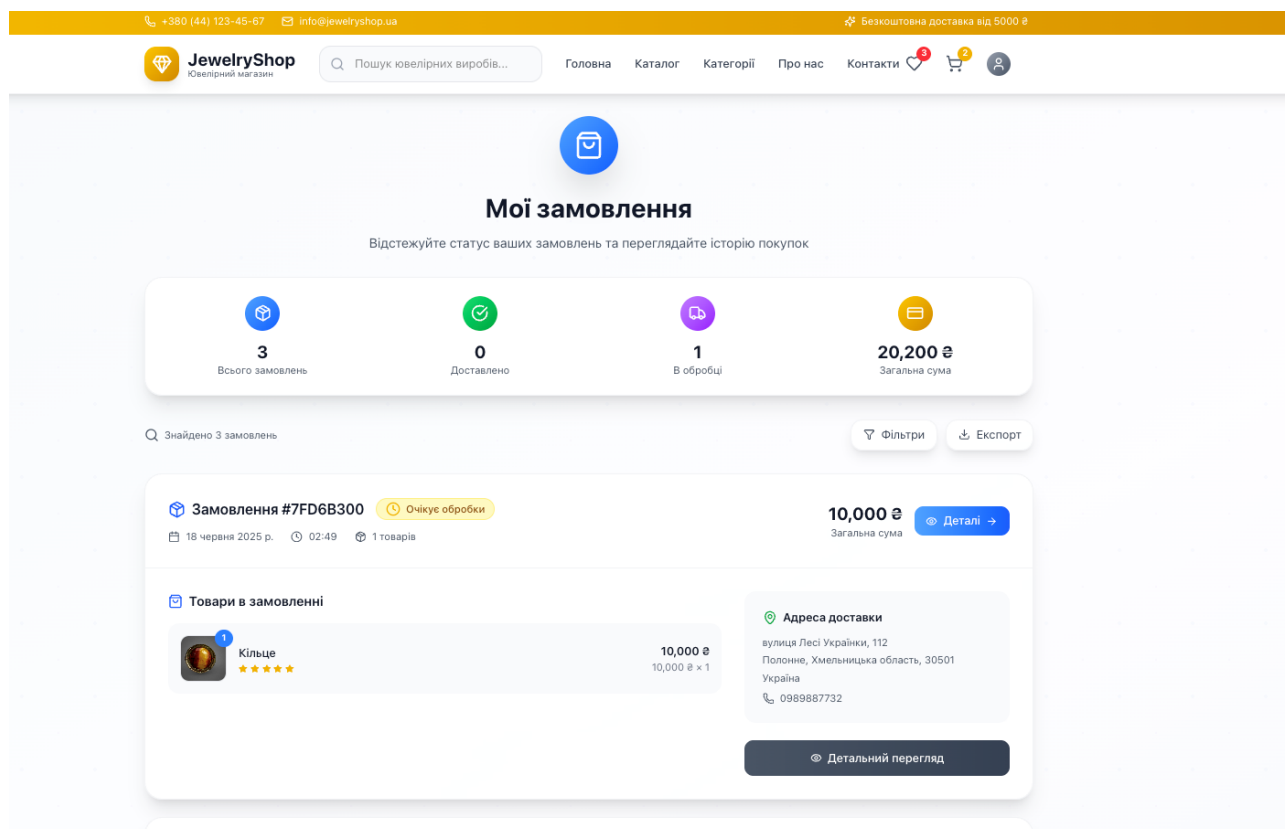


Рис. 3.9 – Сторінка історії замовлень користувача

Користувач має доступ до особистого профілю, де може переглядати й змінювати свої дані (рис. 3.10). На сторінці відображаються контактна інформація, останні замовлення, адреси доставки та залишені відгуки. Також доступні швидкі дії: перехід до каталогу чи історії замовлень.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				54
Змн.	Арк.	№ докум.	Підпис	Дата		

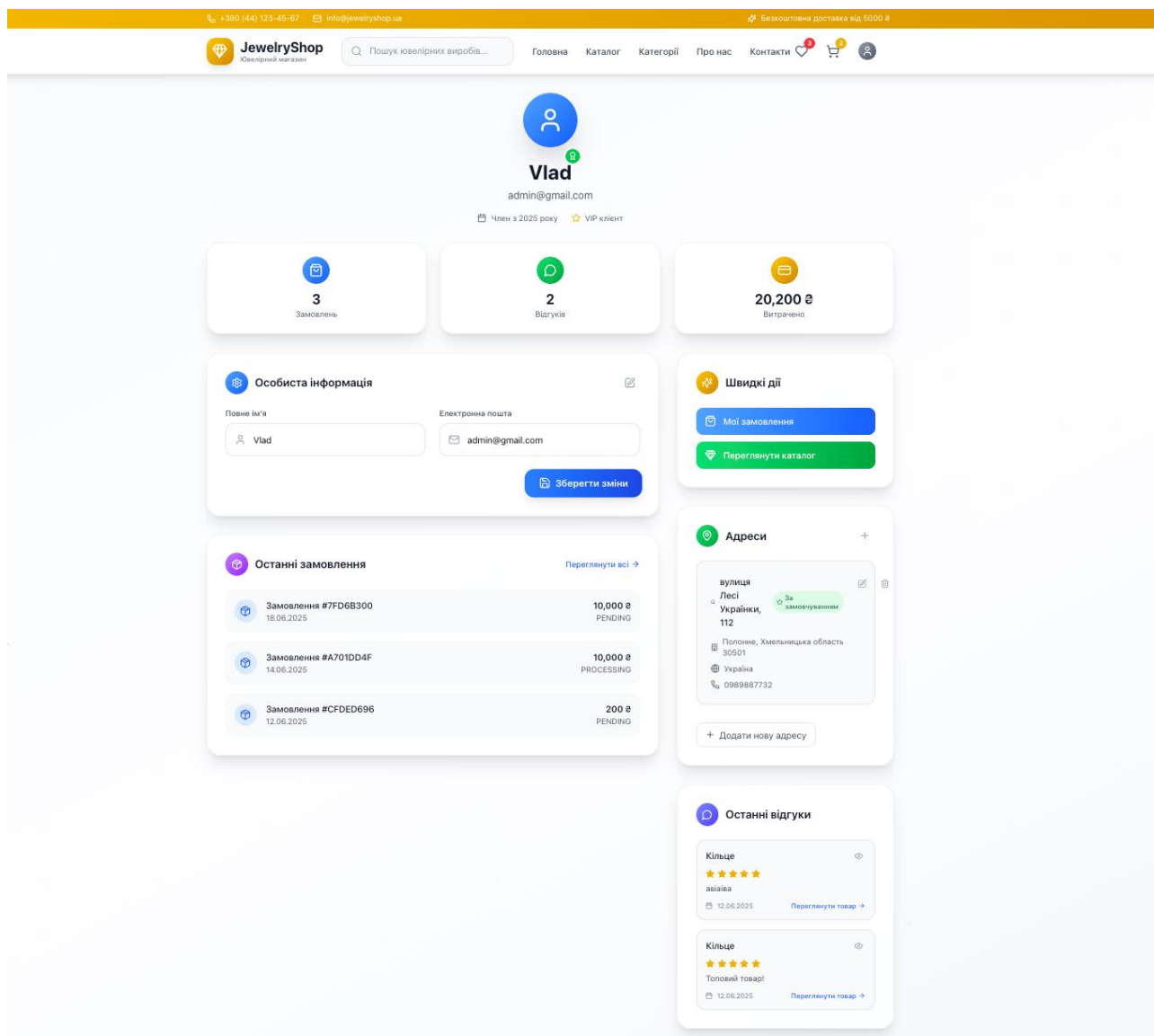


Рис. 3.10 – Особистий кабінет користувача

Якщо користувач має роль адміністратора, йому доступна окрема панель керування (рис. 3.11). Тут він може переглядати статистику магазину, керувати товарами, категоріями, замовленнями, користувачами та відгуками. Інтерфейс адміністратора також містить аналітичні показники щодо продажів, замовлень і доходу за період.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				55
Змн.	Арк.	№ докум.	Підпис	Дата		

