МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю. М. Потебні
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

КАФЕДРА ЕЛЕКТРОНІКИ, ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Кваліфікаційна робота

перший (бакалаврський)
(рівень вищої освіти)

на тему **Розробка веб-додатку для вибору та продажу домашніх тварин**

Виконав: студент 4 курсу, групи 6.1212-пзс
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Програмне забезпечення систем
(код і назва освітньої програми)
В.І. Богуславський
(ініціали та прізвище)
Керівник доцент, А.І. Безверхий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент директор ТОВ «Дісітел»
П.О. Лютий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ім. Ю. М. Потебні ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

Кафедра електроніки, інформаційних систем та програмного забезпечення

Рівень вищої освіти перший (бакалаврський)

Спеціальність 121 Інженерія програмного забезпечення
(код та назва)

Освітня програма Програмне забезпечення систем
(код та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри Тетяна КРИТСЬКА

“ 01 ” березня 2026 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Богуславському Владиславу Ігоровичу
(прізвище, ім’я, по батькові)

1. Тема роботи Розробка веб-додатку для вибору та продажу домашніх тварин

керівник роботи Безверхий Анатолій Ігорович, доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджена наказом ЗНУ від 23.12.2025 р. № 1830-с

Строк подання студентом кваліфікаційної роботи 02.06.2026

2. Вихідні дані бакалаврської роботи
   - комплект нормативних документів ;
   - технічне завдання до роботи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

   - огляд та збір літератури стосовно теми кваліфікаційної роботи;;
   - огляд та аналіз існуючих веб-сервісів і програмних рішень-аналогів;
   - проєктування архітектури клієнт-серверної інформаційної системи;
   - реалізація клієнтської та серверної частини веб-додатку;
   - розробка висновків та пропозицій щодо подальшого розвитку системи;

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) 16 слайдів презентації

6. Консультанти розділів бакалаврської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата
		Завдання прийняв

7. Дата видачі завдання 01.03.2026

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів бакалаврської роботи	Примітка
1	Аналіз предметної області	02.03-01.04.2026	виконано
2	Формулювання основної задачі дипломної роботи та узгодження її з науковим керівником	03.04-08.04.2026	виконано
3	Аналіз існуючих аналогів	11.04-14.04.2026	виконано
4	Проєктування структури бази даних для зберігання даних	15.04-18.04.2026	виконано
5	Узгодження подальших дій з науковим керівником	19.04-20.04.2026	виконано
6	Реалізація серверної частини, REST API та автентифікації	21.04-22.04.2026	виконано
7	Реалізація каталогу оголошень, пошуку та фільтрів	22.04-24.04.2026	виконано
8	Реалізація особистого кабінету користувача	25.04-27.04.2026	виконано
9	Додавання фотографій, обраного та сторінки оголошення	29.04-01.05.2026	виконано
10	Реалізація чату, відгуків, скарг і модерації	02.05-06.05.2026	виконано
11	Розробка адміністративної панелі	07.05-09.05.2026	виконано
12	Доопрацювання адаптивності, службових сторінок і SEO	10.05-14.05.2026	виконано
13	Розгортання веб-додатку на хостингу	15.05-18.05.2026	виконано
14	Тестування та усунення додаткових помилок	19.05-23.05.2026	виконано
15	Створення презентації	24.05-26.05.2026	виконано
16	Оформлення кваліфікаційної роботи	27.05-29.05.2026	виконано

Студент _____ _____В.І. Богуславський

Керівник роботи _____ _____А. І. Безверхий

Нормоконтроль пройдено

Нормоконтролер _____ _____І.А. Скрипник

АНОТАЦІЯ

Сторінок — 66,

Рисунків — 9,

Таблиць — 13,

Джерел — 17.

Богуславський В. І. Розробка веб-додатку для вибору та продажу домашніх тварин: кваліфікаційна робота бакалавра спеціальності 121 «Інженерія програмного забезпечення» / наук. керівник А. І. Безверхий. Запоріжжя : ЗНУ, 2026. 66 с.

У кваліфікаційній роботі розглянуто проблему створення спеціалізованого веб-додатку для вибору, продажу та розміщення оголошень про домашніх тварин. Актуальність роботи зумовлена тим, що універсальні дошки оголошень не завжди забезпечують зручне подання параметрів тварини, пошук за важливими характеристиками та контроль якості контенту.

Метою роботи є розробка веб-додатку PetUA, який поєднує каталог оголошень, пошук і фільтрацію, особистий кабінет, обране, завантаження фотографій, внутрішні повідомлення, відгуки, скарги, модерацію та адміністрування. У процесі виконання роботи проаналізовано предметну область, розглянуто програмні продукти-аналоги, обґрунтовано вибір клієнт-серверної архітектури та сформульовано основні вимоги до системи.

Практичне значення результатів полягає у створенні працездатної основи тематичного веб-сервісу, який спрощує розміщення пропозицій, полегшує пошук домашньої тварини за характеристиками та покращує взаємодію між користувачами.

Ключові слова: *веб-додаток, домашні тварини, оголошення, каталог, React, Node.js, Express, PostgreSQL, Prisma, JWT, Socket.IO, модерація*.

ABSTRACT

Pages — 66,

Drawings — 9,

Tables — 13,

Sources — 17.

Bohuslavskyi V. I. Development of a web application for selecting and selling pets: bachelor qualification work in specialty 121 “Software Engineering” / scientific supervisor A. I. Bezverkhyi. Zaporizhzhia : ZNU, 2026. 66 p.

The qualification work considers the problem of creating a specialized web application for selecting, selling and publishing advertisements about pets. The relevance of the work is caused by the fact that universal classified advertising platforms do not always provide convenient representation of animal parameters, search by important characteristics and quality control of content.

The purpose of the work is to develop the PetUA web application, which combines an advertisement catalog, search and filtering, user profile, favorites, photo uploading, internal messages, reviews, complaints, moderation and administration. During the work, the subject area was analyzed, similar software products were reviewed, the choice of client-server architecture was justified and the main system requirements were formulated.

The practical value of the results lies in creating a working basis for a thematic web service that simplifies publishing offers, makes it easier to search for a pet by characteristics and improves interaction between users.

Keywords: *web application, pets, advertisements, catalog, React, Node.js, Express, PostgreSQL, Prisma, JWT, Socket.IO, moderation.*

ЗМІСТ

ВСТУП 7
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 13
1.1 Опис проблеми вибору та продажу домашніх тварин 13
1.2 Огляд літературних джерел і технологічних основ 15
1.3 Огляд існуючих рішень 17
1.4 Постановка задачі 22
1.5 Висновки до розділу 1 24
2 ДОСЛІДЖЕННЯ ЗАСОБІВ РЕАЛІЗАЦІЇ ТА ФОРМУВАННЯ ВИМОГ 25
2.1 Огляд існуючих рішень 25
2.2 Обґрунтування вибору технологій 30
2.3 Формування функціональних вимог 34
2.3.1 Вимоги до структури оголошення 34
2.4 Нефункціональні вимоги 36
2.5 Висновки до розділу 2 38
3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ 39
3.1 Загальна архітектура системи 39
3.2 Проєктування бази даних 42
3.3 Реалізація клієнтської частини 45
3.4 Реалізація серверної частини та API 49
3.5 Реалізація ролей, модерації та безпеки 51
3.6 Висновки до розділу3 53
4 ТЕСТУВАННЯ, РЕЗУЛЬТАТИ ТА ОЦІНКА ПРАЦЕЗДАТНОСТІ 54
4.1 Підхід до тестування 54
4.2 Перевірка клієнтської частини 55
4.3 Перевірка серверної частини та бази даних 58
4.4 Оцінка відповідності вимогам 60
4.4.1 Фізичні характеристики реалізації 60
4.4.2 Оцінка працездатності та обмежень 61
4.5 Висновки до розділу 4 63
ВИСНОВКИ 64
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 65

ВСТУП

Актуальність теми

Практика пошуку, вибору та розміщення оголошень про домашніх тварин є важливою складовою сучасної цифрової взаємодії між власниками тварин, продавцями, потенційними покупцями та користувачами, які шукають тварину для прийняття у сім'ю. Універсальні дошки оголошень дають змогу швидко опублікувати пропозицію, однак вони не завжди враховують специфіку предметної області. Для користувача, який обирає домашню тварину, важливими є не тільки загальна назва оголошення та ціна, а й вид тварини, порода, вік, стать, стан здоров'я, наявність щеплень, документи, умови утримання, місцезнаходження та можливість уточнити деталі у продавця. Саме ці характеристики потребують структурованого подання в межах спеціалізованої інформаційної системи.

Актуальність теми зумовлена тим, що процес вибору домашньої тварини має підвищені вимоги до достовірності, зручності та повноти інформації. Неповний опис, відсутність фільтрів за суттєвими параметрами, складність порівняння оголошень і необхідність переходити до зовнішніх каналів зв'язку знижують ефективність використання універсальних сервісів. У таких умовах доцільною є розробка окремого веб-додатку, який поєднує каталог оголошень, фільтрацію, особистий кабінет, обране, внутрішні повідомлення, відгуки, скарги та засоби модерації.

Інформаційна система в межах цієї роботи розглядається як сукупність програмних компонентів, даних і засобів взаємодії, які забезпечують збереження, обробку та надання користувачам структурованої інформації. Такий підхід відповідає загальним принципам інженерії програмного забезпечення, згідно з якими програмний продукт має не лише реалізовувати окремі функції, а й бути зручним, підтримуваним і придатним до подальшого розвитку [1]. Для веб-додатку, що працює з оголошеннями, це означає

необхідність чітко відокремити клієнтську частину, серверну логіку, базу даних, механізми автентифікації та роботу з файлами.

Особливе значення має якість інтерфейсу користувача. Згідно з підходом Nielsen Norman Group, користувацький досвід охоплює всі аспекти взаємодії користувача з продуктом, послугою або компанією [2]. У контексті веб-додатку для вибору та продажу домашніх тварин це означає, що система повинна бути зрозумілою для різних категорій користувачів, підтримувати швидкий пошук, не перевантажувати форму створення оголошення та забезпечувати логічну навігацію між головною сторінкою, каталогом, сторінкою оголошення, профілем і повідомленнями.

Не менш важливим є питання доступності сайту для пошукових систем і користувачів, які переходять безпосередньо на внутрішні сторінки. У рекомендаціях Google Search Central SEO Starter Guide зазначено, що пошукова оптимізація пов’язана з покращенням представлення сайту у пошуку та полегшенням індексації й розуміння його вмісту [3]. Тому в межах веб-додатку передбачено використання окремих сторінок для правил, контактів і політики конфіденційності, а також коректних англомовних slug-адрес для маршрутів і оголошень.

Оскільки система передбачає реєстрацію користувачів, збереження контактних даних, автентифікацію, завантаження фотографій і обмін повідомленнями, під час її розробки необхідно враховувати базові вимоги безпеки веб-додатків. У роботі такі вимоги враховано через використання хешування паролів, JWT-автентифікації, refresh-token механізму, валідації вхідних даних, перевірки файлів, розмежування ролей та модерації контенту.

Мета дослідження

Метою кваліфікаційної роботи є розробка веб-додатку для вибору та продажу домашніх тварин, який забезпечує зручне розміщення оголошень, пошук пропозицій за важливими характеристиками, перегляд детальної інформації, комунікацію між користувачами та базові засоби контролю якості контенту.

Завдання дослідження

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати предметну область вибору та продажу домашніх тварин через веб-сервіси;

– розглянути існуючі програмні рішення та визначити їхні переваги й обмеження;

– сформулювати функціональні можливості майбутнього веб-додатку;

– обґрунтувати вибір архітектури клієнт-серверного веб-додатку;

– реалізувати клієнтську частину для каталогу, профілю, обраного, повідомлень і службових сторінок;

– реалізувати серверну частину з REST API, автентифікацією, роботою з базою даних і файлами;

– створити механізми модерації, скарг, відгуків і адміністрування;

– перевірити працездатність системи локально та після розгортання на хостингу.

Об’єкт і предмет дослідження

Об’єктом дослідження є процес організації вибору, публікації та перегляду оголошень про домашніх тварин у веб-середовищі. Об’єкт охоплює взаємодію між користувачами, які створюють оголошення, користувачами, які шукають тварину, а також адміністраторами й модераторами, відповідальними за підтримання впорядкованості сервісу.

Предметом дослідження є програмні засоби, архітектурні рішення та інтерфейсні підходи до розробки веб-додатку для вибору та продажу домашніх тварин. До предмета дослідження належать структура даних оголошення, механізми пошуку і фільтрації, авторизація користувачів, завантаження фотографій, внутрішній чат, відгуки, скарги, модерація та адміністративна панель.

Методи дослідження

У процесі виконання роботи використано методи аналізу, порівняння, системного підходу, моделювання та практичного програмування. Метод

аналізу застосовано для вивчення проблем предметної області та існуючих аналогів. Метод порівняння використано для зіставлення функціональних можливостей універсальних і спеціалізованих сервісів. Системний підхід застосовано для поділу веб-додатку на клієнтську частину, серверну частину, базу даних та допоміжні модулі. Метод моделювання використано під час визначення сутностей бази даних, ролей користувачів і сценаріїв взаємодії. Практичне програмування застосовано для реалізації й перевірки створеної системи.

Наукова новизна

Наукова новизна одержаних результатів полягає в подальшому розвитку підходу до побудови спеціалізованого веб-сервісу оголошень, у якому акцент зроблено не на загальному розміщенні пропозицій, а на структурованому поданні параметрів домашньої тварини, поєднанні каталогу, особистого кабінету, внутрішньої комунікації, рейтингу продавця та модераційних механізмів у єдиній системі. На відміну від універсальних дошок оголошень, запропоноване рішення орієнтоване саме на предметну область домашніх тварин.

Практичне значення

Практичне значення роботи полягає у створенні працездатного веб-додатку, який може використовуватися як основа для тематичного сервісу оголошень про домашніх тварин. Розроблена система дає змогу користувачам публікувати пропозиції, додавати фотографії, шукати оголошення за параметрами, зберігати обране, обмінюватися повідомленнями, залишати відгуки та подавати скарги. Для адміністратора передбачено окремі інструменти керування користувачами, оголошеннями, видами тварин, породами й текстовими сторінками сайту.

Особистий внесок автора полягає у визначенні вимог до системи, виборі технологічного стеку, проєктуванні структури веб-додатку, реалізації клієнтської та серверної частин, створенні сторінок каталогу, профілю, оголошень, повідомлень, модерації та адміністрування, а також у розгортанні

системи на хостингу. Окремо було виконано доопрацювання користувацького досвіду, зокрема автоматичне переміщення сторінки догори під час переходу між маршрутами React Router, що покращує сприйняття навігації в односторінковому веб-додатку.

Апробація результатів роботи

Апробація результатів роботи здійснена шляхом підготовки тез за темою «Розробка веб-додатку для вибору та продажу домашніх тварин». У тезах обґрунтовано доцільність створення спеціалізованої інформаційної системи, наведено основні функціональні можливості та визначено практичну цінність розробки.

