

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 126 «Інформаційні системи та технології»
(освітня програма «Системи бізнес-аналітики»)

на тему:

**«Проектування CRM-системи для управління
діяльністю магазину вінілових платівок»**

Виконав студент групи ІСТ-21-1
ДЕШКОВ Максим Юрійович

Керівник роботи:
ГРАФ Марина Сергіївна

Рецензент:
ФУРІХАТА Денис Вікторович

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ЗАТВЕРДЖУЮ

Завідувач кафедри комп'ютерних наук

Марина ГРАФ

«19» березня 2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу освітнього ступеня «бакалавр»
за спеціальністю 126 «Інформаційні системи та технології»
(освітня програма «Системи Бізнес-аналітики»)

Здобувач вищої освіти: **Дешков Максим Юрійович**

Керівник роботи: **ГРАФ Марина Сергіївна**

Тема роботи: **«Проектування CRM-системи для управління діяльністю магазину
вінілових платівок»,**

затверджена наказом закладу вищої освіти від **«19» березня 2025 р., №100/с**

Термін здачі закінченої роботи: **«10» червня 2025 р.**

Вихідні дані роботи (зазначається об'єкт і предмет дослідження):

Об'єктом дослідження є створення веб-платформи, функціональних алгоритми авторизації, керування користувачами, обробки замовлень та роботи кошика.

Консультанти випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Дата	
		Завдання видав	Завдання прийняв
1	Олександра СВІНЦИЦЬКА	19.03.2025	25.03.2025
2	Олександра СВІНЦИЦЬКА	26.03.2025	20.04.2025
3	Олександра СВІНЦИЦЬКА	21.04.2025	23.05.2025

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі. Опрацювання літературних джерел. Пошук, огляд та аналіз аналогічних розробок.	19.03.25 – 25.03.25	Виконано
2	Проектування структури системи	25.03.25 – 15.04.25	Виконано
3	Написання програмного коду	15.04.25 – 01.05.25	Виконано
4	Тестування системи	01.05.25 – 15.05.25	Виконано
5	Оформлення пояснювальної записки	15.05.25 – 30.05.25	Виконано
6	Попередній захист	10.06.27	Виконано

Здобувач вищої освіти **Максим ДЕШКОВ**

Керівник **Марина ГРАФ**

РЕФЕРАТ

Основний зміст кваліфікаційної роботи викладено на 57 сторінках тексту, робота містить 20 ілюстрацій, 7 таблиць, 3 додатки та список із 20 найменувань використаних літературних джерел.

Метою роботи є створення веб-платформи для магазину вінілових платівок, що об'єднує публічний сайт для користувачів та CRM-систему для адміністраторів і менеджерів. Загальна частина дозволяє переглядати каталог платівок, додавати товари до кошика, оформлювати замовлення та керувати особистим кабінетом користувача. CRM-система призначена для обробки замовлень, управління платівками, користувачами та контролю роботи персоналу.

У межах роботи розроблено архітектуру платформи, створено публічний застосунок та CRM-систему, а також реалізовано серверну частину для обробки запитів і завантаження файлів. CRM реалізовано можливість відстеження історії зміни статусів замовлень, додавання коментарів до замовлень та прикріплення супровідних файлів. Проєкт побудовано з використанням сучасних інструментів та технологій: React для створення клієнтських застосунків, Node.js і Express для серверної частини, а також json-server для тестування API.

Ключові слова: EXPRESS, CRM, ЗАМОВЛЕННЯ, ПЛАТІВКИ, ВЕБ-ПЛАТФОРМА.

					ІСТ.КР.Б – 126 – 25 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	«Проектування та розробка CRM-системи для управління діяльністю магазину вінілових платівок» Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		М.Ю. Дешков						
Керівник		М.С. Граф					3	90
Рецензент		Д.В. Фуріхата				Житомирська політехніка, група ІСТ-21-1		
Н. контр.								
Зав. каф.		М.С. Граф						

ABSTRACT

The main content of the qualification work is presented on 57 pages of text, includes 20 illustrations, 7 tables, 3 appendices, and lists 20 references.

The purpose of this work is to design and develop a web platform for managing a vinyl records store and supporting online sales. The project consists of two key parts: a public-facing web application for customers, and a CRM system intended for administrators and managers. The public application provides access to a vinyl records catalog, shopping cart functionality, order placement, and user account management. The CRM system allows staff to handle orders, manage the product catalog and user data, and oversee personnel operations.

The project defines the platform's architecture, implements both the public interface and the CRM, and develops a server-side component responsible for handling API requests and file uploads. The CRM part features order status tracking, including a history of status changes, the ability to add comments to orders, and attach supplementary files. The solution was built using modern development tools and technologies: React for the frontend applications, Node.js and Express for the backend API, and json-server for API mock data during development.

Keywords: EXPRESS, CRM, VINYL RECORDS, ORDERS, WEB PLATFORM.

		М.Ю. Дешков			ICT.KP.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				4
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	7
ВСТУП	8
РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ ТА ЗАСОБІВ ВИРІШЕННЯ ПІД ЧАС РОЗРОБКИ.....	10
1.1 Аналіз існуючого програмного забезпечення за тематикою кваліфікаційної роботи	10
1.2 Обґрунтування вибору інструментальних засобів та вимог до апаратного забезпечення	15
1.3 Постановка задач	16
Висновки до першого розділу.....	19
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА CRM-СИСТЕМИ ДЛЯ УПРАВЛІННЯ МАГАЗИНОМ ВІНІЛОВИХ ПЛАТІВОК.....	22
2.1 Варіанти використання системи	22
2.2 Розробка структури веб-платформи	25
2.3 Проєктування архітектури веб-платформи.....	28
2.4 Проєктування бази даних системи	30
2.5 Проєктування функціональних алгоритмів роботи веб-платформи ...	32
Висновки до другого розділу	37
РОЗДІЛ 3. ОПИС РОБОТИ З ВЕБ-ПЛАТФОРМОЮ ТА ЇЇ ТЕСТУВАННЯ .	39
3.1 Огляд інтерфейсу веб-платформи.....	39
3.2 Проблематика під час розробки	48
3.3 Тестування системи.....	51
Висновки до третього розділу.....	54
ВИСНОВКИ.....	56
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	59
ДОДАТКИ.....	62
Додаток А.....	63

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				5
Змн.	Арк.	№ докум.	Підпис	Дата		

Додаток Б	70
Додаток В	80

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CRM — англ. Customer Relationship Management, система управління взаємовідносинами з клієнтами.

СУБД — система управління базами даних.

БД — база даних.

API — англ. Application Programming Interface, програмний інтерфейс додатка.

UI — англ. User Interface, користувацький інтерфейс.

UX — англ. User Experience, користувацький досвід.

JSON — англ. JavaScript Object Notation, формат обміну даними.

JS — англ. JavaScript, мова програмування.

HTML — англ. HyperText Markup Language, мова розмітки гіпертексту.

CSS — англ. Cascading Style Sheets, каскадні таблиці стилів.

HTTP — англ. HyperText Transfer Protocol, протокол передачі гіпертексту.

HTTPS — англ. HyperText Transfer Protocol Secure, захищений протокол передачі гіпертексту.

ID — англ. Identifier, ідентифікатор.

CRUD — англ. Create, Read, Update, Delete, основні операції з даними.

MVC — англ. Model-View-Controller, архітектурний шаблон.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. Сучасна торгівля дедалі активніше інтегрується в цифровий простір, де онлайн-платформи стають невід’ємною частиною ведення бізнесу. Особливо це стосується нішевих магазинів, зокрема магазинів вінілових платівок, які об’єднують спільноти шанувальників музики й колекціонерів. Використання CRM-систем та публічних сайтів дозволяє не лише автоматизувати бізнес-процеси, а й підвищити ефективність взаємодії з клієнтами, забезпечити якісний сервіс і розширити ринки збуту. Для України ця тема є актуальною з огляду на зростання популярності вінілових платівок та необхідність адаптації вітчизняного бізнесу до сучасних стандартів електронної комерції. Світові тенденції розвитку онлайн-торгівлі й CRM-рішень підтверджують доцільність розробки індивідуальних платформ для конкретних типів бізнесу.

Метою дослідження є розробка та обґрунтування проєкту CRM-системи для магазину вінілових платівок, яка поєднує публічну платформу для клієнтів та адміністративний інтерфейс для менеджерів. Система має забезпечити автоматизацію та оптимізацію основних бізнес-процесів, зокрема взаємодії з клієнтами, оформлення та обробки замовлень, управління асортиментом та формування звітності.

Клієнти отримують можливість переглядати каталог платівок, додавати товари до кошика, оформлювати замовлення, залишати відгуки та керувати своїм обліковим записом. веб-платформи дає змогу менеджерам обробляти замовлення, оновлювати їх статуси, керувати даними користувачів і товарами, а також аналізувати ключові показники ефективності для покращення якості обслуговування та підвищення лояльності клієнтів.

Для реалізації цієї мети передбачено розробку гнучкої архітектури системи, створення зручного інтерфейсу для клієнта та користувача CRM-системи, впровадження механізмів пошуку, фільтрації товарів і обліку

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				8
Змн.	Арк.	№ докум.	Підпис	Дата		

замовлень, а також налаштування серверної частини для обробки запитів і зберігання даних.

Об’єктом дослідження є процеси взаємодії з клієнтами та ефективне управління продажами в магазині вінілових платівок.

Предметом дослідження є методи та засоби проєктування, розробки та впровадження CRM-систем, що є орієнтованими на специфіку роздрібно́ї торгівлі, з акцентом на автоматизацію збору та аналізу даних для бізнес аналітики.

У процесі розробки планується використовувати засоби для створення інтерфейсів (React), налаштування серверної логіки (Node.js, Express), роботи з тестовими даними (json-server) та створення API для взаємодії компонентів платформи.

В межах теми кваліфікаційної роботи бакалавра було підготовлено й опубліковано тезу ДешковМ.Ю. Огляд систем проєктного менеджменту. Тези доповідей XIV Міжнародної науково-технічної конференції “Інформаційно комп’ютерні технології” 28-29 березня 2024 року. Житомир : "Житомирська політехніка", 2024 [1].

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ ТА ЗАСОБІВ ВИРІШЕННЯ ПІД ЧАС РОЗРОБКИ

1.1 Аналіз існуючого програмного забезпечення за тематикою кваліфікаційної роботи

У ході проєктування складних інформаційних систем, зокрема таких, що поєднують функціонал управління клієнтами, товарними позиціями, замовленнями та персоналом, як CRM-системи для роздрібної торгівлі, надзвичайно важливим етапом є попередній аналіз. Його доцільність зумовлена потребою в оцінці вже існуючих рішень на ринку, вивченні їх функціональних можливостей, архітектурних моделей, підходів до реалізації та використаних технологій. Такий аналіз дозволяє уникнути дублювання типових помилок, врахувати переваги й недоліки конкурентних рішень, а також запозичити ефективні ідеї для побудови власної системи.

Значення CRM-систем (англ. Customer Relationship Management systems) в роздрібному бізнесі є ключовим, адже саме вони забезпечують централізоване управління взаємодією з клієнтами, підвищують ефективність обробки замовлень, дають змогу контролювати комунікацію персоналу, аналізувати результати продажів та адаптувати стратегії обслуговування до потреб аудиторії. Особливої актуальності такі системи набувають в умовах розвитку електронної комерції, де цифрові процеси взаємодії з клієнтом є основою конкурентоспроможності.

У зв'язку з цим, для побудови власного рішення доцільно звернутися до вивчення сучасних CRM-систем, що вже зарекомендували себе у сфері онлайн-торгівлі. Далі представлено огляд найбільш поширених і функціонально насичених аналогів, які стали основою для формування вимог до розроблюваної системи.

Shopify — одна з найпопулярніших платформ для створення інтернет-магазинів. Вона включає базовий функціонал CRM-системи, зокрема управління клієнтами, облік замовлень, повідомлення, історію покупок, тощо

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				10
Змн.	Арк.	№ докум.	Підпис	Дата		

(рисунок 1.1) [2]. Shopify пропонує інтерфейс для адміністраторів, де можна відстежувати статистику продажів, редагувати товари, керувати знижками.

Переваги:

- Простота використання;
- Можливість інтеграції з платіжними сервісами;
- Гнучка система дизайну.

Недоліки:

- Обмежені можливості кастомізації без підписки на розширені пакети;
- Відсутність можливості повної модифікації бекенду;
- Значна вартість щомісячного обслуговування.

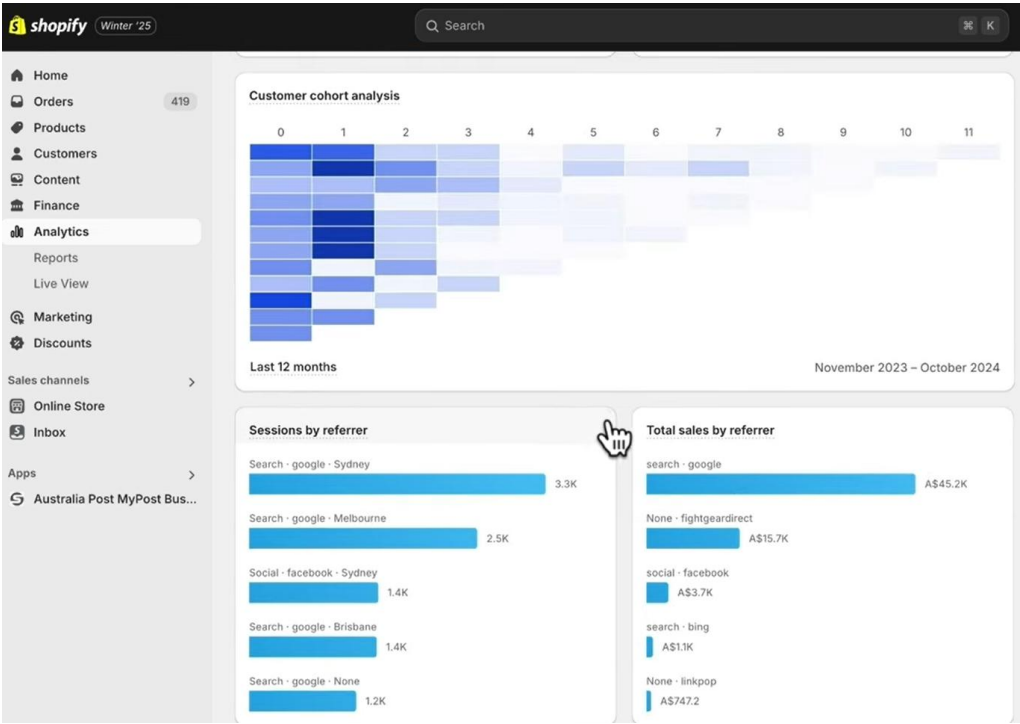


Рисунок 1.1 – Приклад панелі менеджера Shopify

Odoo — модульна ERP/CRM-система, яка включає широкий набір функціоналу для ведення торгівлі, маркетингу, логістики та фінансів (рисунок 1.2) [3]. Її CRM-модуль дозволяє гнучко управляти клієнтами, замовленнями, аналітикою, а також інтегруватися з модулем "Продажі" та "Склад".

Переваги:

- Відкритий вихідний код;
- Великий набір модулів;
- Можливість локального розгортання.

Недоліки:

- Складність первинного налаштування;
- Надмірна для невеликих магазинів складність інтерфейсу;
- Високі вимоги до технічного обслуговування.

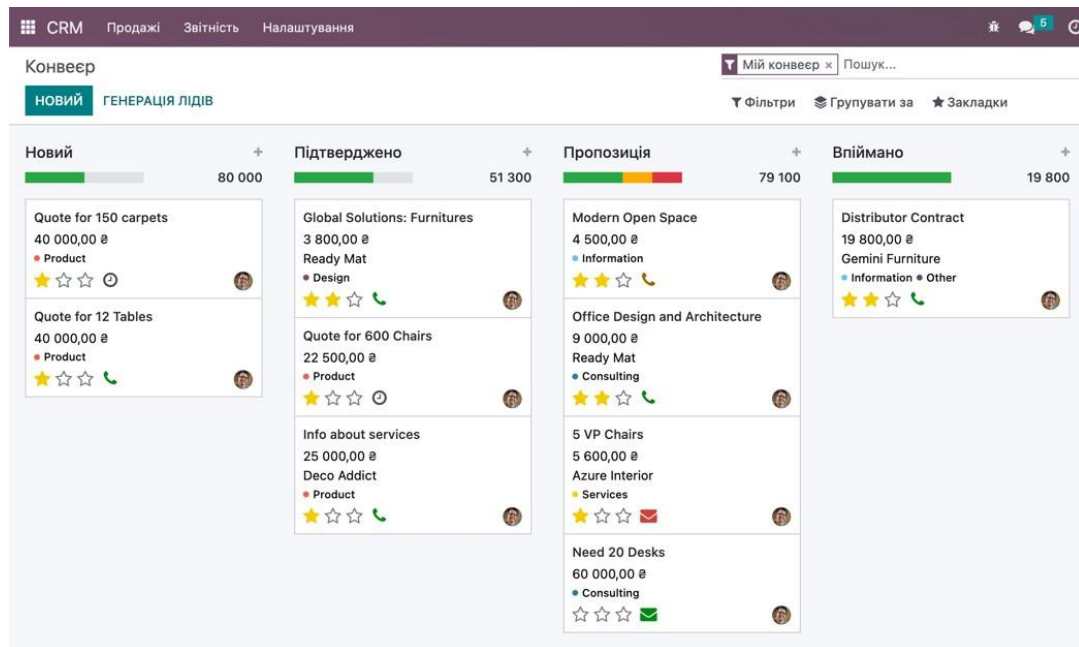


Рисунок 1.2 – Приклад панелі менеджера Odoo CRM

Zoho CRM орієнтований на малі та середні бізнеси, включаючи компанії, що займаються торгівлею фізичними товарами (рисунок 1.6) [4]. Його функціонал охоплює управління контактами, лійку продажів, аналітику, комунікацію з клієнтами через email та месенджери.