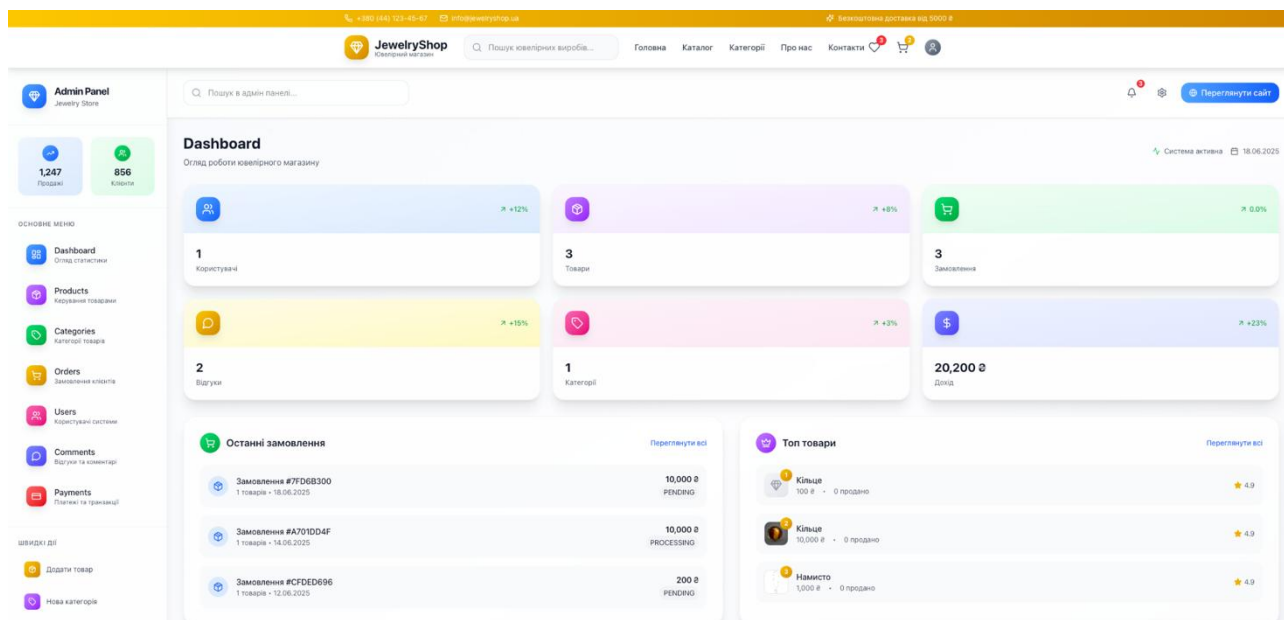


Рис. 3.11 – Адмін-панель для управління інтернет-магазином ювелірних виробів

З адміністративної панелі користувач з правами адміністратора має змогу повноцінно керувати продукцією магазину (рис. 3.12). Він може переглядати поточні товари, фільтрувати їх за категоріями, ціною або датою створення. Також передбачено можливість створення нового товару через зручну форму з полями для назви, опису, характеристик, зображень і 3D-моделі (рис. 3.13). При необхідності кожну позицію можна редагувати — змінювати характеристики, оновлювати фото або видаляти з асортименту (рис. 3.14).

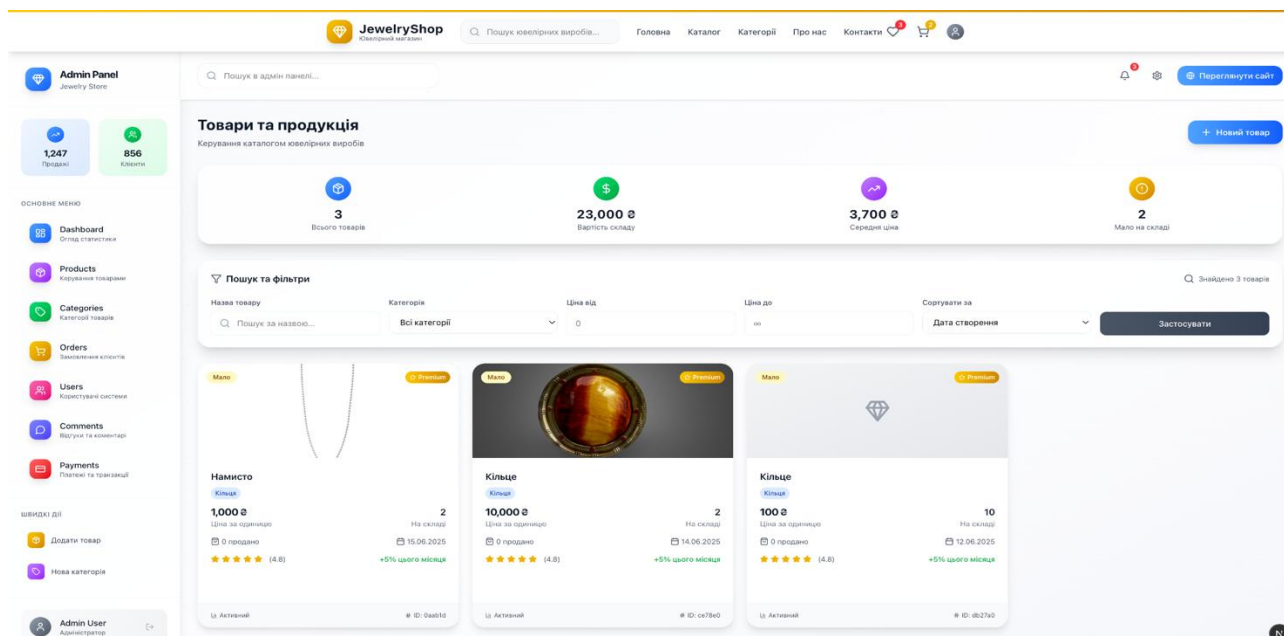


Рис. 3.12 – Список товарів у панелі адміністратора

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				56
Змн.	Арк.	№ докум.	Підпис	Дата		

Пошук ювелірних виробів...

