ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Веб-застосунок для підтримки комерційної діяльності фірми з продажу харчування для домашніх тварин»

(тема кваліфікаційної роботи українською мовою)

«Web application to support the commercial activities of a pet food company»

(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач денної форми навчання
спеціальності 122 Комп'ютерні науки
(код, назва спеціальності)
Освітня програма Комп'ютерні науки
(назва)
ПАНЧЕНКО Денис Станіславович
(прізвище, ім'я, по-батькові здобувача)

Керівник старш. викл. Штефан Н. З. _____
(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент к.т.н., доцент кафедри інженерії ПЗ національного університету "Одеська політехніка", Тройніна А.С.
(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій
№ ___ від ___ _____ 2025 р.

Завідувачка кафедри

_____ КАЗАКОВА Надія
(підпис) (прізвище,ім'я)

Захищено на засіданні ЕК № _____
протокол № ___ від _____ 2025 р.

Оцінка _____ / _____ / _____
(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

_____ КОПИЧЕНКО Іван
(підпис) (прізвище,ім'я)

Одеса 2025

АНОТАЦІЯ

Метою кваліфікаційної роботи є забезпечення зручного онлайн-доступу до персоналізованих рекомендацій з вибору харчування для домашніх тварин шляхом інтеграції інтернет-магазину з консультаційним сервісом у форматі чат-бота.

Методи розробки базуються на використанні модульної архітектури вебзастосунку з поділом на фронтенд (React) і бекенд (Node.js з Express), зберіганні даних у реляційній базі PostgreSQL, а також інтеграції з NLP-сервісом Dialogflow для реалізації інтелектуального чат-бота, здатного надавати рекомендації на основі аналізу вхідних користувацьких запитів.

Результатом роботи є функціональний вебзастосунок, що поєднує інтернет-магазин товарів для домашніх тварин з інтерактивним консультаційним сервісом. Система надає користувачам персоналізовані рекомендації щодо вибору корму на основі вхідних даних про тварину (тип, вік, вага, алергії) за допомогою чат-бота. Вебзастосунок має зручний інтерфейс, безпечну аутентифікацію, історію покупок та підтримку адміністрування бази знань.

Ключові слова: вебзастосунок, інтернет-магазин, чат-бот, Dialogflow, персоналізація, React, Node.js, PostgreSQL, NLP, консультаційний сервіс, модульна архітектура.

ABSTRUCT

The goal of the qualification work is to provide convenient online access to personalized recommendations for choosing pet food by integrating an online store with a chatbot-style consulting service.

The development methods are based on the use of a modular web application architecture with a division into a frontend (React) and a backend (Node.js with Express), data storage in a PostgreSQL relational database, as well as integration with the Dialogflow NLP service to implement an intelligent chatbot capable of providing recommendations based on the analysis of incoming user requests.

The result of the work is a functional web application that combines an online pet store with an interactive consulting service. The system provides users with personalized recommendations for choosing food based on input data about the animal (type, age, weight, allergies) using a chatbot. The web application has a user-friendly interface, secure authentication, purchase history, and support for knowledge base administration.

Keywords: web application, online store, chatbot, Dialogflow, personalization, React, Node.js, PostgreSQL, NLP, consulting service, modular architecture.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

ERD – Entity-Relationship diagram

JWT – JSON Web Token

NLP – Natural Language Processing

SSR – Server-Side Rendering

XML – eXtensible Markup Language

Backend – серверна частина

Frontend – клієнтська частина

Next.js – бібліотека React

Notification Module – модуль повідомлень

PostgreSQL – реляційна база даних

Redux – бібліотека React

Repository – шар роботи з даними

Sequence Diagram – діаграма послідовності

UI – інтерфейс користувача

Use Case Diagram – діаграма варіантів використання

User Module – модуль користувача

ВСТУП

Ринок товарів для домашніх тварин демонструє стабільне зростання, оскільки все більше людей ставляться до своїх улюбленців як до членів сім’ї, прагнучи забезпечити їм найкращі умови життя. Одним із ключових аспектів догляду за тваринами є правильне харчування, що безпосередньо впливає на їхнє здоров’я, активність та довголіття. Проте широкий вибір кормів, різноманітність брендів, складових і рекомендацій може ускладнювати процес вибору оптимального продукту для конкретного улюбленця.

Традиційні способи вибору корму, такі як консультації у зоомагазинах або відвідування ветеринарів, часто вимагають значного часу або можуть бути недоступними через географічні чи фінансові обмеження. Водночас онлайн-магазини пропонують широкий асортимент продукції, проте покупцям бракує персоналізованих рекомендацій, що відповідають індивідуальним потребам їхніх тварин.

У зв’язку з цим виникає потреба у сучасному рішенні, яке поєднувало б зручність онлайн-шопінгу та професійні рекомендації щодо підбору корму. Розробка веб-застосунку, що включає інтернет-магазин і інтерактивний консультаційний сервіс у форматі чатбота, є актуальним кроком у вирішенні цієї проблеми. Такий підхід дозволить не лише спростити процес вибору, а й підвищити рівень довіри клієнтів до бренду, забезпечуючи індивідуальний підхід до кожного покупця.

Метою кваліфікаційної роботи є забезпечення зручного онлайн-доступу до персоналізованих рекомендацій з вибору харчування для домашніх тварин шляхом інтеграції інтернет-магазину з консультаційним сервісом у форматі чат-бота. Система враховує індивідуальні характеристики тварини – породу, вік, вагу та стан здоров’я – для формування оптимальних пропозицій.

Об'єктом розробки є веб-застосунок для продажу харчування для домашніх тварин, який включає інтернет-магазин та інтерактивний сервіс для консультацій з підбору корму.

Предметом розробки є програмні компоненти веб-застосунку, включаючи систему управління товарами, механізм обробки замовлень, інтеграцію з платіжними та логістичними сервісами, а також реалізацію чатбота для надання рекомендацій щодо вибору корму.

1 ВИЗНАЧЕННЯ ВИМОГ ДО ОБ'ЄКТУ РОЗРОБКИ

1.1 Опис предметної області та актуальність розробки

В умовах зростання популярності онлайн-покупок та підвищення вимог до персоналізованого обслуговування впровадження веб-застосунку, що поєднує інтернет-магазин та інтелектуальну систему рекомендацій, є надзвичайно актуальним. Це дозволить не лише спростити процес вибору корму для власників тварин, а й покращити якість догляду за улюбленцями.

Довгостроковий прогноз для ринку домашніх тварин позитивний. Посилена увага до здоров'я та добробуту домашніх тварин сприяє цьому зростанню. Попит на високоякісні продукти зростає, і почалася тенденція до інноваційних іграшок, здорової їжі та екологічних іграшок. І ця тенденція є основною опорою розвитку нашої продукції [1].

Інтернет-магазин, що продає корм для домашніх тварин, пропонує широкий ринок, забезпечуючи клієнтам зручність робити покупки, не виходячи з дому. Ця зручність дозволяє власникам домашніх тварин порівнювати ціни та робити покупки, не виходячи з дому, незалежно від графіка роботи магазину. Управління онлайн-магазином зазвичай потребує менших накладних витрат порівняно з фізичною вітриною, заощаджуючи такі витрати, як оренда, комунальні послуги та персонал.

Гнучкість і масштабованість також є ключовими перевагами онлайн-бізнесу домашніх тварин, оскільки вони надають ефективні маркетингові можливості. Цифровий маркетинг можна використовувати для націлювання на конкретні демографічні показники та інтереси, а інструменти аналітики дають цінну інформацію про поведінку клієнтів і тенденції продажів.

Термін «Веб-сайти з каталогами кормів для домашніх тварин» зазвичай стосується онлайн-платформ або баз даних, які перераховують і класифікують різні види кормів для домашніх тварин, бренди та супутні товари. Ці веб-сайти часто надають детальну інформацію про інгредієнти корму для домашніх тварин, поживний вміст, відгуки клієнтів, а іноді навіть порівняння цін. Вони

можуть бути цінним ресурсом для власників домашніх тварин, які хочуть знайти найкращі варіанти їжі для своїх домашніх тварин на основі конкретних дієтичних потреб, уподобань або бюджету [2].

Веб-сервіси з продажу харчування для тварин стають все більш популярними через кілька ключових тенденцій на ринку. Розглянемо кілька причин, чому ці сервіси актуальні:

1. Зростання кількості домашніх тварин.

Кількість людей, які утримують домашніх тварин, постійно зростає. Це стосується не лише собак і котів, а й інших тварин, таких як птахи, гризуни, рептилії тощо. Люди стають дедалі більш відповідальними до здоров'я своїх улюбленців, що підвищує попит на якісне та збалансоване харчування.

2. Зручність онлайн-покупок.

Все більше споживачів обирають покупки в Інтернеті через зручність і швидкість процесу. Вебсервіси дозволяють користувачам вибирати серед великого асортименту продуктів, без необхідності відвідувати магазини, що особливо актуально для людей з обмеженим часом або доступом до фізичних точок продажу.

3. Персоналізовані рекомендації.

Онлайн-сервіси часто використовують алгоритми для надання рекомендацій щодо харчування, що можуть враховувати вік, породу, розмір, стан здоров'я тварини та інші фактори. Це дозволяє власникам тварин отримати найкраще харчування для своїх улюбленців без необхідності консультуватися з ветеринаром кожного разу.

4. Важливість здорового харчування для тварин.

Власники тварин усе більше усвідомлюють важливість правильного харчування для здоров'я своїх улюбленців. Це включає вибір продуктів без консервантів, з натуральними інгредієнтами або спеціалізованих кормів для тварин з алергіями чи іншими проблемами зі здоров'ям.

5. Збільшення асортименту та різноманіття.

Онлайн-магазини часто мають широкий асортимент продуктів, включаючи преміум-клас, органічне або спеціалізоване харчування, що може бути недоступним у звичайних магазинах.

6. Автоматизація замовлень та підписки.

Багато веб-сервісів дозволяють налаштувати регулярні замовлення через підписки, що забезпечує зручність і економію часу для власників тварин, які регулярно потребують корму.

7. Наскрізна доставка та знижки.

Онлайн-платформи часто пропонують доставку на дім, що особливо цінується у великих містах. Також зазвичай є знижки, акції та програми лояльності для постійних клієнтів.

Ці фактори роблять веб-сервіси для продажу харчування для тварин важливою та динамічною частиною ринку. Веб-сервіси надають зручні, персоналізовані та доступні варіанти для задоволення потреб власників тварин у правильному харчуванні своїх улюбленців.

Автоматизація підбору на основі даних про тварину підвищить точність рекомендацій, зменшить кількість помилкових покупок та сприятиме зростанню рівня довіри клієнтів до бренду. Таким чином, розробка такого сервісу стане важливим кроком у розвитку ринку зоотоварів та підвищенні рівня комфорту для власників домашніх тварин.

1.2 Огляд існуючих аналогів

На першому етапі розробки такого сервісу, перед проектуванням системи, завжди корисно провести огляд існуючи рішень або конкурентів. Це дасть змогу зрозуміти ринок, тобто можливість зібрати інформацію як виглядають поточні рішення, що вже є на ринку, і що з них користується популярністю. Це допоможе уникнути дублювання існуючих функцій і зрозуміти, чого не вистачає в поточних сервісах.

Огляд конкурентів дозволить виявити переваги і недоліки існуючих сервісів. Це дасть можливість створити продукт, що не тільки відповідає потребам ринку, а й перевершує конкурентів у деяких аспектах. Для розробника необхідно визначити можливості для додавання інноваційних або унікальних функцій, які не присутні в конкурентних рішеннях.

Оцінка користувацького досвіду (UX) також заслуговує уваги розробника на перших етапах роботи над створенням веб-сервісу. Аналіз інших сервісів дасть можливість вивчити, як вони взаємодіють із користувачами, як організований процес покупки, навігація, оформлення замовлення та інше. Це допоможе створити більш зручний та привабливий інтерфейс для ваших користувачів.

Огляд конкурентних сервісів дозволить зрозуміти, які технології вони використовують для реалізації своїх рішень, та які бізнес-моделі (наприклад, підписка, разова оплата, додаткові послуги) вони застосовують. Аналіз відгуків та рецензій користувачів на конкурентні платформи дасть уявлення про те, що саме потребують споживачі і як можна задовольнити ці потреби.

Враховуючи ці аспекти, проведення огляду аналогів є важливою частиною етапу планування перед розробкою проекту. Це дозволяє мінімізувати ризики і створити продукт, який буде успішно конкурувати на ринку.

Отже, розглянуто декілька існуючих аналогів та проаналізовано їх сильні та слабкі сторони. Перший аналог «eWorldTrade» [3] (рис. 1.1). Це відомий ринок B2B для промисловості кормів для домашніх тварин. Платформа забезпечує глобальну торговельну мережу, яка з’єднує виробників, постачальників і дистриб’юторів кормів для домашніх тварин із покупцями з усього світу.