Переваги:

- Простий та інтуїтивний інтерфейс;
- Багато налаштувань під конкретні бізнес-процеси;
- Інтеграція з Google Workspace, Facebook, Shopify.

Недоліки:

- Обмеження у безкоштовній версії;

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				12
Змн.	Арк.	№ докум.	Підпис	Дата		

- Складність глибокої кастомізації.

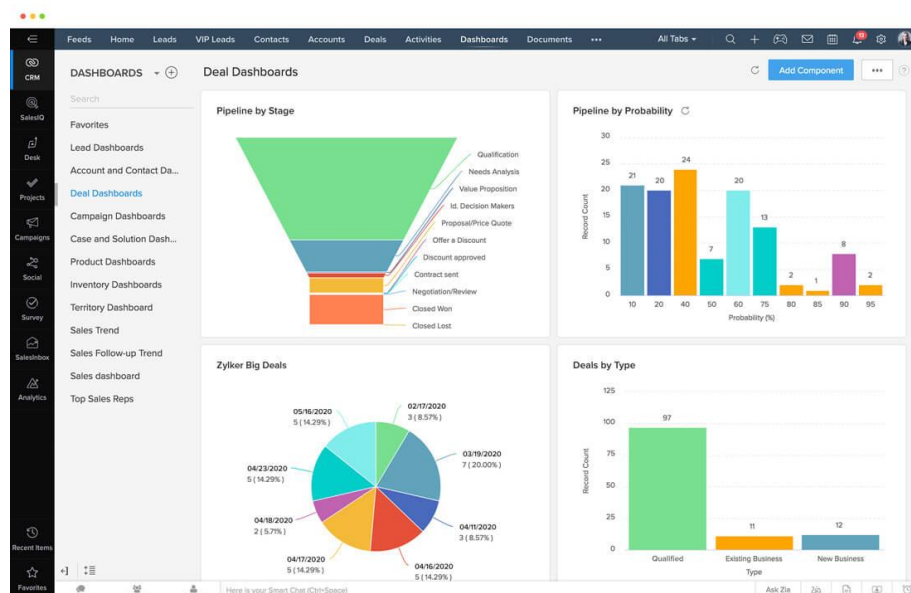


Рисунок 1.3 – Приклад аналітичного звіту ZohoCRM

Це потужне хмарне рішення для великих торгових мереж, що має розвинену інфраструктуру для CRM та e-commerce [5]. Проте його використання є доцільним переважно для масштабних компаній через складність реалізації та вартість.

Недоліки для малого магазину вінілових платівок:

- Надлишковий функціонал;
- Висока вартість ліцензій;
- Потреба у спеціалістах для впровадження.

Таблиця 1.1

Порівняння власної CRM-системи з аналогами

Критерій / Система	Shopify	Odoo	Zoho CRM	Власна CRM-система
Модель розгортання	Хмарна (SaaS)	Хмарна або локальна	Хмарна	Локальна або хмарна (гнучка)
Відкритість коду	Закритий	Відкритий	Закритий	Відкритий
Можливість глибокої кастомізації	Обмежена, платна	Легка для кастомізації	Частково, через модулі	Адаптація під специфіку магазину та їх дизайн

Продовження таблиці 1.1

Заточення під фізичні товари	E-commerce-фокус	Гнучка, підходить для всі видів	B2B	Фокус на продажу вінілових платівок
Система ролей користувачів	Базова	Адмін, менеджер тощо	Гнучка	Адмін, менеджер
Обробка замовлень	Гнучка, глибока	Гнучка, глибока	Обмежена	Базова
Аналітика та статистика	Дашборди	Дашборди, звіти	Звіти	Специфічно адаптована аналітика)
Коментування замовлень	Відсутня	Потребує дод. модулів	Відсутня	З фіксацією автора
Інтеграція з платіжними системами	Вбудована	Потребує дод. модулів	Вбудована	Обмежена
Простота інтерфейсу для користувача	Зручний, мінімалістичний	Складний інтерфейс	Зручна, мінімалістична	Зручний, мінімалістичний
Технічна підтримка	Присутня	Обмежена в безкоштовній версії	Присутня	Відсутня
Вартість використання	Висока (підписка)	Безкоштовна (Community) або платна (Enterprise)	Безкоштовна з обмеженнями або платна	Безкоштовна, внутрішнє обслуговування

Проведений аналіз трьох найбільш поширених CRM-рішень (таблиця 1.1) — Shopify, Odoo та Zoho CRM — дозволив виявити як їхні сильні сторони, так і обмеження у контексті задач, що постають перед магазином вінілових платівок [6]. Розгляд функціональних можливостей цих систем у порівняльній формі виявив, що жодна з них не задовольняє комплексно всі потреби малого торговельного бізнесу з вузькою спеціалізацією на музичній продукції.

Так, Shopify надає широкий набір функцій для онлайн-продажу, проте її закритий код, платна підписка та обмежені можливості кастомізації значно знижують придатність для гнучкої адаптації. Odoo, попри відкритість коду та велику кількість модулів, вимагає від користувача високого рівня технічної компетентності, має складний інтерфейс і може бути надмірною для малого

бізнесу. У свою чергу, ZoHo CRM зосереджується переважно на управлінні клієнтами в сфері B2B та не орієнтована безпосередньо на продаж фізичних товарів, що обмежує її функціональність у сфері роздрібної торгівлі.

На фоні розглянутих рішень, власна CRM-система, яка розробляється в рамках цієї кваліфікаційної роботи, вигідно вирізняється можливістю точкового налаштування під специфіку діяльності магазину вінілових платівок. Зокрема, реалізація гнучкої багаторівневої системи ролей, спеціалізованого каталогу платівок, фільтрації за жанрами, коментування замовлень із фіксацією автора, а також розширеної аналітики, дає змогу створити адаптовану, економічно ефективну та технологічно зручну CRM-систему. Враховуючи те, що система буде повністю кастомною, без щомісячних ліцензійних витрат, її впровадження є доцільним з огляду на довгострокову економічну доцільність і незалежність від сторонніх сервісів.

Таким чином, результати порівняння чітко демонструють доцільність розробки власного CRM-рішення, яке відповідатиме актуальним потребам бізнесу, буде масштабованим, контрольованим та економічно вигідним у перспективі.

1.2 Обґрунтування вибору інструментальних засобів та вимог до апаратного забезпечення

Повна назва програмного забезпечення – «CRM-система для управління діяльністю магазину вінілових платівок» (надалі – веб-платформа). Скорочена назва – Vinyl CRM.

Розробка призначена для автоматизації ключових бізнес-процесів магазину, пов'язаних із торгівлею вініловими платівками та електронною комерцією. Веб-платформа поєднує дві функціональні підсистеми: публічний інтерфейс для клієнтів і внутрішню CRM-систему для менеджерів та адміністраторів. Користувацька частина платформи забезпечує можливість перегляду каталогу товарів із фільтрацією, пошуком, доступом до детальних описів і зображень, додаванням товарів до кошика, оформленням замовлень,

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				15
Змн.	Арк.	№ докум.	Підпис	Дата		

залишенням відгуків та керуванням особистим кабінетом. CRM-інтерфейс дозволяє обробляти замовлення, змінювати їх статуси, управляти асортиментом товарів, адмініструвати облікові записи користувачів, додавати коментарі й супровідні файли до замовлень, а також здійснювати формування звітності й аналіз ключових показників ефективності [7].

Функціонал веб-додатку передбачає зручну та зрозумілу навігацію, мінімізацію кількості дій для здійснення замовлення та інтерактивну взаємодію користувача з системою. Час реакції інтерфейсу не перевищує 0,25 секунди, виконання команд – до 1 секунди, а обробка запитів до бази даних – до 3 секунд. Система забезпечує стабільну роботу з рівнем доступності не менше ніж 90% часу в режимі 24/7. Мінімальні технічні вимоги включають 1 ГБ оперативної пам'яті, до 300 МБ дискового простору для клієнтської частини та до 20 ГБ для серверної частини.

Розробка системи здійснюється в межах виконання кваліфікаційної роботи здобувача освітнього ступеня «бакалавр» за спеціальністю 126 «Інформаційні системи та технології». Вибір методів реалізації, архітектурних рішень та інструментів розробки залишається за розробником. Основними технологіями виступають React для реалізації клієнтської частини, Node.js із використанням Express для створення серверного API, бібліотеки axios для обміну даними та json-server на етапі прототипування. Середовище розробки – Visual Studio Code, підтримувані браузерери – Google Chrome, Mozilla Firefox або інші сучасні браузерери. Серверна частина потребує Node.js версії 16 або вище [5].

1.3 Постановка задач

У сучасних умовах стрімкого розвитку цифрових технологій та електронної комерції, зростає попит на ефективні програмні рішення, здатні оптимізувати управлінські процеси в торгівельній сфері. Магазины, які спеціалізуються на реалізації фізичних носіїв, зокрема вінілових платівок, також стикаються з викликами цифрової трансформації. Вони потребують не

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				16
Змн.	Арк.	№ докум.	Підпис	Дата		

лише якісного представлення асортименту продукції для клієнтів в інтернет-просторі, але й ефективних інструментів для внутрішнього адміністрування бізнес-процесів, обліку товарів, замовлень та взаємодії з покупцями.

Головна мета кваліфікаційної роботи полягає у створенні функціональної CRM-системи, яка дозволяє організувати та автоматизувати основні процеси управління діяльністю магазину вінілових платівок. І хоча загальна архітектура рішення передбачає наявність інтерфейсів, доступних через веббраузер, суть завдання полягає не у створенні окремої веб-платформи, а у розробці комплексного засобу управління (Customer Relationship Management system) із додатковими можливостями обслуговування клієнтів у цифровому середовищі.

Для досягнення поставленої мети було визначено, що CRM-система повинна поєднувати у собі дві ключові частини:

1. Інтерфейс користувача (клієнтська частина), який дозволяє потенційним покупцям ознайомитися з асортиментом, здійснювати покупки онлайн, переглядати історію замовлень, зберігати особисті дані та взаємодіяти з магазином у зручному форматі. Це дозволить автоматизувати значну частину процесів, та полегшити роботу менеджерам.
2. Адміністративно-керуючий функціонал (внутрішня частина системи), розрахований на персонал магазину, зокрема адміністраторів і менеджерів, для ефективного контролю процесів, ведення баз даних та моніторингу динаміки продажів.

В рамках даної роботи було сформульовано низку підзадач, кожна з яких відповідає за окрему функціональну область:

Клієнтська частина CRM-системи включає:

- Каталог продукції — повноцінний розділ, який дає змогу користувачам ознайомлюватися з доступними вініловими платівками. Реалізовано можливості фільтрації за жанром, ціною, алфавітом, а також пошук за назвою або виконавцем. Додатково передбачено функцію сортування для покращення користувацького досвіду.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				17
Змн.	Арк.	№ докум.	Підпис	Дата		

- Кошик замовлень — інструмент для формування переліку бажаних до придбання платівок. Користувач може редагувати кількість товарів, видаляти позиції, бачити підсумкову вартість.
- Сторінка оформлення замовлення — містить форму для введення контактної інформації (ім'я, email, телефон, адреса доставки), з обов'язковою валідацією введених даних, щоб забезпечити коректність обробки заявки.
- Особистий кабінет користувача — окремий простір для збереження історії покупок, редагування профілю та перегляду актуального статусу замовлень.

Адміністративна частина CRM-системи охоплює:

- Модуль обробки замовлень — забезпечує менеджерів і адміністраторів інструментами для перегляду, редагування та модифікації замовлень.
- Передбачає зміну статусу ("очікує", "в обробці", "відправлено", "завершено"), додавання службових коментарів, прикріплення файлів (наприклад, товарних чеків або супровідних документів).
- Інтерфейс управління каталогом — дозволяє додавати нові позиції до асортименту, редагувати існуючі записи, змінювати описи, ціни, зображення та наявність товару.
- Управління користувачами — інструменти для адміністрування зареєстрованих клієнтів, призначення ролей, видалення або редагування облікових записів.
- Контроль діяльності персоналу — функціонал для відстеження змін у системі, фіксація авторства коментарів або оновлень у замовленнях, що дозволяє забезпечити прозорість внутрішніх процесів.
- Аналітичний модуль — формує візуалізацію статистики продажів, популярності товарів, кількості замовлень, активності користувачів тощо.

Серверна частина системи передбачає:

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				18
Змн.	Арк.	№ докум.	Підпис	Дата		

- Реалізацію API — набір маршрутів для обробки запитів від клієнтської та адміністративної частини. Зокрема, створення/редагування замовлень, отримання переліку товарів, авторизація користувачів тощо.
- Базу даних — центральне сховище для всіх сутностей системи: платівок, користувачів, замовлень, коментарів і метаданих.
- Збереження та обробку файлів — завантаження зображень обкладинок, прикріплених документів, резервне копіювання даних.

Додаткові функції:

- Механізм коментування — надання можливості залишати службові примітки до кожного замовлення з автоматичним зазначенням автора коментаря.
- Мультирівнева система ролей — диференціація доступу відповідно до ролі користувача: адміністратор, менеджер, клієнт. Це дозволяє обмежити права доступу до критичних частин системи.
- Підтримка кількох способів оплати — що дозволяє розширити цільову аудиторію магазину та покращити користувацький досвід.

Таким чином, система, що розробляється в рамках кваліфікаційної роботи, є цілісним, структурованим і функціонально багатим рішенням, орієнтованим як на потреби кінцевих споживачів (клієнтів магазину), так і на ефективне забезпечення внутрішніх процесів управління. Взаємозв'язок між усіма її компонентами дозволяє досягти високого рівня інтегрованості, гнучкості в адмініструванні та зручності у використанні. Реалізація саме CRM-системи, а не просто веб-платформи, забезпечує глибину функціоналу, адаптованого під конкретні бізнес-задачі магазину вінілових платівок.

Висновки до першого розділу

У першому розділі було здійснено всебічне дослідження особливостей розробки CRM-системи для управління діяльністю магазину вінілових платівок. Особливу увагу приділено вивченню теоретичних основ побудови таких систем, специфіки їх функціонування в галузі електронної комерції, а

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				19
Змн.	Арк.	№ докум.	Підпис	Дата		

також виявленню основних вимог, які має задовольняти програмне забезпечення для забезпечення ефективного обслуговування клієнтів і внутрішніх бізнес-процесів.

На основі аналізу ринку сучасних CRM-рішень, таких як Shopify, Odoo, Zoho CRM та спеціалізованих платформ, зокрема Discogs, було сформовано уявлення про актуальні підходи до організації онлайн-продажів, ведення клієнтської бази, управління замовленнями та товарними позиціями. Кожне з рішень проаналізовано за низкою критеріїв — модель розгортання, можливість кастомізації, наявність ролей користувачів, зручність інтерфейсу, підтримка аналітики, інтеграція платіжних систем тощо.

Результати аналізу були зведені у порівняльну таблицю, що дозволило наочно оцінити сильні та слабкі сторони кожної системи. Зокрема, Shopify вирізняється простотою та готовим e-commerce-функціоналом, але є комерційним і закритим рішенням; Odoo надає розширені можливості та відкритий код, але вимагає глибоких технічних знань; Zoho CRM орієнтований більше на B2B-ринок і не має достатньої гнучкості у сфері торгівлі фізичними товарами.

На основі цього дослідження були сформульовані основні завдання майбутньої системи:

- створення повнофункціональної CRM-системи для адміністраторів та менеджерів з можливістю керування замовленнями (включно зі зміною статусів, коментарями, файлами), редагуванням асортименту, обліком користувачів та збором аналітичної інформації;
- побудова серверної частини для обробки запитів та зберігання даних;
- забезпечення гнучкості, масштабованості та розширюваності розроблюваної системи.
- реалізація веб-додатку (каталог товарів, пошук, фільтрація, замовлення, обліковий запис користувача) та інтеграція як клієнтської частини веб-платформи;

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		20

Окремо було розглянуто технологічні аспекти, проте без порівняння — лише на рівні констатації вибору. Для реалізації клієнтської частини обрано бібліотеку React, яка дозволяє створювати швидкі та адаптивні інтерфейси. Серверна логіка будується з використанням Node.js та Express, а збереження даних відбувається у MongoDB [9].