Головна

Каталог

Категорії

Про нас

Контакти

Назад до товарів

Створити новий товар

Додайте новий ювелірний виріб до вашої колекції

Основа інформація

Назва товару *

Введіть назву ювелірного виробу

Опис

Детальний опис товару, його особливості та переваги

Ціна (\$) *

Кількість на складі *

\$ 0.00

0

Характеристики

Матеріал

Вага (г)

Розміри

Золото, срібло, платина...

0.00

25x15x5 мм

Категорії

Кільця

Зображення товару

Паретягніть зображення сюди

або

Вибрати файл

Підтримка: JPG, PNG, GIF (макс. 10MB на файл)

3D Моделі

Додайте 3D моделі (опціонально)

Формати: GLB, GLTF

Вибрати файл

Створити товар

Рис. 3.13 – Форма створення нового товару

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				57
Змн.	Арк.	№ докум.	Підпис	Дата		

Пошук ювелірних виробів...

Головна

Каталог

Категорії

Про нас

Контакти

← Назад до товару

Редагувати товар

Оновити інформацію про ювелірний виріб

Основа інформація

Назва товару *

Кільце

Опис

Гарне кільце

Ціна (гр) *

\$ 10000

Кількість на склад *

2

Характеристики

Матеріал

Вага (г)

Розміри

Золото, срібло, платина...

0.00

20x15x5 мм

Категорії

Кільце

Поточні зображення

Додати нові зображення

Перетягніть зображення сюди

або

Вибрати файли

Підтримуються: JPG, PNG, GIF (макс. 10MB кожен)

Додати 3D моделі

Додайте нові 3D моделі (опціонально)

Формати: OBJ, STL

Вибрати файли

Зберегти зміни

Видалити товар

Рис. 3.14 – Форма редагування існуючого товару

Аналогічні дії адміністратор може виконувати і з категоріями товарів (рис. 3.15). Панель керування дозволяє переглядати всі наявні категорії, фільтрувати їх, створювати нові або редагувати існуючі. Це забезпечує зручну організацію структури каталогу та швидке оновлення асортименту.

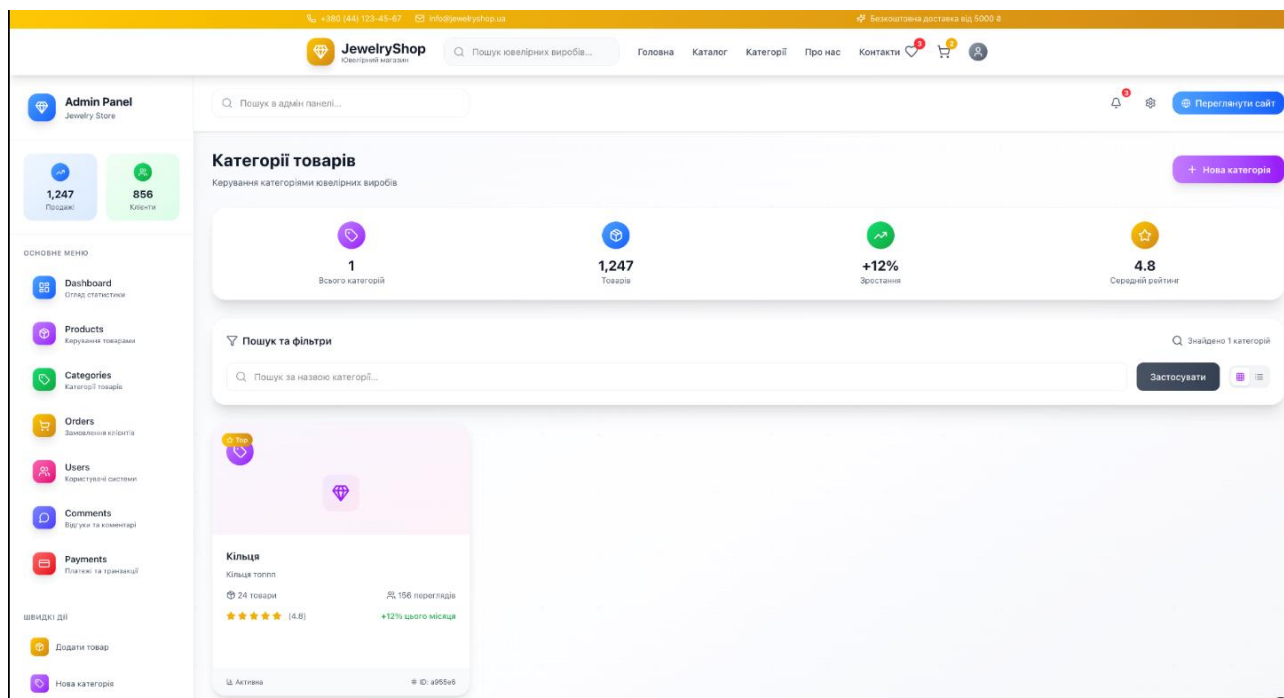


Рис. 3.15 – Керування категоріями ювелірних виробів в адмін-панелі

У вкладці «Замовлення» адміністратор має змогу переглядати всі оформлені покупки користувачів (рис. 3.16). Тут відображаються дані про замовника, сума, статус оплати та прогрес обробки кожного замовлення. Також передбачено фільтри для пошуку за статусом, датою створення та іменем користувача, що спрощує керування великим обсягом даних.