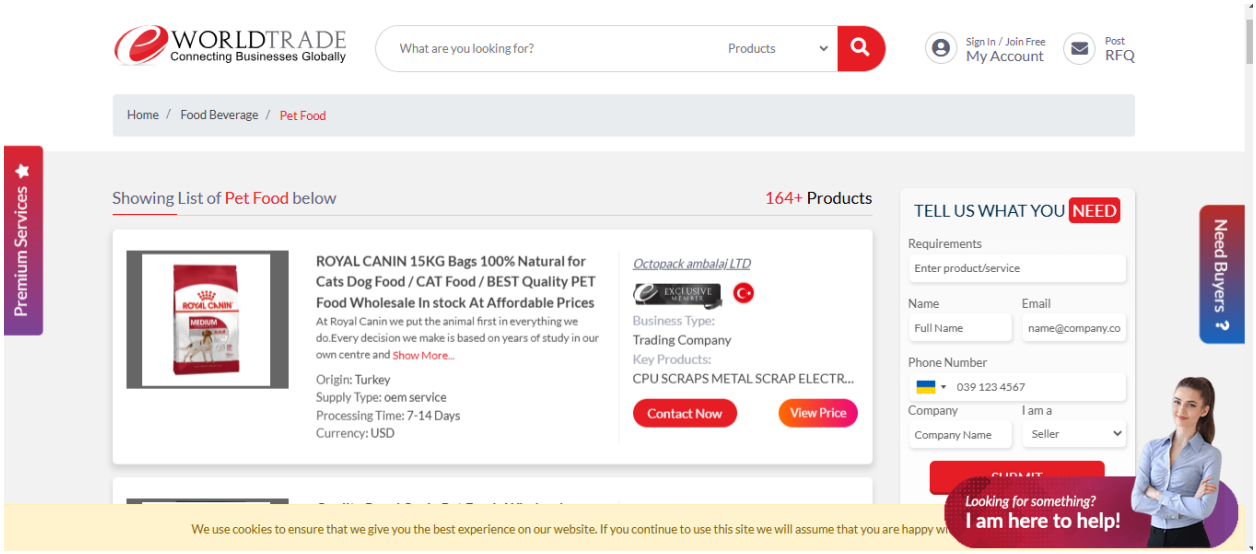

Рисунок 1.1 – Скріншот головної сторінки вебсервісу «eWorldTrade»

Переваги вибору «eWorldTrade»:

1. eWorldTrade пропонує широкий спектр продуктів і послуг, пов’язаних з промисловістю кормів для домашніх тварин, включаючи інгредієнти кормів для домашніх тварин, упаковку кормів для домашніх тварин, обладнання для обробки тощо.

2. Ще однією перевагою використання eWorldTrade є його пошукова система продуктів на основі ШІ. Це дозволяє користувачам швидко та легко знаходити потрібні їм продукти.

3. Ще однією перевагою eWorldTrade є програма перевірених постачальників, яка гарантує, що всі постачальники на платформі є законними та надійними. Це допомагає запобігти шахрайству та захистити покупців від шахрайства, забезпечуючи їм спокій під час ведення бізнесу на платформі.

Відомі корми для домашніх тварин, доступні в eWorldTrade. На eWorldTrade користувачам легко знайти оптові товари за оптовою ціною, зокрема корми для домашніх тварин, наприклад закуски для собак, здорові напої та дієтичне ветеринарне харчування.

Наступний аналог – це веб-сервіс «DogFoodAdvisor» (рис. 1.2), нішевий каталог, присвячений виключно кормам для собак. Сайт було створено, щоб

допомогти власникам собак приймати зважені рішення щодо їжі, яку вони дають своїм улюбленцям. «DogFoodAdvisor» пропонує докладні огляди та рейтинги різних брендів кормів для собак, приділяючи значну увагу якості та безпечності інгредієнтів. Сайт оцінює кожен продукт на основі комплексного аналізу його інгредієнтів, поживної цінності та потенційних ризиків для здоров'я.

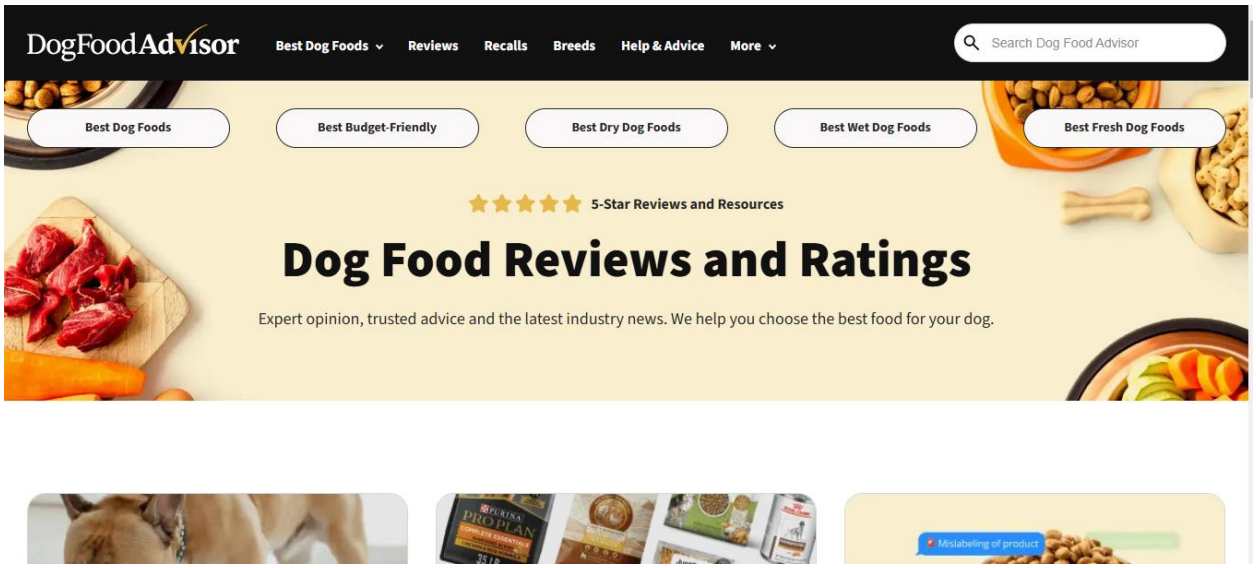

Рисунок 1.2 – Скріншот головної сторінки сервісу «DogFoodAdvisor»

«DogFoodAdvisor» також відомий своєю актуальною інформацією про відкликання кормів для собак, гарантуючи, що користувачі інформуються про будь-які проблеми з безпекою, щойно вони виникають. Крім того, на сайті проводяться форуми спільноти, де користувачі можуть обговорювати теми, пов'язані з харчуванням собак, і ділитися своїм досвідом.

Ключові характеристики:

1. Детальні огляди та рейтинги брендів кормів для собак, зосереджені на якості інгредієнтів, поживній цінності та безпеці.

2. Комплексний аналіз інгредієнтів, включаючи виявлення потенційних алергенів або шкідливих речовин.

3. Регулярні оновлення щодо відкликань кормів для собак, надаючи своєчасну інформацію щодо проблем безпеки.

4. Форуми спільноти, де користувачі можуть брати участь у дискусіях, ставити запитання та ділитися порадами щодо харчування собак.

Плюси даного веб-сервісу:

– дуже докладні та добре досліджені огляди, які дають глибоке розуміння продуктів харчування для собак;

– регулярні оновлення щодо відкликань і галузевих новин, що інформує користувачів про потенційні ризики;

– активні форуми спільноти пропонують додаткові перспективи та поради від інших власників собак.

Недоліки:

– сайт зосереджений виключно на кормах для собак, що обмежує його корисність для власників інших домашніх тварин;

– рівень деталізації може бути надзвичайно високим для користувачів, які шукають швидких або простих рекомендацій.

«Pet Food Sherpa» це вичерпний каталог кормів для домашніх тварин, який наголошує на прозорості процесу пошуку інгредієнтів і виробництва (рис. 1.3). Платформа має на меті розширити можливості власників домашніх тварин, надаючи їм детальну інформацію про продукти, якими вони годують своїх домашніх тварин.

«Pet Food Sherpa» охоплює широкий спектр кормів для домашніх тварин, пропонуючи детальні списки інгредієнтів, аналіз поживності та відомості про те, звідки та як отримані інгредієнти. На сайті також є інструмент порівняння, який дозволяє користувачам оцінювати кілька брендів поруч на основі таких ключових факторів, як якість інгредієнтів, ціна та поживний вміст [4].

Крім того, «Pet Food Sherpa» регулярно публікує дописи в блогах і статті про тенденції в харчуванні домашніх тварин, надаючи користувачам експертну думку та останні дослідження.

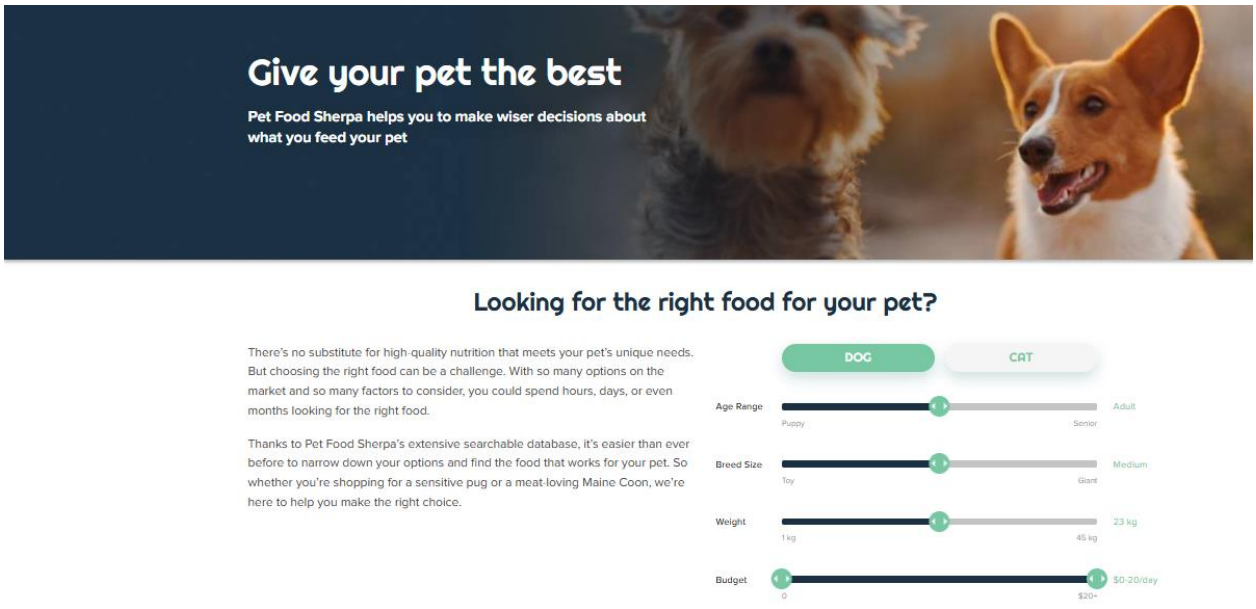

Рисунок 1.3 – Скріншот головної сторінки сервісу «Pet Food Sherpa»

Ключові характеристики «Pet Food Sherpa»:

– прозорість процесів пошуку та виробництва інгредієнтів, надання користувачам інформації про те, як і звідки беруться інгредієнти корму для домашніх тварин;

– детальний аналіз вмісту поживних речовин у широкому асортименті кормів для домашніх тварин;

– інструмент порівняння, який дозволяє користувачам оцінювати кілька брендів поруч;

– регулярні оновлення блогу та статті про тенденції в харчуванні домашніх тварин, які пропонують експертну думку та останні дослідження.

До від'ємностей цього веб-сервісу слід віднести наступні покажчики:

– акцент на прозорості та пошуку інгредієнтів, надання користувачам цінної інформації про продукти, які вони вибирають;

– інструмент порівняння є корисною функцією для користувачів, які хочуть оцінити різні марки;

– регулярні оновлення та висновки експертів допомагають користувачам бути в курсі тенденцій у харчуванні домашніх тварин.

До недоліків слід віднести наступне:

– сайт в основному зосереджений на преміальних і популярних брендах, з меншим охопленням нішевих або нових продуктів;

– деяка інформація може бути занадто технічною для випадкових користувачів, які не знайомі з харчуванням домашніх тварин.

1.3 Опис функціональних вимог

Функціональні вимоги для такого веб-застосунку можна поділити на кілька категорій відповідно до ключових компонентів системи: інтернет-магазин, консультаційний сервіс (чатбот) та управління користувачами [5]. Для цього випадку зручно використати діаграму варіантів використання, де акторами виступають:

– клієнт – користувач, який купує корм і отримує рекомендації;

– чатбот – віртуальний асистент для консультацій;

– адміністратор – керує товарами, обробляє замовлення.