Таким чином, перший розділ заклав концептуальні та технологічні основи проєкту. Було визначено, які функції слід реалізувати, що саме варто запозичити з готових рішень, а що — розробити з нуля відповідно до потреб магазину вінілових платівок. Власна CRM-система дозволить уникнути обмежень сторонніх платформ і забезпечити повну відповідність технічним та бізнес-вимогам магазину вінілових платівок.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				21
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА CRM-СИСТЕМИ ДЛЯ УПРАВЛІННЯ МАГАЗИНОМ ВІНІЛОВИХ ПЛАТІВОК

2.1 Варіанти використання системи

Веб-платформа «CRM-система для управління діяльністю магазину вінілових платівок» забезпечує цілісний функціонал для трьох категорій користувачів — клієнтів, менеджерів і адміністраторів — а також передбачає інтеграцію з зовнішніми системами. Клієнт користується публічною частиною платформи: він має змогу переглядати каталог вінілових платівок, скористатися пошуком і фільтрацією за виконавцями, жанрами чи роком випуску. Обравши бажані товари, користувач додає їх до кошика та оформлює замовлення — вводить контактні дані й обирає спосіб доставки. В особистому кабінеті клієнт відстежує статуси замовлень, залишає коментарі та оцінює придбані платівки. Додатково реалізовано можливість підписки на розсилки з інформацією про нові релізи, акції або зміну наявності товарів.

Менеджери будуть використовувати модулі CRM-системи для обробки замовлень: вони будуть отримувати сповіщення про нові запити, змінювати статуси (наприклад, "нове", "в обробці", "відправлене"), додавати коментарі та прикріплювати документи — рахунки, накладні, акти. У каталозі платівок менеджери матимуть змогу наповнювати магазин асортиментом: додавати і редагувати позиції, заповнювати описи, завантажувати зображення і актуалізувати наявність. Крім того, окремий модуль системи буде вести лог змін, що дозволить відслідковувати, хто і коли вносив правки до замовлення чи каталогу. Менеджери також формують статистичні звіти за продажами, жанрами, залишками на складі — важливі для управлінських рішень.

Адміністратор системи має повний доступ до налаштувань: він призначає ролі користувачам і визначає права доступу (адміністратори, менеджери). Адміністратор конфігурує параметри оплати, доставки й акцій, встановлює мінімальні суми або лімітні акції. Технічний контроль забезпечує

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		22

моніторинг часу відповіді API, навантаження сервера та регулярне резервне копіювання бази даних із можливістю відновлення.

Розглянемо типові сценарії користувацької взаємодії із CRM-системою для управління магазином вінілових платівок. Всі дії в системі умовно поділяються на три основні ролі: клієнт, менеджер та адміністратор. Кожна з них виконує специфічні функції, що забезпечують повноцінну роботу веб-платформи.

Користувач є ключовим елементом публічної частини системи. Його взаємодія розпочинається з перегляду каталогу товарів, який представлено у вигляді інтерактивного списку із можливістю фільтрації за жанром, виконавцем, назвою, ціною або роком випуску (рисунок 1.4). У разі потреби, користувач може скористатися функцією пошуку для швидкого знаходження необхідної позиції.

Після перегляду товару користувач має змогу додати вибрані платівки до кошика. У кошику можна редагувати кількість обраного товару, видаляти позиції або повністю скасувати сформоване замовлення. Далі користувач переходить до оформлення покупки: вводить контактну інформацію, обирає спосіб оплати, підтверджує замовлення та проводить онлайн-оплату через інтегровану платіжну систему.

Після підтвердження оплати одним з доступних методів, система генерує відповідне замовлення, зберігає його в базі даних та надсилає повідомлення на електронну пошту або в особистий кабінет користувача. Надалі клієнт має змогу відслідковувати статус замовлення (наприклад, «очікує», «в обробці», «відправлено», «завершено»), а також отримувати додаткові коментарі, які може залишити менеджер.

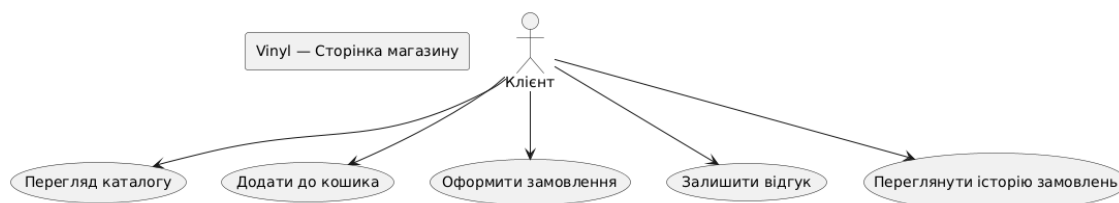


Рисунок 1.4 – Діаграма варіантів використання

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		23

Менеджер працює з внутрішньою частиною системи – CRM-інтерфейсом. Його головне завдання – обробка замовлень. Коли клієнт оформлює замовлення, система автоматично створює відповідний запис у базі даних і генерує повідомлення для менеджера. Повідомлення може бути представлено у вигляді push-сповіщення або просто маркером «нове» у списку замовлень (рисунк 2.1.).

Менеджер відкриває картку замовлення, переглядає його склад, контактні дані користувача, обраний метод доставки та оплати. У разі потреби він має змогу залишити службовий коментар (наприклад, уточнення інформації або повідомлення про зміну умов доставки), який буде видимим для клієнта.

Після перевірки даних менеджер змінює статус замовлення згідно з поточним етапом обробки. Це може бути, наприклад, перехід зі статусу «очікує» до «в обробці», а згодом — до «відправлено». Усі зміни супроводжуються автоматичним сповіщенням користувача, що дозволяє забезпечити прозору комунікацію між клієнтом і магазином.



Рисунок 2.1 – Діаграма варіантів використання

Адміністратор системи виконує функції контролю, налаштування та підтримки. Він має повний доступ до конфігурацій системи, ролей користувачів, а також до журналів активності та службових повідомлень. Основне його завдання — забезпечити стабільну, безпечну та коректну роботу CRM-системи (рисунк 2.2.).

Адміністратор перевіряє, що всі ролі у системі налаштовані відповідно до політики доступу: користувачі мають лише базові права, менеджери —

обмежений доступ до адміністративної частини, а адміністратори — повний доступ до конфігурацій. Він також стежить за технічним станом системи: проводить тестування працездатності модулів, контролює завантаження бази даних і працездатність API.

Крім цього, адміністратор відповідає за оновлення налаштувань, резервне копіювання даних, додавання нових користувачів з особливими правами доступу або розширенням функціоналу відповідно до потреб бізнесу.



Рисунок 2.2. – Діаграма варіантів використання

2.2 Розробка структури веб-платформи

Для візуалізації основних функціональних можливостей платформи було створено діаграму архітектури веб платформи, яка відобрадає основну структуру CRM-системи та її функціонал (рисунок 2.3).

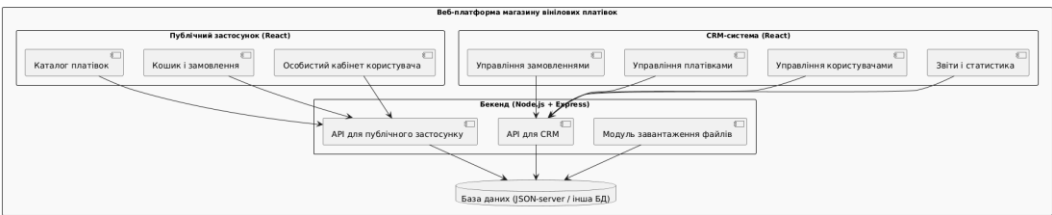


Рисунок 2.3 – Загальна схема архітектури веб-платформи магазину вінілових платівокОсновними акторами виступають клієнт, менеджер і

адміністратор. Після входу в систему клієнт може переглядати каталог платівок, додавати товари до кошика, оформлювати замовлення, залишати відгуки та переглядати історію покупок. Менеджер обробляє замовлення: змінює статуси, додає коментарі та прикріплює документи. Адміністратор керує ролями, налаштовує платіжні та доставочні методи, контролює технічні параметри системи. Діаграма описує загальну структуру сценаріїв використання і ключові функції платформи.

Для формалізації структури інформаційного потоку була побудована IDEF0-діаграма, яка відображає процес обробки інформації як функціональну одиницю з вхідними, вихідними, механізмами та елементами управління (рисунок 2.4).

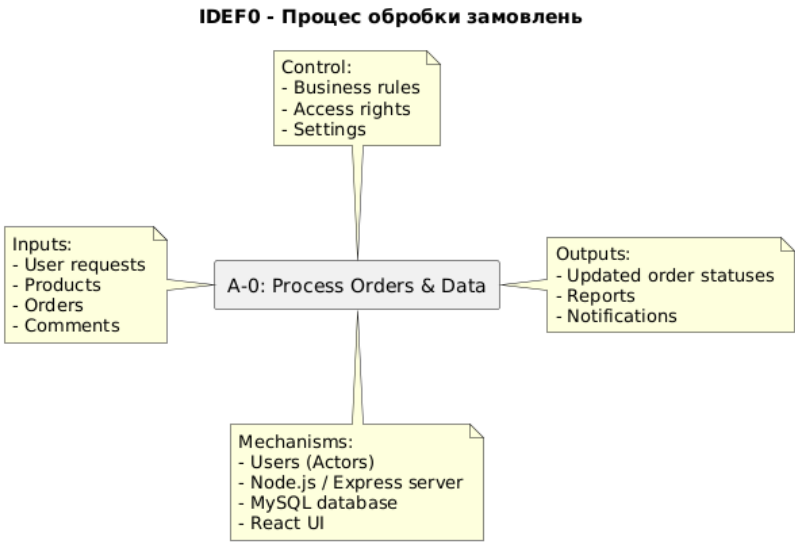


Рисунок 2.4 – IDEF0-діаграма

У центрі знаходиться функція «Процес обробки замовлень і управління даними». Зліва — вхідні дані: запити користувачів, інформація про товари, замовлення і коментарі. Зверху — управління: бізнес-правила обробки замовлень, права доступу, налаштування. Знизу — механізми: користувачі (актори), сервер Express/Node.js, база даних MySQL, клієнтська частина на React. Праворуч — вихідні результати: оновлені статуси замовлень, звіти, сповіщення клієнтам. Така нотація дозволяє структуровано представити засоби, що забезпечують роботу платформи.

Для детальнішого опису системи була створена діаграма декомпозиції, яка демонструє, з яких логічних модулів складається система та як вони пов'язані (рисунок 2.5).



Рисунок 2.5 – Діаграма декомпозиції системи

Головним елементом є «CRM-система магазину», яка розбивається на підсистеми: клієнтський веб-додаток (каталог, кошик, замовлення, кабінет), CRM-модуль (обробка замовлень, каталог товарів, користувачі, статистика), адміністративна панель (ролі, налаштування), інтерфейси інтеграції (платежі, склад, маркетинг) та сервісні функції (авторизація, сповіщення.). Така структура демонструє логіку роботи та взаємодію компонентів на концептуальному рівні.

Також була створена діаграма станів для моделювання статусів замовлення в залежності від дій користувачів та системних подій (рисунок 2.6).

Замовлення переходить між станами: «Нове» → «В обробці» → «Відправлено» → «Завершено» чи «Скасовано». Перехід залежить від виконання умов: підтвердження менеджером, оплатою, доставкою або відміною клієнтом.

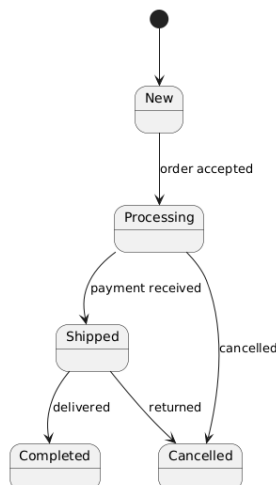


Рисунок 2.6 – Діаграма станів замовлення

2.3 Проектування архітектури веб-платформи

У процесі розробки веб-платформи надзвичайно важливим етапом є формалізація логіки предметної області та її подальше відображення в архітектурі програмного забезпечення. Для цього доцільно застосовувати діаграму класів — один із ключових інструментів об'єктно-орієнтованого моделювання, який дозволяє наочно представити структуру системи, взаємозв'язки між її компонентами, обсяг відповідальності кожного з елементів та характер їхньої взаємодії (рисунок 2.6).

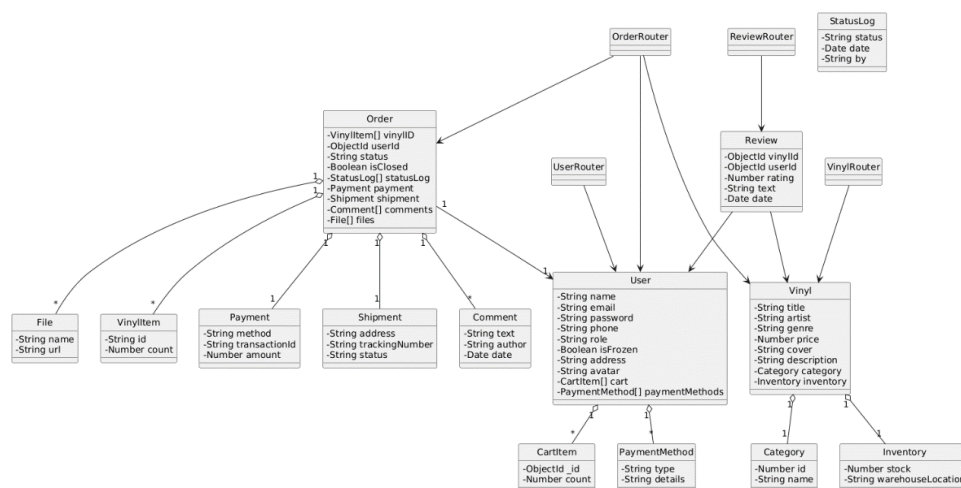


Рисунок 2.6 – Діаграма класів веб-платформи

У контексті проєкту створення CRM-системи для магазину вінілових платівок, діаграма класів відіграє роль "архітектурної мапи", яка дозволяє не

лише краще зрозуміти логіку роботи системи, а й забезпечити її масштабованість, підтримуваність і подальшу розробку. Вона охоплює як основні сутності — користувачі, платівки, замовлення, оплати, відгуки — так і допоміжні структури, які підтримують цілісність даних та логіку бізнес-процесів (наприклад, доставка, статуси, коментарі, прикріплені файли тощо).

Завдяки цій діаграмі можна відстежити, які саме об'єкти беруть участь у кожному з основних сценаріїв взаємодії користувачів із системою: від перегляду каталогу до оформлення замовлення, обробки його менеджером, додавання відгуків і формування звітів. Крім того, діаграма демонструє, як структура моделі відповідає REST-архітектурі, закладеній у бекенді на основі Express та MongoDB.

Основні класи діаграми:

- *User*: Представляє користувача системи. Має такі поля, як name, email, phone, password, role, address, avatar, а також cart для збереження вибраних товарів. Ролі користувача (admin, manager, client) визначають рівень доступу. Зв'язаний із замовленнями, відгуками та повідомленнями.
- *Vinyl*: Описує платівку. Містить title, artist, genre, price, cover, description. Має зв'язки з категорією, відгуками та залишками на складі.
- *Order*: Модель замовлення. Містить масив товарів (vinylID з полем id і count), користувача (userId), статуси, statusLog, доставку (shipment), оплату (payment), коментарі, файли, isClosed.
- *OrderItem*: Відображений у масиві vinylID в Order. Включає id платівки та кількість.
- *Payment* (вкладений в Order): Зберігає спосіб оплати (method), ідентифікатор транзакції (transactionId), суму (amount), статус.
- *Shipment* (вкладений в Order): Містить адресу (address), трек-номер (trackingNumber), статус доставки (status).
- *Comment* (вкладений в Order): Масив коментарів з text, author, date.
- *File* (вкладений в Order): Масив прикріплень з name, url.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				29
Змн.	Арк.	№ докум.	Підпис	Дата		

- *Review*: Модель відгуку з *vinylId*, *userId*, *rating*, *text*, *date*. Зв'язаний з платівками та користувачами через *populate*.
- *Inventory*: Інформація про залишки платівок на складі. Ймовірно, буде реалізовано через окрему модель або розширення *Vinyl*.

Сервісні класи (або логіка всередині роутів):

- *userRoutes* — CRUD-доступ до користувачів.
- *vinylRoutes* — робота з платівками, включно з */vinyls/by-ids*.
- *orderRoutes* — створення, оновлення, перевірка замовлень.
- *reviewRoutes* — додавання, отримання та видалення відгуків.

Таким чином, діаграма класів є невід'ємною частиною архітектурного опису системи, яка забезпечує як високий рівень абстракції, так і технічну точність, необхідну для реалізації, тестування та підтримки веб-платформи магазину вінілових платівок.

2.4 Проектування бази даних системи

Для реалізації CRM-системи магазину вінілових платівок у якості бази даних обрано документно-орієнтовану систему управління базами даних MongoDB. Такий вибір обумовлений низкою переваг цієї технології: гнучка схема зберігання даних дозволяє швидко адаптувати структуру документів під нові бізнес-вимоги без необхідності складних міграцій чи зміни схеми, що є особливо актуальним для динамічних веб-проектів. MongoDB забезпечує високу масштабованість завдяки вбудованим механізмам горизонтального шардінгу та реплікації, що гарантує стабільну роботу системи навіть при збільшенні кількості даних і користувачів. Додатковою перевагою є природна інтеграція з JavaScript-орієнтованими стеком технологій (наприклад, Node.js), що спрощує розробку та обслуговування CRM-системи.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				30
Змн.	Арк.	№ докум.	Підпис	Дата		

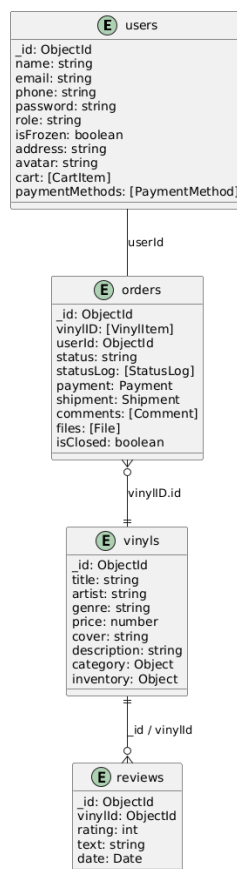


Рисунок 2.7. – Структура бази даних

У базі даних «diploma» організовано чотири основні колекції: users, vinyls, orders і reviews (рисунок 2.7). У колекції users зберігаються документи користувачів із полями name, email, password, role, phone, address, avatar та paymentMethods, а також прапор isFrozen для позначення заблокованих облікових записів. Колекція vinyls містить інформацію про вініли — title, artist, genre, price, cover (URL зображення), description, category і inventory (кількість на складі). У колекції orders кожен документ описує замовлення користувача: масив vinylID, що складається з об'єктів із полями id (посилання на _id платівки) і count (кількість штук), поле userId, статус замовлення, масив statusLog для історії змін статусу, вкладений документ payment із деталями оплати, вкладений документ shipment із даними доставки, масив comments для внутрішніх приміток і поле files для збережених файлів. Колекція reviews акумулює відгуки клієнтів із полями productId, userId, rating, text і date.