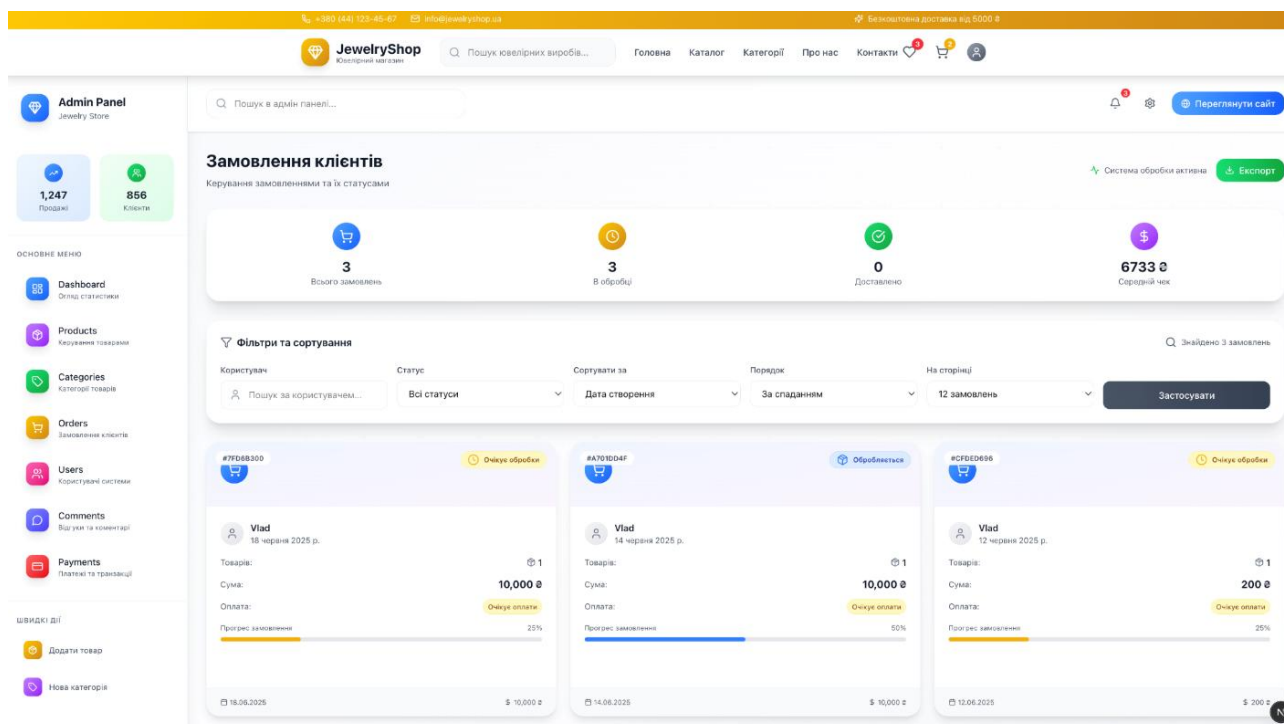


Рис. 3.16 – Вкладка перегляду замовлень у панелі адміністратора

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				59
Змн.	Арк.	№ докум.	Підпис	Дата		

У вкладці «Користувачі» адміністратор може переглядати всі зареєстровані облікові записи, їхні ролі, статуси та активність (рис. 3.17). Передбачено зручну систему фільтрації за ім'ям, електронною поштою, роллю, датою реєстрації й станом обліку. Це дозволяє швидко знайти потрібного користувача, переглянути його історію замовлень, залишені відгуки та при потребі — змінити права доступу.

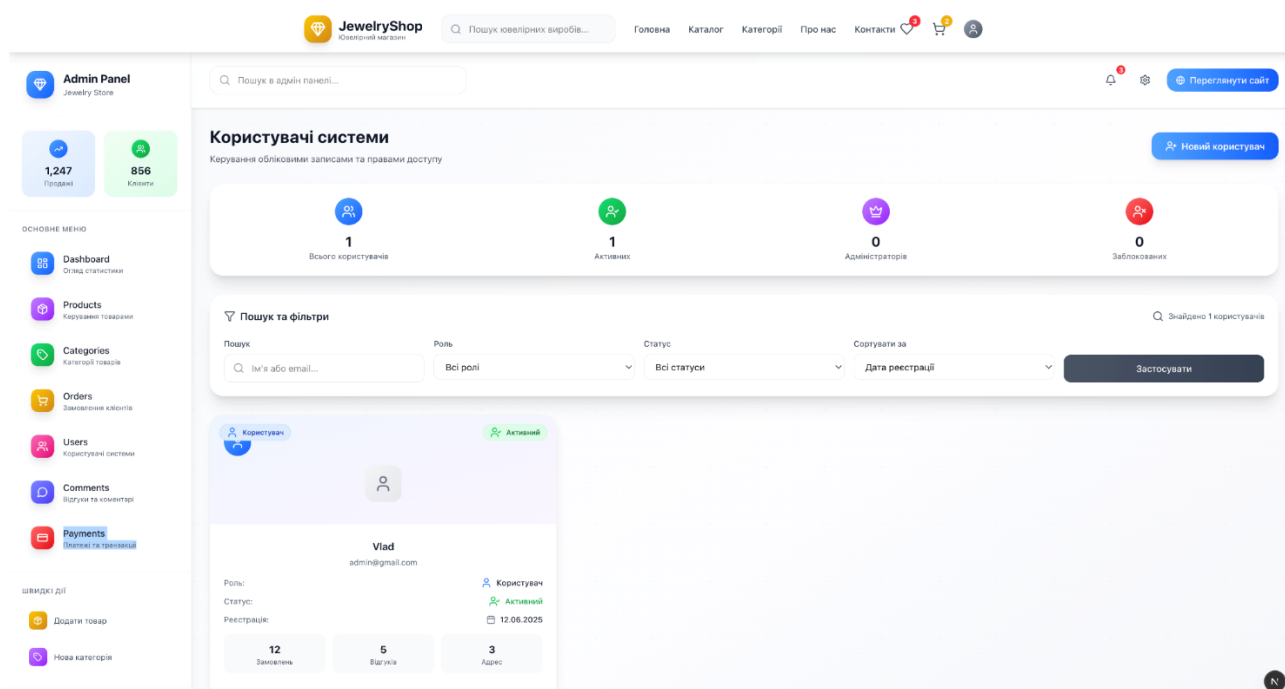


Рис. 3.17 – Панель керування користувачами в системі

У розділі «Коментарі» адміністратор має змогу переглядати всі відгуки користувачів про товари, а також модерувати їхній статус (рис. 3.18). Передбачено фільтрацію за користувачами, товарами та статусами: схвалено, очікує модерації, відхилено. Це дозволяє забезпечити контроль якості контенту та запобігти появі неприйнятних повідомлень на сайті.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				60
Змн.	Арк.	№ докум.	Підпис	Дата		

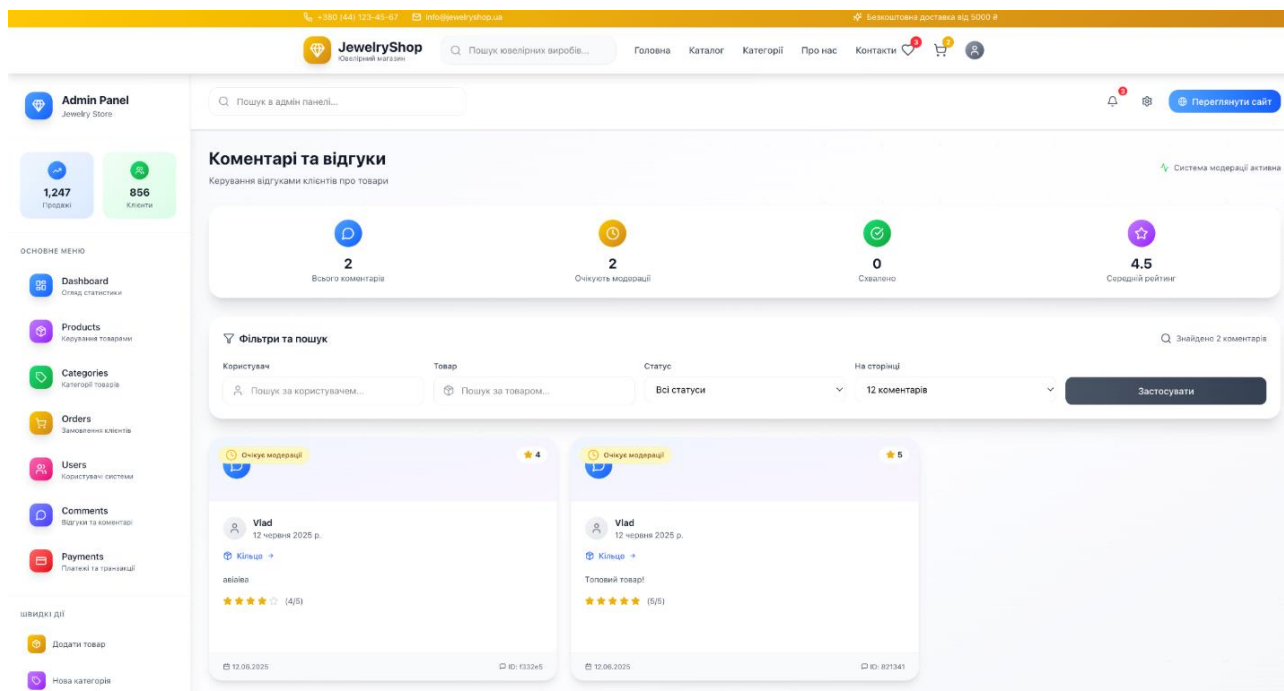


Рис. 3.18 – Панель модерації коментарів користувачів

Окремий розділ адміністративної панелі присвячено обліку платежів і транзакцій (рис. 3.19). Тут адміністратор може переглядати всі фінансові операції, фільтрувати їх за статусом, методом оплати та часовим діапазоном. Інтерфейс надає загальну інформацію про суму, кількість і середній розмір платежів, що дозволяє відстежувати фінансову активність магазину.