Структура роботи відповідає логіці розробки програмного забезпечення. У вступі обґрунтовано актуальність теми, визначено мету, завдання, об’єкт, предмет, методи дослідження та практичне значення. У першому розділі подано аналіз предметної області, огляд літературних джерел і програмних продуктів-аналогів, а також постановку задачі. У наступних розділах подано обґрунтування вибору технологій, проєктування архітектури, опис структури даних, реалізацію основних модулів та результати тестування веб-додатку.

Глосарій

API – інтерфейс прикладного програмування, через який клієнтська частина веб-додатку обмінюється даними із серверною частиною.

Backend – серверна частина програмної системи, що відповідає за обробку запитів, бізнес-логіку, роботу з базою даних, файлами та автентифікацією.

Frontend – клієнтська частина веб-додатку, яка працює у браузері користувача та забезпечує відображення інтерфейсу і взаємодію з API.

JWT – JSON Web Token, токен для передавання підтвердження автентичності користувача під час звернення до захищених маршрутів API.

ORM – технологія об’єктно-реляційного відображення, яка спрощує взаємодію програмного коду з реляційною базою даних.

PostgreSQL – реляційна система керування базами даних, використана для зберігання користувачів, оголошень, повідомлень, відгуків та інших структурованих даних.

Prisma – ORM-засіб для Node.js, який використовується для опису схеми бази даних, виконання міграцій і формування запитів до PostgreSQL.

Refresh token – токен оновлення, що використовується для продовження сесії користувача без повторного введення логіна та пароля.

REST API – стиль побудови програмного інтерфейсу, у якому дані передаються між клієнтом і сервером через HTTP-запити.

SEO – пошукова оптимізація, що охоплює технічні та змістові рішення для кращого розуміння сторінок пошуковими системами.

Socket.IO – бібліотека для обміну подіями між клієнтом і сервером у режимі, близькому до реального часу, використана для внутрішнього чату.

Веб-додаток – програмна система, доступна через браузер і побудована з клієнтської та серверної частин.

Каталог – розділ веб-додатку, у якому користувач переглядає оголошення та застосовує пошук, фільтрацію, сортування і пагінацію.

Модерація – перевірка оголошень, скарг, користувачів або повідомлень з метою підтримання якості та безпечності контенту.

Оголошення – основна інформаційна сутність системи, що описує пропозицію щодо домашньої тварини та містить її характеристики, фото, ціну, місцезнаходження і статус.

Роль користувача – набір прав доступу, що визначає можливості гостя, зареєстрованого користувача, модератора або адміністратора.

Фільтрація – відбір оголошень за параметрами: видом тварини, породою, містом, статтю, віком, типом оголошення та ціною.

Чат – внутрішній механізм обміну повідомленнями між користувачами в межах веб-додатку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис проблеми вибору та продажу домашніх тварин

Предметна область веб-додатку пов’язана з публікацією, пошуком і переглядом оголошень про домашніх тварин. У межах цієї області взаємодіють кілька груп користувачів: власники або продавці тварин, потенційні покупці, користувачі, які шукають тварину для безоплатного прийняття, модератори та адміністратори. Кожна група має власні потреби, однак усі вони залежать від якості подання інформації та зручності інтерфейсу.

Для продавця або власника тварини важливо швидко створити оголошення, додати фотографії, вказати основні характеристики та отримати звернення від зацікавлених користувачів. Якщо форма створення оголошення перевантажена або нелогічна, користувач може не заповнити важливі поля, що погіршує якість каталогу. Якщо ж форма надто спрощена, у системі з’являються неповні оголошення, які складно порівнювати між собою.

Для покупця або користувача, який шукає тварину, важливими є швидкість пошуку та можливість відфільтрувати нерелевантні пропозиції. Наприклад, користувач може шукати кошеня певної породи у конкретному місті, собаку певного віку або оголошення про тварину, яку віддають у добрі руки. Універсальна дошка оголошень може містити категорію тварин, але її структура зазвичай не орієнтована на повний набір параметрів, притаманних саме цій предметній області.

Окремою проблемою є довіра між сторонами. Оголошення про тварин можуть містити неповну або недостовірну інформацію, повторюваний текст, підозрілі формулювання, відсутні фотографії або контакти, що змушують користувача переходити до зовнішніх месенджерів. Для зменшення таких ризиків доцільно передбачити внутрішній чат, відгуки про продавця, скарги на оголошення або повідомлення та базову модерацію. Наявність цих механізмів

не гарантує повного усунення ризиків, але підвищує контрольованість інформаційного середовища.

Також важливим є питання локалізації та зрозумілості інтерфейсу. Для українського користувача доцільно реалізувати інтерфейс українською мовою, а технічні маршрути та slug-адреси залишити англомовними. Такий підхід дає змогу зробити сайт зрозумілим для користувача, але водночас забезпечити охайну структуру URL-адрес, придатну для пошукової оптимізації та подальшого розгортання.

Отже, основна проблема полягає у відсутності простого спеціалізованого середовища, яке поєднує структурований каталог оголошень про домашніх тварин, зручний пошук, особистий кабінет, обране, внутрішню комунікацію, рейтинг продавця, скарги, модерацію та адміністрування. Саме ця проблема визначає необхідність створення окремого веб-додатку.

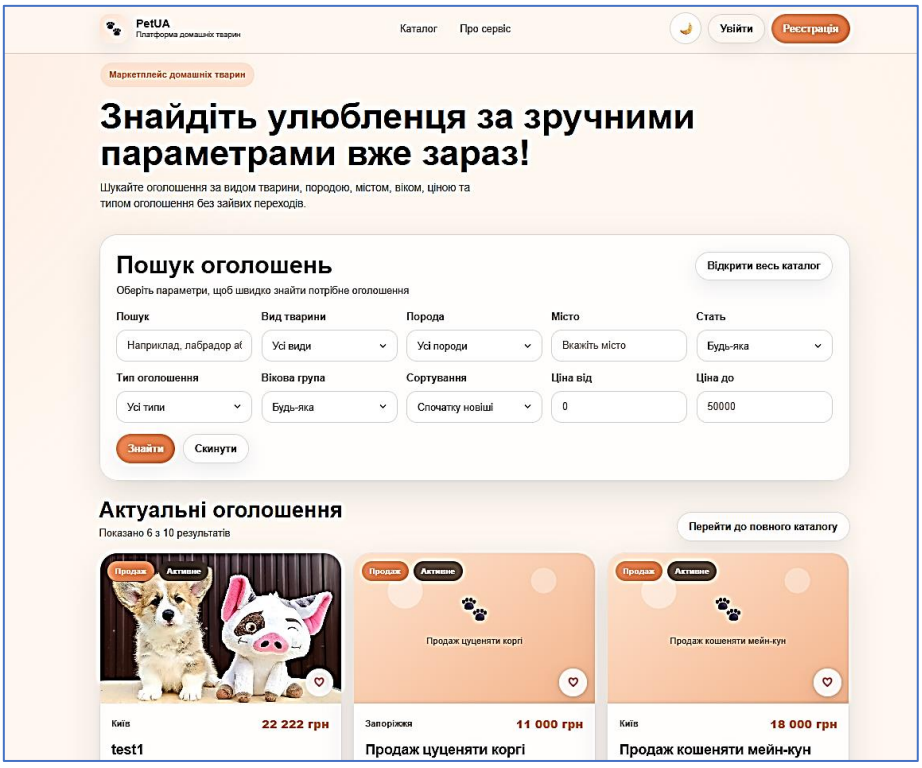


Рисунок 1 – *Загальний вигляд головної сторінки веб-додатку PetUA*

1.2 Огляд літературних джерел і технологічних основ

Під час аналізу предметної області було враховано літературні та електронні джерела, пов’язані з проєктуванням веб-додатків, користувацьким досвідом, безпекою, базами даних і сучасними інструментами розробки. Оскільки робота має прикладний характер, основну увагу приділено тим джерелам, які безпосередньо впливають на архітектуру та реалізацію системи.

Загальні принципи розробки програмного забезпечення розглянуто на основі праці M. T. Valente «Software Engineering: A Modern Approach» [1]. У контексті цієї роботи важливими є положення щодо структурованого підходу до вимог, проєктування, реалізації та тестування. Веб-додаток для оголошень не можна розглядати лише як набір сторінок, оскільки він містить дані, ролі користувачів, сценарії взаємодії, правила доступу та механізми контролю якості контенту.

Для оцінки значення інтерфейсу використано матеріали Nielsen Norman Group [2]. З них випливає, що користувацький досвід формується не тільки зовнішнім виглядом сторінки, а й усім шляхом користувача: від першого переходу на сайт до створення оголошення, перегляду пропозиції, додавання в обране або початку діалогу. Тому в системі важливо забезпечити зрозумілі назви сторінок, помітні кнопки дій, адаптивність для різних екранів і передбачувану поведінку під час переходу між сторінками.

Питання пошукової оптимізації та коректної структури сторінок розглянуто на основі рекомендацій Google Search Central [3]. Для веб-додатку з каталогом оголошень це має практичне значення, оскільки кожне оголошення може бути окремою сторінкою з унікальною адресою. У розробленій системі це враховано через використання людино-зрозумілих англомовних slug-адрес, окремих сторінок правил, контактів і політики конфіденційності, а також SEO-компонента на клієнтській частині.

Під час проєктування серверної частини окрему увагу приділено безпеці автентифікації, контролю доступу, перевірці вхідних даних і обмеженню

файлів, що завантажуються. У системі передбачено ролі USER, MODERATOR та ADMIN, а доступ до адміністративних і модераційних маршрутів обмежено за ролями.

Клієнтську частину реалізовано з використанням React. Офіційна документація React визначає цю бібліотеку як засіб побудови користувацьких інтерфейсів із компонентів [4]. Для розробленої системи компонентний підхід є доцільним, оскільки картка оголошення, форма фільтрів, форма скарги, елементи профілю, хедер, футер та інші частини інтерфейсу можуть бути винесені в окремі повторно використовувані компоненти.

Для швидкого створення та збирання клієнтського додатку використано Vite. Офіційні матеріали Vite описують його як сучасний інструмент frontend-збирання, орієнтований на швидку розробку веб-додатків [5]. У межах дипломного проєкту це рішення дало змогу швидко запускати локальне середовище, перевіряти зміни в інтерфейсі та збирати production-версію для розгортання на статичному хостингу.

Серверну частину реалізовано на Node.js та Express. Node.js є асинхронним подієво-орієнтованим середовищем виконання JavaScript, придатним для створення масштабованих мережевих застосунків [6]. Express, у свою чергу, є мінімалістичним веб-фреймворком для Node.js, який надає засоби для створення веб-додатків і API [7]. Така комбінація є достатньою для реалізації REST API, роботи з middleware, обробки помилок, авторизації та підключення окремих маршрутів.

Для збереження даних обрано PostgreSQL. Офіційні матеріали PostgreSQL характеризують його як потужну об’єктно-реляційну систему керування базами даних з відкритим кодом [8]. У цьому веб-додатку PostgreSQL використовується для збереження користувачів, оголошень, видів тварин, порід, фотографій, обраного, діалогів, повідомлень, відгуків, скарг і службових текстових сторінок.

Для роботи з базою даних використано Prisma ORM. Згідно з документацією Prisma, цей інструмент забезпечує доступ до баз даних,

міграції та зручний клієнт для Node.js [9]. У межах системи Prisma використовується для опису схеми даних, створення міграцій, виконання запитів і підтримання відповідності між моделями додатку та таблицями бази даних.

Автентифікація користувачів побудована з використанням JSON Web Token. Стандарт RFC 7519 визначає JWT як компактний URL-безпечний спосіб передавання claims між сторонами [10]. У розробленій системі JWT використовується для access-token механізму, а refresh-token зберігається з хешуванням для підтримання сесії користувача після перезавантаження сторінки.

Функцію внутрішнього чату реалізовано з використанням Socket.IO. Офіційна документація визначає Socket.IO як бібліотеку для низьколатентної двонапрямної подієвої комунікації між клієнтом і сервером [11]. Це відповідає потребам системи, де користувачі повинні швидко отримувати нові повідомлення без постійного ручного оновлення сторінки.

1.3 Огляд існуючих рішень

Оскільки завдання роботи пов’язане зі створенням веб-додатку для вибору та продажу домашніх тварин, було розглянуто існуючі рішення, які частково або повністю виконують подібні функції. До аналізу включено універсальні дошки оголошень, тематичні сервіси про тварин і платформу, орієнтовану на адопцію. Це дало змогу порівняти не лише наявність каталогу, а й зручність пошуку, структурованість даних, можливості комунікації, довіру до продавців і придатність рішення для українського користувача.

OLX Україна — універсальна дошка оголошень, яка має окрему категорію для продажу домашніх тварин. На сторінці категорії сервіс зазначає, що користувач може швидко купити домашнього улюбленця [12]. Рішення є відомим серед українських користувачів і має велику кількість оголошень у різних містах.

Переваги:

– велика аудиторія користувачів і продавців;

– знайомий для більшості користувачів формат дошки оголошень;

– можливість швидко опублікувати пропозицію та отримати звернення.

Недоліки:

– сервіс є універсальним, тому предметна область домашніх тварин не є його основним напрямом;

– опис тварини залежить переважно від тексту продавця, а не від спеціалізованої структури характеристик;

– тематична модерація, внутрішній рейтинг продавця саме для цієї предметної області та контроль якості оголошень не є центральними функціями сервісу.

RIA.com — український сервіс оголошень, у якому є розділ «Домашні тварини та зоотовари». На сторінці розділу подано пропозиції щодо домашніх і сільськогосподарських тварин, а також зоотоварів [13]. Сервіс охоплює широку предметну область і може використовуватися для пошуку тварин у різних регіонах України.

Переваги:

– наявність окремого розділу для тварин і зоотоварів;

– підтримка різних категорій пропозицій;

– можливість переглядати оголошення за містами та типами тварин.

Недоліки:

– домашні тварини поєднані з товарами для тварин, що зменшує тематичну зосередженість сервісу;

– система не орієнтована саме на повний сценарій вибору домашньої тварини;

– відсутній акцент на внутрішньому чаті, відгуках, скаргах і модерації як єдиному процесі в межах спеціалізованого сервісу.

Lapki.pet — тематичний сервіс оголошень, пов’язаний із продажем і купівлею домашніх тварин. У матеріалах сервісу зазначено, що він надає інформацію про вартість, вік і місцезнаходження тварини, а також орієнтований на покупців, продавців, розплідники та заводчиків [14]. Це

рішення є найближчим до теми кваліфікаційної роботи, оскільки має спеціалізовану спрямованість.

Переваги:

– тематична орієнтація саме на домашніх тварин;

– наявність характеристик, важливих для вибору тварини;

– підтримка оголошень від приватних осіб і розплідників.

Недоліки:

– функціональність сервісу не може бути адаптована під навчальну задачу та власну структуру бази даних;

– архітектура, внутрішня логіка модерації та реалізація інтерфейсу не є відкритими для аналізу в межах цієї роботи;

– у власному веб-додатку необхідно самостійно реалізувати кабінет користувача, чат, обране, скарги, відгуки, модерацію та адміністративну панель.