Структура даних у MongoDB забезпечує високу продуктивність при читанні й записі великих обсягів інформації, дає змогу застосовувати

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		31

текстовий пошук за полями title, artist і genre та використовувати compound-індекси для оптимізації запитів за статусом замовлення та датою створення. Завдяки вбудованим механізмам реплікації та шардінгу система здатна витримувати високі навантаження й забезпечувати відмовостійкість. Безпека реалізована через зберігання хешів паролів, TLS-шифрування з'єднань і рольовий доступ, коли адміністратори мають повні права, а клієнти — тільки права на перегляд і зміну власних даних і замовлень.

2.5 Проткування функціональних алгоритмів роботи веб-платформи

У процесі розробки веб-платформи для інтернет-магазину вінілових платівок ключовими завданнями стало створення цілісних, безпечних та зрозумілих алгоритмів, які б забезпечували стабільну взаємодію між клієнтом, менеджером та адміністратором. Основні логічні блоки включають авторизацію користувачів, керування замовленнями, редагування профілю, взаємодію з кошиком, адміністрування користувачів, а також централізоване обслуговування API-запитів.

Алгоритмічна реалізація базується на використанні React для клієнтської частини, що відповідає за візуалізацію інтерфейсу та взаємодію з користувачем, і Express/Node.js у поєднанні з MongoDB для реалізації серверної логіки та збереження даних. У подальших підрозділах наведено ключові фрагменти реалізації відповідних функцій з короткими коментарями до їх логіки.

Авторизація є критично важливим елементом будь-якої сучасної системи, оскільки забезпечує контроль доступу до персональних даних, замовлень і захищених сторінок. Процес авторизації розпочинається з отримання облікових даних (email та пароль), введених користувачем у компоненті Login.js. Асинхронна функція handleLogin надсилає POST-запит на бекенд-ендпоінт /users/login, отримує у відповідь токен доступу та об'єкт

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		32

користувача, після чого зберігає ці дані у localStorage. Також відбувається налаштування глобального заголовку Authorization, який автоматично додається до всіх подальших запитів.

Лістинг коду:

```
async function handleLogin(e) {  
  e.preventDefault();  
  const response = await api.post('/users/login', { email, password });  
  const { token, user } = response.data;  
  localStorage.setItem('user', JSON.stringify({ ...user, token }));  
  api.defaults.headers.common['Authorization'] = `Bearer ${token}`;  
  navigate('/profile'); }  

```

Після успішної авторизації користувач має доступ до персоналізованих сторінок. У компоненті App.js реалізовано логіку маршрутизації: якщо користувач вже авторизований (тобто у state збережено токен і дані користувача), то доступ до сторінки профілю надається автоматично. У разі спроби доступу без авторизації система виконує редирект на сторінку входу.

Лістинг коду:

```
<Routes> <Route path="/login" element={user ? <Navigate to="/profile"/> : <Login  
onLogin={handleLogin}/>}/>  
  
<Route path="/profile" element={user ? <Profile user={user}/> : <Navigate to="/login"/>}/>  
  
<Route path="*" element={<Navigate to="/" />}/></Routes>  

```

Компонент Profile містить логіку отримання актуальних даних про клієнта та його замовлення. Запит на /users/me повертає об'єкт із ключовими полями: ім'я, email, телефон, адресу доставки тощо. Після цього виконується другий запит — /orders/with-vinyls, який повертає всі замовлення з розширеною інформацією про платівки та користувача.

Лістинг коду:

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		33

```

useEffect(() => {
  api.get('/users/me').then(res => setFormData(res.data));
  api.get('/orders/with-vinyls')
    .then(res => {
      const myOrders = res.data.filter(
        o => o.userInfo && o.userInfo._id === user._id );
      setOrders(myOrders);
    }); }, [user]);

```

Редагування профілю здійснюється через PATCH-запит. Після оновлення даних ці значення синхронізуються з локальним сховищем браузера (localStorage), щоб уникнути втрати інформації при оновленні сторінки.

Лістинг коду:

```

async function handleSubmit(e) {
  e.preventDefault();
  const { data: updated } = await api.patch('/users/me', formData);
  setFormData(updated);
  const stored = JSON.parse(localStorage.getItem('user'));
  localStorage.setItem('user', JSON.stringify({ ...stored, ...updated }));
  setEditing(false);
  setMessage('Дані успішно оновлено'); }

```

На стороні менеджера реалізовано функціонал обробки замовлень. Компонент Orders завантажує всі замовлення за запитом /orders/with-vinyls, після чого формує масив об'єктів vinylInfos, який містить назви, виконавців та кількість замовлених платівок. Менеджер має можливість змінювати статус замовлення за допомогою PATCH-запиту.

Лістинг коду:

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				34
Змн.	Арк.	№ докум.	Підпис	Дата		

```

function updateStatus(id, status) {
  api.patch(`/orders/${id}`, { status, byRole: 'manager' })
    .then(() => loadOrders());
}

function loadOrders() {
  api.get('/orders/with-vinyls').then(res => {
    setOrders(res.data.map(o => ({
      ...o,
      vinylInfos: o.vinylInfos.map(v => ({
        title: v.title, artist: v.artist, count: v.count })))
    )));
  });
}

```

Серед додаткових алгоритмів, що стосуються адміністрування, — функція «розмороження» користувача. Це дозволяє тимчасово обмежити доступ без повного видалення акаунта. Відповідна кнопка в адмін-панелі викликає PATCH-запит до `/users/:id/freeze`, який змінює булеве значення прапорця `isFrozen`.

Лістинг коду:

```

function toggleFreeze(id, isFrozen) {
  api.patch(`/users/${id}/freeze`, { isFrozen: !isFrozen })
    .then(() => loadUsers()); }

```

Алгоритм роботи з кошиком базується на локальному стані в React (`useState`) та синхронізації з `localStorage`, щоб дані не втрачались між оновленнями сторінки. Функція `addToCart` або `updateQuantity` змінює відповідні значення, а дані зберігаються у контексті, доступному всім компонентам.

Лістинг коду:

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		35

```
function addToCart(vinyl) {
  setCartItems(prev => {
    const item = prev.find(i => i.id === vinyl.id);
    if (item) item.quantity += 1;
    else prev.push({ ...vinyl, quantity: 1 });
    return [...prev]; });
}
```

Усі запити винесено в окремий модуль `api.js` на базі бібліотеки `axios` [8]. Використання перехоплювачів (`interceptors`) дозволяє автоматично додавати токен авторизації до кожного запиту, що значно спрощує обслуговування API та підвищує безпеку взаємодії.

Лістинг коду:

```
import axios from 'axios';

const api = axios.create({ baseURL: 'http://localhost:3001' });

api.interceptors.request.use(config => {
  const user = JSON.parse(localStorage.getItem('user') || '{}');
  if (user.token) config.headers.Authorization = `Bearer ${user.token}`;
  return config; });

export default api;
```

Побудовані алгоритми є основою функціонування клієнтської та адміністративної частини CRM-системи. Кожен алгоритм реалізовано з урахуванням принципів модульності, читабельності, повторного використання коду та безпеки даних. Особливу увагу приділено зручності користувача, що забезпечено завдяки автоматичним оновленням стану, збереженню даних у `localStorage` та розподіленню логіки на окремі компоненти та сервіси.

Завдяки цьому реалізована система є масштабованою, гнучкою та готовою до подальшого розширення — як у межах функціоналу (нові ролі, додаткові дії), так і в технічному плані (підключення нових API, мобільної версії тощо). Розмежування логіки між фронтендом і бекендом,

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		36

централізоване управління API та використання сучасних підходів у маршрутизації та управлінні станом формують якісну інженерну основу системи.

Висновки до другого розділу

Було здійснено комплексне проектування веб-платформи «CRM-система для управління діяльністю магазину вінілових платівок», що охоплює як веб-додаток для клієнтів, так і технічну реалізацію функціональних елементів CRM-системи для менеджерів та адміністраторів. Розроблено архітектуру, яка поєднує модулі CRM-системи для управління каталогом товарів, обробки замовлень, збору аналітики та взаємодії з клієнтами, що забезпечує гнучкість та масштабованість платформи.

Для формалізації логіки роботи системи побудовано низку UML-діаграм: діаграма варіантів використання, діаграма станів, IDEF0-діаграма, діаграма декомпозиції та діаграма класів. Вони відображають структуру, поведінку та функціональні можливості системи на різних рівнях абстракції. Зокрема, діаграма класів продемонструвала модульну архітектуру бекенд-коду та взаємозв'язки між сутностями, а діаграма станів відобразила послідовність змін статусів замовлень у процесі їх обробки.

Було реалізовано основні механізми роботи системи: управління каталогом товарів, ведення обліку залишків на складі, формування замовлень та їх обробка менеджерами, управління користувачами та ролями, створення та надсилання сповіщень клієнтам. Детально описано логіку основних компонентів, зокрема обробку замовлень, взаємодію клієнта з публічною частиною сайту, роботу з кошиком та історією покупок, формування звітів.

Також було реалізовано дизайн у мінімалістичному стилі з інтуїтивно зрозумілим інтерфейсом користувача. Інтерфейс публічної частини розроблено з використанням React, а серверна частина реалізована на Node.js із застосуванням Express. Усі компоненти платформи взаємодіють між собою

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				37
Змн.	Арк.	№ докум.	Підпис	Дата		

через REST API, що забезпечує надійність та можливість подальшої інтеграції з іншими сервісами.

Отже, на цьому етапі завершено логічне та технічне проєктування системи, визначено її структуру, функціональні модулі та взаємодії, що забезпечує основу для подальшої реалізації, тестування та вдосконалення платформи.

РОЗДІЛ 3. ОПИС РОБОТИ З ВЕБ-ПЛАТФОРМОЮ ТА ЇЇ ТЕСТУВАННЯ

3.1 Огляд інтерфейсу веб-платформи

Інтерфейс веб-платформи спроектовано з урахуванням принципів зручності використання, інтуїтивної навігації та сучасних тенденцій веб-дизайну, що забезпечує комфортну роботу для всіх категорій користувачів: адміністраторів, менеджерів магазину та звичайних клієнтів. Кожна група користувачів отримує доступ до функціоналу, який відповідає її ролі у системі, що сприяє ефективному виконанню основних бізнес-процесів та гарантує захищеність даних.

В адміністративній панелі адміністратор має розширені можливості для управління обліковими записами користувачів: перегляд, додавання нових користувачів, редагування наявних профілів, блокування або розблокування акаунтів за необхідності (рисунок 3.1). Також реалізовано функції швидкого пошуку та фільтрації за різними параметрами (роль, статус, активність тощо), що дозволяє оперативно знаходити потрібну інформацію і підтримувати порядок у системі. Такий підхід забезпечує повний контроль над доступом до функцій платформи й сприяє безперебійному адмініструванню веб-платформи.

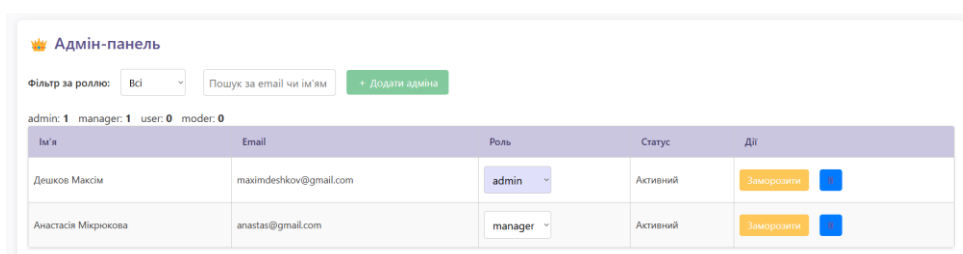


Рисунок 3.1 – Управління користувачами (адмін)

Шапка сторінки містить основні навігаційні елементи, логотип магазину, меню переходу до основних розділів, а також індикатор авторизації користувача. Завдяки цьому забезпечується інтуїтивно зрозумілий доступ до всіх ключових функцій платформи.

Рисунок 3. – Вигляд шапки сторінки

На головній сторінці інтерфейсу менеджера магазину розміщено інтерактивний дашборд із ключовими аналітичними показниками (рисунок 3.2.). Тут у зручній та наочній формі відображаються загальна кількість замовлень за обраний період, графік динаміки продажів, поточна та накопичена виручка, а також статистика активності користувачів системи. Завдяки цьому менеджер може в реальному часі оперативно відстежувати основні бізнес-показники, своєчасно реагувати на зміни в попиті та приймати обґрунтовані управлінські рішення для підвищення ефективності роботи магазину. оцінювати стан бізнесу.

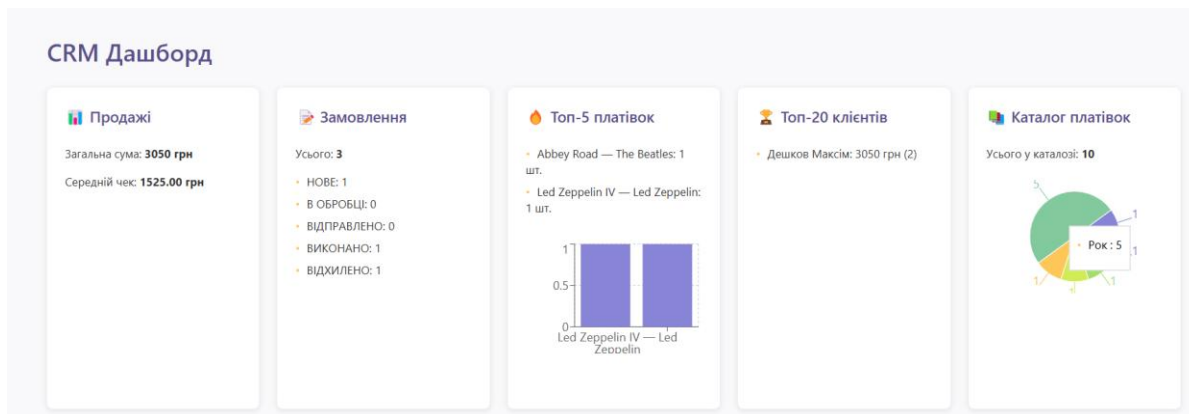


Рисунок 3.2 – Дашборд, аналітика за замовч. (менеджер)

Для проведення поглибленого аналізу в системі реалізовано аналітичний конструктор, який надає менеджеру гнучкі інструменти для самостійного формування звітів. Завдяки інтерактивному інтерфейсу користувач може будувати різноманітні графіки та таблиці, застосовуючи необхідні фільтри, групування та обираючи довільні часові періоди(рисунок 3.3). Це дає змогу детально досліджувати динаміку продажів, порівнювати ефективність різних категорій товарів, аналізувати поведінку клієнтів і виявляти тенденції. Така аналітична функціональність суттєво підвищує якість управлінських рішень та

дозволяє своєчасно адаптувати стратегію розвитку магазину відповідно до отриманих даних.

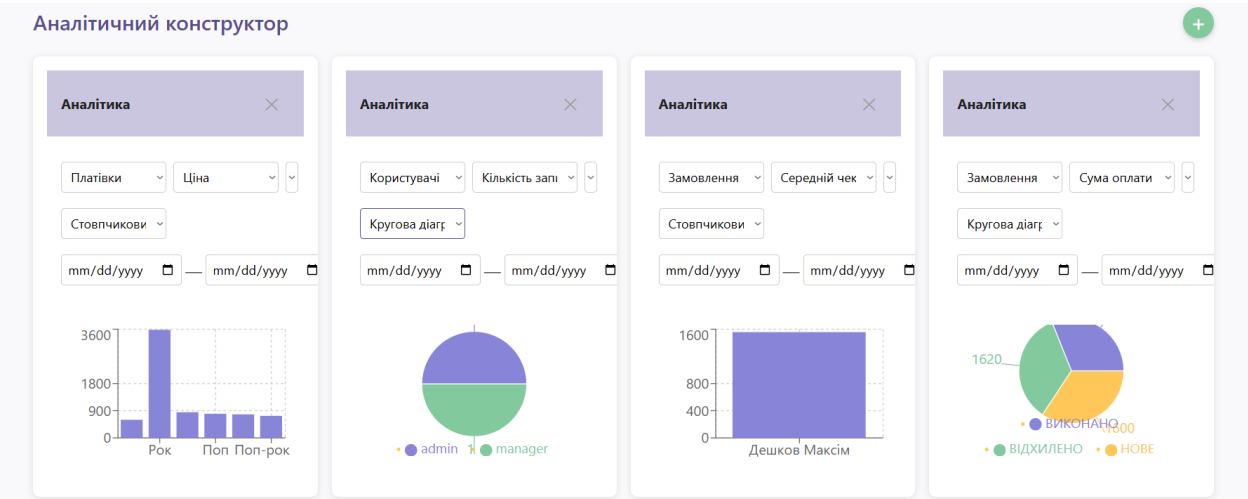


Рисунок 3.3 – Дашборд, налітичний конструктор (менеджер)

Менеджер магазину має можливість переглядати повний перелік поточних та минулих замовлень через спеціалізований інтерфейс управління. У цьому розділі доступна функція зміни статусу кожного замовлення (наприклад, “обробляється”, “відправлено”, “доставлено” тощо). За потреби менеджер може ініціювати прямий зв’язок із клієнтом через доступні контактні дані. Для зручності роботи передбачено відкриття детальної інформації про кожне замовлення у окремому діалоговому вікні, де відображаються всі супутні дані — склад замовлення, історія змін статусу, фінансова інформація та коментарі. Такий функціонал забезпечує повний контроль над процесом обробки замовлень і підвищує якість сервісу для клієнтів.