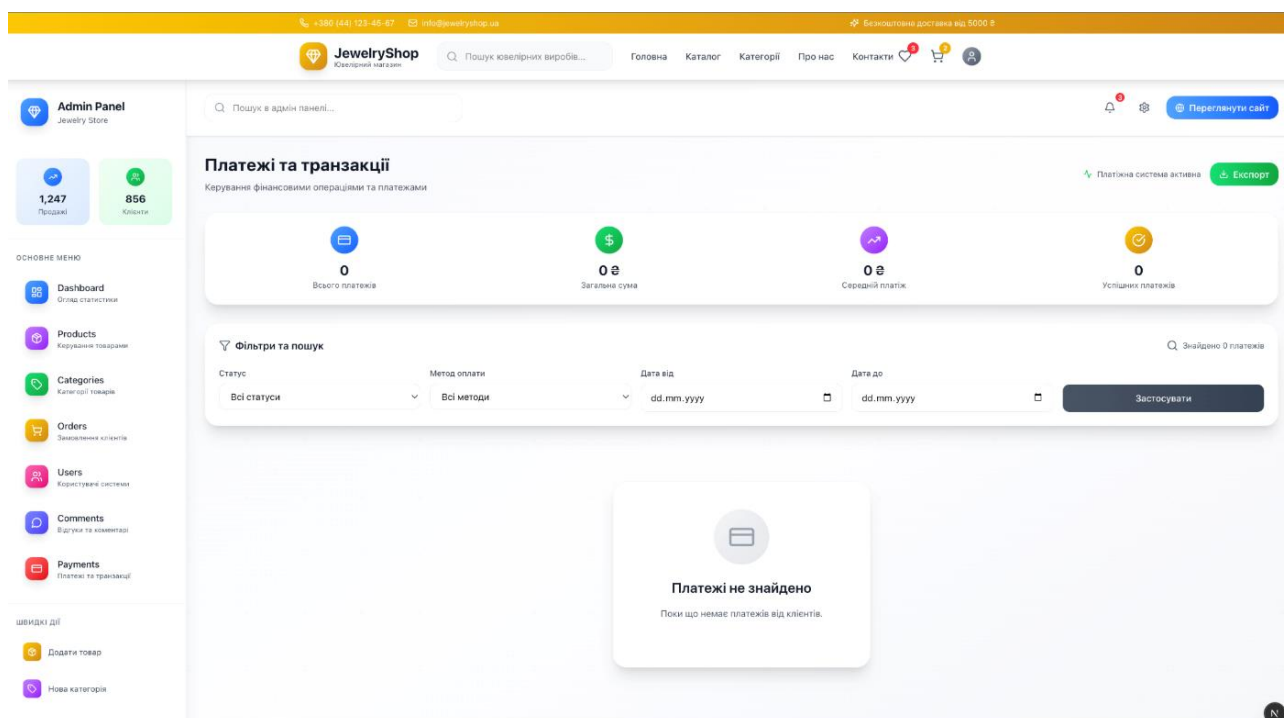


Рис. 3.19 – Облік платежів і транзакцій у панелі адміністратора

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				61
Змн.	Арк.	№ докум.	Підпис	Дата		

3.3 Порядок встановлення та налаштування системи

Розгортання інтернет-магазину ювелірних виробів потребує попередньої підготовки середовища та виконання ряду кроків з інсталяції залежностей, налаштування змінних середовища та запуску сервісів. Зважаючи на використання фреймворку Next.js і бази даних PostgreSQL, система реалізована як повноцінний full-stack застосунок з серверними API-роутами та клієнтською частиною на React.

Для локального запуску достатньо встановити Node.js, PostgreSQL та виконати команду `npm install` для завантаження всіх залежностей. Далі необхідно налаштувати файл `.env`, у якому вказуються змінні середовища: зокрема, `DATABASE_URL` для підключення до бази даних, `NEXTAUTH_SECRET` для захисту сесій, ключі Stripe для оплати та інші конфіденційні параметри. Після цього виконується ініціалізація бази даних за допомогою Prisma командою `prisma migrate dev`, що створює всі таблиці відповідно до визначеної схеми. Для продакшн-середовища рекомендується використовувати Vercel як хостинг-платформу, яка нативно інтегрується з Next.js і підтримує автоматичне розгортання з Git.

Такий підхід забезпечує простоту встановлення, масштабованість системи та легкість подальшого супроводу. Усі залежності, команди запуску та приклад змінних середовища описано у файлі `README.md`, що входить до складу репозиторію.

Після успішного розгортання системи було проведено ручне функціональне тестування з метою перевірки працездатності основних модулів. Особливу увагу приділено сценаріям взаємодії з кошиком, реєстрацією та входом користувача, переглядом товарів, оформленням замовлення, обробкою платежів, а також адмініструванням товарів і категорій. Тестування дозволило переконатися в стабільності роботи інтерфейсу, відповідності дій очікуваній логіці, а також у коректній роботі API-запитів на клієнті та сервері. При цьому використовувався набір ручних тест-кейсів, приклад яких наведено в таблиці 3.1.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				62
Змн.	Арк.	№ докум.	Підпис	Дата		

Таблиця 3.1

Перелік тест-кейсів для перевірки працездатності інтернет-магазину

№	Назва тесту	Початкові умови	Дії користувача	Очікуваний результат
1	Реєстрація нового користувача	Користувач неавторизований	Перейти на сторінку реєстрації, заповнити форму	Створюється новий обліковий запис, відбувається вхід
2	Авторизація з правильними даними	Користувач уже зареєстрований	Увести коректний email і пароль	Система авторизує користувача та перенаправляє в профіль
3	Перегляд товару	Є наявні товари в системі	Перейти до каталогу, відкрити сторінку товару	Відображаються зображення, опис, ціна, 3D-модель
4	Додавання товару до кошика	Користувач авторизований	На сторінці товару натиснути «Додати до кошика»	Товар з'являється в кошику, оновлюється сума
5	Оформлення замовлення з оплатою	Кошик заповнений, користувач авторизований	Ввести адресу, обрати доставку та оплату, підтвердити	Створюється замовлення, відображається в історії

6	Створення нового товару (адмін)	Увійдено під адміністратором	Перейти до панелі, натиснути «Додати товар», зберегти	Товар з'являється в каталозі з відповідними даними
7	Перегляд списку замовлень (адмін)	Є принаймні 1 замовлення	В адмін-панелі перейти до вкладки «Orders»	Відображаються дані про всі замовлення
8	Модерація коментарів (адмін)	Є відгуки зі статусом «Очікує»	Перейти до вкладки «Comments», змінити статус	Відгук стає видимим або відхиленням

Висновки до третього розділу

У третьому розділі було детально розглянуто архітектуру, інтерфейс та функціональні можливості інтернет-магазину ювелірних виробів. На основі діаграми розгортання описано основні компоненти системи та їх взаємодію в середовищі виконання. Продемонстровано, як реалізовано клієнтську і серверну частину застосунку, налаштовано маршрутизацію, API-запити, інтеграцію з базою даних, платіжною системою Stripe та хмарним сховищем файлів.