Petfinder — міжнародна платформа для пошуку тварин із притулків і рятувальних організацій. На головній сторінці сервіс пропонує знайти нового домашнього улюбленця та переглядати тварин із мережі притулків і rescue-організацій [15]. Основний акцент цього рішення зроблено на адопції, а не на продажу або змішаних типах оголошень.

Переваги:

– соціальна спрямованість і підтримка притулків;

– зручний пошук тварин за базовими параметрами;

– великий досвід організації тематичного каталогу тварин.

Недоліки:

– сервіс орієнтований переважно на адопцію, а не на продаж, в’язку або оголошення про знайдених тварин;

– географія та модель роботи не відповідають локальному українському сервісу;

– функціональність не покриває всі сценарії, передбачені у веб-додатку PetUA.

Таблиця *1.1 – Порівняння існуючих рішень з вимогами до веб-додатку PetUA*

Рішення	**Тип рішення**	**Сильні сторони**	**Обмеження щодо задачі роботи**
OLX Україна	Універсальна дошка оголошень із категорією тварин	Велика аудиторія, швидка публікація, знайомий формат роботи	Не має вузької спеціалізації на параметрах тварини, тематичній модерації, відгуках і внутрішньому сценарії вибору домашнього улюбленця
RIA.com	Універсальний каталог оголошень із розділом тварин і зоотоварів	Наявність розділу для тварин, широкий набір пропозицій, підтримка регіонального пошуку	Тварини поєднані із зоотоварами, тому сервіс не зосереджений лише на виборі тварини та не покриває повний набір функцій PetUA
Lapki.pet	Тематичний сервіс оголошень про домашніх тварин	Орієнтація саме на тварин, наявність відомостей про вік, вартість і місцезнаходження	Є зовнішнім готовим сервісом; у межах роботи необхідна власна реалізація архітектури, бази даних, кабінету, чату, скарг, модерації й адміністрування
Petfinder	Міжнародний сервіс пошуку тварин для адопції	Соціальна спрямованість, робота з притулками, тематичний каталог тварин	Орієнтований переважно на адопцію, має іншу географію та не підтримує змішану модель продажу, передачі, в’язки й знайдених тварин

Порівняльний аналіз показав, що розглянуті рішення вирішують лише частину поставленої задачі. Універсальні дошки оголошень мають велику аудиторію, але не забезпечують достатньої предметної деталізації. Тематичні сервіси краще відповідають потребам пошуку тварин, однак не дають змоги описати, змінити та захистити власну архітектурну реалізацію. Сервіси адопції мають сильну соціальну спрямованість, але не підтримують усі типи оголошень, необхідні для цієї системи.

Отже, створення власного веб-додатку PetUA є доцільним. Розробка дає змогу поєднати каталог домашніх тварин, фільтри, детальну сторінку оголошення, особистий кабінет, обране, внутрішній чат, відгуки, скарги, модерацію та адміністративне керування в єдиному інформаційному середовищі. Саме це відрізняє запропоновану систему від універсальних платформ і робить її придатною для розв’язання поставленої задачі.

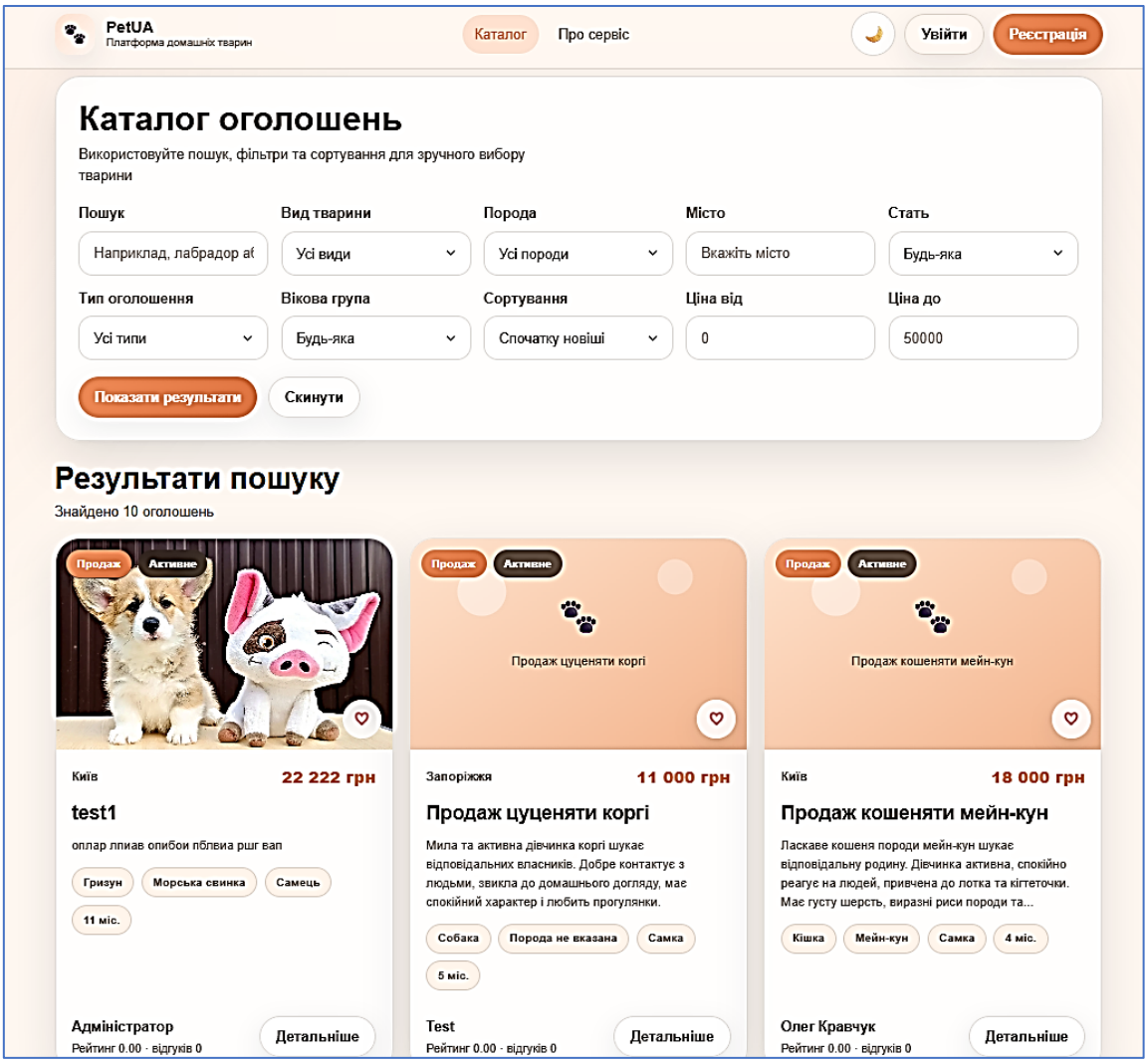


Рисунок *2 – Сторінка каталогу оголошень із фільтрами пошуку*

1.4 Постановка задачі

На основі аналізу предметної області та існуючих рішень сформульовано задачу розробки веб-додатку PetUA для вибору та продажу домашніх тварин. Система має працювати як клієнт-серверний веб-додаток із React/Vite клієнтською частиною, Node.js/Express серверною частиною, PostgreSQL базою даних, Prisma ORM, JWT-автентифікацією, механізмом refresh-token, Socket.IO чатом та локальним сховищем файлів для фотографій.

Система повинна дозволяти незареєстрованому користувачу переглядати головну сторінку, каталог оголошень, сторінку окремого оголошення, сторінку продавця, контакти, правила сайту та політику конфіденційності. Для каталогу мають бути доступні пошук, фільтрація за видом тварини, породою, містом, статтю, типом оголошення, віковою групою, ціною та сортуванням.

Зареєстрований користувач повинен мати можливість створювати, редагувати та видаляти власні оголошення, завантажувати фотографії, переглядати власні оголошення у профілі, додавати оголошення до обраного, писати продавцю повідомлення, переглядати діалоги, залишати відгуки та подавати скарги на оголошення або повідомлення.

Модератор повинен мати доступ до скарг і засобів перевірки проблемного контенту. Для модератора доцільно передбачити перегляд статистики скарг, зміну статусу скарги, приховування та відновлення оголошення. Адміністратор повинен мати ширші можливості: керувати користувачами, ролями, статусами, оголошеннями, видами тварин, породами та службовими текстовими сторінками.

Форма створення оголошення має містити поля для назви, опису, типу оголошення, виду тварини, породи, статі, віку, ціни, міста, області, інформації про здоров’я, щеплення, документи, умови утримання та фотографії. При цьому система повинна виконувати валідацію обов’язкових полів і базову

перевірку текстового вмісту на підозрілі формулювання або неприродні повтори символів.

Для покращення користувацького досвіду веб-додаток повинен підтримувати адаптивну верстку, світлу й темну тему, бургер-меню на малих екранах, коректну роботу у різних браузерах, зрозумілі повідомлення про помилки, автоматичне оновлення профілю після входу та збереження сесії після перезавантаження сторінки. Окремою вимогою є автоматичне прокручування сторінки догори під час переходу між маршрутами, оскільки в односторінкових React-додатках без цього користувач може залишатися внизу сторінки та не помічати переходу.

З погляду даних система повинна зберігати користувачів, refresh-токени, види тварин, породи, оголошення, фотографії, обране, діалоги, учасників діалогів, повідомлення, відгуки, скарги та текстові блоки сайту. Взаємозв’язки між сутностями мають бути реалізовані через зовнішні ключі, а видалення пов’язаних даних повинно виконуватися контрольовано через правила Prisma та PostgreSQL.

З погляду розгортання система повинна працювати локально у середовищі розробки та бути придатною до розміщення на хостингу. Серверна частина має працювати як Render Web Service, база даних — як Render PostgreSQL, а клієнтська частина — як Render Static Site. Для коректної роботи маршрутизації клієнтського додатку на статичному хостингу необхідно налаштовувати rewrite-правила, оскільки React Router обробляє внутрішні маршрути на стороні клієнта [16].

Отже, поставлена задача передбачає не лише створення окремих сторінок, а повну реалізацію тематичної інформаційної системи з каталогом, ролями, базою даних, API, файлами, автентифікацією, повідомленнями, модерацією та адміністративним керуванням.

1.5 Висновки до розділу 1

У першому розділі проаналізовано предметну область вибору та продажу домашніх тварин через веб-додаток. Встановлено, що універсальні дошки оголошень частково вирішують задачу публікації пропозицій, однак не забезпечують повного спеціалізованого середовища для структурованого вибору тварини, збереження обраного, внутрішньої комунікації, відгуків, скарг і модерації.

Огляд літературних джерел і технологічних основ показав доцільність використання клієнт-серверної архітектури з React, Vite, Node.js, Express, PostgreSQL, Prisma, JWT та Socket.IO. Такий стек відповідає задачі розробки сучасного веб-додатку, у якому потрібно поєднати зручний інтерфейс, API, базу даних, автентифікацію, завантаження файлів і обмін повідомленнями.

Аналіз аналогів показав, що OLX Україна та RIA.com мають велику аудиторію, але є універсальними сервісами; Petfinder орієнтований переважно на адопцію; Lapki.pet є тематичним рішенням, однак у межах кваліфікаційної роботи необхідно створити власну систему з повним описом архітектури й реалізації. На підставі цього сформульовано задачу розробки веб-додатку PetUA як спеціалізованої інформаційної системи для вибору та продажу домашніх тварин.

2 ДОСЛІДЖЕННЯ ЗАСОБІВ РЕАЛІЗАЦІЇ ТА ФОРМУВАННЯ ВИМОГ

2.1 Огляд існуючих рішень

Огляд існуючих рішень виконано з метою визначення функцій, які доцільно реалізувати у веб-додатку PetUA, а також для виявлення обмежень наявних платформ. Для порівняння розглянуто універсальні дошки оголошень, тематичні каталоги тварин та сервіси, орієнтовані на адопцію. Такий підхід дає змогу оцінити не лише зовнішній інтерфейс, а й модель даних, спосіб пошуку, комунікацію між користувачами та наявність модерації.

Під час огляду враховано не лише наявність категорії тварин у конкретному сервісі, а й те, наскільки така категорія підтримує предметну структуру даних. Для веб-додатку PetUA важливо, щоб оголошення не було звичайним текстовим повідомленням із фотографією, а мало впорядкований набір полів. Саме тому під час порівняння аналізувалися фільтри, спосіб подання картки, наявність сторінки продавця, внутрішніх повідомлень, обраного, рейтингу, скарг і модераційних механізмів.

Окремо враховано відмінність між платформами, які працюють як загальні дошки оголошень, і платформами, орієнтованими на адопцію або спеціалізований пошук тварин. Універсальні сервіси мають перевагу у великій аудиторії, однак їхня інформаційна модель зазвичай не розроблена спеціально під характеристики тварини. Тематичні сервіси краще враховують предметну область, але можуть бути обмежені за регіоном, мовою інтерфейсу або набором доступних функцій.

Для оцінювання аналогів доцільно виділити такі критерії: тематична спрямованість, зручність каталогу, повнота параметрів оголошення, можливість зв’язку між користувачами, наявність обраного, контроль контенту, локалізація, адаптивність та зрозумілість інтерфейсу. Такий набір критеріїв дає змогу оцінити не тільки зовнішній вигляд сервісу, а й

відповідність його функціональності задачам розроблюваної інформаційної системи.

OLX Україна є одним із найбільш відомих сервісів оголошень. Його перевагою є велика аудиторія, наявність пошуку, категорій, фільтрів і повідомлень між користувачами. Водночас платформа є універсальною, тому оголошення про тварин розміщуються поряд з великою кількістю інших товарів і послуг. Для предметної області домашніх тварин це створює обмеження: частина важливих характеристик не має окремих структурованих полів, а користувач змушений шукати потрібні відомості у довільному описі.

Перевагами універсальної моделі OLX є велика кількість користувачів, швидка публікація оголошення, зрозумілий механізм пошуку і широка впізнаваність платформи. Для користувача це зменшує поріг входу, оскільки він уже знайомий із загальним принципом роботи сервісу.

Недоліком такої моделі є те, що тварина в ній розглядається як один із багатьох типів товарів або пропозицій. Через це сервіс не може повністю врахувати специфіку вибору живої істоти: історію здоров’я, умови утримання, наявність щеплень, документів, особливості поведінки та відповідальність майбутнього власника.

Для PetUA з такого аналога доцільно запозичити принцип швидкого доступу до каталогу, зрозумілі картки та простий шлях до зв’язку з автором. Водночас слід уникнути надмірної універсальності та зробити структуру оголошення спеціалізованою саме для домашніх тварин.

RIA.com також містить розділ для тварин і зоотоварів, проте за логікою роботи залишається загальною платформою оголошень. Перевагами є зрозуміла структура сторінок, підтримка пошуку та фільтрів. Недоліком є те, що сервіс не зосереджений саме на сценарії відповідального вибору домашньої тварини, де важливі не лише ціна й місто, а й стан здоров’я, документи, умови утримання, інформація про продавця, рейтинг і можливість подати скаргу.