Список замовлень				
Статус:		Пошук:		
Всі		ID / ПІБ / Телефон / Email		
ID	Платівки	Статус	Контакти замовника	Дії
68515a1415295d10ada736c7	Abbey Road — The Beatles (1 шт.)	ВИКОНАНО Замовлення закрито	ПІБ: Дешков Максим Телефон: maximdeschkov@gmail.com Email: maximdeschkov@gmail.com Адреса: Адмін вул. 1 м. Житомир	Відкрити (адмін)
6852042415295d10ada736e7	Back in Black — AC/DC (1 шт.), The Dark Side of the Moon — Pink Floyd (1 шт.)	ВІДХИЛЕНО Замовлення закрито	ПІБ: Дешков Максим Телефон: maximdeschkov@gmail.com Email: maximdeschkov@gmail.com Адреса: Адмін вул. 1 м. Житомир	Відкрити (адмін)
6852043015295d10ada736e8	Led Zeppelin IV — Led Zeppelin (1 шт.)	НОВЕ	ПІБ: Дешков Максим Телефон: maximdeschkov@gmail.com Email: maximdeschkov@gmail.com Адреса: Адмін вул. 1 м. Житомир	

Рисунок 3.4 – Управління замовленнями (менеджер)

В інтерфейсі управління платівками менеджер має можливість ефективно працювати з товарним асортиментом магазину. Система дозволяє додавати нові позиції у каталог, вказуючи їх основні характеристики: назву, виконавця, жанр, ціну, опис та ілюстративні матеріали. Для вже існуючих товарів передбачено оперативне редагування даних — зміна ціни, оновлення опису або завантаження нових зображень. Завдяки зручним інструментам пошуку та фільтрації менеджер може швидко знаходити потрібні платівки за різними параметрами, що суттєво полегшує роботу навіть з великим каталогом. Окрім цього, у інтерфейсі відображається інформація про залишки товару на складі, що дає змогу своєчасно відслідковувати рівень запасів та приймати рішення щодо їх поповнення.







Дашбордин	Замовлення	Платівки	Користувачі	Адмін панель	Вийти
Каталог платівок					
Додати платівку					
Назва	Виконавець	Жанр	Ціна	Обкладинка	Дії
Abbey Road	The Beatles	Рок	700		Редагувати Відкрити
Back in Black	AC/DC	Рок	700		Редагувати Відкрити
Born to Run	Bruce Springsteen	Рок	690		Редагувати Відкрити
Hotel California	Eagles	Рок	770		Редагувати Відкрити
Kind of Blue	Miles Davis	Джаз	600		Редагувати Відкрити
Led Zeppelin IV	Led Zeppelin	Хард-рок	780		Редагувати Відкрити

Рисунок 3.4 – Управління платівками (менеджер)

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				42
Змн.	Арк.	№ докум.	Підпис	Дата		

Для менеджера в системі реалізовано спеціальний розділ, що надає повний перелік усіх зареєстрованих користувачів платформи. Інтерфейс цього модуля містить зручні інструменти для швидкого пошуку користувачів за різними параметрами — такими як ім'я, електронна пошта, роль або статус облікового запису (рисунок 3.5). Крім того, реалізовано функцію сортування списку за обраними критеріями, наприклад, за датою реєстрації, останньою активністю чи алфавітом. Це дозволяє менеджеру оперативно знаходити потрібну інформацію, здійснювати аналіз структури користувачів, контролювати доступ та ефективно управляти клієнтською базою.

Список користувачів								
Фільтр за ім'ям		Фільтр за email		Фільтр за телефоном		Усі ролі		Усі статуси
ID	Ім'я	Email	Телефон	Роль	Статус	Адреса	Аватар	Кошик
685159d915295d10ada736ba	Дешков Максим	maximdeshev@gmail.com	0960234274	admin	Активний	Адмін вул. 1 м. Житомир	—	<ul style="list-style-type: none"> ID: 685184eb15295d10ada736db, кількість: 1 ID: 685184e215295d10ada736da, кількість: 1
689959d915295d10ada736ba	Анастасія Мікрюкова	anastas@gmail.com	+380960234274	manager	Активний	Адмін вул. 1 м. Житомир	—	—

Рисунок 3.5 – Сторінка список користувачів (менеджер)

Для підтримки високої якості сервісу та дотримання корпоративних стандартів у системі передбачено окремий модуль модерування відгуків та коментарів клієнтів. Менеджер може оперативно переглядати всі залишені користувачами відгуки, а також застосовувати необхідні дії щодо кожного запису: приховувати відображення неприйнятних або некоректних повідомлень (рисунок 3.6.). Такий функціонал дозволяє ефективно контролювати інформаційне середовище магазину, своєчасно реагувати на порушення правил користування платформою та забезпечувати позитивний досвід для всіх клієнтів.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		43

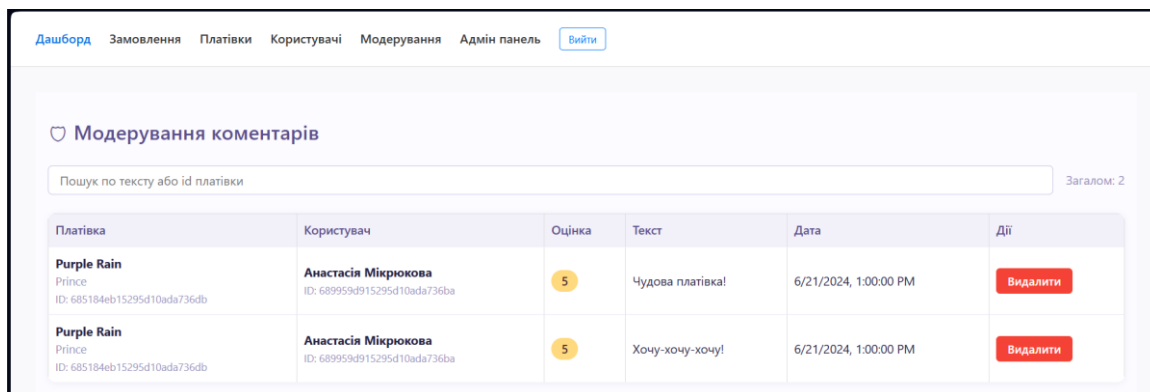


Рисунок 3.6 – Модерування коментарів (менеджер)

Головна сторінка інтерфейсу для клієнта розроблена з урахуванням зручності та інформативності. Тут розміщено коротку презентацію магазину, яка знайомить відвідувачів з основними перевагами сервісу та його асортиментом (рисунок 3.7).

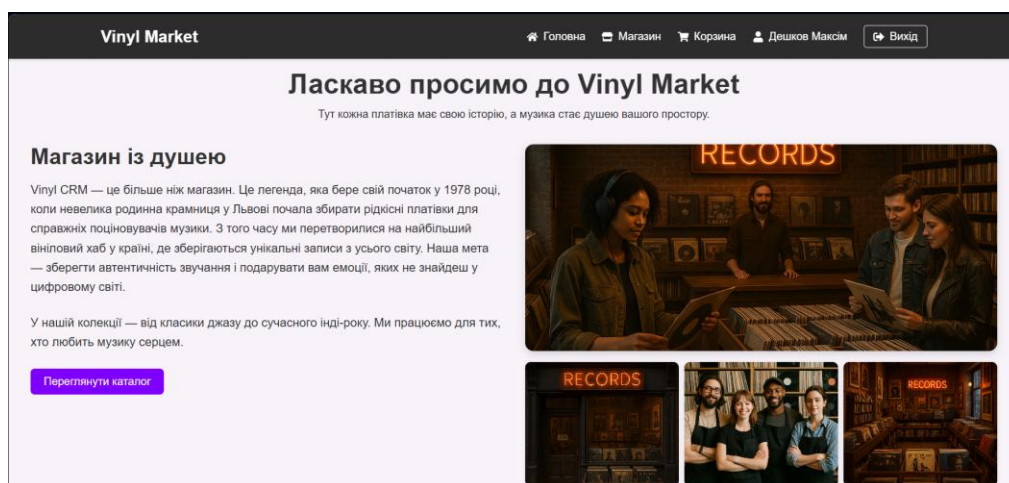


Рисунок 3.7 – Домашня сторінка магазину (користувач)

У каталозі магазину клієнтам доступний повний перелік усіх наявних платівок із детальним описом характеристик кожної позиції. Для зручності користувачів реалізовано функціонал багатокритеріального фільтрування, який дозволяє швидко знаходити потрібні товари за виконавцем, жанром, ціною та іншими параметрами (рисунок 3.8.). Також передбачено можливість текстового пошуку по каталогу, що особливо актуально для постійних клієнтів

або поціновувачів окремих виконавців. Такий підхід суттєво спрощує навігацію, дозволяє оперативно орієнтуватися у великому асортименті й забезпечує індивідуальний підхід до кожного покупця.

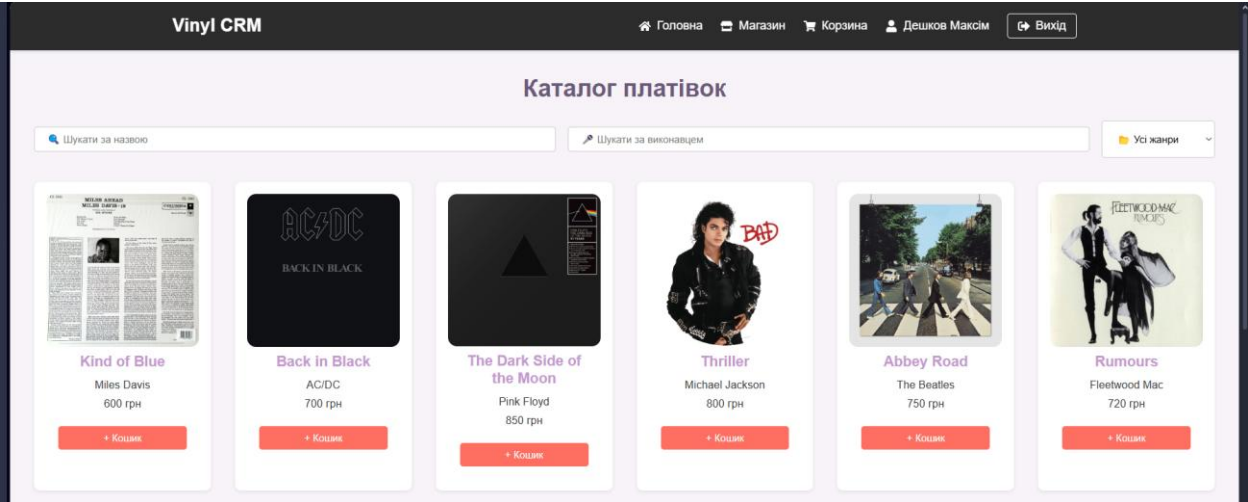


Рисунок 3.8 – Каталог магазину (користувач)

У кошику відображається повний перелік вибраних товарів. Для кожного товару вказується його назва, зображення, ціна за одиницю, вибрана кількість та підсумкова сума за дану позицію. Всі дані оновлюються автоматично при зміні кількості або видаленні товару з кошика. Окрім цього, в нижній частині кошика відображається загальна сума замовлення, яка також змінюється динамічно залежно від дій користувача (рисунок 3.9).

Функціонал кошика передбачає можливість змінювати кількість товару безпосередньо у вікні кошика. Для цього передбачені кнопки для збільшення або зменшення кількості, а також поле для прямого введення числа. Будь-які зміни кількості миттєво відображаються у підсумковій вартості конкретного товару та загальній сумі замовлення.

Також передбачено можливість видалення товару з кошика. Біля кожної позиції є кнопка для видалення, після натискання на яку товар миттєво прибирається з переліку, а загальна сума автоматично перераховується.

Кошик реалізований з використанням контексту React для збереження стану даних протягом сеансу роботи користувача. Це дозволяє зберігати дані навіть під час переходів між сторінками. У підсумку реалізовано інтерактивний і зручний для користувача модуль кошика, що забезпечує повний контроль над вибраними товарами та спрощує процес оформлення замовлення.

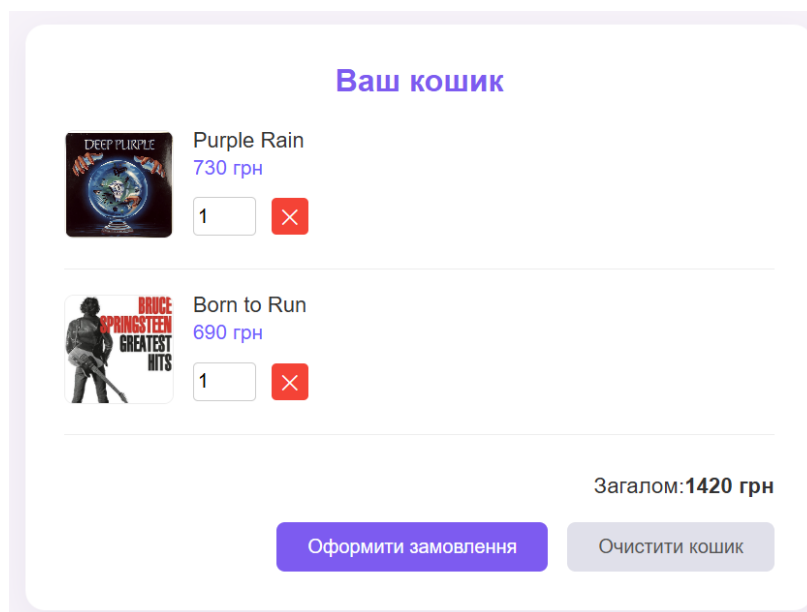


Рисунок 3.9 – Корзина (користувач)

У профілі відображаються основні дані користувача, зокрема ім'я, електронна пошта, адреса доставки та номер телефону. Усі ці дані користувач може змінювати безпосередньо на сторінці профілю за допомогою інтерактивної форми. Для цього передбачено окремі поля вводу, заповнені поточними значеннями, а також кнопка для збереження змін після редагування. Внесені зміни зберігаються у базі даних і автоматично оновлюють дані користувача в системі. Додатково профіль містить таблицю із переліком замовлень користувача. Для кожного замовлення в таблиці відображаються:

- ID замовлення;
- дата оформлення;
- статус замовлення (наприклад, «Виконано», «Нове»);
- перелік придбаних платівок із зазначенням виконавця та назви альбому;

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				46
Змн.	Арк.	№ докум.	Підпис	Дата		

– підсумкова сума замовлення.

Реалізація профілю виконана з використанням React для побудови інтерфейсу та забезпечення динамічного оновлення даних. Інформація про замовлення завантажується з бекенду через API під час відкриття сторінки профілю. Це дозволяє користувачу в реальному часі бачити актуальний стан своїх замовлень.

Результатом реалізації є функціональний модуль профілю користувача (рис 3.10), який дозволяє зручно управляти особистими даними, відстежувати історію покупок та підтримувати зв'язок із магазином для подальшої роботи з замовленнями.

ID	Дата	Статус	Платівки	Сума
68515a1415295d10ada736c7	—	ВИКОНАНО	• Abbey Road — The Beatles (x1)	1450 грн
6852042415295d10ada736e7	—	ВІДХИЛЕНО	• Back in Black — AC/DC (x1) • The Dark Side of the Moon — Pink Floyd (x1)	1620 грн
6852043015295d10ada736e8	—	НОВЕ	• Led Zeppelin IV — Led Zeppelin (x1)	1800 грн

Рисунок 3.10 – Профіль та замовлення (користувач)

На сторінці оформлення замовлення користувач має можливість завершити покупку обраних платівок у зручному та зрозумілому інтерфейсі. Тут відображається підсумковий список усіх товарів із зазначенням кількості, вартості та загальної суми до сплати (3.11). Користувач заповнює персональні контактні дані (ім'я, електронну адресу, номер телефону), а також адресу доставки. Передбачено вибір способу оплати з-поміж доступних варіантів. Для фінального підтвердження замовлення потрібно перевірити введену інформацію й натиснути кнопку оформлення. Після завершення процесу користувач отримує відповідне повідомлення про успішне прийняття замовлення, а дані про нове замовлення автоматично зберігаються в

особистому кабінеті. Такий підхід до організації сторінки забезпечує швидкість і зручність оформлення покупки, підвищує рівень користувацького досвіду та сприяє зростанню лояльності клієнтів.