Особливу увагу приділено інтерфейсу користувача — послідовно описано головні сценарії взаємодії: перегляд каталогу, робота з кошиком, оформлення замовлення, перегляд профілю, вподобані товари. Крім цього, описано можливості адміністратора щодо управління товарами, категоріями, замовленнями, користувачами та коментарями. Також наведено порядок встановлення системи, налаштування клієнта Prisma, проведення ручного тестування та приклади тест-кейсів, що підтверджують працездатність платформи.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				64
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У процесі виконання бакалаврської кваліфікаційної роботи було розроблено повноцінний інтернет-магазин ювелірних виробів, що відповідає сучасним вимогам до функціональності, зручності та безпеки вебзастосунків. На основі попереднього аналізу ринку електронної комерції та цільової аудиторії було сформовано вимоги до системи та обґрунтовано вибір технологій реалізації.

У ході реалізації проєкту створено структуровану базу даних на основі PostgreSQL і Prisma, спроектовано зручний та адаптивний інтерфейс, що підтримує усі ключові сценарії користувача: перегляд товарів, пошук, фільтрація, додавання до списку вподобаного, керування кошиком, оформлення замовлень і перегляд історії покупок. Також реалізовано адміністративну панель з можливістю керування товарами, категоріями, слайдерами, користувачами та відгуками.

Платформа інтегрована з платіжною системою Stripe, забезпечує безпечну обробку транзакцій, а також підтримує зберігання зображень і 3D-моделей у хмарному середовищі. Завдяки проведеному ручному тестуванню підтверджено стабільність та працездатність усіх модулів. Система показала надійність, масштабованість і високу готовність до використання у виробничому середовищі.

Результати роботи можуть бути використані як готове комерційне рішення або база для подальшого розширення функціоналу, зокрема впровадження мобільного застосунку, аналітики продажів чи персоналізованих рекомендацій. Таким чином, створена система є прикладом ефективного вебінструменту для цифрової торгівлі ювелірними виробами.

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				65
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. James Allen — американський інтернет-магазин ювелірних виробів, відомий широким асортиментом каблучок для заручин і можливістю 3D-візуалізації [Електронний ресурс]. – Режим доступу: <https://www.jamesallen.com>
2. Pandora — міжнародний бренд ювелірних прикрас, що спеціалізується на виготовленні браслетів, каблучок, сережок і намистин [Електронний ресурс]. – Режим доступу: <https://www.pandora.net>
3. Kuyashii Jewelry — сучасний інтернет-магазин дизайнерських ювелірних прикрас, орієнтований на молодіжну аудиторію [Електронний ресурс]. – Режим доступу: <https://kuyashii.com>
4. Parts of Four — бренд ювелірних виробів і аксесуарів ручної роботи з унікальним бруталним стилем та індустріальною естетикою [Електронний ресурс]. – Режим доступу: <https://partsoffour.com>
5. NextJS [Електронний ресурс] - Режим доступу до ресурсу: <https://nextjs.org>
6. NextJS SSR [Електронний ресурс] - Режим доступу до ресурсу: <https://nextjs.org/docs/pages/building-your-application/rendering/server-side-rendering>
7. NextAuth.js [Електронний ресурс] - Режим доступу до ресурсу: <https://next-auth.js.org/>
8. Prisma [Електронний ресурс]. – Режим доступу: <https://www.prisma.io>
9. Tanstack query [Електронний ресурс] - Режим доступу до ресурсу: <https://tanstack.com/query/latest/>
10. node-telegram-bot-api [Електронний ресурс]. – Режим доступу: <https://github.com/yagop/node-telegram-bot-api>
11. Zod [Електронний ресурс]. – Режим доступу: <https://zod.dev>
12. NextJS SSG [Електронний ресурс] - Режим доступу до ресурсу: <https://nextjs.org/docs/pages/building-your-application/rendering/static-site-generation>

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				66
Змн.	Арк.	№ докум.	Підпис	Дата		

13. TypeScript [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.typescriptlang.org/>
14. NextJS CSR [Електронний ресурс] - Режим доступу до ресурсу:
<https://nextjs.org/docs/pages/building-your-application/rendering/client-side-rendering>
15. PostgreSQL- [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.postgresql.org/>
16. MySql [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.mysql.com/>
17. MongoDB [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.mongodb.com/>
18. Strapi's documentation [Електронний ресурс] –
Режим доступу до ресурсу: <https://docs.strapi.io/>
19. ReactJS [Електронний ресурс] - Режим доступу до ресурсу:
<https://react.dev/reference/react>
20. Tailwind CSS [Електронний ресурс] - Режим доступу до ресурсу:
<https://tailwindcss.com/docs>
21. NextJS Serever actions _ [Електронний ресурс] - Режим доступу до
ресурсу: <https://nextjs.org/docs/app/building-your-application/data-fetching/server-actions-and-mutations>
22. Tanstack query [Електронний ресурс] - Режим доступу до ресурсу:
<https://tanstack.com/query/latest/>
23. Vercel AI SDK [Електронний ресурс] - Режим доступу до ресурсу:
<https://ai-sdk.dev/docs/introduction>
24. Joana Almeida. Node js vs. PHP: An In-Depth Comparison Guide for Web
Development – DistantJob [Електронний ресурс] / Joana Almeida – 9 червня 2022
– Режим доступу до ресурсу: <https://distantjob.com/blog/nodejs-vsphp/>.
25. What is middleware? [Електронний ресурс] – Режим доступу
до ресурсу: <https://www.redhat.com/en/topics/middleware/what-is-middleware>

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				67
Змн.	Арк.	№ докум.	Підпис	Дата		