Сильною стороною RIA.com є поділ оголошень на категорії та наявність базової структури для пошуку. Це корисно для користувача, який приблизно розуміє, що шукає, і хоче обмежити кількість результатів. Проте в межах предметної області домашніх тварин важливими є не лише категорія та місцезнаходження, а й сукупність детальніших параметрів.

У PetUA ці обмеження враховано через виділення окремих сутностей для виду тварини та породи, а також через додаткові поля оголошення. Такий підхід дозволяє не втрачати важливі характеристики у вільному описі та формує кращу основу для подальшої фільтрації, сортування і модерації.

Lapki.pet є ближчим до предметної області, оскільки орієнтований на оголошення про тварин. Сильними сторонами такого рішення є тематичний каталог, прості картки та зрозуміле подання інформації. Однак для кваліфікаційної роботи важливим є не лише використання готового тематичного сервісу, а створення власної інформаційної системи з відкритою для аналізу архітектурою, моделлю даних, серверною логікою, ролями користувачів і власними адміністративними функціями.

Тематичні сервіси на кшталт Lapki.pet підтверджують доцільність окремого підходу до оголошень про тварин. Якщо користувач заходить саме на такий сайт, він очікує не випадкової стрічки, а предметно впорядкованого каталогу. Тому в PetUA передбачено головну сторінку з акцентом на швидкий пошук, повний каталог, сторінку оголошення, сторінку продавця та особистий кабінет.

Разом із цим для дипломної розробки важливо показати не лише користувацьку частину, а й інженерну повноту системи. Через це у PetUA реалізовано серверну логіку, базу даних, API-маршрути, автентифікацію, завантаження файлів, ролі та адміністративні можливості. Саме ці складові відрізняють інформаційну систему від простої статичної сторінки або прототипу інтерфейсу.

Petfinder демонструє приклад розвиненого сервісу для пошуку тварин, однак його основна модель пов’язана з адопцією та притулками. Це рішення

корисне як орієнтир щодо зручності карток і пошуку, але воно не відповідає локальному сценарію продажу, публікації приватних оголошень, обробки відгуків про продавця та побудови власної модераційної моделі.

Petfinder доцільно розглядати як приклад предметно орієнтованого сервісу, у якому центральним об’єктом є не загальний товар, а тварина. Проте логіка адопції та робота з притулками відрізняються від моделі PetUA. У розроблюваній системі підтримуються різні типи оголошень: продаж, передача у добрі руки, в’язка, знайдено або загублено, резерв чи продано. Це робить модель даних ширшою і потребує більш універсального механізму статусів.

На основі огляду аналогів можна зробити висновок, що кожне з рішень має корисні елементи, однак жодне з них не поєднує у потрібному вигляді локалізований український інтерфейс, тематичну структуру оголошень, внутрішню комунікацію, модерацію, ролі та адміністративне керування. Тому розробка власного веб-додатку є не дублюванням існуючих сервісів, а практично обґрунтованою спробою створити систему з потрібним набором функцій. У таблиці 2.1 наведено порівняння існуючих рішень з вимогами PetUA.

Таблиця *2.1 – Порівняння існуючих рішень з вимогами PetUA*

Рішення	Сильні сторони	Обмеження для задачі
OLX Україна	велика аудиторія, пошук, повідомлення	універсальна платформа, недостатньо предметних полів
RIA.com	структуровані оголошення, фільтри	немає повної спеціалізації на тваринах
Lapki.pet	тематична спрямованість, прості картки	закрита реалізація, обмежена керованість архітектури
Petfinder	розвинена логіка пошуку тварин	орієнтація на адопцію, інша модель ринку

Проведений аналіз показав, що наявні рішення мають корисні функції, але не повністю відповідають поставленій задачі. Тому доцільною є розробка власного веб-додатку, у якому тематичний каталог поєднується з ролями

користувачів, структурованою базою даних, внутрішнім чатом, відгуками, скаргами та модерацією.

Порівняння підтвердило, що ключовою відмінністю PetUA має бути не лише оформлення сторінок, а й предметна модель даних. Універсальна дошка оголошень може дозволити створити текстову пропозицію про тварину, але вона не гарантує однакового заповнення важливих параметрів. У спеціалізованій системі ці параметри стають частиною структури оголошення, що покращує якість пошуку та сприйняття інформації.

Крім того, для предметної області домашніх тварин важливо не відокремлювати каталог від комунікації. Користувач часто хоче уточнити додаткові деталі: стан здоров’я, поведінку, можливість перегляду тварини, документи або умови передачі. Тому внутрішній чат, відгуки і сторінка продавця є не другорядними прикрасами, а функціями, що безпосередньо підтримують реальний сценарій використання.

Окремо слід зазначити роль модерації. Навіть базовий механізм скарг і перевірки підозрілого контенту підвищує керованість системи. Для дипломної роботи це також демонструє, що система розглядається як повноцінне інформаційне середовище, у якому передбачено не лише створення записів, а й контроль їхнього життєвого циклу.

На рисунку 3 наведено приклад сторінки каталогу існуючого сервісу оголошень, який використовується для розміщення пропозицій про домашніх тварин. Такий приклад дає змогу порівняти універсальний підхід до подання оголошень зі спеціалізованою структурою, передбаченою у веб-додатку PetUA.

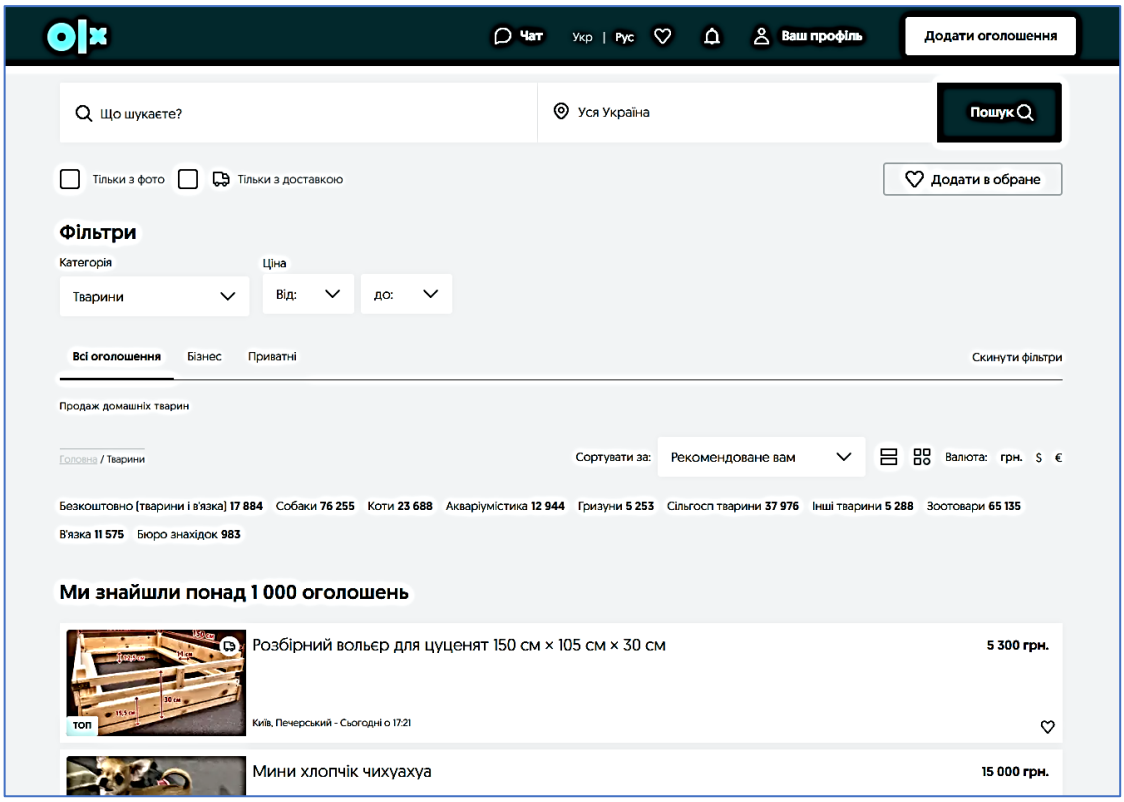


Рисунок *3 – Приклад сторінки каталогу існуючого сервісу оголошень про домашніх тварин OLX*

2.2 Обґрунтування вибору технологій

Клієнтську частину реалізовано з використанням React і Vite. React обрано через компонентний підхід, який дає змогу розділити інтерфейс на незалежні елементи: шапку сайту, картку оголошення, форму фільтрів, форму авторизації, сторінку профілю, модальні вікна, галерею фотографій і службові повідомлення. Така структура спрощує підтримку коду й дає змогу повторно використовувати компоненти на різних сторінках.

Компонентний підхід є важливим саме для такого веб-додатку, оскільки багато елементів інтерфейсу повторюються на різних сторінках. Наприклад, картка оголошення використовується на головній сторінці, у каталозі, в обраному та в подібних оголошеннях на сторінці детального перегляду. Якщо

логіку такої картки винесено в окремий компонент, її легше підтримувати, змінювати та перевіряти.

React Router використовується для маршрутизації між сторінками односторінкового веб-додатку. Такий підхід створює відчуття повноцінного сайту з окремими адресами, але без повного перезавантаження документа браузером. Для покращення навігації додатково реалізовано компонент автоматичного прокручування сторінки догори після переходу між маршрутами. Фрагмент цього рішення наведено у лістингу 1.

Лістинг 1 – *Компонент прокручування сторінки догори після зміни маршруту*

```
import { useEffect } from 'react'
import { useLocation } from 'react-router-dom'

function ScrollToTop() {
    const { pathname, search } = useLocation()

    useEffect(() => {
        window.scrollTo({ top: 0, left: 0, behavior:
'instant' })
    }, [pathname, search])

    return null
}

export default ScrollToTop
```

Vite використано як інструмент збирання клієнтської частини. Його застосування є доцільним для навчального та практичного веб-проєкту, оскільки він забезпечує швидкий запуск середовища розробки, зручну роботу з модулями та просте формування виробничої збірки. Для реалізованого додатку це важливо, оскільки інтерфейс містить багато окремих сторінок і компонентів, які потрібно перевіряти під час розробки.

Серверну частину реалізовано на Node.js із використанням Express. Такий вибір відповідає задачі створення REST API для клієнтської частини. Express забезпечує маршрутизацію, обробку запитів, підключення middleware,

перевірку токенів, обробку помилок і роботу з файловими завантаженнями. Для системи PetUA сервер відповідає за автентифікацію, оголошення, користувачів, обране, чат, відгуки, скарги та адміністративні дії.

Для зберігання даних використано PostgreSQL. Реляційна модель є придатною для цієї предметної області, оскільки дані мають чіткі зв’язки: користувач створює оголошення, оголошення належить до виду та породи, має фотографії, може бути додане до обраного, отримувати скарги, а продавець може мати відгуки й рейтинг. Використання зовнішніх ключів забезпечує цілісність даних.

Prisma ORM застосовано як проміжний шар між серверною логікою та базою даних. Він дає змогу описати структуру сутностей у схемі, виконувати міграції, генерувати клієнт для запитів і зменшити кількість ручного SQL-коду. Для дипломного проєкту це також спрощує пояснення структури бази даних, оскільки схема Prisma відображає основні сутності системи.

Для автентифікації використано JWT і refresh token. Access token забезпечує доступ до захищених маршрутів протягом обмеженого часу, а refresh token дає змогу відновлювати сесію без повторного введення логіна й пароля. Такий підхід є доречним для веб-додатку, у якому частина функцій доступна тільки зареєстрованим користувачам, модераторам або адміністраторам.

Socket.IO використано для реалізації внутрішнього чату. На відміну від звичайних HTTP-запитів, такий механізм дає змогу передавати повідомлення між учасниками діалогу майже в реальному часі. Це важливо для сценарію, коли покупець хоче швидко уточнити деталі про тварину, документи, умови передачі або місцезнаходження.

Фотографії оголошень зберігаються локально у каталозі uploads. Для поточного стану проєкту це практичне рішення, оскільки додаток працює локально та не вимагає підключення зовнішнього файлового сховища. Водночас архітектура не забороняє у майбутньому замінити локальне зберігання на хмарне сховище.

Використання локального каталогу uploads також спрощує демонстрацію роботи системи в навчальному середовищі. Файл зберігається на сервері, а у базі даних фіксується шлях до нього. Клієнтська частина отримує цей шлях через API та формує URL для відображення зображення у картці або галереї. Такий підхід не потребує складної інфраструктури, але водночас дозволяє показати повний цикл роботи з файлами.

Під час завантаження фотографій важливо перевіряти тип і розмір файлу. Це зменшує ризик завантаження некоректних або небезпечних даних. Для PetUA такі перевірки виконуються на рівні серверного middleware, після чого фотографії зв’язуються з конкретним оголошенням. У результаті кожне оголошення може мати набір зображень, а не лише одну ілюстрацію.

Окремим практичним рішенням стала реалізація галереї на сторінці оголошення. Користувач може натиснути на основне фото, відкрити модальне вікно, переглядати зображення кнопками або клавішами та закривати галерею. Це підвищує зручність перегляду і робить сторінку оголошення більш завершеною. У таблиці 2.2 наведено призначення основних технологій у веб-додатку.

Таблиця *2.2 – Призначення основних технологій у веб-додатку*

Технологія	Рівень системи	Призначення
React	frontend	побудова компонентного інтерфейсу
Vite	frontend	середовище розробки та збірка
Node.js	backend	виконання серверної логіки
Express	backend	маршрути API та middleware
PostgreSQL	дані	реляційне зберігання сутностей
Prisma	дані	ORM, міграції, доступ до БД
JWT	безпека	перевірка доступу до захищених маршрутів
Socket.IO	комунікація	чат між користувачами

2.3 Формування функціональних вимог

Функціональні вимоги сформовано з урахуванням ролей користувачів і основних сценаріїв роботи. Гість повинен мати можливість переглядати головну сторінку, каталог, сторінку оголошення, сторінку продавця та службові сторінки. Реєстрація потрібна лише для дій, які змінюють дані або пов’язані з персональною взаємодією.

Поділ вимог за ролями дає змогу уникнути ситуації, коли всі функції системи стають доступними будь-якому користувачу. Наприклад, гість може переглядати відкриту інформацію, але не повинен створювати оголошення, надсилати повідомлення, додавати до обраного або залишати відгуки. Такі дії потребують автентифікації, оскільки мають бути прив’язані до конкретного облікового запису.

Зареєстрований користувач є основною робочою роллю системи. Він створює оголошення, керує власними даними, редагує профіль, завантажує аватар, переглядає власні оголошення, спілкується через чат і взаємодіє з продавцями. Саме для цієї ролі важливо забезпечити зручний особистий кабінет, оскільки через нього проходить більшість операцій після входу.

Модератор і адміністратор мають розширені права, але їхні задачі відрізняються. Модератор зосереджений на перевірці контенту, скарг і проблемних записів. Адміністратор відповідає за ширше керування системою: користувачів, оголошення, види тварин, породи, текстові сторінки та статистику. Такий поділ відповідає принципу мінімально необхідних прав.