Рисунок 3.11 – Оформлення замовлення (користувач)

3.2 Проблематика під час розробки

У процесі створення веб-платформи «Vinyl CRM» виникло декілька ключових технічних і організаційних викликів, які потребували ретельного аналізу та поетапного вирішення. Нижче описано основні з них із наведеними ілюстраціями (скріншотами) та прикладами коду, що допомогли подолати ці труднощі.

В першу чергу, при спробі зібрати фронтенд виникла помилка з відсутнім модулем `zlib` у пакеті `body-parser`. Webpack 5 за замовчуванням не надає поліфіли для Node.js-модулів, тому під час запуску збірки з'являється повідомлення:

Вирішення полягало в додаванні в конфігурацію Webpack наступного блоку у `webpack.config.js`:

Лістинг коду:

```
module.exports = {  
  
  // ...
```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				48
Змн.	Арк.	№ докум.	Підпис	Дата		


```

resolve: {
  fallback: {
    zlib: require.resolve("browserify-zlib"),
    querystring: require.resolve("querystring-es3")
  }
}

```

Друга проблема стосувалась механізму авторизації. Після видалення AuthContext увесь потік логіну/логауту довелося реалізувати в App.js та Login.js. Спочатку при спробі зберегти user у localStorage й одразу редіректити в /profile виникав безкінечний цикл перенаправлень.

Щоб виправити, було впроваджено умову в маршрутизації в App.js

Лістинг коду:

```

<Route
  path="/login"
  element={user
    ? <Navigate to="/profile" replace />
    : <Login onLogin={handleLogin}/>
  }
/>

<Route
  path="/profile"
  element={user
    ? <Profile user={user}/>
    : <Navigate to="/login" replace />
  }
/>

```

Це гарантувало, що при наявності user ви ніколи не потрапите на форму входу, а без user — не зможете зайти в профіль.

Третім викликом стало некоректне вивантаження даних профілю та замовлень у компоненті Profile.js. Спочатку застосовувався маршрут /users/me,

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		49

але на сервері його не було, і при спробі GET /users/me відповідь була 404, тому дані не завантажувались.

Вирішили це так:

Замінити GET /users/me на GET /users/\${user._id}

Після PATCH-запиту оновлювати не тільки стан компонента, а й localStorage.

Лістинг коду:

```
// правильний виклик
api.get(`/users/${user._id}`)
  .then(res => setFormData(res.data));

// синхронізація після оновлення
api.patch(`/users/${user._id}`, formData)
  .then(res => {
    const upd = res.data;
    localStorage.setItem('user', JSON.stringify({ ...user, ...upd }));
  });
```

Четвертий виклик виник у момент інтеграції фільтрації замовлень. Маршрут /orders повертав лише ID та статус, а потрібні були дані про платівки. Тому на бекенді був створений новий маршрут /orders/with-vinyls, який робить агрегацію у MongoDB із \$lookup по колекції vinyls. На фронті замінили api.get('/orders') на:

Лістинг коду:

```
api.get('/orders/with-vinyls')
  .then(res => {
    const myOrders = res.data
    .filter(o => String(o.userInfo._id) === String(user._id));
    setOrders(myOrders);
  });
```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				50
Змн.	Арк.	№ докум.	Підпис	Дата		

П'ятий момент стосувався стилізації та резиновості інтерфейсу: спочатку шаблон SCSS для Header і Footer використовував кольори та змінні, проте на мобільних екранах лейаут “ламається”. Додали медіа-запити:

Лістинг коду:

```
@media (max-width: 768px) {  
  .header .nav { display: none; }  
  .footer .footer-container { flex-direction: column; }  
}
```

Вирішення кожної з цих проблем вимагало ретельного аналізу стеку помилок, зміни конфігурацій збірки та коригування маршрутів API. Виконані кроки дозволили створити стабільну та функціональну CRM-систему з чітким механізмом авторизації, захищеними маршрутами й коректною обробкою даних у клієнт- та сервер-частинах.

3.3 Тестування системи

Тестування є невід’ємною частиною процесу розробки веб-платформи, оскільки дозволяє перевірити коректність реалізації функціоналу, відповідність системи вимогам замовника та її готовність до експлуатації. Метою проведення тестування є виявлення можливих помилок у роботі окремих модулів і системи загалом, а також підтвердження стабільності та надійності роботи під час виконання ключових бізнес-процесів.

Тестування проводиться для того, щоб гарантувати:

- правильність роботи основних сценаріїв використання системи;
- відповідність реалізованого функціоналу технічному завданню та очікуванням користувача;
- стабільність роботи під час взаємодії між окремими компонентами системи (публічною частиною та CRM);
- зручність і безпеку роботи кінцевого користувача.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				51
Змн.	Арк.	№ докум.	Підпис	Дата		

У процесі тестування було розроблено набір тест-кейсів для перевірки найбільш критичних функцій системи, зокрема роботи авторизації, додавання товарів у кошик, оформлення замовлень, управління замовленнями менеджером, блокування користувачів адміністратором та роботи пошуку у каталозі. Проведення тестування дозволяє впевнитися, що веб-платформа є працездатною, стабільною та готовою до подальшого впровадження.

Таблиця 3.1

Тест кейс №1

Назва	Авторизація користувача
Опис	Перевірка можливості входу зареєстрованого користувача з валідними даними
Кроки	1. Перейти на сторінку входу 2. Ввести email та пароль 3. Натиснути кнопку “Sign in”
Очікуваний результат	Користувач успішно входить у систему, відкривається головна сторінка профілю
Результат	Успішно

Таблиця 3.2

Тест кейс №2

Назва	Додавання платівки до кошика
Опис	Перевірка можливості додати товар у кошик із каталогу
Кроки	1. Перейти у каталог 2. Обрати товар 3. Натиснути “Додати у кошик”
Очікуваний результат	Товар з’являється у кошику, кількість і сума замовлення оновлюються
Результат	Успішно

Таблиця 3.3

Тест кейс №3

Назва	Оформлення замовлення
Опис	Перевірка можливості оформлення замовлення з кошика
Кроки	1. Перейти у кошик 2. Натиснути “Оформити замовлення” 3. Заповнити контактні дані 4. Підтвердити замовлення
Очікуваний результат	Система створює нове замовлення, з’являється повідомлення про успішне оформлення
Результат	Успішно

Таблиця 3.4

Тест кейс №4

Назва	Зміна статусу замовлення менеджером
Опис	Перевірка можливості зміни статусу замовлення у CRM
Кроки	1. Авторизуватися як менеджер 2. Перейти у розділ “Замовлення” 3. Обрати замовлення 4. Змінити статус (наприклад, на “Виконано”)
Очікуваний результат	Статус замовлення оновлюється, клієнт бачить зміну у своєму профілі
Результат	Успішно

Таблиця 3.5

Тест кейс №5

Назва	Заморозка користувача адміністратором
Опис	Перевірка функції блокування акаунту користувача через адмін-панель
Кроки	1. Авторизуватися як адміністратор 2. Перейти у розділ “Користувачі” 3. Знайти потрібного користувача 4. Натиснути “Блокувати”
Очікуваний результат	Статус акаунту змінюється на “Заблоковано”, користувач не може авторизуватися
Результат	Успішно

Тест кейс №6

Назва	Пошук платівки у каталозі
Опис	Перевірка роботи пошуку платівок за ім'ям виконавця в каталозі
Кроки	1. Перейти у розділ “Каталог” 2. Ввести у поле пошуку ім'я виконавця 3. Натиснути кнопку “Пошук”
Очікуваний результат	Відображаються лише ті платівки, які відповідають введеному виконавцю
Результат	Успішно

Висновки до третього розділу

У третьому розділі було детально розглянуто інтерфейс веб-платформи «CRM-система для управління діяльністю магазину вінілових платівок», описано особливості взаємодії різних категорій користувачів із системою та наведено приклади ключових екранів. Зроблено акцент на зручності, функціональності й інтуїтивності розробленого інтерфейсу, що забезпечує ефективну роботу адміністраторів, менеджерів і клієнтів магазину.

В ході розробки системи було вирішено низку технічних та організаційних проблем, зокрема питання авторизації користувачів, синхронізації даних, відображення інформації про замовлення, а також оптимізації інтерфейсу для різних пристроїв. Описані труднощі та способи їх вирішення свідчать про глибоке розуміння особливостей сучасних вебтехнологій і гнучкий підхід до реалізації функціональних вимог.

Окрему увагу приділено тестуванню системи: було складено перелік тест-кейсів для основних бізнес-сценаріїв, що охоплюють функціональність реєстрації, авторизації, управління кошиком, оформлення замовлення, зміни статусу, блокування користувачів, а також пошуку в каталозі. Результати тестування підтвердили працездатність основних модулів системи, їх

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		54

відповідність поставленим вимогам, а також забезпечили високий рівень зручності й безпеки для кінцевих користувачів.

Реалізована веб-платформа відповідає сучасним вимогам до CRM-рішень для електронної комерції, демонструє високу якість виконання інтерфейсних і функціональних рішень, а також готовність до подальшого масштабування й розвитку.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				55
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У межах кваліфікаційної роботи було створено веб-платформу «CRM-система для управління діяльністю магазину вінілових платівок», яка поєднує веб-додаток для клієнтів магазину вінілових платівок та CRM-систему для менеджерів і адміністратора. Реалізована система охоплює повний спектр функціональності, необхідної для забезпечення основних бізнес-процесів магазину, включаючи каталог товарів, кошик, систему замовлень, особистий кабінет користувача, а також засоби адміністрування, аналітики та управління контентом.

У першому розділі було здійснено аналіз предметної області, визначено функціональні потреби користувачів, сформульовано вимоги до системи та обґрунтовано вибір технологічного стеку. Було розглянуто типові проблеми управління у сфері онлайн-торгівлі, проведено аналіз існуючих аналогів, визначено їхні переваги та недоліки, а також показано, як запропоноване рішення дозволяє ефективно вирішити виявлені проблеми.

У другому розділі розроблено архітектуру інформаційної системи, змодельовано основні бізнес-процеси із застосуванням UML-діаграм, спроектовано структуру бази даних, а також функціона користувача для клієнтського веб-додатку, та адміністративного модуля. Архітектура була побудована за принципами модульності, розширюваності та інкапсуляції.

У третьому розділі реалізовано функціональні компоненти системи, налагоджено маршрутизацію, механізми авторизації та доступу, організовано взаємодію між модулями за допомогою REST API. Проведено модульне й інтеграційне тестування, а також тестування типових сценаріїв користувача для перевірки стабільності роботи й відповідності функціоналу вимогам.

Розроблена система є відповіддю на актуальні виклики цифровізації роздрібної торгівлі, зокрема — потребу в оптимізації внутрішніх процесів, мінімізації людського фактору та підвищенні клієнтського досвіду. В умовах зростаючої конкуренції на ринку e-commerce, використання подібних CRM-рішень стає необхідною складовою ефективної комерційної діяльності.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				56
Змн.	Арк.	№ докум.	Підпис	Дата		

У процесі розробки було спроектовано архітектуру застосунку з використанням технологій React для клієнтської частини, Express та Node.js для серверної логіки, MongoDB для зберігання даних, а також REST API для взаємодії між модулями. Рольовий доступ, механізми авторизації, налаштування прав і робота з токенами були реалізовані відповідно до вимог щодо безпеки даних. У результаті аналізу предметної області було змодельовано ключові бізнес-процеси, після чого здійснено їх реалізацію у вигляді функціональних компонентів, відповідальних за обробку замовлень, модерування відгуків, керування асортиментом товарів та обліковими записами.

Архітектура проєкту побудована таким чином, щоб забезпечити гнучкість у масштабуванні функціоналу, адаптацію до змін бізнес-процесів та подальшу інтеграцію з мобільними або сторонніми сервісами. Завдяки використанню модульного підходу та сучасних вебтехнологій, платформа може слугувати

У процесі створення платформи виникли технічні труднощі, зокрема конфлікти з модулями Node.js під час збирання проєкту, потреба у переналаштуванні механізмів авторизації після видалення глобального контексту, помилки під час завантаження даних користувачів через некоректні маршрути, а також необхідність реалізації додаткових запитів для отримання повної інформації щодо замовлень. Вирішення цих питань потребувало модифікації конфігураційного файлу збірки, впровадження додаткової логіки маршрутизації, коригування обробників API та налаштування агрегованих запитів до бази даних. Окрему увагу приділено забезпеченню адаптивності інтерфейсу для різних типів пристроїв, що реалізовано через медіа-запити та налаштування стилів.

Функціональність системи було перевірено за допомогою прикладних сценаріїв користувача. Тестування підтвердило стабільність роботи всіх модулів, відповідність функціоналу заданим вимогам, коректність маршрутизації та безпечність взаємодії з даними. Реалізована платформа може

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				57
Змн.	Арк.	№ докум.	Підпис	Дата		

бути розгорнута для використання в комерційних умовах або адаптована до потреб інших типів роздрібно́ї торгівлі.

В результаті виконання роботи створено цифровий інструмент, що дозволяє централізовано керувати ключовими процесами магазину вінілових платівок, знижує ризик помилок у роботі з клієнтськими замовленнями та забезпечує структуроване середовище для роботи персоналу. Реалізований підхід є прикладом практичного застосування сучасних технологій у розв’язанні конкретних бізнес-завдань.

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				58
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дешков М.Ю. Огляд систем проєктного менеджменту. Тези доповідей XIV Міжнародної науково-технічної конференції “Інформаційно комп’ютерні технології” 28-29 березня 2024 року. Житомир : "Житомирська політехніка", 2024. <https://conf.ztu.edu.ua/wp-content/uploads/2024/05/106.pdf>
2. Shopify [Електронний ресурс] — Режим доступу: <https://www.shopify.com/> — Дата звернення: 01.06.2025
3. Odoo CRM [Електронний ресурс] — Режим доступу: <https://www.odoo.com/page/crm> — Дата звернення: 01.06.2025
4. Zoho CRM [Електронний ресурс] — Режим доступу: <https://www.zoho.com/crm/> — Дата звернення: 01.06.2025
5. Ігнатенко О.І. Перспективи використання CRM-систем в Україні за сучасних умов. Економічний вісник університету, 2024. URL: https://www.researchgate.net/publication/371111741_Prospects_of_using_CRM_systems_in_Ukraine_under_modern_conditions (дата звернення: 1.05.2025) researchgate.net
6. Коташева Д.В. Використання CRM-систем для розробки та впровадження стратегій цифрового бренд-менеджменту й інтернет-маркетингу (досвід країн ЄС). Маркетинг і менеджмент, 2025. URL: https://www.researchgate.net/publication/370627770_Using_Crm_Systems_for_the_Development_and_Implementation_of_Communication_Strategies_for_Digital_Brand_Management_and_Internet_Marketing_eu_Experience (дата звернення: 14.06.2025) researchgate.net
7. OpenAI. ChatGPT. ChatGPT. URL: <https://chatgpt.com> (дата звернення: 15.06.2025).
8. GeeksforGeeks. Customer Relationship Management (CRM) System with Node.js and Express.js. GeeksforGeeks. URL: <https://www.geeksforgeeks.org/customer-relationship-management-crm-system-with-node-js-and-express-js/> (дата звернення: 19.06.2025) geeksforgeeks.org

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				59
Змн.	Арк.	№ докум.	Підпис	Дата		

9. Dynamic Pixels. Full Stack Development with React, Node.js, and MongoDB. Programmers.io Blog. URL: <https://programmers.io/blog/fullstack-development-react-nodejs-mongodb/> (дата звернення: 13.06.2025) programmers.io
10. Express.js. Express – мінімальний і гнучкий фреймворк для Node.js. Express. URL: <https://expressjs.com/> (дата звернення: 12.04.2025) expressjs.com
11. Axios. Interceptors | Axios Docs. Axios. URL: <https://axios-http.com/docs/interceptors> (дата звернення: 13.05.2025) axios-http.com
12. MongoDB, Inc. A Comprehensive Guide to Data Modeling. MongoDB. URL: <https://www.mongodb.com/resources/basics/databases/data-modeling> (дата звернення: 19.06.2025) mongodb.com
13. SanjayTTG. JWT Authentication in React with react-router. DEV Community. URL: <https://dev.to/sanjayttg/jwt-authentication-in-react-with-react-router-1d03> (дата звернення: 12.06.2025) dev.to
14. React Router. Declarative routing for React. React Router. URL: <https://reactrouter.com/> (дата звернення: 19.06.2025) reactrouter.com
15. React. React – бібліотека для побудови користувацьких інтерфейсів. React. URL: <https://react.dev/> (дата звернення: 11.04.2025) react.dev
16. Eleken. How to Design a CRM System: All You Need to Know about Custom CRM. Eleken Blog. URL: <https://www.eleken.co/blog-posts/how-to-design-a-crm-system-all-you-need-to-know-about-custom-crm> (дата звернення: 25.05.2025) eleken.co
17. Slashdev.io. How To Build A Custom CRM System In NodeJS In 2024. Slashdev.io. URL: <https://slashdev.io/-how-to-build-a-custom-crm-system-in-nodejs-in-2024> (дата звернення: 29.05.2025) slashdev.io
18. MDN Web Docs. Introduction – Express/Node.js. MDN. URL: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Express_Nodejs/Introduction (дата звернення: 19.06.2025) developer.mozilla.org