Для основних варіантів використання нижче надано опис: назва, актори, основний сценарій та альтернативні сценарії (якщо вони мають місце бути). Діаграму варіантів використання представлено на рис. 1.3.

ВВ№1 «Перегляд каталогу та пошук товару»

Актор: Клієнт

Основний сценарій:

1. Клієнт заходить на сайт/у застосунок.
2. Обирає категорію кормів або використовує пошук.
3. Переглядає список товарів, може застосовувати фільтри (за брендом, віком тварини тощо).
4. Відкриває сторінку товару для перегляду деталей.

Альтернативні сценарії:

А1 «Каталог порожній»: якщо немає товарів за заданими фільтрами, система повідомляє клієнта.

А2 «Немає в наявності»: якщо товару немає, клієнту пропонуються альтернативи.

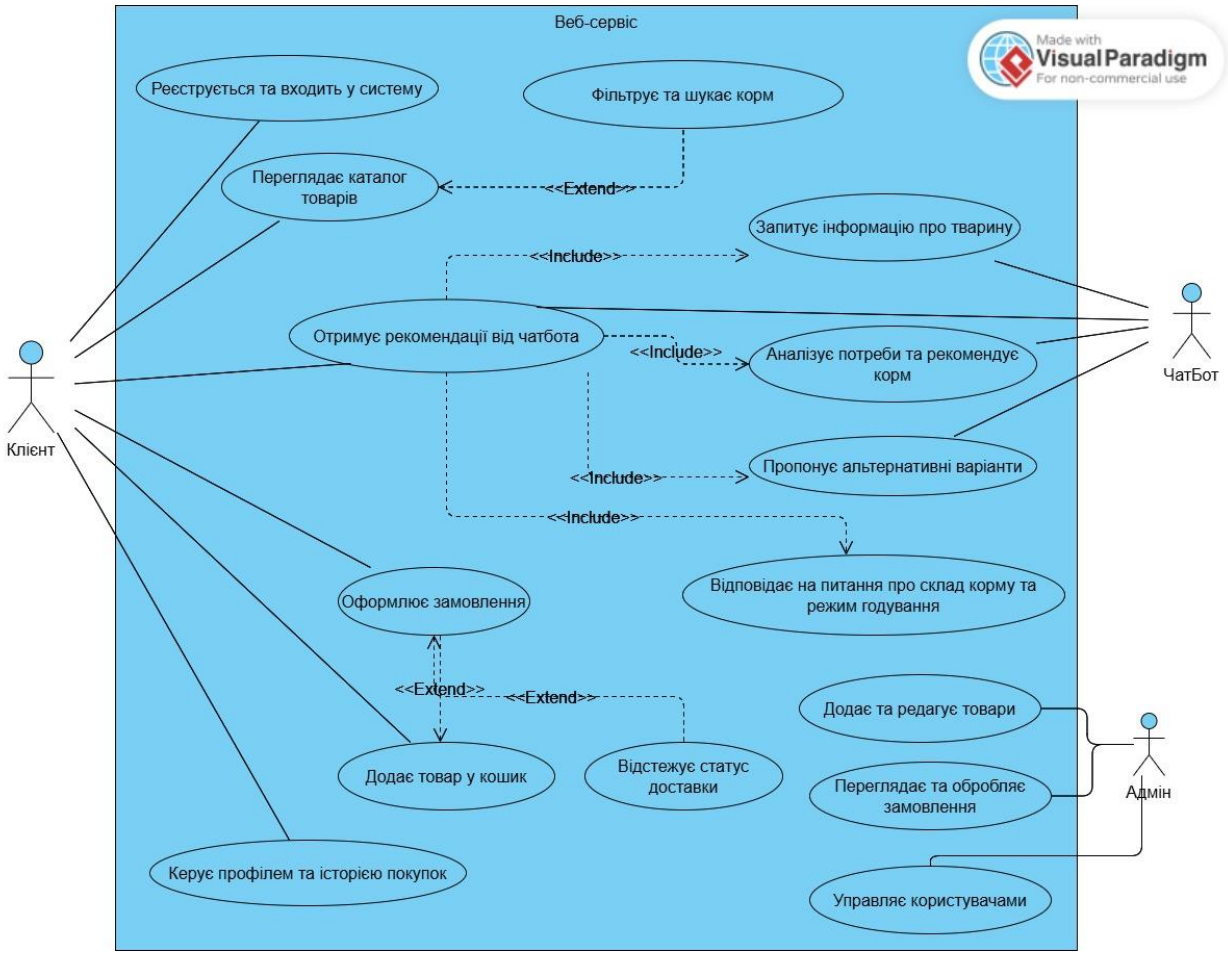

Рисунок 1.4 – Діаграма варіантів використання для веб-сервісу

ВВ№2 «Отримання рекомендацій від чатботу»

Актори: Клієнт, Чатбот.

Основний сценарій:

1. Клієнт запускає чатбота.
2. Чатбот запитує інформацію про тварину (порода, вік, вага, алергії тощо).
3. Клієнт вводить відповіді.
4. Чатбот аналізує дані та рекомендує корм.

5. Клієнт отримує список відповідних товарів з посиланнями на сторінки магазину.

Альтернативні сценарії:

А1 «Недостатньо інформації»: чатбот запитує уточнюючі дані, якщо введено неповну інформацію.

А2 «Немає відповідного корму»: чатбот пропонує клієнту звернутися до консультанта або залишити контакт для зворотного зв’язку.

А3 «Помилка в даних»: якщо клієнт вводить некоректні дані (наприклад, вага у від’ємному значенні), бот просить виправити.

ВВ№3 «Додавання товару в кошик і оформлення замовлення»

Актор: Клієнт

Основний сценарій:

1. Клієнт відкриває сторінку товару.
2. Натискає кнопку «Додати в кошик».
3. Переходить у кошик і перевіряє замовлення.
4. Натискає «Оформити замовлення».
5. Вибирає спосіб оплати та доставки.
6. Підтверджує замовлення та отримує сповіщення.

Альтернативні сценарії:

А1 «Недостатньо товару на складі»: клієнт отримує попередження та може вибрати альтернативний товар.

А2«Користувач не авторизований»: система пропонує зареєструватися/увійти для збереження історії замовлень.

А3 «Помилка оплати»: якщо оплата не проходить, клієнт отримує сповіщення і може вибрати інший спосіб оплати.

ВВ№4 «Керування каталогом товарів»

Актор: Адміністратор

Основний сценарій:

1. Адміністратор входить у систему.
2. Відкриває панель управління товарами.

3. Додає новий товар або редагує існуючий.
4. Вказує детальну інформацію: назву, склад, категорію, ціну, фото тощо.
5. Зберігає зміни.

Альтернативні сценарії:

А1 «Помилка введення даних»: система перевіряє правильність заповнених полів (наприклад, ціна не може бути від’ємною).

А2 «Товар вже існує»: система попереджає, якщо намагаються додати дублікат.

А3 «Товар не відображається»: якщо товар тимчасово недоступний, адміністратор може приховати його в каталозі.

ВВ№5 «Управління замовленнями»

Актор: Адміністратор.

Основний сценарій:

1. Адміністратор переглядає список нових замовлень.
2. Змінює їхній статус (в обробці → відправлено → доставлено).
3. Відправляє клієнту повідомлення про оновлення статусу.

Альтернативні сценарії:

А1 «Клієнт скасував замовлення»: адміністратор отримує запит на скасування та повертає кошти.

А2 «Проблеми з доставкою»: якщо доставка затримується, адміністратор зв’язується з клієнтом і пропонує альтернативи.

ВВ№6 «Запитує інформацію про тварину»

Актори: Чатбот, Клієнт.

Основний сценарій:

1. Клієнт відкриває чатбота у веб-застосунку або месенджері.
2. Чатбот вітає клієнта та запитує, чи хоче він отримати персоналізовану рекомендацію щодо корму.
3. Якщо клієнт погоджується, чатбот починає послідовно ставити питання:

- тип тварини (кіт, собака, інше);
- порода;
- вік (у місяцях або роках);
- вага (в кг);
- рівень активності (низький, середній, високий);
- чи є алергії або особливі потреби (спеціальне харчування, хвороби).

4. Клієнт вводить відповіді, чатбот перевіряє коректність введених даних.
5. Якщо всі дані отримані, чатбот переходить до аналізу інформації та підбору відповідного корму.

Альтернативні сценарії:

А1 «Клієнт вводить некоректні дані»: чатбот повідомляє про помилку та просить ввести коректні дані.

А2 «Клієнт не знає точну інформацію»: чатбот пропонує варіанти «Приблизно» або «Не знаю» та підбирає корм за середніми показниками.

А3 «Клієнт відмовляється надавати інформацію»: чатбот пропонує стандартні категорії кормів або рекомендує звернутися до менеджера.

А4 «Клієнт залишає діалог без відповіді»: через певний час чатбот може надіслати нагадування або запитати, чи потрібна допомога.

ВВ№7 «Керування профілем та історією покупок»

Актори: Клієнт

Основний сценарій:

1. Клієнт входить у систему (або реєструється, якщо ще не має облікового запису).
2. Переходить до розділу «Мій профіль».
3. Може змінювати особисті дані (ім’я, адресу доставки, контактний номер, email).
4. Відкриває розділ «Історія покупок».

5. Переглядає список своїх минулих замовлень із деталями (дата, сума, статус доставки, список товарів).

6. За бажанням може повторити замовлення, натиснувши кнопку «Купити знову».

7. Якщо замовлення ще не відправлено, клієнт може скасувати його або змінити адресу доставки.

Альтернативні сценарії:

А1 «Клієнт вводить некоректні дані» (наприклад, email без @): система повідомляє про помилку та просить виправити введені дані.

А2 «Клієнт хоче видалити акаунт»:

- система запитує підтвердження та попереджає про втрату всіх даних (історії покупок, бонусів тощо);
- після підтвердження обліковий запис видаляється.

А3 «Клієнт хоче відстежити статус замовлення»:

- у розділі «Історія покупок» відображається поточний статус кожного замовлення (очікує оплату, обробляється, відправлено, доставлено);
- якщо доставка підтримує відстеження, клієнт отримує трек-номер та посилання на службу доставки.

А4 «Клієнт забув пароль»:

- на сторінці входу натискає «Забули пароль?»;
- вводить email для відновлення;
- отримує посилання на скидання пароля та вводить новий пароль.

1.4 Нефункціональні вимоги до веб-застосунку

До нефункціональних вимог веб-застосунку для продажу корму та консультацій клієнтів слід віднести в першу чергу продуктивність, а саме час відкриття сторінки каталогу товарів не повинен перевищувати 2 секунд. Крім

цього відповідь чатбота на запит користувача має надходити протягом 1 секунди [5].

Наступний покажчик – це масштабованість: архітектура повинна дозволяти горизонтальне масштабування для обробки великої кількості користувачів, крім цього вона повинна бути можливість розширення каталогу товарів без зниження продуктивності.

З точки зору зручності використання (юзабіліті):

– інтерфейс повинен бути адаптивним для всіх типів пристроїв (мобільні, планшети, ПК);

– навігація має бути інтуїтивно зрозумілою;

– чатбот повинен мати простий та зрозумілий діалог з можливістю повернення до попередніх кроків.

Код має бути структурованим та добре документованим. Адміністратор повинен мати панель керування для оновлення каталогу товарів без потреби змінювати код. Модульна архітектура для легкого оновлення та додавання функцій.

1.5 Постановка завдання

Після огляду предметної області та аналізу існуючих аналогів йде етап формулювання функціональних та нефункціональних вимог, визначення цільової аудиторії. Наступник крок – це постановка задач для реалізації веб-сервісу:

1. Проектування архітектури:
   - визначення структури веб-застосунку (Frontend, Backend, БД);
   - розробка UML-діаграм (варіантів використання, класів, послідовностей);
   - визначення API для взаємодії між компонентами системи.

2. Вибір інструментів розробки:
   - Frontend;
   - Backend;
   - база даних;
   - Чатбот.
3. Розробка бази даних.
4. Розробка чатбота (консультаційного сервісу):
   - навчання бота;
   - інтеграція NLP (Natural Language Processing);
   - інтеграція чатбота у веб-сервіс.
5. Розробка веб-сервісу:
   - розробка клієнтської частини (Frontend);
   - розробка серверної частини (Backend).
   - адміністративна панель.
6. Тестування сервісу:
   - функціональне тестування;
   - тестування UI/UX.

2 ПРОЕКТУВАННЯ ВЕБ-ЗАСТОСУНКУ

2.1 Графік планування робот

Для формування графіку виконання етапів робот над створенням веб-сервісу використана діаграма Ганта – це інструмент для візуалізації графіка виконання проекту, який показує завдання, терміни їх виконання, залежності між ними та ключові етапи. Вона використовується в управлінні проектами для планування та контролю роботи.