26. Cloudinary for Node.js [Електронний ресурс] - Режим доступу до ресурсу: https://cloudinary.com/documentation/node_integration
27. Building your own hooks [Електронний ресурс] - Режим доступу до ресурсу: <https://legacy.reactjs.org/docs/hooks-custom.html>
28. Документація JavaScript [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
29. Write a simple Telegram Bot in Next.js [Електронний ресурс]–Режим доступу до ресурсу: <https://blog.devgenius.io/write-a-simpletelegram-bot-in-next-js-b49379e77163>
30. Prisma Documentation - Офіційна документація Prisma [Електронний ресурс]. – Режим доступу до ресурсу: <https://www.prisma.io/docs/>
31. JOURNAL I. ReactJS: A Comprehensive Analysis of its features, Performance, and Suitability for Modern Web Development. INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT. 2023. Vol. 07, no. 09. P. 1. URL: <https://doi.org/10.55041/ijssrem25667>.
32. Clean Architecture [Електронний ресурс] – Режим доступу до ресурсу: <https://medium.com/codenx/clean-architecture-4c23dfa4c698>
33. Роберт М. Чиста архітектура: Мистецтво створення програмного забезпечення. Роберт М. [Видання друге] – Харків: Фабула, 2023 – 368 с
34. Learning React: Modern Patterns for Developing React Apps, 2nd edition / Alex Banks, Eve Porcello, 2020 – 307 p.
35. React Explained: Your Step-by-Step Guide to React, 1st edition / Zac Gordon, 2019 – 366 p.
36. NextAuth.js Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://next-auth.js.org/getting-started/introduction>
37. Many-to-many relations in Prisma [Електронний ресурс] – Режим доступу до ресурсу: <https://www.prisma.io/docs/orm/prisma-schema/datamodel/relations/many-to-many-relations>

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				68
Змн.	Арк.	№ докум.	Підпис	Дата		

38. TypeScript for JavaScript - [Електронний ресурс] - Режим доступу до ресурсу: <https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html>

39. Трофименко О. Г. Веб-технології та веб-дизайн : навч. посібник / О. Г. Трофименко, О. Б. Козін, О. В. Задерейко, О. Є. Плачінда. – Одеса : Фенікс, 2019. – 284 с.

40. Тест-кейс – один із ключових інструментів тестувальників ПЗ. [Електронний ресурс] – Режим доступу до ресурсу: <https://foxminded.ua/test-keis/>.

41. Документація HTML5 [Електронний ресурс] – Режим доступу до ресурсу: <https://webplatform.github.io/docs/html/html5/>

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				69
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		В.О. Мирончук			ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
		А.В. Морозов				70
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг authOption.ts:

```

import NextAuth, { NextAuthOptions } from "next-auth";
import CredentialsProvider from "next-auth/providers/credentials";
import { PrismaAdapter } from "@next-auth/prisma-adapter";
import { prisma } from "@lib/prisma";
import bcrypt from "bcrypt";

export const authOptions: NextAuthOptions = {
  adapter: PrismaAdapter(prisma),
  session: {
    strategy: "jwt",
  },
  secret: process.env.NEXTAUTH_SECRET,
  providers: [
    CredentialsProvider({
      name: "Email & Password",
      credentials: {
        email: { label: "Email", type: "email" },
        password: { label: "Password", type: "password" },
      },
      async authorize(credentials) {
        if (!credentials?.email || !credentials.password) {
          return null;
        }
        const user = await prisma.user.findUnique({
          where: { email: credentials.email },
        });
        if (!user) return null;

        const isValid = await bcrypt.compare(
          credentials.password,
          user.passwordHash
        );
        if (!isValid) return null;
        return {
          id: user.id,
          email: user.email,
          name: user.fullName,
          role: user.role,
        };
      },
    }),
  ],
});

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				71
Змн.	Арк.	№ докум.	Підпис	Дата		

```

],
callbacks: {
  async jwt({ token, user }) {
    if (user) {
      token.role = (user as any).role;
    }
    return token;
  },
  async session({ session, token }) {
    session.user = {
      id: token.sub!,
      email: session.user?.email!,
      name: session.user?.name!,
      role: token.role as string,
    };
    return session;
  },
},
};

const handler = NextAuth(authOptions);
export { handler as GET, handler as POST };

```

Лістинг schema.prisma

```

// This is your Prisma schema file,
// learn more about it in the docs: https://pris.ly/d/prisma-schema

// Looking for ways to speed up your queries, or scale easily with your serverless or edge functions?
// Try Prisma Accelerate: https://pris.ly/cli/accelerate-init

generator client {
  provider = "prisma-client-js"
  output   = "../src/generated/prisma"
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
}

enum Role {
  USER
  ADMIN
}

```

		В.О. МIRONCHUK			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				72
Змн.	Арк.	№ докум.	Підпис	Дата		

```
enum OrderStatus {
    PENDING
    PROCESSING
    SHIPPED
    DELIVERED
    CANCELLED
}
```

```
enum PaymentStatus {
    PENDING
    PAID
    FAILED
    REFUNDED
}
```

```
enum PaymentMethod {
    CREDIT_CARD
    PAYPAL
    BANK_TRANSFER
}
```

```
enum CommentStatus {
    PENDING
    APPROVED
    REJECTED
}
```

```
model User {
    id      String  @id @default(uuid())
    email   String  @unique
    passwordHash String
    fullName String
    role    Role    @default(USER)
    isBlocked Boolean @default(false)
    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt

    addresses Address[]
    cartItems CartItem[]
    orders Order[]
    comments Comment[]
    Account Account[]
    Session Session[]
}
```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				73
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    Favorite Favorite[]
}

model Address {
    id      String  @id @default(uuid())
    user    User    @relation(fields: [userId], references: [id])
    userId  String
    street  String
    city    String
    state   String
    zipCode String
    country String
    phone   String
    isDefault Boolean @default(false)
    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt

    orders Order[] @relation("AddressToOrder")
}

model Category {
    id      String      @id @default(uuid())
    name    String      @unique
    description String?
    products ProductCategory[]
}

model Product {
    id      String  @id @default(uuid())
    name    String
    description String
    price    Float
    stockQuantity Int
    soldCount Int    @default(0)
    averageRating Float @default(0)
    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt

    categories ProductCategory[]
    images      ProductImage[]
    models3D    ProductModel[]
    cartItems   CartItem[]
    orderItems  OrderItem[]
    comments    Comment[]

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				74
Змн.	Арк.	№ докум.	Підпис	Дата		

```
Favorite Favorite[]
}
```

```
model ProductCategory {
  id      String  @id @default(uuid())
  product Product @relation(fields: [productId], references: [id])
  productId String
  category Category @relation(fields: [categoryId], references: [id])
  categoryId String

  @@unique([productId, categoryId])
}
```

```
model ProductImage {
  id      String  @id @default(uuid())
  product Product @relation(fields: [productId], references: [id])
  productId String
  url      String
  altText  String?
  order    Int    @default(0)
}
```

```
model ProductModel {
  id      String  @id @default(uuid())
  product Product @relation(fields: [productId], references: [id])
  productId String
  url      String // посилання на 3D-модель
  description String?
}
```

```
model CartItem {
  id      String  @id @default(uuid())
  user    User    @relation(fields: [userId], references: [id])
  userId   String
  product Product @relation(fields: [productId], references: [id])
  productId String
  quantity Int
  addedAt  DateTime @default(now())

  @@unique([userId, productId])
}
```

```
model Order {
  id      String  @id @default(uuid())
```

		В.О. МIRONЧУК			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. МОРОЗОВ				75
Змн.	Арк.	№ докум.	Підпис	Дата		