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				60
Змн.	Арк.	№ докум.	Підпис	Дата		

19. LogRocket Blog. React Context tutorial: Complete guide with practical examples. LogRocket. URL: <https://blog.logrocket.com/react-context-tutorial/> (дата звернення: 19.06.2025) blog.logrocket.com

20. Solace. How To Use REST APIs With Microservices. Solace. URL: <https://solace.com/resources/developer/how-to-use-rest-apis-with-microservices-video> (дата звернення: 19.06.2025)

(не на всю літературу розставлено посилання в тексті, виправити)

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				61
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Лістинг коду App.js:

```

import React, { useState, useEffect } from "react";
import { BrowserRouter as Router, Routes, Route, Navigate } from "react-router-dom";
import Header from "../components/Header";
import Footer from "../components/Footer";
import CartProvider from "../context/CartContext";
import Home from "../pages/Home";
import Catalog from "../pages/Catalog";
import VinylDetails from "../pages/VinylDetails";
import CartPage from "../pages/CartPage";
import Checkout from "../pages/Checkout";
import Profile from "../pages/Profile";
import Login from "../pages/Login";
import Register from "../pages/Register";
import api from "../services/api";
const App = () => {
  const [user, setUser] = useState(null);
  useEffect(() => {
    const stored = localStorage.getItem('user');
    if (stored) {
      const userData = JSON.parse(stored);
      setUser(userData);
      api.defaults.headers.common['Authorization'] = `Bearer ${userData.token}`;
    } else {
      setUser(null);
      delete api.defaults.headers.common['Authorization'];
    }
  }, []);
  const handleLogin = (userData) => {
    setUser(userData);
    localStorage.setItem('user', JSON.stringify(userData));
    api.defaults.headers.common['Authorization'] = `Bearer ${userData.token}`;
  };
  const handleLogout = () => {
    setUser(null);
    localStorage.removeItem('user');
    delete api.defaults.headers.common['Authorization'];
    window.location.reload();
  };
  return (
    <CartProvider>
    <Router>
    <Header user={user} logout={handleLogout} />
    <main className="main-content">
    <Routes>
    { /* Доступно лише якщо не залогінені */ }
    <Route
    path="/login"
    element={

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		63

```

user
? <Navigate to="/profile" replace />
: <Login onLogin={handleLogin} />
}
/>
<Route
path="/register"
element={
user
? <Navigate to="/profile" replace />
: <Register />
}
/>
{/* Публічні сторінки */}
<Route path="/" element={<Home />} />
<Route path="/catalog" element={<Catalog />} />
<Route path="/vinyl/:id" element={<VinylDetails />} />
<Route path="/cart" element={<CartPage />} />
<Route path="/checkout" element={<Checkout />} />
{/* Профіль — тільки для залогінених */}
<Route
path="/profile"
element={
user
? <Profile user={user} />
: <Navigate to="/login" replace />
}
/>
{/* Все інше — на головну */}
<Route path="*" element={<Navigate to="/" replace />} />
</Routes>
</main>
<Footer />
</Router>
</CartProvider>
);
};
export default App;

```

Лістинг коду: Catalog.js:

```

import React, { useEffect, useState, useContext } from 'react';
import { Link } from 'react-router-dom';
import api from '../services/api';
import { CartContext } from '../context/CartContext';
const Catalog = () => {
const [vinyls, setVinyls] = useState([]);
const [filtered, setFiltered] = useState([]);
const [search, setSearch] = useState("");
const [authorSearch, setAuthorSearch] = useState("");
const [genre, setGenre] = useState("");
const { addToCart } = useContext(CartContext);

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				64
Змн.	Арк.	№ докум.	Підпис	Дата		


```

const API_ROOT = api.defaults.baseURL.replace(/\/api\/?$/, "");
useEffect(() => {
  api.get('/vinyls')
    .then(res => {
      setVinyls(res.data);
      setFiltered(res.data);
    })
    .catch(console.error);
}, []);
useEffect(() => {
  let data = [...vinyls];
  if (search) data = data.filter(v => v.title.toLowerCase().includes(search.toLowerCase()));
  if (authorSearch) data = data.filter(v =>
    v.artist.toLowerCase().includes(authorSearch.toLowerCase()));
  if (genre) data = data.filter(v => v.genre === genre);
  setFiltered(data);
}, [search, authorSearch, genre, vinyls]);
const genres = Array.from(new Set(vinyls.map(v => v.genre))).filter(g => g);
return (
  <div className="catalog-container">
    <h2>Каталог платівок</h2>
    <div className="filters simple-filters">
      <input type="text" placeholder="🔍 Шукати за назвою" value={search} onChange={e =>
        setSearch(e.target.value)} />
      <input type="text" placeholder="🖋 Шукати за виконавцем" value={authorSearch}
        onChange={e => setAuthorSearch(e.target.value)} />
      <select value={genre} onChange={e => setGenre(e.target.value)}>
        <option value="">📁 Усі жанри</option>
        {genres.map(g => <option key={g} value={g}>{g}</option>)}
      </select>
    </div>
    <div className="catalog-grid">
      {filtered.map(v => (
        <div key={v._id} className="vinyl-card simple-card">
          <Link to={`/vinyl/${v._id}`}>
            <img src={v.cover ? `${API_ROOT}/uploads/${v.cover}` : '/default-cover.png'} alt={v.title} />
            <div className="card-info">
              <h3>{v.title}</h3>
              <p className="artist">{v.artist}</p>
              <p className="price">{v.price} грн</p>
            </div>
          </Link>
          <button onClick={() => addToCart(v)} className="add-btn">+ Кошик</button>
        </div>
      ))}
    </div>
  </div>
);
};
export default Catalog;

```

Лістинг коду: Checkout.js:

```
import React, { useEffect, useState } from 'react';
import api from '../services/api';
const Checkout = () => {
  const [user, setUser] = useState(null);
  const [cartItems, setCartItems] = useState([]);
  const [vinylMap, setVinylMap] = useState({});
  const [form, setForm] = useState({
    name: "",
    phone: "",
    email: "",
    address: "",
    paymentMethod: 'cash' // cash = при отриманні, bank = по реквізітам
  });
  const [fileLinks, setFileLinks] = useState([]);
  const [fileInput, setFileInput] = useState("");
  useEffect(() => {
    const storedUser = JSON.parse(localStorage.getItem('user'));
    if (storedUser && storedUser._id) {
      setUser(storedUser);
      api.get(`/users/${storedUser._id}`).then(async res => {
        const u = res.data;
        setForm({
          name: u.name || "",
          phone: u.phone || "",
          email: u.email || "",
          address: u.address || "",
          paymentMethod: 'cash'
        });
        const cart = u.cart || [];
        setCartItems(cart);
        await loadVinylDetails(cart);
      });
    } else {
      const localCart = JSON.parse(localStorage.getItem('cart')) || [];
      setCartItems(localCart);
      loadVinylDetails(localCart);
    }
  }, []);
  const extractId = (item) => {
    if (item._id && typeof item._id === 'object' && item._id._id) {
      return item._id._id.toString();
    }
    if (item._id) {
      return item._id.toString();
    }
    if (item.productId) {
      return item.productId.toString();
    }
    return null;
  };
};
```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		66

```

const loadVinylDetails = async (cart) => {
const ids = Array.from(new Set(cart.map(extractId).filter(Boolean)));
if (!ids.length) return;
try {
const res = await api.post('/vinyls/by-ids', { ids });
const map = { };
res.data.forEach(v => {
map[v._id?.toString()] = v;
map[v.id] = v;
});
setVinylMap(map);
} catch (err) {
console.error('Помилка завантаження деталей платівок', err);
}
};

const handleChange = e => {
setForm({ ...form, [e.target.name]: e.target.value });
};

const handleAddFileLink = () => {
if (fileInput.trim()) {
setFileLinks([...fileLinks, fileInput.trim()]);
setFileInput("");
}
};

const handleSubmit = async e => {
e.preventDefault();
try {
await api.post('/orders', {
vinylID: cartItems.map(i => ({
id: extractId(i),
count: getQuantity(i)
})),
userId: user?._id || 'анонім',
shipment: { address: form.address },
payment: {
method: form.paymentMethod
},
files: fileLinks.map(link => ({
name: link.split('/').pop(),
url: link
}))
});
alert("Замовлення оформлено!");
localStorage.removeItem('cart');
setCartItems([]);
setFileLinks([]);
} catch (err) {
console.error('Помилка при оформленні замовлення', err);
if (err.response) {
alert(`Не вдалося оформити замовлення: ${err.response.data.message || 'Невідома помилка'}`);
} else {

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		67

```

alert('Не вдалося оформити замовлення: немає відповіді від сервера');
}
}
};
const getQuantity = (item) => item.quantity || item.count || 1;
return (
<div className="checkout-page">
<div className="checkout-cart">
<h3>Ваші платівки</h3>
{cartItems.length === 0 ? (
<p>Корзина порожня</p>
) : (
<ul>
{cartItems.map((item, idx) => {
const id = extractId(item);
const v = vinylMap[id] || {};
const imgSrc = v.cover
? `${api.defaults.baseURL.replace(/\/api\/?$/, "")}/uploads/${v.cover}`
: '/default-cover.png';
return (
<li key={id + '-' + idx} className="checkout-item">
<img src={imgSrc} alt={v.title || 'Платівка'} style={{ width: 40, height: 40, objectFit: 'cover' }} />
<div>
<div><strong>{v.title || 'Платівка'}</strong></div>
<div>{v.artist || ""}</div>
<div>{getQuantity(item)} шт.</div>
</div>
</li>
);
}}
</ul>
)}
</div>
<div className="checkout-details">
<h3>Деталі замовлення</h3>
<form onSubmit={handleSubmit}>
<label>
Ім'я:
<input name="name" value={form.name} onChange={handleChange} required />
</label>
<label>
Телефон:
<input name="phone" value={form.phone} onChange={handleChange} required />
</label>
<label>
Email:
<input name="email" value={form.email} onChange={handleChange} required type="email" />
</label>
<label>
Адреса:
<input name="address" value={form.address} onChange={handleChange} required />

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		68

```

</label>
<label>
Метод оплати:
<select name="paymentMethod" value={ form.paymentMethod } onChange={ handleChange }>
<option value="cash">При отриманні</option>
<option value="bank">По реквізітам</option>
</select>
</label>
<label>
Посилання на файл (квитанція, скріншот):
<input
type="text"
value={ fileInput }
onChange={ e => setFileInput(e.target.value) }
placeholder="Вставте URL файлу"
/>
<button type="button" onClick={ handleAddFileLink }>Додати файл</button>
</label>
{ fileLinks.length > 0 && (
<ul>
{ fileLinks.map((f, i) => (
<li key={ i }><a href={ f } target="_blank" rel="noopener noreferrer">{ f }</a></li>
))}
</ul>
)}
<button type="submit">Оформити замовлення</button>
</form>
</div>
</div>
);
};
export default Checkout;

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		69

Лістинг коду: : App.js

```

import React, { useState, useEffect } from 'react';
import { BrowserRouter as Router, Routes, Route, Link, Navigate } from 'react-router-dom';
import Dashboard from './pages/Dashboard';
import Orders from './pages/Orders';
import Vinyls from './pages/Vinyls';
import Users from './pages/Users';
import AdminPanel from './pages/AdminPanel';
import ModerateReviews from './pages/ModerateReviews';
import Login from './pages/Login';
const App = () => {
  const [user, setUser] = useState(null);
  useEffect(() => {
    const storedUser = localStorage.getItem('user');
    if (storedUser) {
      setUser(JSON.parse(storedUser));
    }
  }, []);
  const handleLogin = (userData) => {
    setUser(userData);
    localStorage.setItem('user', JSON.stringify(userData));
  };
  const handleLogout = () => {
    setUser(null);
    localStorage.removeItem('user');
  };
  // Витягуємо role для зручності:
  const userRole = user?.user?.role || user?.role || null;
  return (
    <Router>
      {user ? (
        <>
          <header className="app-header">
            <nav className="app-nav">
              <Link to="/">Дашборд</Link>
              <Link to="/orders">Замовлення</Link>
              <Link to="/vinyls">Платівки</Link>
              <Link to="/users">Користувачі</Link>
              {/* Дозволяємо модерування admin та manager */}
              {(userRole === 'admin' || userRole === 'manager') && (
                <Link to="/moderate">Коментарі</Link>
              )}
              {userRole === 'admin' && <Link to="/admin">Адмін панель</Link>}
            <button onClick={handleLogout} className="logout-button">Вийти</button>
          </nav>
        </header>
        <main className="app-main">
          <Routes>
            <Route path="/" element={<Dashboard />} />
            <Route path="/orders" element={<Orders />} />

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				70
Змн.	Арк.	№ докум.	Підпис	Дата		

```

<Route path="/vinyls" element={<Vinyls />} />
<Route path="/users" element={<Users />} />
{/* ===== Сторінка модерування ===== */}
<Route
  path="/moderate"
  element={
    (userRole === 'admin' || userRole === 'manager')
    ? <ModerateReviews />
    : <Navigate to="/" />
  }
/>
{/* ===== Адмін панель ===== */}
<Route path="/admin" element={userRole === 'admin' ? <AdminPanel /> : <Navigate to="/" />}
/>
<Route path="*" element={<Navigate to="/" />} />
</Routes>
</main>
</>
) : (
  <Routes>
    <Route path="*" element={<Login onLogin={handleLogin} />} />
  </Routes>
)
</Router>
);
};
export default App;

```

Лістинг коду: Dashboard.js

```

import React, { useEffect, useState } from 'react';
import api from '../services/api';
import {
  ResponsiveContainer,
  BarChart, Bar, XAxis, YAxis, Tooltip, CartesianGrid,
  PieChart, Pie, Cell, Legend,
  LineChart, Line
} from 'recharts';
const COLORS = ['#8884d8', '#82ca9d', '#ffc658', '#d0ed57', '#a4de6c'];
const DATA_SOURCES = [
  { label: 'Замовлення', value: 'orders' },
  { label: 'Платівки', value: 'vinyls' },
  { label: 'Користувачі', value: 'users' }
];
const CHART_TYPES = [
  { label: 'Стовпчиковий', value: 'bar' },
  { label: 'Кругова діаграма', value: 'pie' },
  { label: 'Лінійний', value: 'line' }
];
const ALL_STATUSES = ['НОВЕ', 'В ОБРОБЦІ', 'ВІДПРАВЛЕНО', 'ВИКОНАНО', 'ВІДХИЛЕНО'];

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				71
Змн.	Арк.	№ докум.	Підпис	Дата		

```

function median(values) {
  if (!values.length) return 0;
  const s = [...values].sort((a, b) => a - b);
  const mid = Math.floor(s.length / 2);
  return s.length % 2 === 0 ? ((s[mid - 1] + s[mid]) / 2).toFixed(2) : s[mid].toFixed(2);
}

// Генерує ключ-місяць для агрегації (YYYY-MM)
function getMonthKey(date) {
  const d = new Date(date);
  if (isNaN(d)) return 'невідомо';
  return `${d.getFullYear()}-${String(d.getMonth() + 1).padStart(2, '0')}`;
}

// Динаміка продажів та замовлень по місяцях
function getMonthlyStats(orders, type = 'amount') {
  const map = {};
  orders.forEach(o => {
    if (!o.statusLog || !o.statusLog.length) return;
    const d = o.statusLog.slice(-1)[0].date;
    const month = getMonthKey(d);
    if (!map[month]) map[month] = { name: month, amount: 0, count: 0 };
    if (type === 'amount') map[month].amount += (o.payment?.amount || 0);
    map[month].count += 1;
  });
  return Object.values(map).sort((a, b) => a.name.localeCompare(b.name));
}

// Динаміка нових користувачів по місяцях
function getUserMonthlyStats(users) {
  const map = {};
  users.forEach(u => {
    if (!u.createdAt) return;
    const month = getMonthKey(u.createdAt);
    if (!map[month]) map[month] = { name: month, count: 0 };
    map[month].count += 1;
  });
  return Object.values(map).sort((a, b) => a.name.localeCompare(b.name));
}

const Dashboard = () => {
  const [orderStats, setOrderStats] = useState({});
  const [salesStats, setSalesStats] = useState({});
  const [vinylSalesStats, setVinylSalesStats] = useState({});
  const [clientStats, setClientStats] = useState({});
  const [vinylCatalogStats, setVinylCatalogStats] = useState({});
  const [userStats, setUserStats] = useState({});
  const [rawData, setRawData] = useState({ orders: [], vinyls: [], users: [] });
  const [analytics, setAnalytics] = useState([
    {
      id: Date.now(),
      source: 'orders', metric: '', groupBy: '', chartType: 'bar',
      startDate: '', endDate: ''
    }
  ]);
  useEffect(() => {
    Promise.all([
      api.get('/orders/with-vinyls'),

```