Вона дозволяє виконати планування етапів розробки, розбити проєкт на логічні завдання (проектування, розробка, тестування тощо) та визначити дати початку і завершення кожного етапу [6]. Це допомагає слідкувати за часом та контролювати процес. Структуру декомпозиції робіт з зазначенням проміжків часу на виконання кожного етапу представлено в табл. 2.1.

Таблиця 2.1 – Структура декомпозиції робіт

№	Назва задачі	Початок	Завершення
1	Специфікація вимог	03.02.2025	20.02.2025
1.1	Характеристика предметної області та аналіз актуальності розробки	03.02.2025	06.02.2025
1.2	Аналіз конкурентних рішень	07.02.2025	13.02.2025
1.3	Функціональні вимоги	14.02.2025	18.02.2025
1.4	Нефункціональні вимоги	19.02.2025	20.02.2025
2	Проектування архітектури	21.02.2025	10.03.2025
2.1	Визначення структури вебсервісу	21.02.2025	25.02.2025
2.2	Розробка UML-діаграм	26.02.2025	05.03.2025
2.3	Визначення API для взаємодії між компонентами системи	06.03.2025	10.03.2025

Продовження таблиці 2.1

№	Назва задачі	Початок	Завершення
3	Вибір інструментів розробки	11.03.2025	24.03.2025
3.1	Frontend і Backend	11.03.2025	13.03.2025
3.2	База даних	14.03.2025	18.03.2025
3.3	Бібліотеки для Чатботу	19.03.2025	24.03.2025
4	Розробка бази даних	25.03.2025	04.04.2025
5	Розробка чатбота	07.04.2025	22.04.2025
5.1	Навчання бота	07.04.2025	11.04.2025
5.2	Інтеграція NLP	14.04.2025	17.04.2025
5.3	Інтеграція чатбота у веб-сервіс	18.04.2025	22.04.2025
6	Розробка веб-сервісу	23.04.2025	19.05.2025
6.1	Розробка клієнтської частини	23.04.2025	02.05.2025
6.2	Розробка серверної частини	05.05.2025	09.05.2025
6.3	Адміністративна панель	12.05.2025	19.05.2025
7	Тестування веб-сервісу	20.05.2025	
7.1	Функціональне тестування	20.05.2025	26.05.2025
7.2	Тестування UI/UX	27.05.2025	30.05.2025

На рисунку 2.1 представлено діаграму Ганта, яка допомагає розробникам у візуалізації залежностей між завданнями. Наприклад, розробку API не можна почати, поки не спроектовано базу даних. Крім цього, вона служить для контролю термінів виконання робот, тут легко відстежувати, чи проєкт рухається за графіком.

Якщо одна задача затримується, можна оцінити вплив на весь проєкт. Також корисно описати основні ризики під час роботи над розробкою веб-сервісу.

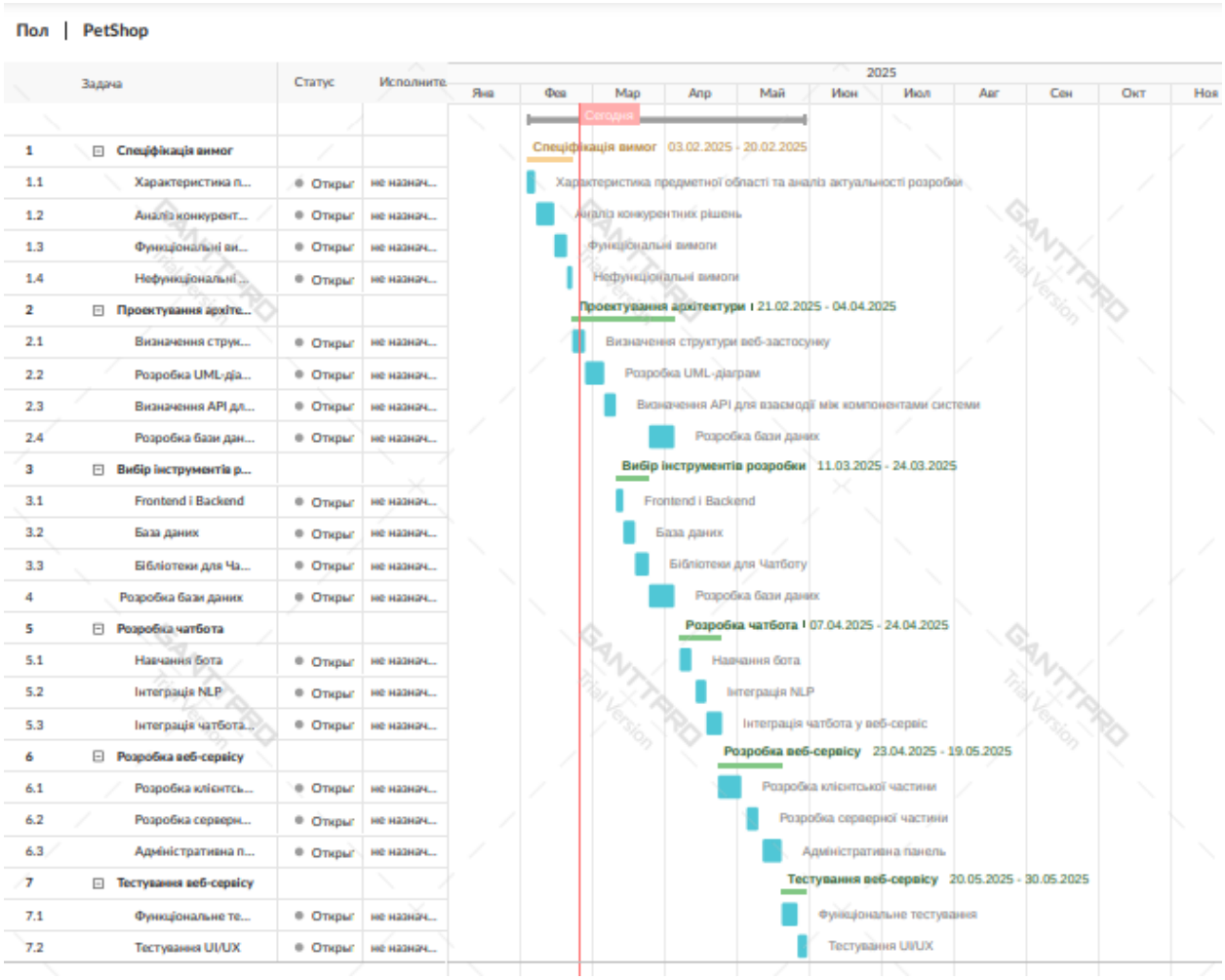

Рисунок 2.1 – Діаграма Ганта для розробки веб-сервісу

2.2 Вибір архітектури для веб-застосунку

Архітектура веб-застосунку визначає структуру компонентів, взаємодію між ними та принципи їх роботи. У цьому проєкті оптимальним буде клієнт-серверний підхід з мікросервісними або модульними рішеннями. Модульний підхід дозволяє розділити систему на незалежні частини, які взаємодіють між собою через API [7]. Це підвищує гнучкість, масштабованість і зручність підтримки. Архітектура містить наступні модулі:

1. Модуль користувача (User Module), функціями якого є реєстрація/авторизація користувачів (JWT), управління профілем (редагування, зміна пароля), перегляд історії покупок та взаємодія з чатботом.

З точки зору взаємодії з іншими модулями:

– надсилає запити до модуля БД для збереження інформації;

– отримує відповіді від модуля замовлень щодо історії покупок;

– взаємодіє з модулем чату для персоналізованих консультацій.

2. Модуль каталогу товарів (Product Module). До його функцій входить управління товарами (CRUD), фільтрація та пошук корму за параметрами, оновлення даних про наявність та ціну.

Взаємодія з іншими модулями:

– отримує дані від модуля БД;

– взаємодіє з модулем адмін-панелі для редагування товарів;

– передає інформацію чатботу для консультацій.

3. Модуль замовлень (Order Module): оформлення замовлення та обробка статусів (очікується, оплачене, відправлене), а також інтеграція з платіжними системами

Взаємодія з іншими модулями наступна:

– використовує модуль БД для збереження замовлень;

– отримує платежі від модуля оплати;

– надсилає повідомлення в модуль повідомлень.

4. Модуль повідомлень (Notification Module): надсилання email, SMS або push-повідомлень, оповіщення про зміну статусу замовлення та нагадування про знижки або акції.

Взаємодія з іншими модулями:

– отримує дані від модуля замовлень;

– надсилає повідомлення через зовнішні сервіси;

– використовується чатботом для оповіщень.

5. Адмін-панель (Admin Panel Module): управління каталогом товарів, перегляд замовлень та статусів та аналіз статистики продажів.

Взаємодія з іншими модулями:

– взаємодіє з модулем каталогу для редагування товарів;

– отримує дані про замовлення з модуля БД;

– генерує аналітику для модуля аналітики.

На рисунку 2.2 представлено архітектуру застосунку.

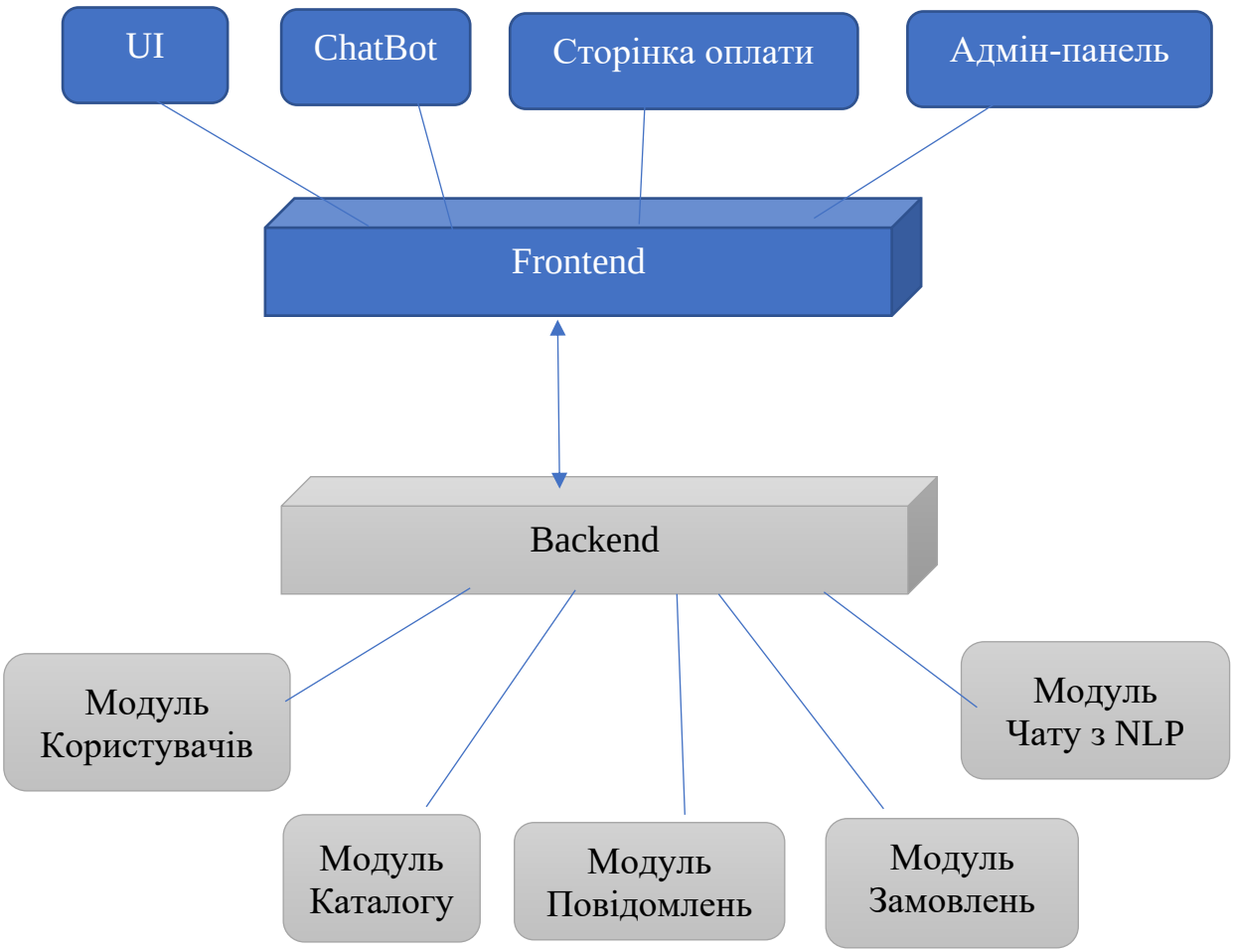

Рисунок 2.2 – Архітектура веб-застосунку

2.3 Обґрунтування вибору засобів розробки

2.3.1 Інструменти для реалізації фронтенда

Для розробки такого веб-застосунку, що включає інтернет-магазин та сервіс консультації (чатбот), оптимальним буде стек технологій, який забезпечить гнучкість, масштабованість та легкість підтримки. Для розробки клієнтської частини веб-застосунку можна обрати React.js, Vue.js або Angular. Для аргументованого вибору краще зробити аналіз цих інструментів у вигляді таблиці порівнянь їх основних характеристик та можливостей (табл. 2.2).