2.3.1 Вимоги до структури оголошення

Оголошення у PetUA повинно бути не довільним текстовим блоком, а структурованим записом. Це дозволяє одночасно вирішити кілька задач: забезпечити зручне заповнення форми, сформувати картку оголошення,

реалізувати фільтрацію, показати характеристики у детальному перегляді та зберегти дані в базі у придатному для подальшої обробки вигляді.

До обов’язкових предметних полів належать назва, тип оголошення, вид тварини, порода, стать, вік, ціна або ознака безоплатної передачі, місто, область і опис. Додаткові поля містять інформацію про стан здоров’я, щеплення, документи, родовід, умови утримання і фотографії. Усі ці дані мають пряме практичне значення для користувача, який обирає домашню тварину.

Статус оголошення використовується для відображення життєвого циклу запису. Активне оголошення доступне у каталозі; зарезервоване або продане може залишатися видимим для історії або підтвердження завершення пропозиції; позначене модерацією оголошення потребує перевірки. Така модель дозволяє не видаляти дані без потреби, а керувати доступністю записів.

Зареєстрований користувач повинен створювати оголошення, редагувати власні записи, завантажувати фотографії, додавати оголошення до обраного, вести діалоги, залишати відгуки про продавців і подавати скарги. Модератор повинен мати інструменти для обробки скарг і приховування проблемного контенту. Адміністратор повинен керувати користувачами, довідниками, статусами, оголошеннями та службовими текстовими сторінками.

Оголошення є центральною сутністю системи. Воно повинно містити назву, вид тварини, породу, стать, вік, ціну, місцезнаходження, опис, стан здоров’я, наявність щеплень, документи або родовід, умови утримання, фотографії, дату публікації, автора та статус. Наявність окремих полів забезпечує якісніший пошук, ніж зберігання всієї інформації лише в довільному описі. У таблиці 2.3 наведено основні функціональні вимоги за ролями.

Таблиця *2.3 – Основні функціональні вимоги за ролями*

Роль	Основні дії
Гість	перегляд каталогу, пошук, фільтрація, перегляд сторінок
Користувач	створення оголошень, обране, чат, відгуки, скарги
Модератор	перегляд скарг, зміна статусів, приховування контенту
Адміністратор	керування користувачами, ролями, довідниками та сторінками

2.4 Нефункціональні вимоги

Нефункціональні вимоги визначають якість роботи системи. Для PetUA важливими є зручність інтерфейсу, адаптивність, доступність основних дій, стабільність локального запуску, зрозуміла структура коду, контроль доступу, валідація даних і можливість подальшого розширення.

Нефункціональні вимоги мають не менше значення, ніж перелік функцій, оскільки саме вони визначають придатність системи до реального використання. Веб-додаток може формально підтримувати створення оголошення, але якщо форма незручна, не працює на мобільному екрані або втрачає введені дані без зрозумілого повідомлення, така реалізація не буде якісною.

Для PetUA важливими є такі групи нефункціональних вимог:

– інтерфейс повинен бути українською мовою та використовувати зрозумілі назви полів і дій;

– сторінки мають коректно відображатися на настільних і мобільних екранах;

– дані форм повинні перевірятися до запису в базу даних;

– захищені дії мають бути доступними лише після автентифікації;

– кодова база повинна бути поділена на логічні модулі, придатні до підтримки;

– система має мати можливість подальшого розширення без повної зміни архітектури.

Підтримуваність забезпечується не лише вибором технологій, а й структурою проєкту. Клієнтська частина поділена на сторінки, компоненти, API-модулі, сховища стану, стилі та утиліти. Серверна частина поділена на маршрути, контролери, middleware, схеми валідації, утиліти й модулі роботи з базою даних. Такий поділ спрощує пошук помилок і внесення змін.

Важливою вимогою є коректна робота у різних браузерах. Під час доопрацювання системи було виявлено відмінність у відображенні елемента select на macOS, тому для нього додано окремі CSS-правила нормалізації. Це показує, що адаптивність у веб-додатку охоплює не лише ширину екрана, а й поведінку нативних елементів у різних середовищах.

Адаптивність реалізується через гнучкі CSS-сітки, медіа-запити та коректне відображення елементів на різних розмірах екрана. Окремо враховано роботу нативних елементів форми, зокрема select, оскільки різні операційні системи можуть відображати їх неоднаково. Для усунення цієї проблеми стилі select нормалізовано, а стандартну стрілку замінено на контрольований графічний елемент.

Безпека системи передбачає хешування паролів, використання токенів, розмежування доступу за ролями, перевірку належності ресурсу власнику, валідацію вхідних даних, обмеження дій для незареєстрованих користувачів і контроль скарг. З огляду на навчальний формат проєкту реалізовано базовий рівень захисту, достатній для демонстрації принципів безпечної роботи клієнт-серверної системи. У таблиці наведено 2.4 – Нефункціональні вимоги до системи.

Таблиця *2.4 – Нефункціональні вимоги до системи*

Вимога	Зміст
Зручність	простий інтерфейс, зрозумілі форми, повідомлення про помилки
Адаптивність	коректне відображення на ПК і мобільних екранах
Безпека	JWT, ролі, валідація, контроль доступу
Підтримуваність	модульна структура frontend і backend
Розширюваність	можливість додавання нових ролей, статусів і фільтрів

2.5 Висновки до розділу 2

У другому розділі розглянуто існуючі рішення, обґрунтовано вибір технологій та сформовано основні вимоги до веб-додатку PetUA. Встановлено, що для поставленої задачі доцільно використовувати React, Vite, Node.js, Express, PostgreSQL, Prisma, JWT та Socket.IO. Визначені вимоги стали основою для подальшого проєктування архітектури, бази даних, API та користувацького інтерфейсу.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ

3.1 Загальна архітектура системи

Веб-додаток PetUA побудовано за клієнт-серверною архітектурою. Клієнтська частина відповідає за відображення сторінок, взаємодію з користувачем, форми, фільтри, маршрутизацію, перемикання теми та роботу з локальним станом. Серверна частина обробляє бізнес-логіку, перевіряє права доступу, взаємодіє з базою даних, зберігає файли та надає API для frontend.

Клієнт-серверна модель є доцільною для PetUA, оскільки система має не лише показувати статичний контент, а й обробляти персональні дії користувача. Реєстрація, створення оголошення, редагування профілю, завантаження фотографій, додавання до обраного, надсилання повідомлень і модерація не можуть бути реалізовані лише на стороні браузера. Вони потребують серверної перевірки та централізованого зберігання даних.

У межах архітектури виділено кілька основних потоків даних. Перший потік пов’язаний із відкритим каталогом: клієнт надсилає запит з параметрами пошуку, сервер формує умову вибірки, Prisma виконує запит до PostgreSQL, після чого клієнт отримує список оголошень і дані пагінації. Другий потік пов’язаний із захищеними діями: клієнт додає access token у заголовок запиту, сервер перевіряє токен і роль, а потім виконує дію. Третій потік пов’язаний із файлами: фотографія передається через multipart/form-data, зберігається на сервері, а шлях записується у базу даних.

Для демонстрації логіки підключення маршрутів у серверній частині у лістингу 2 наведено фрагмент маршрутизатора API. Він показує, що система не є набором випадкових endpoints, а має згруповані модулі для автентифікації, профілю, обраного, чату, скарг, модерації, власних оголошень, відкритих даних і адміністрування.

Лістинг 2 – *Групування основних API-маршрутів серверної частини*

```
router.use('/health', healthRoutes)
router.use('/auth', authRoutes)
router.use('/profile', profileRoutes)
router.use('/favorites', favoritesRoutes)
router.use('/chats', chatsRoutes)
router.use('/complaints', complaintsRoutes)
router.use('/moderation', moderationRoutes)
router.use('/my/ads', myAdsRoutes)
router.use('/', reviewsRoutes)
router.use('/', publicUsersRoutes)
router.use('/', adsRoutes)
router.use('/admin', adminRoutes)
```

На рисунку 4 показано загальну архітектуру системи (див. рис. 4). Вона відображає взаємодію між браузером користувача, React-додатком, сервером Express, базою даних PostgreSQL, файловим каталогом uploads та допоміжними модулями. Така схема є важливою для розуміння того, як окремі функції веб-додатку пов’язані між собою.

База даних PostgreSQL зберігає структуровані дані системи. Prisma використовується для опису схеми, міграцій і виконання запитів. Окремий каталог uploads застосовується для фотографій оголошень. Чат працює через Socket.IO, що доповнює REST API механізмом подій у реальному часі.

Архітектура поділена на логічні рівні: презентаційний рівень, рівень API, рівень бізнес-логіки, рівень доступу до даних і рівень зберігання. Такий поділ дозволяє розвивати окремі частини без повного переписування всієї системи.

На рисунку 4 наведено загальну архітектуру веб-додатку PetUA. Схема відображає взаємодію клієнтської частини, серверної логіки, бази даних, файлового сховища, модулів автентифікації, оголошень, чату, скарг, модерації та адміністрування.

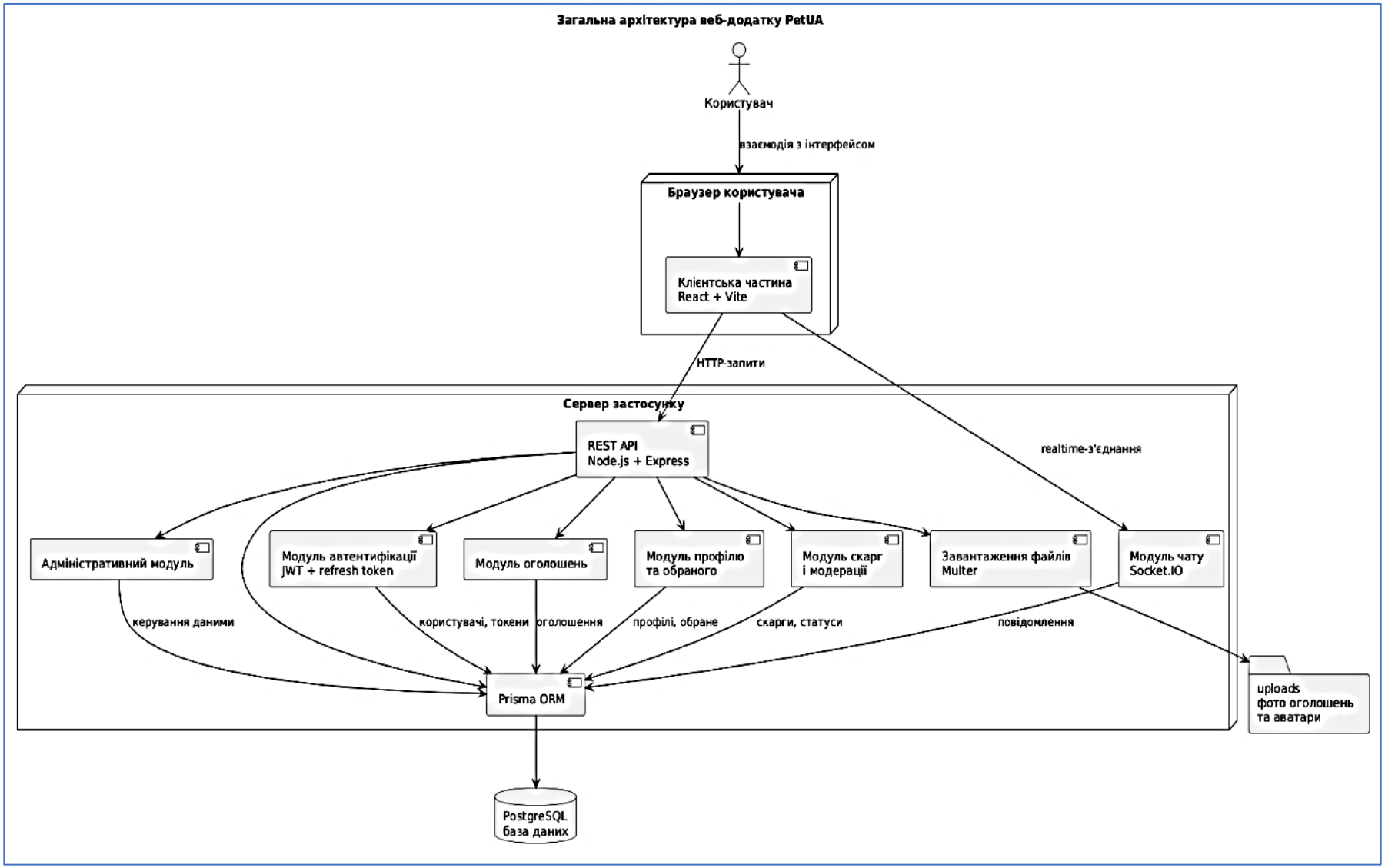


Рисунок *4 – Загальна архітектура веб-додатку PetUA*

У таблиці 3.1 наведено логічні рівні архітектури PetUA.

Таблиця *3.1 – Логічні рівні архітектури PetUA*

Рівень	Компоненти	Відповідальність
Презентаційний	React, CSS, маршрути	інтерфейс користувача
API	Express routes	приймання та обробка HTTP-запитів
Бізнес-логіка	контролери, middleware	перевірка правил і прав доступу
Дані	Prisma Client	операції з базою даних
Зберігання	PostgreSQL, uploads	дані та фотографії

3.2 Проєктування бази даних

База даних спроєктована навколо сутностей, які безпосередньо відповідають предметній області. Основними є користувач, оголошення, вид тварини, порода, фотографія, обране, діалог, повідомлення, відгук, скарга, refresh-токен і текстова сторінка. Така структура підтверджує, що PetUA є інформаційною системою, а не лише набором статичних сторінок.

Реляційна модель добре підходить для цієї системи, оскільки більшість даних має чіткі зв’язки. Один користувач може створити багато оголошень, одне оголошення належить одному виду тварини і одній породі, одне оголошення може мати багато фотографій, один користувач може мати багато записів обраного, а один діалог містить кілька учасників і повідомлень. Такі зв’язки природно описуються зовнішніми ключами.

У базі даних також відображено службові аспекти роботи системи. Refresh-токени потрібні для підтримки сесій; скарги потрібні для модерації; текстові сторінки дозволяють керувати змістом політики конфіденційності, правил і контактів; статуси користувача й оголошення допомагають

реалізувати життєвий цикл записів. Тому база даних є не лише сховищем каталогу, а основою всієї інформаційної системи.

Фрагмент опису сутності оголошення у Prisma наведено у лістингу 3. Він демонструє, що основна сутність містить як предметні поля, так і службові атрибути для статусів, модерації, лічильника переглядів і зв'язків з іншими сутностями.