```

api.get('/vinyls'),
api.get('/users')
])
.then(([o, v, u]) => {
  setRawData({ orders: o.data, vinyls: v.data, users: u.data });
  buildDefault(o.data, v.data, u.data);
})
.catch(console.error);
}, []);
function buildDefault(orders, vinyls, users) {
  const valid = orders.filter(o => (o.status || "").toUpperCase() !== 'ВІДХИЛЕНО');
  const totalOrders = orders.length;
  const ordersByStatus = { };
  ALL_STATUSES.forEach(st => ordersByStatus[st] = 0);
  orders.forEach(o => {
    const st = (o.status || "").toUpperCase();
    if (ALL_STATUSES.includes(st)) {
      ordersByStatus[st]++;
    }
  });
  const totalSales = valid.reduce((s, o) => s + (o.payment?.amount || 0), 0);
  const avgOrderValue = valid.length ? (totalSales / valid.length).toFixed(2) : 0;
  const medianOrderValue = median(valid.map(o => o.payment?.amount || 0));
  const countMap = { };
  valid.forEach(o => o.vinylInfos.forEach(v => {
    const key = `${v.title} — ${v.artist}`;
    countMap[key] = (countMap[key] || 0) + v.count;
  }));
  const topVinyls = Object.entries(countMap)
    .sort((a, b) => b[1] - a[1])
    .slice(0, 5);
  const vinylChart = topVinyls.map(([name, quantity]) => ({ name, quantity }));
  const clientMap = { };
  valid.forEach(o => {
    const uid = o.userId, name = o.userInfo?.name || '—';
    if (!clientMap[uid]) clientMap[uid] = { userId: uid, name, total: 0, orders: 0 };
    clientMap[uid].total += (o.payment?.amount || 0);
    clientMap[uid].orders += 1;
  });
  const topClients = Object.values(clientMap)
    .sort((a, b) => b.total - a.total)
    .slice(0, 20);
  const byGenre = vinyls.reduce((a, v) => {
    const g = v.genre || '—';
    a[g] = (a[g] || 0) + 1;
    return a;
  }, { });
  const totalVinyls = vinyls.length;
  // Додатково: користувачі по ролях, та "нові" за останній місяць (якщо є createdAt)
  const rolesMap = { };
  users.forEach(u => {
    const r = u.role || 'user';

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				
Змн.	Арк.	№ докум.	Підпис	Дата		73

```

rolesMap[r] = (rolesMap[r] || 0) + 1;
});
const lastMonth = () => {
const now = new Date();
return users.filter(u => {
const d = u.createdAt ? new Date(u.createdAt) : null;
return d && d > new Date(now.getFullYear(), now.getMonth() - 1, now.getDate());
}).length;
})();
setOrderStats({ totalOrders, ordersByStatus, avgOrderValue, medianOrderValue });
setSalesStats({ totalSales });
setVinylSalesStats({ topVinyls, vinylChart });
setClientStats({ topClients });
setVinylCatalogStats({ totalVinyls, byGenre });
setUserStats({ rolesMap, lastMonth });
}
const addPanel = () => setAnalytics(a => [
...a,
{ id: Date.now(), source: 'orders', metric: '', groupBy: '', chartType: 'bar', startDate: '', endDate: ''
}
]);
const removePanel = id => setAnalytics(a => a.filter(p => p.id !== id));
const updatePanel = (id, field, value) =>
setAnalytics(a => a.map(p => p.id === id ? { ...p, [field]: value } : p));
// === ОНОВЛЕНА логіка makeData (універсальна) ===
const makeData = (p) => {
let data = rawData[p.source] || [];
// --- Фільтр по даті ---
if ((p.source === 'orders' || p.source === 'users' || p.source === 'vinyls') && p.startDate &&
p.endDate) {
data = data.filter(item => {
let d = null;
if (p.source === 'orders') {
d = new Date(item.statusLog?.slice(-1)[0]?.date);
} else if (item.createdAt) {
d = new Date(item.createdAt);
}
return d && d >= new Date(p.startDate) && d <= new Date(p.endDate);
});
}
// --- Агрегація ---
const map = {};
data.forEach(item => {
// Групування
let key = 'Всі';
if (p.groupBy) {
if (p.groupBy === 'month' && p.source === 'orders') {
key = getMonthKey(item.statusLog?.slice(-1)[0]?.date || new Date());
} else if (p.source === 'orders' && p.groupBy === 'userId' && item.userInfo) {
key = item.userInfo.name || item.userId || '—';
} else {
key = item[p.groupBy] || '—';
}
}
}

```

```

    }
    }
    // Метрика
    let val = 1;
    if (p.metric) {
    if (p.metric === 'average' && p.source === 'orders') {
    // Тут буде розрахунок середнього чеку — зробимо після заповнення map
    } else {
    val = (p.metric.split('.').reduce((o, f) => o?.[f], item) || 0);
    }
    }
    map[key] = (map[key] || 0) + Number(val);
    });
    // Середній чек — особливий випадок
    if (p.metric === 'average' && p.source === 'orders') {
    // Групуємо масиви сум для кожної групи, потім рахуємо середнє
    const arrMap = { };
    data.forEach(item => {
    let key = 'Bci';
    if (p.groupBy) {
    if (p.groupBy === 'month' && p.source === 'orders') {
    key = getMonthKey(item.statusLog?.slice(-1)[0]?.date || new Date());
    } else if (p.groupBy === 'userId' && item.userInfo) {
    key = item.userInfo.name || item.userId || '—';
    } else {
    key = item[p.groupBy] || '—';
    }
    }
    const val = item.payment?.amount || 0;
    if (!arrMap[key]) arrMap[key] = [];
    arrMap[key].push(val);
    });
    return Object.entries(arrMap).map(([name, arr]) => ({
    name,
    value: arr.length ? (arr.reduce((s, v) => s + v, 0) / arr.length).toFixed(2) : 0
    }));
    }
    return Object.entries(map).map(([name, value]) => ({ name, value }));
    };
    return (
    <div className="dashboard">
    <h2>CRM Дашборд</h2>
    <div className="dashboard-grid">
    <div className="card">
    <h3>📊 Продажі</h3>
    <p>Загальна сума: <strong>{salesStats.totalSales} грн</strong></p>
    <p>Середній чек: <strong>{orderStats.avgOrderValue} грн</strong></p>
    <p>Медіанний чек: <strong>{orderStats.medianOrderValue} грн</strong></p>
    </div>
    <div className="card">
    <h3>📅 Замовлення</h3>




```

```

<p>Усього: <strong>{orderStats.totalOrders}</strong></p>
<ul>
{ALL_STATUSES.map(st => (
<li key={st}>{st}: {orderStats.ordersByStatus?.[st] || 0}</li>
))}
</ul>
</div>
<div className="card">
<h3>🔥 Топ-5 платівок</h3>
<ul>
{vinylSalesStats.topVinyls?.map(([name, qty]) => (
<li key={name}>{name}: {qty} шт.</li>
))}
</ul>
<div className="chart-wrapper">
<ResponsiveContainer width="100%" height={150}>
<BarChart data={vinylSalesStats.vinylChart || []}>
<CartesianGrid strokeDasharray="3 3" />
<XAxis dataKey="name" />
<YAxis />
<Tooltip />
<Bar dataKey="quantity" fill={COLORS[0]} />
</BarChart>
</ResponsiveContainer>
</div>
</div>
<div className="card">
<h3>🏆 Топ-20 клієнтів</h3>
<ul>
{clientStats.topClients?.map(c => (
<li key={c.userId}>
{c.name}: {c.total} грн ({c.orders})
</li>
))}
</ul>
</div>
<div className="card">
<h3>📀 Каталог платівок</h3>
<p>Усього у каталозі: <strong>{vinylCatalogStats.totalVinyls}</strong></p>
<div className="chart-wrapper">
<ResponsiveContainer width="100%" height={150}>
<PieChart>
<Pie
data={Object.entries(vinylCatalogStats.byGenre || {}).map(([name, v]) => ({ name, value: v })))}
dataKey="value" nameKey="name" outerRadius={60} label
>
{Object.keys(vinylCatalogStats.byGenre || {}).map((_, i) => (
<Cell key={i} fill={COLORS[i % COLORS.length]} />
))}
</Pie>
<Tooltip />

```

```

</PieChart>
</ResponsiveContainer>
</div>
</div>
{ /* Динаміка продажів по місяцях */ }
<div className="card">
<h3> Динаміка продажів (місяці)</h3>
<div className="chart-wrapper">
<ResponsiveContainer width="100%" height={ 160 }>
<LineChart data={ getMonthlyStats(rawData.orders, 'amount') }>
<CartesianGrid strokeDasharray="3 3" />
<XAxis dataKey="name" />
<YAxis />
<Tooltip />
<Line type="monotone" dataKey="amount" stroke={ COLORS[1] } />
</LineChart>
</ResponsiveContainer>
</div>
</div>
{ /* Динаміка замовлень по місяцях */ }
<div className="card">
<h3> Динаміка замовлень (місяці)</h3>
<div className="chart-wrapper">
<ResponsiveContainer width="100%" height={ 160 }>
<BarChart data={ getMonthlyStats(rawData.orders, 'count') }>
<CartesianGrid strokeDasharray="3 3" />
<XAxis dataKey="name" />
<YAxis />
<Tooltip />
<Bar dataKey="count" fill={ COLORS[0] } />
</BarChart>
</ResponsiveContainer>
</div>
</div>
{ /* Динаміка нових користувачів */ }
<div className="card">
<h3> Нові користувачі (місяці)</h3>
<div className="chart-wrapper">
<ResponsiveContainer width="100%" height={ 160 }>
<LineChart data={ getUserMonthlyStats(rawData.users) }>
<CartesianGrid strokeDasharray="3 3" />
<XAxis dataKey="name" />
<YAxis />
<Tooltip />
<Line type="monotone" dataKey="count" stroke={ COLORS[2] } />
</LineChart>
</ResponsiveContainer>
</div>
</div>
</div>
{ /* === Аналітичний конструктор === */ }

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				77
Змн.	Арк.	№ докум.	Підпис	Дата		

```

<div className="dashboard-analytics">
  <div className="analytics-header">
    <h3>Аналітичний конструктор</h3>
    <button className="add-analytics" onClick={addPanel}>+</button>
  </div>
  <div className="analytics-grid">
    {analytics.map(p => {
      const data = makeData(p);
      return (
        <div key={p.id} className="analytics-panel card">
          <div className="panel-header">
            <strong>Аналітика</strong>
            <button className="remove-analytics" onClick={() => removePanel(p.id)}>X</button>
          </div>
          <div className="panel-controls">
            <select value={p.source} onChange={e => updatePanel(p.id, 'source', e.target.value)}>
              {DATA_SOURCES.map(ds => (
                <option key={ds.value} value={ds.value}>{ds.label}</option>
              ))}
            </select>
            <select value={p.metric} onChange={e => updatePanel(p.id, 'metric', e.target.value)}>
              <option value="">Кількість записів</option>
              {p.source === 'orders' && <option value="payment.amount">Сума оплати</option>}
              {p.source === 'orders' && <option value="average">Середній чек</option>}
              {p.source === 'vinyls' && <option value="price">Ціна</option>}
            </select>
            <select value={p.groupBy} onChange={e => updatePanel(p.id, 'groupBy', e.target.value)}>
              <option value="">— без групування —</option>
              {p.source === 'orders' && <
                <option value="status">Статус</option>
                <option value="userId">Клієнт</option>
                <option value="month">Місяць</option>
              </>}
              {p.source === 'vinyls' && <option value="genre">Жанр</option>}
              {p.source === 'users' && <option value="role">Роль</option>}
            </select>
            <select value={p.chartType} onChange={e => updatePanel(p.id, 'chartType', e.target.value)}>
              {CHART_TYPES.map(ct => (
                <option key={ct.value} value={ct.value}>{ct.label}</option>
              ))}
            </select>
            <div className="date-range">
              <input type="date" value={p.startDate} onChange={e => updatePanel(p.id, 'startDate',
                e.target.value)} />
              <span>—</span>
              <input type="date" value={p.endDate} onChange={e => updatePanel(p.id, 'endDate',
                e.target.value)} />
            </div>
          </div>
          <div className="chart-wrapper">
            {data.length === 0 ? (

```

```

<div className="dashboard-empty">Даних для побудови аналітики не знайдено</div>
) : p.chartType === 'bar' ? (
  <ResponsiveContainer width="100%" height="100%">
    <BarChart data={data}>
      <CartesianGrid strokeDasharray="3 3" />
      <XAxis dataKey="name" />
      <YAxis allowDecimals={false} />
      <Tooltip />
      <Bar dataKey="value" fill={COLORS[0]} />
    </BarChart>
  </ResponsiveContainer>
) : p.chartType === 'pie' ? (
  <ResponsiveContainer width="100%" height="100%">
    <PieChart>
      <Pie data={data} dataKey="value" nameKey="name" outerRadius={60} label>
        {data.map((_, i) => (
          <Cell key={i} fill={COLORS[i % COLORS.length]} />
        ))}
      </Pie>
      <Tooltip />
      <Legend verticalAlign="bottom" />
    </PieChart>
  </ResponsiveContainer>
) : (
  <ResponsiveContainer width="100%" height="100%">
    <LineChart data={data}>
      <CartesianGrid strokeDasharray="3 3" />
      <XAxis dataKey="name" />
      <YAxis />
      <Tooltip />
      <Line type="monotone" dataKey="value" stroke={COLORS[1]} />
    </LineChart>
  </ResponsiveContainer>
)
  </div>
</div>
);
}}}
</div>
</div>
</div>
);
};
export default Dashboard;

```

Лістинг коду: Server.js:

```
// server.js
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const path = require('path');
// --- Routes ---
const usersRouter = require('./routes/users');
const vinylsRouter = require('./routes/vinyls');
const ordersRouter = require('./routes/orders');
const reviewsRouter = require('./routes/reviews');
const uploadsRouter = require('./routes/uploads');
const app = express();
app.use(cors());
app.use(express.json());
// Serve uploads publicly
app.use('/uploads', express.static(path.join(__dirname, 'uploads')));
// API routes
app.use('/api/users', usersRouter);
app.use('/api/vinyls', vinylsRouter);
app.use('/api/orders', ordersRouter);
app.use('/api/reviews', reviewsRouter);
app.use('/api/uploads', uploadsRouter);
// Not found for API
app.use((req, res, next) => {
  if (req.path.startsWith('/api/')) {
    return res.status(404).json({ message: 'API route not found' });
  }
  next();
});
// Global error handler
app.use((err, req, res, next) => {
  console.error('✖ Unhandled error:', err);
  res.status(500).json({ message: 'Internal server error' });
});
mongoose
  .connect('mongodb://127.0.0.1:27017/diploma', {
    useNewUrlParser: true,
    useUnifiedTopology: true
  })
  .then(() => console.log('✔ Connected to MongoDB'))
  .catch(err => console.error('✖ MongoDB connection error:', err));
const PORT = process.env.PORT || 3001;
app.listen(PORT, () => {
  console.log('🚀 Server listening on port ${PORT}');
});
```

Лістинг коду: Роут Upload.js

```
const express = require('express');
```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				80
Змн.	Арк.	№ докум.	Підпис	Дата		


```

const multer = require('multer');
const path = require('path');
const router = express.Router();
const storage = multer.diskStorage({
  destination: (req, file, cb) => cb(null, 'uploads/'),
  filename: (req, file, cb) => cb(null, Date.now() + '-' + file.originalname)
});
const upload = multer({ storage });
router.post('/:type', upload.single('file'), (req, res) => {
  const { type } = req.params;
  if (!['vinyl', 'user'].includes(type)) {
    return res.status(400).json({ message: 'Непідтримуваний тип' });
  }
  res.json({ filePath: `uploads/${req.file.filename}`, type });
});
module.exports = router;

```

Лістинг коду: Роут Reviews.js:

```

const express = require('express');
const router = express.Router();
const mongoose = require('mongoose');
const reviewSchema = new mongoose.Schema({
  vinylId: { type: mongoose.Schema.Types.ObjectId, ref: 'Vinyl', required: true },
  userId: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  rating: { type: Number, min: 1, max: 5, required: true },
  text: { type: String, required: true },
  date: { type: Date, default: Date.now }
});
const Review = mongoose.models.Review || mongoose.model('Review', reviewSchema, 'reviews');
// Отримати коментарі
router.get('/', async (req, res) => {
  try {
    const filter = {};
    if (req.query.vinylId) {
      filter.vinylId = req.query.vinylId;
    }
    const reviews = await Review.find(filter)
      .populate('userId', 'name')
      .populate('vinylId', 'title artist');
    res.json(reviews);
  } catch (err) {
    console.error('Помилка отримання відгуків:', err);
    res.status(500).json({ message: 'Серверна помилка' });
  }
});
// Додати коментар
router.post('/', async (req, res) => {
  try {
    const { vinylId, rating, text, userId } = req.body;
    if (!vinylId || !rating || !text || !userId) {

```

		М.Ю. Дешков			ІСТ.КР.Б – 126 – 25 – ПЗ	Арк.
		М.С. Граф				81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

return res.status(400).json({ message: 'vinylId, rating, text, userId обов'язкові' });
}
const newReview = new Review({
  vinylId,
  userId,
  rating,
  text
});
await newReview.save();
const populatedReview = await Review.findById(newReview._id)
  .populate('userId', 'name')
  .populate('vinylId', 'title artist');
res.json(populatedReview);
} catch (err) {
  console.error('Помилка додавання відгуку:', err);
  res.status(500).json({ message: 'Серверна помилка' });
}
});
// Видалити коментар
router.delete('/:id', async (req, res) => {
  try {
    if (!mongoose.Types.ObjectId.isValid(req.params.id)) return res.status(400).send('Invalid ID');
    const result = await Review.findByIdAndDelete(req.params.id);
    if (result) res.send('Видалено');
    else res.status(404).send('Не знайдено');
  } catch (err) {
    res.status(500).send('Серверна помилка');
  }
});
module.exports = router;

```