Таблиця 2.2 – Порівняння фронтенд-фреймворків [8]

Критерій	React.js	Vue.js	Angular
Автор	Facebook (Meta)	Evan You	Google
Кривина навчання	Середня	Низька	Висока
Продуктивність	Висока	Висока	Середня
Гнучкість	Дуже висока	Висока	Обмежена
Підхід	Компонентний (JSX)	Компонентний (Options API/ Composition API)	Повноцінний MVC-фреймворк
Складність у масштабуванні	Низька	Середня	Висока
Популярність та екосистема	Найбільша спільнота, багато готових бібліотек	Менша спільнота, але швидко зростає	Корпоративний рівень, складна екосистема
Інструменти розробки	Create React App, Next.js, Gatsby	Vue CLI, Nuxt.js	Angular CLI
Підтримка SEO	Next.js / Gatsby	Nuxt.js	Вбудований SSR
Підходить для малих проектів	Так	Так	Ні (занадто важкий)

Аргументація вибору React.js наступна:

1. Гнучкість – React не нав’язує архітектурних рішень, дозволяючи обирати потрібні інструменти (Redux, Zustand або Context API для управління станом).

2. Продуктивність – React використовує віртуальний DOM, що мінімізує оновлення реального DOM і підвищує швидкість роботи додатка.

3. Популярність та підтримка – React має найбільшу спільноту серед фронтенд-фреймворків, що забезпечує доступ до великої кількості бібліотек, готових компонентів та документації. Багато компаній використовують React (Meta, Instagram, Airbnb, Netflix), що гарантує стабільний розвиток технології.

4. Компонентний підхід – а саме використання компонентів дозволяє легко повторно використовувати код, що знижує витрати на розробку та підтримку [9].

5. Масштабованість – React підходить як для малих додатків (за рахунок простого синтаксису), так і для великих веб-систем (завдяки бібліотекам Next.js та Redux).

6. SSR та SEO – за допомогою Next.js React дозволяє реалізувати серверний рендеринг (SSR), що підвищує SEO-оптимізацію магазину.

Отже, можна зробити висновок, що React.js – найбільш універсальний для комерційного веб-додатка з магазином і чатботом.

2.3.2 Інструменти для реалізації бекенда

Для розробки бекенд-частини веб-застосунку можна обрати Node.js (Express.js), Python (Django/FastAPI) або PHP (Laravel). Розглянемо їх основні особливості, переваги та недоліки, а також обґрунтуємо вибір Node.js + Express.js (табл. 2.3).

Таблиця 2.3 – Порівняння бекенд-фреймворків

Критерій	Node.js + Express.js	Python + Django/FastAPI	PHP + Laravel
Автор	OpenJS Foundation	Python Software Foundation	Taylor Otwell
Кривина навчання	Середня	Низька (Django), Середня (FastAPI)	Низька
Продуктивність	Висока (асинхронна)	Середня (синхронна у Django)	Середня
Гнучкість	Дуже висока	Django – обмежена, FastAPI – висока	Висока, але потребує більше коду
Масштабованість	Висока (асинхронний підхід)	Висока (FastAPI)	Середня

Продовження таблиці 2.3

Розмір коду	Менший код (JS)	Django – більше коду, FastAPI – легший	Laravel – багато коду
Складність підтримки	Середня	Django – низька, FastAPI – середня	Висока (Laravel потребує більше ресурсів)
Серверний рендеринг	Підтримує (Next.js + Express)	Django має шаблонізатор	Laravel має Blade
Популярність	Дуже висока	Висока	Висока серед PHP-проектів

Для даного веб-сервісу обрано Node.js + Express.js. Аргументація вибору наступна:

1. Висока продуктивність – Node.js працює на асинхронній моделі (Event Loop), що дозволяє одночасно обробляти велику кількість запитів. Це критично для інтернет-магазину, який має багато запитів від користувачів одночасно.

2. З точки зору гнучкості Express.js – це мінімалістичний фреймворк, який не нав’язує жорстких правил. Його легко налаштовувати під специфічні потреби:

- API для магазину (товари, замовлення, користувачі);
- чатбот для підбору корму;
- інтеграція з платіжними сервісами.

3. Єдина мова (JavaScript), що дає змогу розробникам працювати як на фронтенді, так і на бекенді без необхідності змінювати мову програмування. Це прискорює розробку і спрощує підтримку проекту.

4. Підтримка WebSockets (Socket.io) допомогає реалізувати чат-бот (швидкий двосторонній зв’язок без перевантаження сервера).

5. Масштабованість – Node.js підтримує мікросервісну архітектуру, що дозволяє легко розширювати функціонал. Наприклад, можна винести обробку платежів або чатбота в окремий сервіс.

6. Легка інтеграція з базою даних , а саме, Express.js добре працює з PostgreSQL. Також можна легко інтегрувати MongoDB (якщо потрібна NoSQL БД для чату).

7. Швидкість розробки – Node.js + Express.js має багато готових модулів в NPM, що дозволяє швидко додати автентифікацію (JWT, OAuth), логування, кешування.

Для цього проекту Node.js + Express.js – найкращий вибір, бо він швидкий, легко інтегрується з фронтендом на React і дозволяє ефективно працювати з чатом та магазином одночасно.

2.3.3 Вибір бази даних

Вибір бази даних є одним із ключових етапів проектування веб-застосунку. База даних відповідає за збереження, обробку та управління інформацією про користувачів, товари, замовлення та інші сутності. У випадку веб-застосунку для комерційної діяльності фірми з продажу корму для тварин база даних повинна:

– забезпечувати надійне зберігання даних про клієнтів, замовлення, товари;

– підтримувати швидкий доступ до інформації та ефективне виконання складних SQL-запитів (наприклад, аналіз історії покупок, рекомендації);

– мати можливість масштабування для обробки зростаючої кількості користувачів і транзакцій;

– підтримувати гнучкі схеми даних, оскільки частина інформації може мати динамічну структуру (наприклад, налаштування чату-бота або характеристики тварин у JSON-форматі).

Для даного проекту розглядаються різні варіанти баз даних, зокрема реляційні (PostgreSQL, MySQL) та NoSQL (MongoDB). Нижче буде наведено

порівняння цих баз даних і обґрунтовано вибір найбільш підходящого варіанту (табл. 2.4) [10].

Таблиця 2.4 – Порівняння баз даних

Критерій	PostgreSQL	MySQL	MongoDB
Тип БД	Реляційна (SQL)	Реляційна (SQL)	NoSQL (документна)
Гнучкість схем	Висока (JSONB, XML, HSTORE)	Середня	Дуже висока
Продуктивність	Висока (оптимізований механізм запитів)	Висока (але слабша за PostgreSQL на складних запитах)	Висока (особливо при масштабуванні)
Масштабованість	Висока (горизонтальне і вертикальне масштабування)	Висока	Дуже висока (кластеризація)
Тип БД	Реляційна (SQL)	Реляційна (SQL)	NoSQL (документна)
Робота з JSON	JSONB – потужний вбудований функціонал	Обмежена підтримка	Природна підтримка JSON
Складні запити (JOIN, CTE, Window functions)	Дуже добра підтримка	Середня	Погана (немає JOIN)

Вибір пав на PostgreSQL, яка є ідеальна для інтернет-магазину. Вона реляційна. Крім цього, PostgreSQL швидше обробляє запити з JOIN, вкладеними запитами та аналітикою (наприклад, аналіз покупок клієнтів). Оскільки сервіс має чат-бот для підбору корму, можливо, потрібно буде зберігати гнучкі структури даних (JSON-об’єкти про уподобання тварин).

PostgreSQL має JSONB (двійковий формат JSON), який працює краще, ніж у MySQL. Масштабується та підтримує реплікацію (готовий для великих навантажень).

2.3.4 Вибір інструментів для створення чат-бота

Чат-бот у веб-застосунку для продажу корму для тварин має виконувати такі завдання:

1. Підбір корму на основі інформації про тварину (вік, порода, вага, особливі потреби).
2. Консультації з питань годівлі та догляду.
3. Обробка замовлень – перевірка наявності товару, оформлення замовлення.
4. Взаємодія з користувачем – збереження профілю тварини, історії покупок.
5. Відправка повідомлень – нагадування про повторну покупку.

Natural Language Processing (NLP) – це галузь штучного інтелекту, яка займається обробкою природної мови. Вона дозволяє комп’ютерам аналізувати, розуміти та взаємодіяти з текстовими чи голосовими запитами людини так, як це робить людина. У контексті чат-ботів NLP відіграє ключову роль, оскільки користувачі можуть формулювати запитання різними способами, використовуючи синоніми, скорочення або розмовну мову.

У випадку веб-застосунку для продажу корму для тварин NLP допоможе боту зрозуміти, що саме запитує користувач: шукає він корм для конкретної породи, хоче отримати консультацію з харчування чи перевірити статус замовлення. Завдяки NLP бот не просто реагує на ключові слова, а й визначає контекст розмови, аналізує наміри користувача (intents) та розпізнає сутності (entities) – наприклад, тип тварини, її вік або особливі потреби.

Google Dialogflow використовує сучасні алгоритми NLP для автоматичного визначення намірів користувача та пошуку відповідних відповідей. Його можливості дозволяють створювати бота, який адаптується до різних варіантів формулювання запитів, аналізує їхню структуру та контекст. Це робить взаємодію з ботом більш природною та схожою на діалог із реальною людиною.

Для інтеграції Google Dialogflow у веб-застосунок необхідно використовувати відповідний програмний інструмент. Оскільки Dialogflow надає API для обробки запитів, можна використовувати різні мови програмування для його взаємодії із серверною частиною.

Один із найкращих варіантів для роботи з NLP – це мова Python. Вона має багато бібліотек для обробки природної мови, таких як spaCy, NLTK та Transformers, що дозволяє додатково аналізувати повідомлення користувачів перед їх передачею до Dialogflow. Крім того, для взаємодії з Dialogflow у Python існує офіційний SDK (dialogflow), що спрощує інтеграцію.

Отже, вибір пав на NLP, Python та його бібліотеки для поєднання з Dialogflow. Процес взаємодії між користувачем, чат-ботом, NLP-системою (Dialogflow) і бекендом можна описати за допомогою UML-діаграм. Найкраще для цього підходить діаграма послідовності (Sequence Diagram), оскільки вона показує порядок взаємодії між компонентами у часовій послідовності (рис. 2.3).

Акторами та об’єктами такої діаграми є:

– користувач – взаємодіє з ботом через веб-застосунок або месенджер;

– чат-бот (Frontend) – приймає повідомлення користувача та надсилає їх до NLP-системи;

– Dialogflow (NLP-система) – аналізує введений текст, визначає намір користувача (intent) і передає його далі;

– Backend (Python-сервер) – отримує результат аналізу, перевіряє базу даних та формує відповідь;

– база даних (PostgreSQL) – містить інформацію про корми, характеристики тварин тощо.