```

user      User      @relation(fields: [userId], references: [id])
userId    String
shippingAddress Address @relation("AddressToOrder", fields: [shippingAddressId],
references: [id])
shippingAddressId String
status    OrderStatus @default(PENDING)
totalAmount Float
paymentStatus PaymentStatus @default(PENDING)
createdAt DateTime @default(now())
updatedAt DateTime @updatedAt

items OrderItem[]
payment Payment?
}

```

```

model OrderItem {
  id      String @id @default(uuid())
  order   Order  @relation(fields: [orderId], references: [id])
  orderId String
  product Product @relation(fields: [productId], references: [id])
  productId String
  quantity Int
  unitPrice Float
  totalPrice Float
}

```

```

model Payment {
  id      String @id @default(uuid())
  order   Order  @relation(fields: [orderId], references: [id])
  orderId String @unique
  amount  Float
  method  PaymentMethod
  status  PaymentStatus @default(PENDING)
  transactionId String?
  createdAt DateTime @default(now())
}

```

```

model Comment {
  id      String @id @default(uuid())
  user    User    @relation(fields: [userId], references: [id])
  userId  String
  product Product  @relation(fields: [productId], references: [id])
  productId String
  content String
}

```

		В.О. Мирончук			ІІЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				76
Змн.	Арк.	№ докум.	Підпис	Дата		

```

rating      Int?
status      CommentStatus @default(PENDING)
moderatedById String?
moderatedAt  DateTime?
createdAt   DateTime      @default(now())
updatedAt   DateTime      @updatedAt
}

model Account {
  id          String @id @default(cuid())
  userId      String
  type        String
  provider     String
  providerAccountId String
  refresh_token String? @db.Text
  access_token String? @db.Text
  expires_at   Int?
  token_type   String?
  scope        String?
  id_token     String? @db.Text
  session_state String?

  user User @relation(fields: [userId], references: [id], onDelete: Cascade)

  @@unique([provider, providerAccountId])
}

model Session {
  id          String @id @default(cuid())
  sessionToken String @unique
  userId      String
  expires     DateTime

  user User @relation(fields: [userId], references: [id], onDelete: Cascade)
}

model VerificationToken {
  identifier String
  token      String @unique
  expires    DateTime

  @@unique([identifier, token])
}

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				77
Змн.	Арк.	№ докум.	Підпис	Дата		

```

model Favorite {
  id      String  @id @default(uuid())
  user    User    @relation(fields: [userId], references: [id])
  userId  String
  product Product @relation(fields: [productId], references: [id])
  productId String
  createdAt DateTime @default(now())

  @@unique([userId, productId])
}

```

Лістинг product/route.ts

```

import { NextResponse } from "next/server";
import { prisma } from "@lib/prisma";
import { put } from "@vercel/blob";
import { v4 as uuidv4 } from "uuid";

export async function GET(request: Request) {
  const { searchParams } = new URL(request.url);

  const category = searchParams.get("category") ?? undefined;
  const priceMin = parseFloat(searchParams.get("priceMin") ?? "0");
  const priceMax = parseFloat(
    searchParams.get("priceMax") ?? String(Number.MAX_VALUE)
  );
  const sortBy = (searchParams.get("sortBy") as keyof any) ?? "createdAt";
  const order: "asc" | "desc" = (searchParams.get("order") as any) ?? "desc";

  const page = Math.max(parseInt(searchParams.get("page") ?? "1", 10), 1);
  const perPage = Math.min(
    Math.max(parseInt(searchParams.get("perPage") ?? "10", 10), 1),
    100
  );

  const where: any = { price: { gte: priceMin, lte: priceMax } };
  if (category) {
    where.categories = {
      some: { category: { name: { contains: category, mode: "insensitive" } } },
    };
  }
}

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				78
Змн.	Арк.	№ докум.	Підпис	Дата		

```

const [total, products] = await Promise.all([
  prisma.product.count({ where }),
  prisma.product.findMany({
    where,
    include: {
      images: true,
      models3D: true,
      categories: { include: { category: true } },
    },
    orderBy: { [sortBy]: order },
    skip: (page - 1) * perPage,
    take: perPage,
  }),
]);

return NextResponse.json({
  data: products,
  meta: { page, perPage, total, totalPages: Math.ceil(total / perPage) },
});
}

export async function POST(request: Request) {
  const contentType = request.headers.get("content-type") || "";
  if (!contentType.startsWith("multipart/form-data")) {
    return NextResponse.json(
      { error: "Запит має бути form-data" },
      { status: 400 }
    );
  }

  const form = await request.formData();
  const name = form.get("name") as string;
  const description = form.get("description") as string;
  const price = parseFloat(form.get("price") as string);
  const stockQuantity = parseInt(form.get("stockQuantity") as string, 10);
  const categoryIds = form.getAll("categoryIds") as string[];

  if (!name || isNaN(price) || isNaN(stockQuantity)) {
    return NextResponse.json(
      { error: "name, price та stockQuantity обов'язкові" },
      { status: 400 }
    );
  }
}

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				79
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (categoryIds.length > 0) {
  const existing = await prisma.category.findMany({
    where: { id: { in: categoryIds } },
    select: { id: true },
  });
  if (existing.length !== categoryIds.length) {
    return NextResponse.json(
      { error: "Має бути коректні категорії" },
      { status: 400 }
    );
  }
}

const imageFiles = form.getAll("images") as File[];
const imageData: { url: string; altText: string; order: number }[] = [];
for (let idx = 0; idx < imageFiles.length; idx++) {
  const file = imageFiles[idx];
  const ext = file.type.split("/").pop();
  const key = `products/images/${uuidv4()}.${ext}`;
  const { url } = await put(key, file, { access: "public" });
  imageData.push({ url, altText: file.name, order: idx });
}

const modelFiles = form.getAll("models3D") as File[];
const modelData: { url: string; description: string }[] = [];
for (let idx = 0; idx < modelFiles.length; idx++) {
  const file = modelFiles[idx];
  const ext = file.name.split(".").pop();
  const key = `products/models/${uuidv4()}.${ext}`;
  const { url } = await put(key, file, { access: "public" });
  modelData.push({ url, description: file.name });
}

const product = await prisma.product.create({
  data: {
    name,
    description,
    price,
    stockQuantity,
    categories: {
      create: categoryIds.map((id: string) => ({
        category: { connect: { id } },
      })),
    },
  },
});

```

		В.О. Мирончук			ІПЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				80
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    images: { create: imageData },
    models3D: { create: modelData },
  },
  include: {
    images: true,
    models3D: true,
    categories: { include: { category: true } },
  },
});

return NextResponse.json(product, { status: 201 });
}

```

Лістинг lib/prisma.ts

```
import { PrismaClient } from "../generated/prisma";
```

```
declare global {
  var prisma: PrismaClient | undefined;
}

```

```
export const prisma =
  global.prisma ||
  new PrismaClient({
    log: process.env.NODE_ENV === "development" ? ["query"] : [],
  });

```

```
if (process.env.NODE_ENV === "development") global.prisma = prisma;
```

		В.О. Мирончук			ІІЗ.КР.Б - 121 - 25 - ПЗ	Арк.
		А.В. Морозов				81
Змн.	Арк.	№ докум.	Підпис	Дата		