Лістинг 3 – Фрагмент моделі оголошення у схемі Prisma

```
model Ad {
    id               String   @id @default(cuid())
    title            String
    slug             String   @unique
    description      String
    adType           AdType
    status           AdStatus @default(ACTIVE)
    animalGender     AnimalGender
    ageMonths        Int
    price            Decimal? @db.Decimal(10, 2)
    city             String
    region           String?
    healthInfo       String?
    vaccinationInfo  String?
    documentInfo     String?
    housingInfo      String?
    moderationFlag   ModerationFlag @default(NONE)
    moderationReason String?
    viewsCount       Int @default(0)
}
```

ER-діаграма на рисунку 5 відображає логічну структуру даних (див. рис. 5). Вона підтверджує, що веб-додаток має повноцінну модель предметної

області, а не зберігає всі дані у вигляді одного неструктурованого текстового поля.

Сутність користувача містить дані профілю, роль і службову інформацію для входу. Сутність оголошення зберігає предметні характеристики тварини, статус, автора й дату створення. Фотографії винесено в окрему сутність, що дає змогу зберігати декілька зображень для одного оголошення та реалізувати галерею.

Діалоги та повідомлення виділено окремо, оскільки комунікація між покупцем і продавцем є важливою частиною системи. Відгуки пов’язані з продавцем і автором відгуку, а скарги можуть стосуватися оголошень або повідомлень. Такий підхід дозволяє реалізувати модерацію без змішування різних типів даних. У таблиці наведено основні сутності бази даних.

Таблиця *3.2 – Основні сутності бази даних*

Сутність	Призначення
User	зберігання профілю, ролі та даних автентифікації
Ad	оголошення про тварину
AnimalType	вид тварини
Breed	порода тварини
AdPhoto	фотографії оголошення
Favorite	зв’язок користувача з обраними оголошеннями
Conversation	діалог між користувачами
Message	повідомлення в чаті
Review	відгук і рейтинг продавця
Complaint	скарга для модерації

На рисунку 5 наведено ER-діаграму основних сутностей бази даних веб-додатку PetUA. Діаграма відображає ключові інформаційні об’єкти системи та зв’язки між ними, які забезпечують роботу каталогу оголошень, профілів користувачів, обраного, внутрішнього чату, відгуків і скарг.

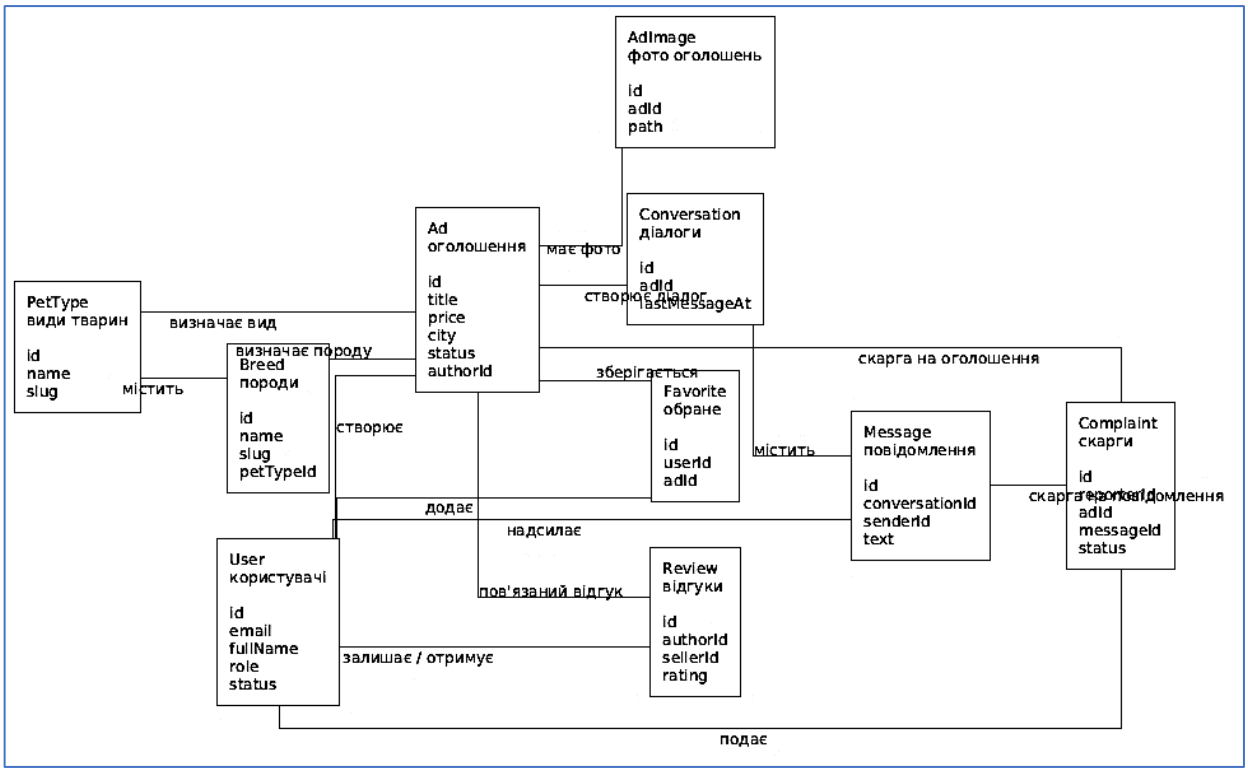


Рисунок *5 – ER-діаграма основних сутностей бази даних*

3.3 Реалізація клієнтської частини

Клієнтська частина реалізована як односторінковий веб-додаток на React. Маршрутизація забезпечує перехід між головною сторінкою, каталогом, сторінкою оголошення, профілем, сторінкою продавця, обраним, чатом, адміністративною панеллю та службовими сторінками. Для покращення зручності переходів додано автоматичне прокручування сторінки догори після зміни маршруту.

Структура клієнтської частини поділена на кілька логічних груп. У папці pages розміщено сторінки, які відповідають основним маршрутам. У папці components зберігаються повторно використовувані елементи: картки оголошень, форми, пагінація, галерея, службові компоненти захисту маршрутів і компоненти макета. Окремо виділено API-модулі, сховища стану, стилі та допоміжні функції.

Такий поділ дозволяє підтримувати код без змішування відповідальностей. Наприклад, компонент AdCard відповідає за відображення короткої картки оголошення, а сторінка AdsCatalogPage керує завантаженням списку, параметрами фільтрів і пагінацією. Компонент MarketplaceFilters відповідає за форму пошуку, а API-модуль ads.api.js приховує деталі HTTP-запитів.

Для сторінки оголошення важливим елементом стала галерея фотографій. Вона реалізована як окремий компонент ImageGallery, що отримує список зображень і керує поточним індексом перегляду. Фрагмент обробки клавіш наведено у лістингу 4.

Лістинг 4 – Обробка клавіатурної навігації у галереї фотографій

```
useEffect(() => {
    function onKey(event) {
        if (!isOpen) return
        if (event.key === 'Escape') setIsOpen(false)
        if (event.key === 'ArrowRight') {
            setIndex((value) => (value + 1) % items.length)
        }
        if (event.key === 'ArrowLeft') {
            setIndex((value) => (value - 1 + items.length)
% items.length)
        }
    }

    window.addEventListener('keydown', onKey)
    return () => window.removeEventListener('keydown',
onKey)
}, [isOpen, items.length])
```

На рисунку 6 показано сторінку детального перегляду оголошення з основними даними про тварину (див. рис. 6). На рисунку 7 наведено модальне вікно галереї, яке відкривається після натискання на фотографію (див. рис. 7). Ці два екрани демонструють завершеність сценарію перегляду: користувач бачить і структуровані характеристики, і візуальний матеріал.

Інтерфейс побудовано українською мовою. Сторінки мають єдиний візуальний стиль, світлу і темну тему, адаптивну структуру та зрозумілі

елементи керування. Каталог підтримує пошук, фільтрацію, сортування та пагінацію. Картки оголошень містять основну інформацію: назву, фото, ціну, місто, вид тварини, короткий опис і статус.

На сторінці оголошення реалізовано детальний перегляд даних про тварину. Відображаються фотографії, опис, характеристики, інформація про здоров’я, документи, щеплення, умови утримання, автор і кнопки взаємодії. Для фотографій створено окремий компонент ImageGallery, який відкриває модальну галерею після натискання на головне фото, дозволяє гортати зображення кнопками, використовує мініатюри та закривається кнопкою або клавішею Escape.

Окремо нормалізовано відображення елемента select для різних платформ, зокрема macOS. Без цього нативний вигляд поля міг відрізнятися від загального дизайну. CSS-правила прибирають стандартний вигляд браузера, задають однакову висоту, відступи, радіус, фон і власну стрілку.

На рисунку 6 наведено сторінку детального перегляду оголошення у веб-додатку PetUA. На цій сторінці користувач може переглянути фотографії тварини, основні характеристики оголошення, ціну, місцезнаходження, інформацію про продавця, а також перейти до комунікації з автором оголошення або подати скаргу на проблемний вміст.

На рисунку 7 наведено модальне вікно галереї фотографій оголошення. Галерея відкривається після натискання на основне зображення тварини та дає змогу переглядати фотографії у збільшеному вигляді, перемикати їх за допомогою кнопок навігації або клавіш клавіатури та закривати перегляд без переходу на іншу сторінку.

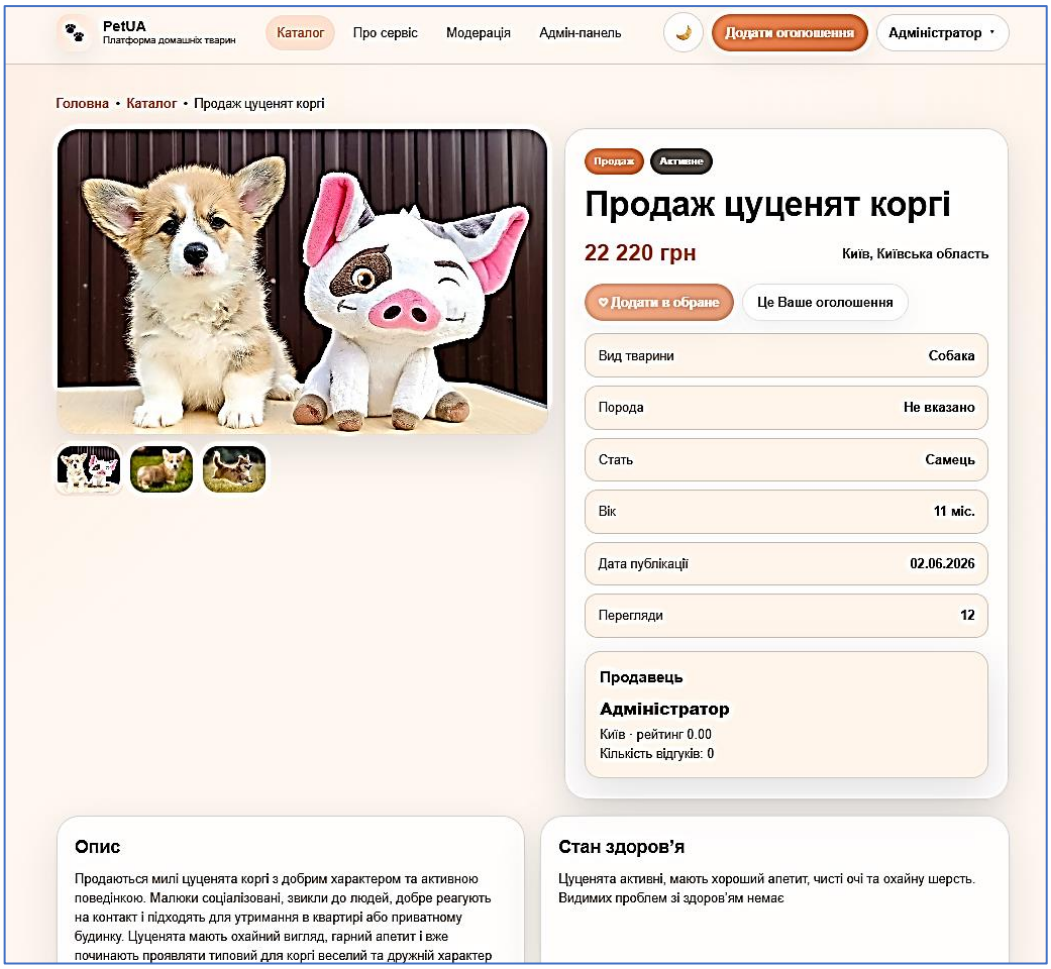


Рисунок 6 – *Сторінка детального перегляду оголошення з галереєю фотографій*



Рисунок *7* – *Модальне вікно галереї фотографій оголошення*

3.4 Реалізація серверної частини та API

Серверна частина реалізована на Express і поділена на маршрути, контролери, middleware, утиліти та сервіси роботи з даними. Такий поділ спрощує підтримку коду: маршрути описують доступні endpoints, контролери містять обробку запитів, middleware перевіряють автентифікацію й ролі, а Prisma Client виконує операції з базою даних.

У серверній частині використано типову для Express структуру, у якій кожен функціональний напрям має окремий файл маршрутів і контролер. Наприклад, публічні оголошення обробляються через ads.routes.js і ads.controller.js, власні оголошення користувача – через my-ads.routes.js і my-ads.controller.js, а модерація – через moderation.routes.js і moderation.controller.js. Це зменшує ризик надмірного зростання одного файлу та спрощує читання коду.

Схеми валідації винесено окремо. Такий підхід дозволяє перевіряти параметри запитів, тіла запитів і route-параметри до потрапляння даних у бізнес-логіку. Наприклад, при отриманні списку оголошень перевіряються параметри сторінки, ліміту, сортування, вікової групи, ціни та інших фільтрів.

Формування умов вибірки для каталогу є одним із прикладів серверної логіки. Користувач може передати кілька фільтрів одночасно, і сервер повинен коректно перетворити їх у запит до бази даних. Фрагмент такої логіки наведено у лістингу 5.

Лістинг 5 – Приклад формування умов вибірки оголошень

```
function buildAdsWhere(query) {
    const where = {
        status: { in: ['ACTIVE', 'RESERVED', 'SOLD',
'FLAGGED'] }
    }

    if (query.city) {
```

```
        where.city = { contains: query.city, mode:
'insensitive' }
    }

    if (query.petTypeSlug) {
        where.petType = { is: { slug: query.petTypeSlug } }
    }

    if (query.breedSlug) {
        where.breed = { is: { slug: query.breedSlug } }
    }

    return where
}
```

Особливістю реалізації стало врахування статусу FLAGGED у сценаріях відображення обраного та каталогу. На практиці це потрібно для коректної роботи з оголошеннями, які потребують перевірки, але не повинні повністю зникати з пов’язаних списків користувача без зрозумілого пояснення.

API забезпечує роботу з автентифікацією, користувачами, оголошеннями, видами тварин, породами, фотографіями, обраним, відгуками, скаргами, діалогами та повідомленнями. Для захищених маршрутів перевіряється access token. Дії, які стосуються ролей модератора або адміністратора, додатково обмежуються перевіркою ролі.

Завантаження фотографій виконується через серверний механізм приймання файлів. Після успішного завантаження файл зберігається у локальному каталозі uploads, а шлях до нього записується у базу даних. Це дозволяє пов’язати одне оголошення з кількома фотографіями та відображати їх у клієнтській галереї.