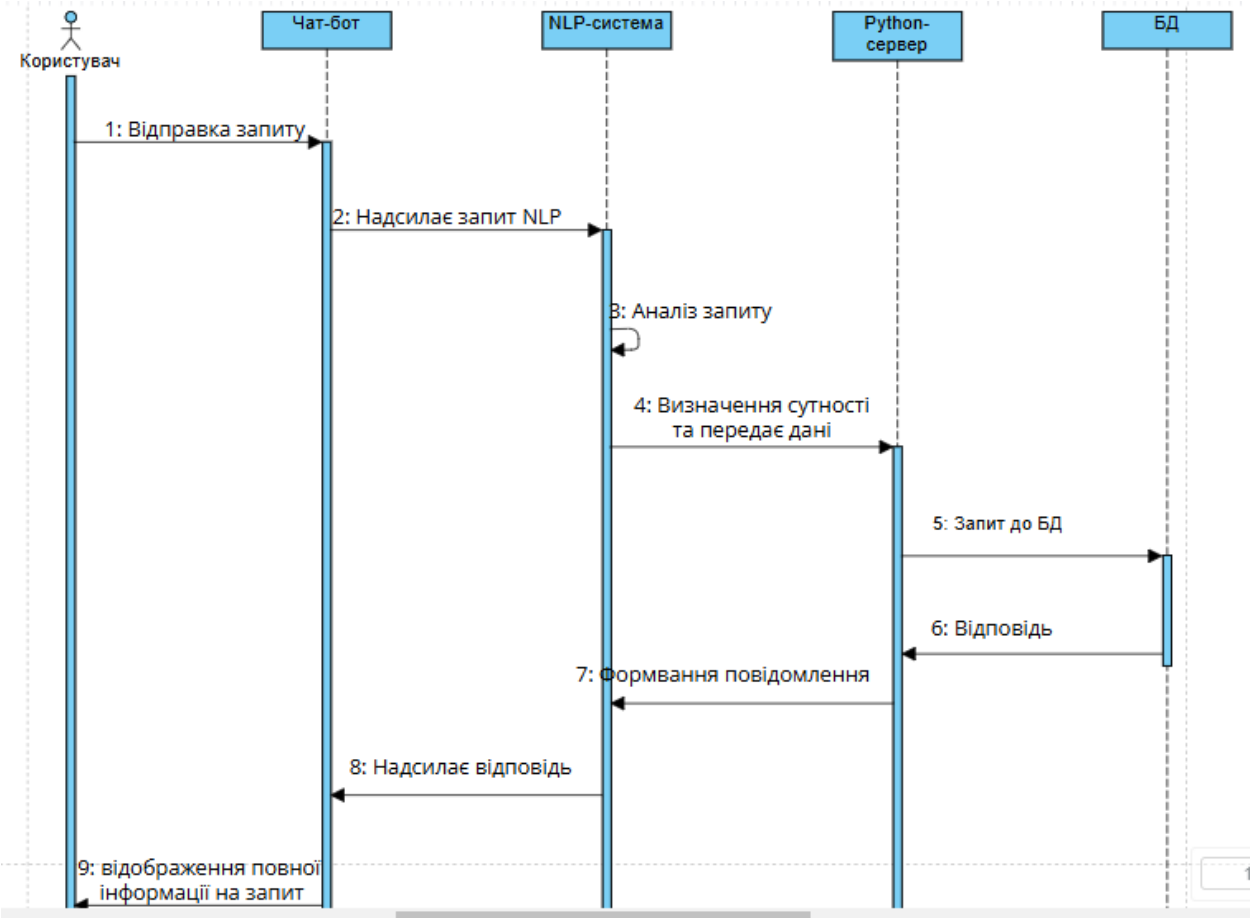

Рисунок 2.3 – Діаграма послідовності дії для роботи чат-бота

Приклад основного сценарію: «Користувач запитує підбір корму».

1. Користувач вводить повідомлення (наприклад, "Який корм підійде для мого кота 3 років?").

2. Чат-бот (Frontend) надсилає це повідомлення у Dialogflow.

3. Dialogflow (NLP-система) аналізує текст, визначає намір (наприклад, "Запит про корм") і витягує сутності (наприклад, "тварина = кіт", "вік = 3 роки").

4. Dialogflow передає дані на бекенд (Python-сервер) через API-запит.

5. Backend приймає запит, взаємодіє з базою даних (PostgreSQL), шукає відповідні корми та формує відповідь.

6. Backend передає сформовану відповідь назад у Dialogflow.

7. Dialogflow надсилає відповідь у чат-бот.

8. Користувач отримує відповідь ("Рекомендований корм для вашого кота: Корм_марка, містить високий вміст білка...").

Альтернативні сценарії роботи бота у такому випадку наступний:

1. Якщо користувач вводить некоректний запит (наприклад, "Хочу корм для мого улюбленця"), бот може уточнити, про яку тварину йдеться.

2. Якщо в базі немає відповідного корму, система може порекомендувати зв’язатися з оператором.

3. Якщо користувач хоче купити рекомендований корм, бот може передати його в кошик магазину.

2.4 Опис структури бази даних

Для зберігання інформації про користувачів, товари, замовлення та рекомендації для чат-бота обрано реляційну базу даних PostgreSQL. Основні таблиці для БД та опис їх полів і типів даних наведено нижче.

Таблиця «Users» (Користувачі) зберігає дані про зареєстрованих користувачів:

– id SERIAL (PK) – унікальний ідентифікатор користувача;

– name VARCHAR(100) – ім'я користувача;

– email VARCHAR(255) UNIQUE – електронна пошта користувача;

– password TEXT – хешований пароль;

– phone VARCHAR(2) – телефон;

– created_at TIMESTAMP DEFAULT – дата реєстрації;

– updated_at TIMESTAMP – дата останнього оновлення;

Таблиця «Pets» (Домашні тварини користувача) описує домашніх улюбленців, для яких підбирається корм.

До її полів належать наступні:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор тварини;

– user_id INTEGER REFERENCES users(id) ON DELETE CASCADE – власник тварини;

– name VARCHAR(100) – ім'я тварини;

– species VARCHAR(50) – вид (кіт, собака, інше);

– breed VARCHAR(100) – порода;

– age INTEGER – вік (роки);

– weight DECIMAL(5,2) – вага (кг);

– health_conditions TEXT – особливості здоров'я;

– created_at TIMESTAMP DEFAULT now() – дата додавання.

Таблиця «Products» (Корми) містить дані про товари, які доступні в магазині. Її поля наступні:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор товару;

– name VARCHAR(255) – назва корму;

– brand VARCHAR(100) – торгова марка;

– category VARCHAR(50) – категорія (сухий, вологий, лікувальний);

– for_species VARCHAR(50) – для яких тварин (собаки, коти);

– age_range VARCHAR(50) – для якого віку (щеня, дорослий, старший);

– weight DECIMAL(5,2) – вага упаковки (кг);

– price DECIMAL(10,2) – ціна;

– stock INTEGER – кількість на складі;

– description TEXT – опис корму;

– created_at TIMESTAMP DEFAULT now() – дата додавання.

Наступна таблиця «Orders» (Замовлення) зберігає інформацію про замовлення користувачів:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор замовлення;

– user_id INTEGER REFERENCES users(id) ON DELETE CASCADE – покупець;

– total_price DECIMAL(10,2) – загальна сума замовлення;

– status VARCHAR(50) статус (очікує, сплачено, відправлено);

– created_at TIMESTAMP DEFAULT now() – дата оформлення.

Таблиця «Order_items» (Товари в замовленні) зв’язує замовлення з товарами та має наступні поля:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор;

– order_id INTEGER REFERENCES orders(id) ON DELETE CASCADE належить до замовлення;

– product_id INTEGER REFERENCES products(id) ON DELETE CASCADE – замовлений товар;

– quantity INTEGER – кількість одиниць товару;

– subtotal_price DECIMAL(10,2) – сума за цей товар.

Таблиця «Chat_sessions» (Сесії чату) відстежує розмови користувачів з ботом:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор сесії;

– user_id INTEGER REFERENCES users(id) ON DELETE CASCADE користувач;

– started_at TIMESTAMP DEFAULT now() – час початку сесії;

– ended_at TIMESTAMP – час закінчення сесії (якщо завершена).

Таблиця «Messages» (Повідомлення в чаті) зберігає історію повідомлень між ботом та користувачем:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор повідомлення;

– session_id INTEGER REFERENCES chat_sessions(id) ON DELETE CASCADE – до якої сесії належить;

– sender VARCHAR(50) – хто відправив (user, bot);

– message TEXT – текст повідомлення;

– created_at TIMESTAMP DEFAULT now() – час відправки.

Наведена структура бази даних охоплює основні функції сервісу, проте для повноцінної роботи веб-застосунку можуть знадобитися додаткові

таблиці, які забезпечать кращу масштабованість, аналітику та ефективність роботи системи.

Таблиця «Reviews» (Відгуки про товари) зберігає відгуки користувачів про корми:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор відгуку;

– user_id INTEGER REFERENCES users(id) ON DELETE CASCADE користувач, що залишив відгук;

– product_id INTEGER REFERENCES products(id) ON DELETE CASCADE – відгук про який товар;

– rating INTEGER CHECK (rating BETWEEN 1 AND 5) – оцінка (від 1 до 5);

– comment TEXT – текст відгуку;

– created_at TIMESTAMP DEFAULT now() – час створення.

Таблиця «Subscriptions» (Підписки на автоматичні покупки) зберігає дані про підписки користувачів на регулярну доставку корму:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор підписки;

– user_id INTEGER REFERENCES users(id) ON DELETE CASCADE – користувач, який оформив підписку;

– product_id INTEGER REFERENCES products(id) ON DELETE CASCADE – корм, що буде доставлятися;

– frequency – VARCHAR(50) – періодичність (щотижня, раз на місяць);

– next_delivery TIMESTAMP – дата наступної доставки;

– status VARCHAR(20) – активна/скасована;

– created_at TIMESTAMP DEFAULT now() – час створення підписки.

Таблиця «Payment_transactions» (Оплати) зберігає інформацію про транзакції оплати (контроль фінансових операцій та можливість повернення коштів у разі проблеми):

– id SERIAL PRIMARY KEY – унікальний ідентифікатор транзакції;

– order_id INTEGER REFERENCES orders(id) ON DELETE CASCADE – до якого замовлення відноситься;

– user_id INTEGER REFERENCES users(id) ON DELETE CASCADE користувач, який оплатив;

– amount DECIMAL(10,2) – сума оплати;

– status VARCHAR(20) – статус (успішно, відхилено);

– payment_method VARCHAR(50) – спосіб оплати (карта, PayPal, Google Pay);

– created_at TIMESTAMP DEFAULT now() – час платежу.

Таблиця «Admin_users» (Адміністратори системи) зберігає дані про адміністраторів, які керують сервісом. Відокремлення адміністративного доступу від звичайних користувачів дасть можливість забезпечити гнучкість у керуванні правами:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор адміністратора;

– email VARCHAR(255) UNIQUE – логін (email);

– password TEXT – захешований пароль;

– role VARCHAR(50) – роль (менеджер, модератор, фінансовий відділ);

– created_at TIMESTAMP DEFAULT now() – дата додавання.

Остання таблиця, «Faq» (База знань для бота) зберігає часті питання та відповіді для роботи чат-бота. Вона забезпечує оптимізацію роботи бота – швидкі відповіді без NLP-обробки:

– id SERIAL PRIMARY KEY – унікальний ідентифікатор питання;

– question TEXT – запитання користувача;

– answer TEXT – відповідь;

– created_at TIMESTAMP DEFAULT now() – дата додавання.

Діаграму «сутність-зв’язок» представлено на рисунку 2.4.

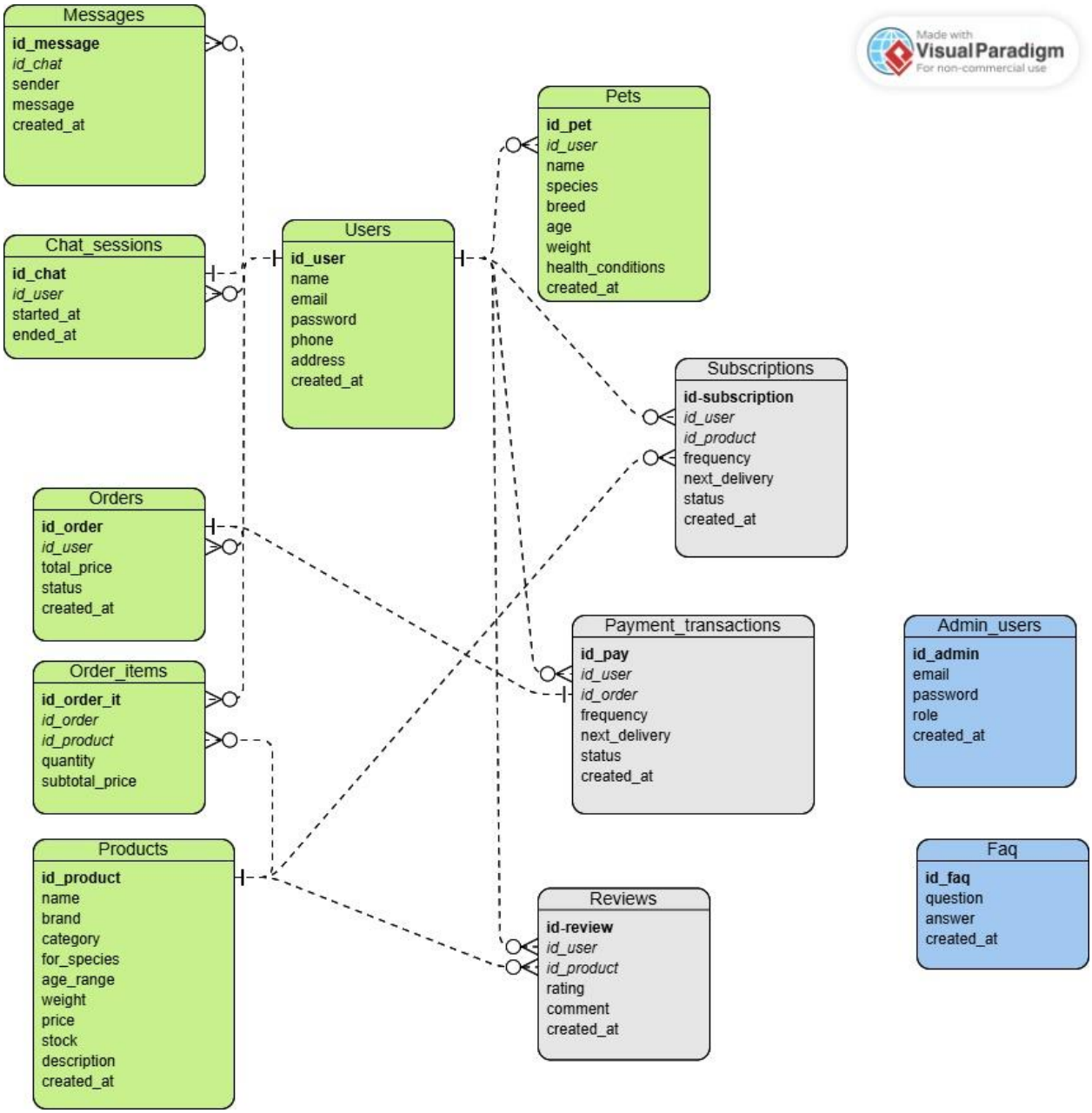

Рисунок 2.4 – Діаграма «сутність-зв’язок» для БД

3 РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ

3.1 Опис інтерфейсу користувача

Цільовою аудиторією даного веб-застосунку є власники домашніх тварин, які шукають корм і консультації щодо їх коректного підбору для своїх улюбленців. Основна мета інтерфейсу полягає у простоті використання, швидкому доступі до продуктів і чату з ботом. З точки зору навігації на сайті представлено головне меню, категорії, пошук, кнопки для підтвердження дій користувача. Стиль обрано мінімалістичний, дружній. Головну сторінку представлено на рисунку 3.1.

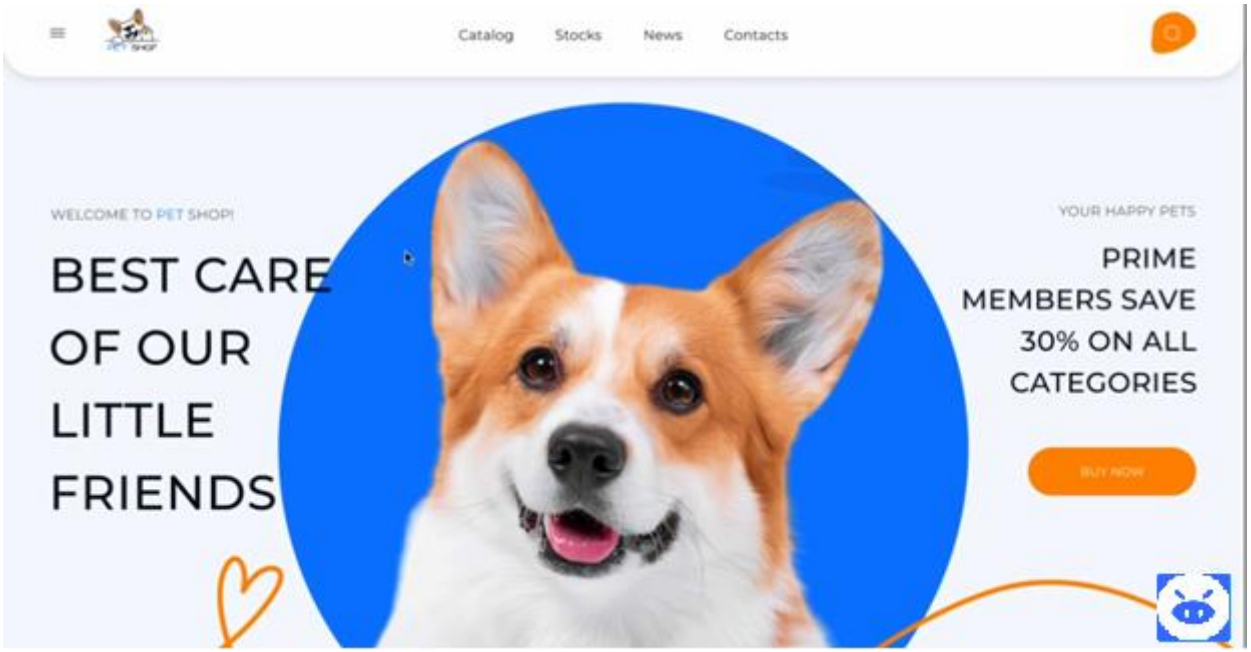

Рисунок 3.1 – Скріншот головної сторінки

Основною мовою обрану англійську з точки зору зручності для подальшого розвитку та продажу веб-сервісу. При роботі із застосунком користувачу слід обрати вид домашньої тваринки (собака, кіт, хом’як або рибка), як показано на рис. 3.2. Весь товар відображається на центральній частині сайту (рис. 3.3).

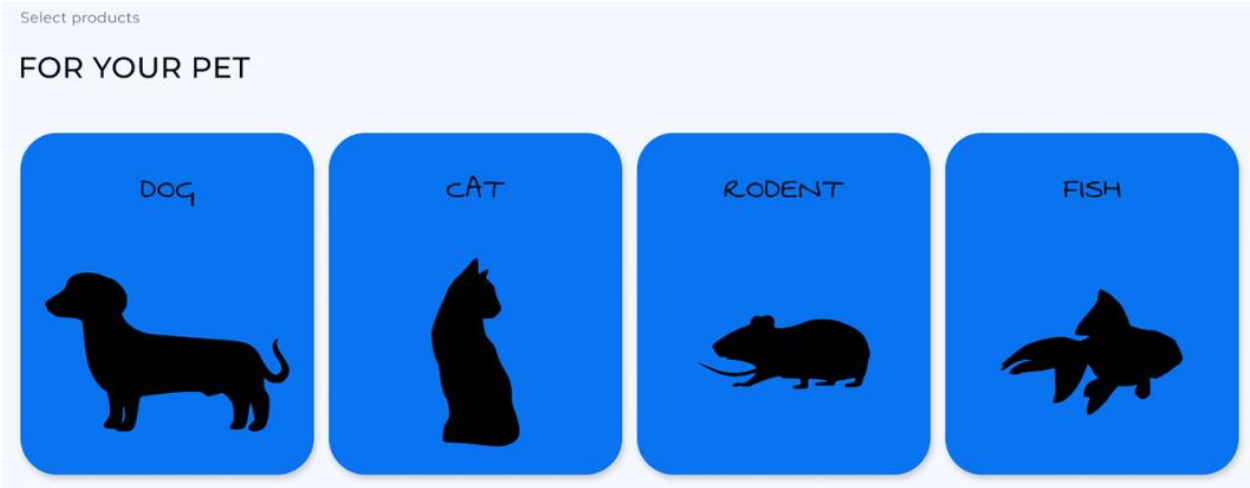

Рисунок 3.2 – Блок вибору тварини

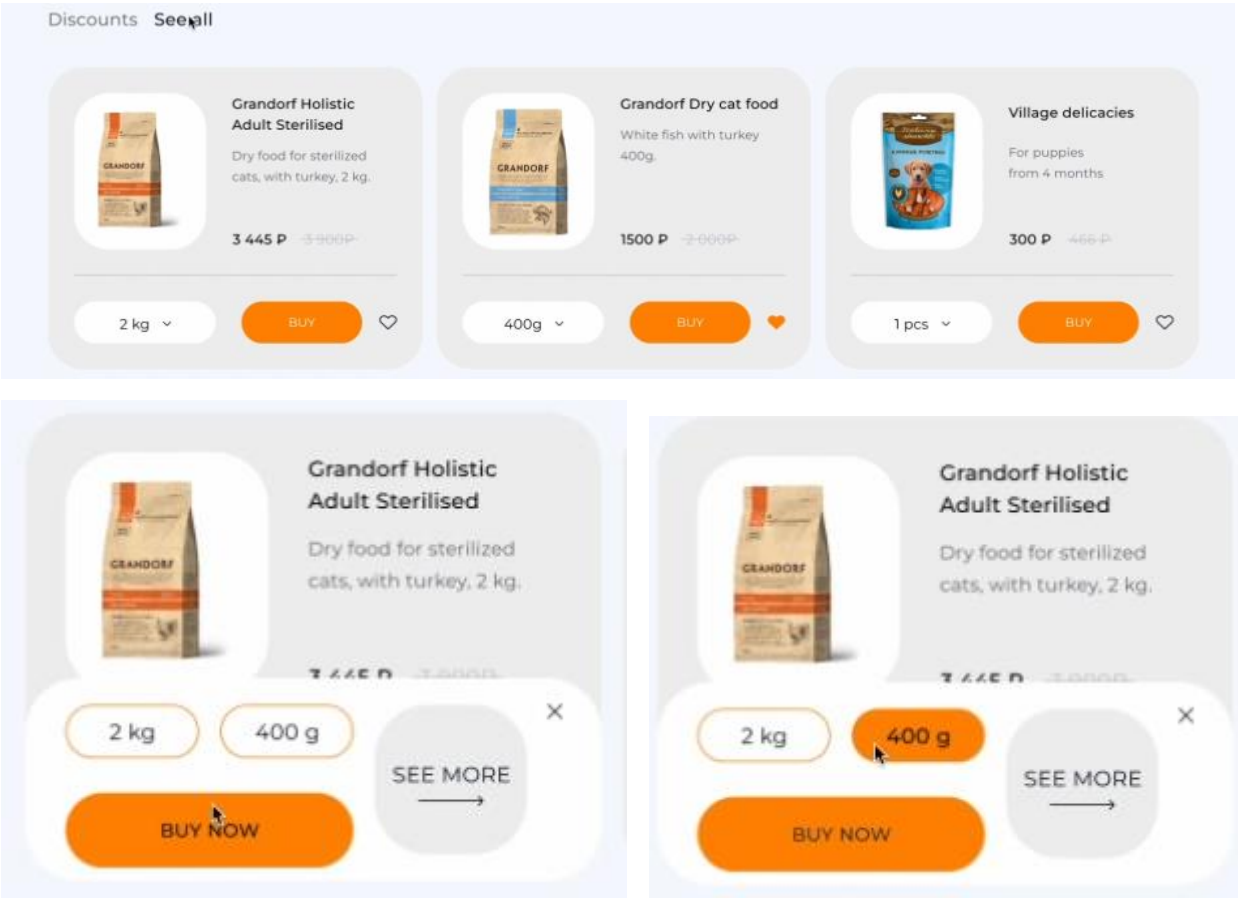

Рисунок 3.3 – Приклад розміщення товару на сторінці

Для кожної одиниці товару є можливість обрати об’єм пакування або кількість пакетів у одному лоті. Користувач бачить вагу та через кнопку «Buy» додає товар до свого кошика. Крім цього, праворуч є функція «додати до обраного». Меню відкриває користувачу перелік фільтрів для зручного вибору товару (рис. 3.4). Тут є можливість обирати за назвою бренда, за основним компонентом складової їжі, віковою категорією тварини та за типом продукту харчування. Для користувачів доступно блок з актуальними новинами та постами щодо догляду за тваринами у домашніх умовах (рис. 3.5).