Для роботи з обраним реалізовано додавання, видалення та перемикання стану. Додатково передбачено обробку ситуації, коли у клієнтській частині для перемикання передано slug замість фактичного ідентифікатора оголошення: у такому випадку клієнт виконує повторне отримання оголошення та використовує реальний id. Це підвищує стійкість інтерфейсу до різних сценаріїв навігації.

У таблиці наведено основні групи API-маршрутів.

Таблиця *3.3 – Основні групи API-маршрутів*

Група	Призначення
/auth	реєстрація, вхід, оновлення токенів, вихід
/ads	каталог, створення, редагування, перегляд оголошень
/favorites	робота з обраними оголошеннями
/conversations	створення та перегляд діалогів
/messages	надсилання й отримання повідомлень
/reviews	відгуки та рейтинг продавця
/complaints	створення й обробка скарг
/admin	адміністративні функції

3.5 Реалізація ролей, модерації та безпеки

Розмежування доступу реалізовано через ролі: гість, користувач, модератор і адміністратор. Гість має доступ лише до відкритих сторінок. Користувач може створювати оголошення, спілкуватися в чаті, додавати оголошення до обраного, залишати відгуки й подавати скарги. Модератор обробляє скарги та проблемний контент. Адміністратор має повний доступ до керування системою.

Механізм ролей потрібен для запобігання неконтрольованим діям. Наприклад, звичайний користувач не повинен мати змоги змінювати статуси чужих оголошень або блокувати інших користувачів. Модератор не обов’язково повинен змінювати довідники порід або службові сторінки, якщо його задача полягає саме у перевірці скарг. Адміністратор має найширший набір прав, але навіть для нього дії повинні проходити через серверну перевірку.

Автентифікація побудована на поєднанні access token і refresh token. Access token використовується для поточних захищених запитів, а refresh token зберігається для поновлення сесії. Такий підхід зменшує необхідність

постійно вводити пароль, але дозволяє обмежити строк дії основного токена доступу.

Для контролю якості контенту реалізовано базовий аналіз тексту оголошення. Він не замінює повноцінну модерацію, але допомагає автоматично позначати підозрілі записи. Приклад перевірки повторів символів і небажаних формулювань наведено у лістингу 6.

Лістинг 6 – Фрагмент базової перевірки змісту оголошення

```
const suspiciousWordsCount = countMatches(
    text,
    /(казино|ставки|кредит|інвестиції|наркотики|зброя)/gi
)
const repeatedCharsCount = countMatches(text,
/(.)\1{5,}/gi)

if (suspiciousWordsCount > 0) {
    reasons.push('Виявлено підозрілі формулювання')
}

if (repeatedCharsCount > 0) {
    reasons.push('Виявлено неприродні повтори символів')
}
```

У результаті система може показувати власнику оголошення причину, чому запис потребує перевірки, а модератору – швидше знаходити проблемні елементи. Такий підхід є достатнім для навчального проєкту і водночас залишає простір для майбутнього розвитку, наприклад для складнішої перевірки текстів або зображень.

Модерація необхідна через специфіку предметної області. Оголошення про тварин можуть містити недостовірну інформацію, некоректні фото, заборонені контакти або образливі повідомлення. Система скарг дозволяє користувачам повідомляти про проблему, а модератору – змінювати статус скарги та реагувати на контент.

Базова безпека включає хешування паролів, перевірку токенів, використання refresh-token для продовження сесії, контроль ролей, перевірку власника ресурсу під час редагування оголошення, валідацію даних форм і

захист адміністративних маршрутів. Такий набір засобів відповідає рівню бакалаврської кваліфікаційної роботи та демонструє практичне застосування принципів безпечної веб-розробки. У таблиці *3.4* наведені права доступу користувачів.

Таблиця *3.4 – Права доступу користувачів*

Функція	Гість	Користувач	Модератор	Адміністратор
Перегляд каталогу	+	+	+	+
Створення оголошення	–	+	+	+
Чат	–	+	+	+
Скарги	–	+	+	+
Обробка скарг	–	–	+	+
Керування користувачами	–	–	–	+

3.6 Висновки до розділу 3

У третьому розділі описано проєктування та реалізацію веб-додатку PetUA. Розглянуто клієнт-серверну архітектуру, структуру бази даних, побудову клієнтської частини, API, завантаження фотографій, галерею, ролі користувачів, модерацію та базові засоби безпеки. Реалізовані рішення підтверджують, що система має повноцінну серверну логіку, структуроване зберігання даних і функції адміністрування.

Додатково у розділі показано, що реалізація має модульну структуру: окремо виділено компоненти інтерфейсу, сторінки, API-клієнти, маршрути серверної частини, контролери, схеми валідації та утиліти. Такий поділ є важливим для подальшого супроводу системи, оскільки кожен модуль відповідає за власну частину функціональності.

Наведені лістинги демонструють не окремі випадкові фрагменти, а ключові технічні рішення: маршрутизацію API, структуру основної сутності, обробку навігації у клієнтському застосунку, галерею фотографій, фільтрацію оголошень і базову перевірку контенту. Це підтверджує, що система реалізована як працездатний веб-додаток, а не лише як описова модель.

4 ТЕСТУВАННЯ, РЕЗУЛЬТАТИ ТА ОЦІНКА ПРАЦЕЗДАТНОСТІ

4.1 Підхід до тестування

Тестування веб-додатку виконано з позиції перевірки основних користувацьких сценаріїв, коректності взаємодії frontend і backend, стабільності роботи бази даних і відповідності інтерфейсу сформованим вимогам. Оскільки система працює локально, перевірка виконувалася у середовищі розробки з локальним сервером, клієнтською частиною та PostgreSQL базою даних.

Ручне тестування для цієї системи є доцільним, оскільки потрібно було перевірити не лише факт отримання відповіді від API, а й послідовність дій користувача. Наприклад, сценарій створення оголошення охоплює відкриття форми, заповнення полів, вибір виду і породи, збереження даних, перехід до перегляду, появу запису в каталозі та відображення у профілі. Помилка на будь-якому з цих етапів впливає на загальну працездатність функції.

Тестування проводилося за принципом чорної скриньки для користувацьких сценаріїв і частково за принципом білої скриньки для перевірки серверної логіки. У першому випадку перевірялася поведінка системи з боку користувача. У другому випадку аналізувалися маршрути, умови вибірки, статуси, відповіді API та зв’язки у базі даних.

Під час перевірки особливу увагу приділено сценаріям, у яких раніше виникали помилки: відкриття сторінки щойно створеного оголошення, відображення фотографій, робота з обраним, збереження сесії після перезавантаження сторінки, CORS-запити між клієнтом і сервером, адаптивність галереї та поведінка фільтрів.

Перевірялися реєстрація, авторизація, оновлення сесії, створення оголошення, редагування, завантаження фотографій, фільтрація каталогу, сортування, пагінація, додавання до обраного, відкриття сторінки оголошення,

робота галереї, створення діалогу, надсилання повідомлень, відгуки, скарги, модераційні дії та доступ до адміністративної панелі.

Окрему увагу приділено адаптивності та браузерним відмінностям. Було враховано відображення елементів форми, зокрема select на macOS, поведінку модальної галереї, кнопки перемикання фотографій, закриття через Escape, роботу мініатюр і розміщення елементів на вузьких екранах. У таблиці 4.1 наведено план перевірки основних сценаріїв.

Таблиця 4.1 – *План перевірки основних сценаріїв*

Сценарій	Очікуваний результат
Реєстрація користувача	створення профілю та можливість входу
Створення оголошення	оголошення зберігається в базі даних
Завантаження фото	файл зберігається в uploads і відображається
Фільтрація каталогу	список змінюється відповідно до параметрів
Додавання до обраного	стан обраного оновлюється для користувача
Чат	повідомлення з’являється у діалозі
Скарга	модератор бачить звернення
Галерея	фото відкриваються в модальному вікні та гортаються

4.2 Перевірка клієнтської частини

Клієнтська частина перевірялася через ручне проходження сторінок і сценаріїв. Головна сторінка повинна коректно відображати навігацію, тематичні блоки та переходи до каталогу. Каталог повинен змінювати список оголошень після введення пошукового запиту, вибору фільтрів або зміни сортування.

Форма створення оголошення перевірялася на заповнення обов’язкових полів, коректне відображення помилок і передавання даних на сервер. Сторінка оголошення перевірялася на відображення всіх структурованих полів, коректне відкриття галереї та роботу кнопок взаємодії.

Перевірка клієнтської частини також охоплювала візуальну узгодженість елементів. Було перевірено, що картки оголошень мають однакову логіку компонування, а нижній блок із продавцем і кнопкою детального перегляду прив’язаний до нижньої частини картки. Це важливо для рівного вигляду сітки каталогу, особливо коли описи оголошень мають різну довжину.

Під час тестування адаптивності перевірялися вузькі екрани, на яких звичайне горизонтальне меню стає незручним. Для таких умов передбачено бургер-меню та компактніше розташування елементів шапки. Окремо перевірено, що кнопки, поля введення, select, картки та модальні вікна не виходять за межі екрана.

На рисунку 8 наведено каталог після застосування фільтрів (див. рис. 8). Цей екран демонструє, що система не просто показує список оголошень, а дозволяє користувачу уточнювати пошук за предметними параметрами. На рисунку 9 показано форму створення оголошення (див. рис. 9), яка підтверджує наявність структурованого введення даних.

Після останніх змін було перевірено адаптивне положення кнопок галереї. Попередня реалізація розміщувала кнопки надто далеко від зображення, що могло створювати проблеми на вузьких екранах. Виправлення зменшило відступи для кнопок переходу між фотографіями, а також вирівняло формат запису кольорів у CSS.

На рисунку 8 наведено приклад роботи каталогу оголошень після застосування фільтрів. Користувач може обмежити результати пошуку за видом тварини, породою, містом, статтю, віковою групою, типом оголошення та діапазоном ціни. Такий підхід спрощує вибір тварини й дає змогу швидко відсіяти нерелевантні пропозиції.

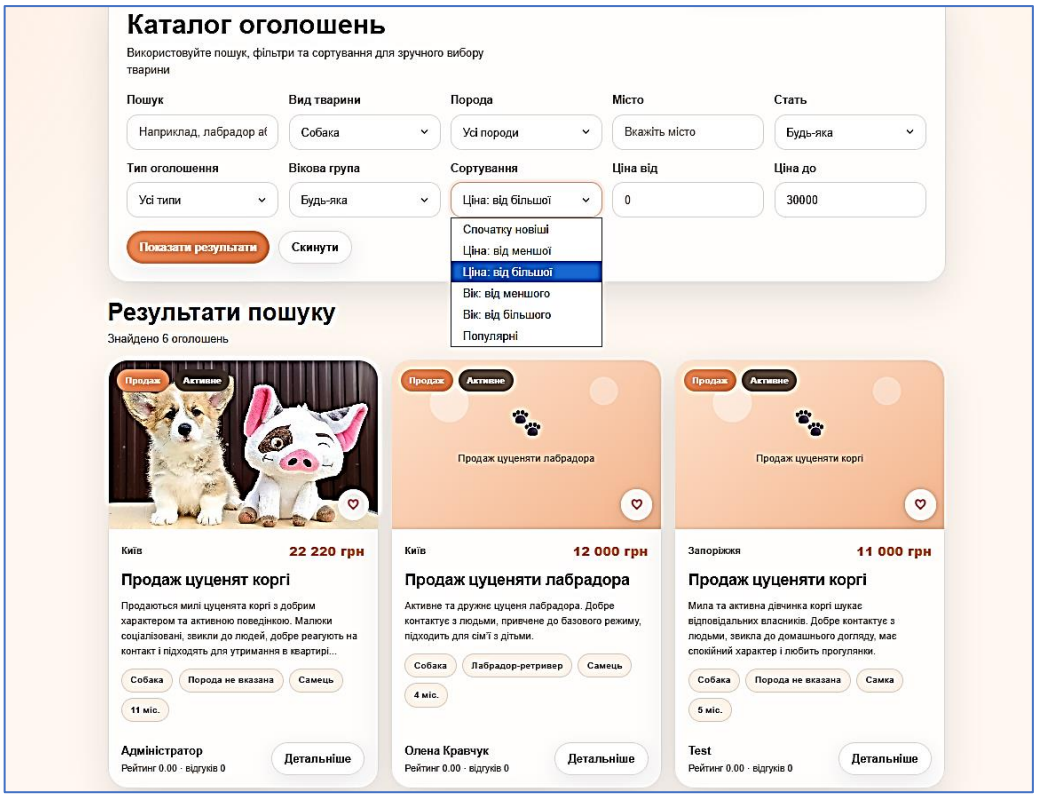


Рисунок *8 – Каталог оголошень після застосування фільтрів*

На рисунку 9 наведено форму створення оголошення про домашню тварину. Форма містить структуровані поля, необхідні для повного опису пропозиції: назву, вид тварини, породу, стать, вік, ціну, місцезнаходження, стан здоров’я, інформацію про щеплення, документи, умови утримання та фотографії. Завдяки цьому дані оголошення зберігаються у впорядкованому вигляді й надалі можуть використовуватися для пошуку, фільтрації та перегляду.

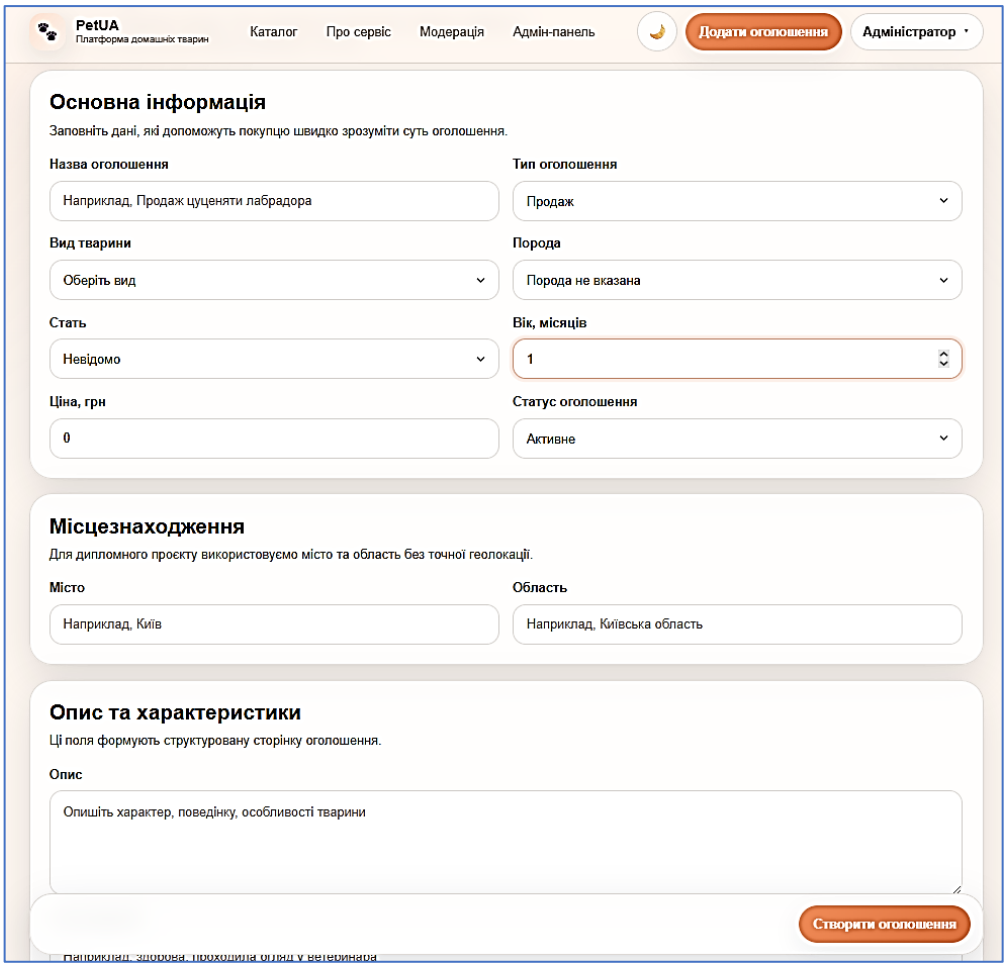


Рисунок *9 – Форма створення оголошення про домашню тварину*

4.3 Перевірка серверної частини та бази даних

Серверна частина перевірялася через виконання HTTP-запитів із клієнтської частини та аналіз відповідей API. Для успішних операцій сервер повертає дані, необхідні клієнту, а для помилок – зрозуміле повідомлення. Захищені маршрути перевіряють наявність токена та роль користувача.

Для відкритих маршрутів перевірялося отримання видів тварин, порід, списку оголошень, окремого оголошення за slug і публічних даних продавця. Для захищених маршрутів перевірялося, що запит без токена або з некоректним токеном завершується помилкою доступу, а запит з коректним токеном виконується відповідно до ролі користувача.

У базі даних перевірялися зв’язки між сутностями. Наприклад, після створення оголошення воно повинно бути пов’язане з автором, видом тварини і породою. Після завантаження фотографії створюється окремий запис з шляхом до файлу і посиланням на оголошення. Після додавання до обраного формується зв’язок між користувачем і оголошенням, який не повинен дублюватися.

Окремо перевірено роботу помилок 404 для неіснуючих оголошень. Якщо користувач переходить за неправильним slug, система повинна показати зрозуміле повідомлення, а не порожню сторінку або технічну помилку. Це особливо важливо після створення оголошення, коли клієнтська частина переходить на сторінку нового запису.

Перевірка бази даних включала створення користувачів, оголошень, фотографій, записів обраного, діалогів, повідомлень, відгуків і скарг. Важливою умовою було збереження зв’язків між сутностями. Наприклад, фотографії повинні належати конкретному оголошенню, відгук – конкретному продавцю, а повідомлення – конкретному діалогу.

Окремо перевірено роботу видимих статусів оголошень для обраного. До списку статусів, які можуть відображатися у відповідному сценарії, додано FLAGGED. Це дає змогу коректніше працювати зі станами оголошень, які вже потрапили в поле модерації, але не повинні спричиняти помилку інтерфейсу.

У таблиці 4.2 наведено– результати перевірки серверних функцій.

Таблиця *4.2 – Результати перевірки серверних функцій*

Функція	Результат перевірки
Авторизація	токени створюються, захищені маршрути працюють
Оголошення	створення, перегляд і редагування виконуються
Фотографії	завантаження та відображення працюють
Обране	додавання й перемикання стану працюють
Чат	діалоги та повідомлення зберігаються
Скарги	створення та перегляд модератором виконуються

4.4 Оцінка відповідності вимогам

Після реалізації основних модулів виконано оцінку відповідності системи сформованим вимогам. Веб-додаток підтримує ключові ролі, каталог, структуровані оголошення, пошук, фільтрацію, сторінку продавця, особистий кабінет, обране, чат, відгуки, скарги, модерацію та адміністративні функції.

4.4.1 Фізичні характеристики реалізації

Окрім функціональної перевірки, для кваліфікаційної роботи важливо оцінити фізичні характеристики створеного програмного продукту. Веб-додаток складається з клієнтської частини, серверної частини, схеми Prisma, міграцій бази даних, seed-даних, стилів і статичних файлів. Орієнтовний обсяг вихідного коду наведено у таблиці 4.3.

Таблиця *4.3 – Орієнтовний обсяг вихідного коду за складовими системи*

Складова	Основні файли	Орієнтовний обсяг
Клієнтська логіка JavaScript/JSX	сторінки, компоненти, API-модулі, сховища стану	приблизно 4300–4700 рядків
Серверна логіка JavaScript	маршрути, контролери, middleware, утиліти, socket-модуль	приблизно 3000–3400 рядків
Стилі CSS	глобальні стилі, стилі кабінету, чату, теми та адаптивності	приблизно 2500–2900 рядків
Prisma та дані ініціалізації	schema.prisma, міграції, seed.js	приблизно 1200–1500 рядків
Разом	клієнтська частина, сервер, база даних і стилі	понад 11000 рядків

Наведена оцінка є приблизною, оскільки кількість рядків може змінюватися після форматування, доопрацювання стилів або додавання нових функцій. Водночас вона показує реальний масштаб розробки: система охоплює не одну сторінку, а повний набір модулів для frontend, backend, бази даних, авторизації, файлів, чату та адміністрування.

За складом модулів проєкт включає понад десять сторінок клієнтської частини, набір повторно використовуваних компонентів, більше десяти груп серверних маршрутів, окремі контролери для адміністратора, модерації, оголошень, профілю, чату, скарг і відгуків, а також декілька міграцій бази даних. Такий обсяг відповідає практичній задачі бакалаврської роботи з інженерії програмного забезпечення.

4.4.2 Оцінка працездатності та обмежень

Система успішно виконує основні сценарії у локальному середовищі розробки. Клієнтська частина запускається через Vite, серверна частина працює на Node.js, база даних PostgreSQL використовується через Prisma. Після застосування міграцій і seed-даних користувач може пройти шлях від реєстрації до створення оголошення, перегляду каталогу, додавання до обраного та комунікації.

До обмежень поточної реалізації належить локальне зберігання фотографій у каталозі uploads. Для навчальної і демонстраційної версії це прийнятно, але для виробничого використання доцільно застосувати зовнішнє файлове сховище, резервне копіювання та додатковий контроль доступу до медіафайлів. Також у майбутньому можна додати email-підтвердження реєстрації, відновлення пароля, розширені правила модерації та автоматизовані тести.

З погляду продуктивності система працює з невеликим демонстраційним набором даних. Для збільшення обсягу каталогу доцільно додати індекси для полів, які часто використовуються у фільтрах, зокрема міста, статусу, виду тварини, породи, типу оголошення та дати публікації. Це не змінює архітектуру, але покращує масштабованість у разі збільшення кількості записів.

Система не є універсальною дошкою оголошень, оскільки модель даних і інтерфейс зосереджені на предметній області домашніх тварин. Це проявляється у наявності полів виду, породи, статі, віку, стану здоров’я,

щеплень, документів, умов утримання та фотографій. Така структура підвищує якість пошуку та зменшує залежність від довільного текстового опису.

Веб-додаток перевірено локально та розгорнуто у тестовому хмарному середовищі Render. Клієнтська частина працює як статичний сайт, серверна частина – як окремий web service, а база даних – як PostgreSQL-сховище. Під час перевірки підтверджено, що основні сторінки, каталог, авторизація, створення оголошень, завантаження фотографій, обране, чат і модераційні функції доступні не лише у локальному середовищі, а й через публічну адресу розгорнутого застосунку. Водночас використання безкоштовного тарифу Render має обмеження: перший запит після простою може виконуватися повільніше, а для довготривалого виробничого використання доцільно перейти на стабільний платний тариф, налаштувати резервне копіювання, постійне файлове сховище, моніторинг і політику обробки персональних даних. У таблиці 4.4 наведено відповідність реалізації основним вимогам.

Таблиця 4.4 – *Відповідність реалізації основним вимогам*

Вимога	Стан
Реєстрація та авторизація	реалізовано
Ролі користувачів	реалізовано
Каталог і фільтри	реалізовано
Структуровані оголошення	реалізовано
Фотографії та галерея	реалізовано
Обране	реалізовано
Чат	реалізовано
Відгуки та рейтинг	реалізовано
Скарги й модерація	реалізовано
Адміністративна панель	реалізовано
Локальна та тестова хмарна працездатність	підтверджено

4.5 Висновки до розділу 4

У четвертому розділі описано підхід до тестування, перевірку клієнтської та серверної частин, роботу бази даних і оцінку відповідності реалізованої системи вимогам. Результати перевірки показали, що веб-додаток PetUA локально працездатний, підтримує основні сценарії користувачів і відповідає поставленій задачі бакалаврської кваліфікаційної роботи.

Під час тестування підтверджено працездатність ключових сценаріїв: реєстрації, входу, оновлення сесії, створення і редагування оголошення, завантаження фотографій, роботи каталогу, фільтрів, пагінації, обраного, сторінки продавця, чату, скарг і модерації. Також перевірено виправлення, пов’язані з відображенням select на macOS, роботою галереї та переходом користувача догори сторінки після зміни маршруту.

Оцінка фізичних характеристик показала, що розробка має достатній обсяг для повноцінної інформаційної системи. У системі наявні клієнтські компоненти, серверні маршрути, контролери, схеми валідації, модель бази даних, міграції, стилі, seed-дані та службові сторінки. Це підтверджує завершеність основної реалізації та можливість подальшого розвитку проєкту.

ВИСНОВКИ

1. У кваліфікаційній роботі розв’язано задачу розробки веб-додатку для вибору та продажу домашніх тварин. Результатом роботи є спеціалізована інформаційна система PetUA, яка поєднує каталог оголошень, структуроване зберігання даних, ролі користувачів, особистий кабінет, чат, відгуки, скарги, модерацію та адміністративне керування.

2. У процесі роботи проаналізовано предметну область і наявні рішення. Встановлено, що універсальні дошки оголошень не повністю враховують специфіку вибору домашніх тварин, оскільки не забезпечують достатньо зручної предметної фільтрації, структурованого подання характеристик тварини та внутрішньої комунікації між користувачами.

3. Для реалізації системи використано стек React, Vite, Node.js, Express, PostgreSQL, Prisma, JWT, refresh token, Socket.IO та локальне зберігання фотографій. Обрані технології дали змогу побудувати клієнт-серверний веб-додаток із REST API, реляційною базою даних, автентифікацією, завантаженням файлів і підтримкою обміну повідомленнями.

4. Реалізований веб-додаток підтримує реєстрацію та авторизацію користувачів, створення й редагування оголошень, завантаження фотографій, пошук, фільтрацію, сортування, обране, перегляд сторінки продавця, чат, відгуки, скарги, модерацію та адміністраторську панель. Інтерфейс виконано українською мовою, передбачено адаптивність, світлу й темну тему та базову SEO-підготовку сторінок.

5. Тестування підтвердило працездатність основних функцій системи у локальному середовищі та після тестового розгортання на Render. Подальший розвиток PetUA доцільно пов’язати з підключенням зовнішнього файлового сховища, додаванням автоматизованих тестів, удосконаленням відновлення пароля, розширенням адміністративної статистики та покращенням правил модерації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Valente M. T. Software Engineering: A Modern Approach. 2024. URL: https://softengbook.org/ (дата звернення: 28.04.2026).

2. Nielsen J., Norman D. The Definition of User Experience (UX). Nielsen Norman Group. 1998. URL: https://www.nngroup.com/articles/definition-user-experience/ (дата звернення: 28.04.2026).

3. Search Engine Optimization (SEO) Starter Guide. Google Search Central. URL: https://developers.google.com/search/docs/fundamentals/seo-starter-guide (дата звернення: 01.05.2026).

4. React Documentation. Meta Open Source. URL: https://react.dev/ (дата звернення: 01.05.2026).

5. Vite Documentation. VoidZero Inc. URL: https://vite.dev/ (дата звернення: 01.05.2026).

6. About Node.js. OpenJS Foundation. URL: https://nodejs.org/en/about (дата звернення: 01.05.2026).

7. Express - Node.js web application framework. OpenJS Foundation. URL: https://expressjs.com/ (дата звернення: 01.05.2026).

8. PostgreSQL Documentation. PostgreSQL Global Development Group. URL: https://www.postgresql.org/docs/ (дата звернення: 01.05.2026).

9. Prisma ORM Documentation. Prisma Data, Inc. URL: https://www.prisma.io/docs (дата звернення: 01.05.2026).

10. Jones M., Bradley J., Sakimura N. RFC 7519: JSON Web Token (JWT). Internet Engineering Task Force, 2015. URL: https://datatracker.ietf.org/doc/html/rfc7519 (дата звернення: 01.05.2026).

11. Socket.IO Documentation. Socket.IO. URL: https://socket.io/docs/v4/ (дата звернення: 01.05.2026).

12. Тварини: продаж домашніх тварин, купити домашнього улюбленця. OLX Україна. URL: https://www.olx.ua/uk/zhivotnye/ (дата звернення: 01.05.2026).

13. Домашні тварини та зоотовари. RIA.com. URL: https://www.ria.com/uk/c/zhivotnyye/ (дата звернення: 01.05.2026).

14. Дошка безкоштовних оголошень про тварин Lapki.pet. URL: https://lapki.pet/uk (дата звернення: 01.05.2026).

15. Petfinder. Find your New Best Friend. URL: https://www.petfinder.com/ (дата звернення: 01.05.2026).

16. Static Site Redirects and Rewrites. Render Docs. URL: https://render.com/docs/redirects-rewrites (дата звернення: 01.05.2026).

17. Богуславський В. Розробка веб-додатку для вибору та продажу домашніх тварин. *Збірник наукових праць студентів, здобувачів PhD, докторантів і молодих вчених, університетська конференція «МОЛОДА НАУКА-2026» Том III «Секції Інженерного навчально-наукового інституту ім. Ю .М. Потебні Запорізького національного університету»* : м. Запоріжжя, 20–24 квітня 2026 р. Запоріжжя, 2026. С 84–85.

Декларація

академічної доброчесності

здобувача ступеня вищої освіти ЗНУ

Я, Богуславський Владислав Ігорович, студент 4 курсу, форми навчання денної, Інженерного навчально-наукового інституту ім. Ю.М. Потебні ЗНУ, спеціальність 121 Інженерія програмного забезпечення освітньої програми «Програмне забезпечення систем», адреса електронної пошти ohotnikegor@gmail.com, — підтверджую, що написана мною кваліфікаційна робота на тему «Розробка веб-додатку для вибору та продажу домашніх тварин» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст.42 Закону України «Про освіту», зі змістом яких ознайомлений.

- заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

- згоден на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою інтернет-системи, а також на архівування моєї роботи в базі даних цієї системи.

Дата 06.06.2026 _____ Богуславський Владислав Ігорович
(студент)

Дата 06.06.2026 _____ Безверхий Анатолій Ігорович
(керівник)