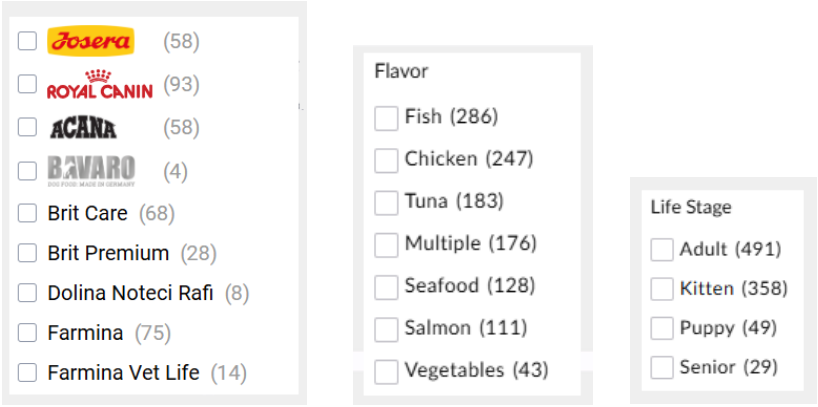

Рисунок 3.4 – Приклади фільтрив для товару

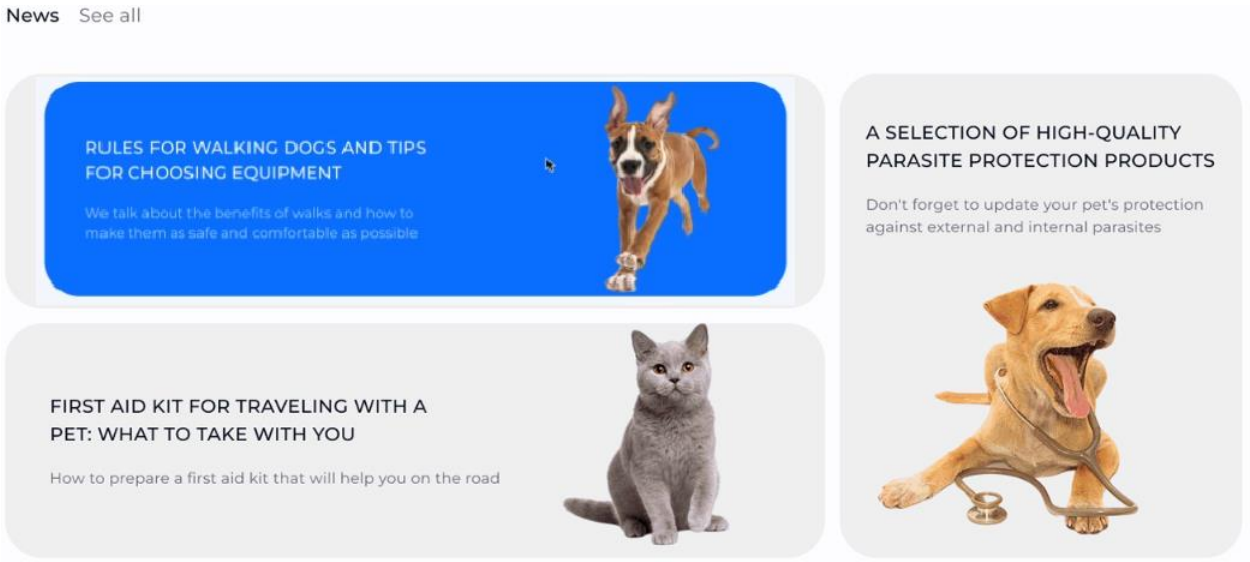

Рисунок 3.5 – Блок з корисними порадами для власників тварин

3.2 Реалізація бази даних застосунку

У спроектованій базі даних є кілька таблиць, які мають залежності одна від одної через зовнішні ключі, тому важливо створювати їх у правильному порядку, щоб уникнути помилок при визначенні зв'язків. Основні (незалежні) таблиці слід створити у першу чергу, вони не містять зовнішніх ключів. До таких таблиць належать «Users», «Pets» та «Products»:

Видаляємо всі таблиці, якщо вони існують (для повторного запуску)

```
    DROP TABLE IF EXISTS faq, admin_users, payment_transactions,
subscriptions, reviews, messages, chat_sessions,
    order_items, orders, products, pets, users CASCADE;


    -- Таблиця користувачів
    CREATE TABLE users (
        id SERIAL PRIMARY KEY,
        name VARCHAR(255) NOT NULL,
        email VARCHAR(255) UNIQUE NOT NULL,
        password TEXT NOT NULL,
        phone VARCHAR(20),
        address TEXT,
        created_at TIMESTAMP DEFAULT now()
    );


    -- Таблиця домашніх тварин
    CREATE TABLE pets (
        id SERIAL PRIMARY KEY,
        user_id INTEGER REFERENCES users(id) ON DELETE CASCADE,
        name VARCHAR(255) NOT NULL,
        species VARCHAR(50) CHECK (species IN ('dog', 'cat',
'bird', 'other')),
        breed VARCHAR(255),
        age INTEGER CHECK (age >= 0),
        weight DECIMAL(5,2),
        health_notes TEXT
    );


    -- Таблиця товарів (кормів)
    CREATE TABLE products (
        id SERIAL PRIMARY KEY,
        name VARCHAR(255) NOT NULL,
        description TEXT,
        price DECIMAL(10,2) CHECK (price >= 0),
```

```
        stock INTEGER CHECK (stock >= 0),
        category VARCHAR(100),
        created_at TIMESTAMP DEFAULT now()
    );
```

Після створення основних таблиць можна створювати ті, що містять зовнішні ключі. Таблиця «Orders» (Замовлення) зберігає дані про замовлення, але не товари в ньому, а в таблиці «Order_items» (Склад замовлення) зберігається список товарів у кожному замовленні.

```
    -- Таблиця замовлень
    CREATE TABLE orders (
        id SERIAL PRIMARY KEY,
        user_id INTEGER REFERENCES users(id) ON DELETE CASCADE,
        total_price DECIMAL(10,2) CHECK (total_price >= 0),
        status VARCHAR(50) CHECK (status IN ('pending', 'paid',
'shipped', 'delivered', 'canceled')),
        created_at TIMESTAMP DEFAULT now()
    );


    -- Таблиця елементів замовлення
    CREATE TABLE order_items (
        id SERIAL PRIMARY KEY,
        order_id INTEGER REFERENCES orders(id) ON DELETE CASCADE,
        product_id INTEGER REFERENCES products(id) ON DELETE
CASCADE,
        quantity INTEGER CHECK (quantity > 0),
        price DECIMAL(10,2) CHECK (price >= 0)
    );
```

Наступні таблиці необхідні для роботи чатбота:

```
    -- Таблиця чат-сесій (бот)
    CREATE TABLE chat_sessions (
        id SERIAL PRIMARY KEY,
        user_id INTEGER REFERENCES users(id) ON DELETE CASCADE,
        started_at TIMESTAMP DEFAULT now()
    );


    -- Таблиця повідомлень (бот)
    CREATE TABLE messages (
        id SERIAL PRIMARY KEY,
        session_id INTEGER REFERENCES chat_sessions(id) ON DELETE
CASCADE,
```

```
        sender VARCHAR(20) CHECK (sender IN ('user', 'bot')),
        message TEXT NOT NULL,
        created_at TIMESTAMP DEFAULT now()
    );
```

Для роботи з діяльністю користувачів необхідні таблиці з відгуками, підписки на автоматичну розсилку новин та інформація щодо їх платежів за товар.

```
    -- Таблиця відгуків користувачів
    CREATE TABLE reviews (
        id SERIAL PRIMARY KEY,
        user_id INTEGER REFERENCES users(id) ON DELETE CASCADE,
        product_id INTEGER REFERENCES products(id) ON DELETE
CASCADE,
        rating INTEGER CHECK (rating BETWEEN 1 AND 5),
        comment TEXT,
        created_at TIMESTAMP DEFAULT now()
    );

    -- Таблиця підписок на автоматичну покупку
    CREATE TABLE subscriptions (
        id SERIAL PRIMARY KEY,
        user_id INTEGER REFERENCES users(id) ON DELETE CASCADE,
        product_id INTEGER REFERENCES products(id) ON DELETE
CASCADE,
        frequency VARCHAR(50) CHECK (frequency IN ('weekly',
'monthly')),
        next_delivery TIMESTAMP,
        status VARCHAR(20) CHECK (status IN ('active',
'canceled')),
        created_at TIMESTAMP DEFAULT now()
    );

    -- Таблиця платежів
    CREATE TABLE payment_transactions (
        id SERIAL PRIMARY KEY,
        order_id INTEGER REFERENCES orders(id) ON DELETE CASCADE,
        user_id INTEGER REFERENCES users(id) ON DELETE CASCADE,
        amount DECIMAL(10,2) CHECK (amount >= 0),
        status VARCHAR(20) CHECK (status IN ('successful',
'failed')),
        payment_method VARCHAR(50),
        created_at TIMESTAMP DEFAULT now()
    );
```

Таблиця «Admins» (Адміністратори) зберігає інформацію про адміністраторів веб-застосунку. Тут вони можуть керувати товарами, замовленнями та базою знань.

```
-- Таблиця адміністраторів
CREATE TABLE admin_users (
    id SERIAL PRIMARY KEY,
    email VARCHAR(255) UNIQUE NOT NULL,
    password TEXT NOT NULL,
    role VARCHAR(50) CHECK (role IN ('manager', 'moderator',
'finance')),
    created_at TIMESTAMP DEFAULT now()
);
```

Питання та відповіді, які бот використовує для відповіді на запити користувачів, зберігатимуться у таблиці «FAQ»:

```
-- Таблиця для FAQ (база знань чат-бота)
CREATE TABLE faq (
    id SERIAL PRIMARY KEY,
    question TEXT NOT NULL,
    answer TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT now()
);
```

Адміністратор керує базою знань наступним чином: так, для додавання нового питання-відповіді виконується такий запит:

```
INSERT INTO knowledge_base (category, question, answer)
VALUES ('харчування', 'Який корм підійде для алергічних котів?',
       'Рекомендуємо гіпоалергенний корм X.');
```

Оновлення відповіді у базі знань:

```
UPDATE knowledge_base
SET answer = 'Рекомендуємо корм Y для алергічних котів.'
WHERE entry_id = 1;
```

3.3 Реалізація функціоналу чатбота

Розробка чат-бота проходить кілька етапів: від збору даних і налаштування моделі обробки мови (NLP) до тестування. Спочатку слід наповнити базу знать для бота з точки зору найчастіших питань, які можуть виникнути у користувача. Для цього в БД є таблиця «Faq».

Якщо бот отримує запит, він може:

- шукати відповідь у таблиці «Faq»;
- якщо не знайдено – передавати запит на обробку NLP-моделі (Dialogflow);
- якщо NLP теж не знаходить відповідь – передавати питання адміністратору для оновлення бази знань.

Для розуміння природної мови використовуємо Dialogflow, який працює з Python через API. Компонентами NLP виступають інтерни, тобто наміри користувача, та сутності – в даному випадку це параметри, які бот повинен розпізнавати у запиті від користувача.

Приклад інтентів:

– «Підібрати корм для кота» – інтент «Find_Food»;

– «Як мені оформити замовлення?» – інтент «Order_Help».

Приклад сутностів:

– «animal_type» – це кіт, собака, папуга, рибка, хом'як;

– «age» – дорослий, кошеня, щеня, молодий та інше;

– «allergy» – на що є алергія у питомця (курка, зернові, риба).

У перелік тренувальних фраз бажано внести як можна більше запитів користувача, це поможе зменшити навантаження на консультанта и користувачу не прийдеться чекати живої відповіді.

Всі питання поділено умовно на каттегорії, так зручніше шукати відповіді. Приклад декількох питань щодо кожної каттегорії, які охоплюють різні тематики, пов’язані з підбором корму, оплатою, доставкою та іншими запитами наведено нижче.

1. Запитання про підбір корму:
   - Який корм найкраще підійде для мого пса?
   - Чим годувати кота, якщо у нього алергія?
   - Підкажіть, який корм підійде для цуценяти маленької породи?
2. Запитання про оплату:
   - Які способи оплати ви приймаєте?
   - Чи можна оплатити замовлення при отриманні?
   - Як оформити оплату карткою онлайн?
3. Запитання про доставку:
   - Скільки коштує доставка в Київ?
   - Як швидко ви доставляєте замовлення?
   - Чи є безкоштовна доставка при великому замовленні?
4. Загальні питання:
   - Як зв’язатися з підтримкою?
   - Чи можна повернути корм, якщо він не підійшов?
   - Які у вас години роботи?

Отже, після встановлення SDK для Dialogflow завантажуємо бібліотеки і налаштувуємо аутентифікації:

```
import dialogflow_v2 as dialogflow
import json
from google.oauth2 import service_account
```

Шлях до файлу з ключем сервісного акаунта Dialogflow:

```
credentials = service_account.Credentials.from_service_account_file(
    "service_account.json"
)
```

Функція для додавання тренувальних фраз «add_training_phrases» реалізована наступним чином:

```
     def add_training_phrases(project_id, intent_id,
training_phrases):
         client =
dialogflow.IntentsClient(credentials=credentials)
         parent = client.intent_path(project_id, intent_id)
```

Отримуємо поточний інтент та додаємо нові фрази у форматі Dialogflow:

```
      intent = client.get_intent(name=parent)
        for phrase in training_phrases:
            part =
dialogflow.Intent.TrainingPhrase.Part(text=phrase)
            training_phrase =
dialogflow.Intent.TrainingPhrase(parts=[part])
            intent.training_phrases.append(training_phrase)
```

Після цього слід оновити інтент у Dialogflow:

```
      response = client.update_intent(intent=intent,
language_code="uk")
        print(f"Додано {len(training_phrases)} нових фраз до
інтенту '{response.display_name}'")
```

Для передачі фраз у бот необхідно вказати ідентифікатор вашого проекту Dialogflow та інтенту, в який додаємо фрази:

```
    PROJECT_ID = "pet-food-assistant"
    INTENT_ID = "3f5a1c6b-8d9f-4b2c-a123-45f67890abcd"
```

Початковий список тренувальних фраз для навчання:

```
    training_phrases = [
        "Який корм найкраще підійде для мого пса?",
        "Чим годувати кота, якщо у нього алергія?",
        "Підкажіть, який корм підійде для цуценяти маленької
породи?",
        "Які способи оплати ви приймаєте?",
        "Чи можна оплатити замовлення при отриманні?",
        "Як швидко ви доставляєте замовлення?",
        "Чи є безкоштовна доставка при великому замовленні?",
        "Як зв’язатися з підтримкою?",
        "Чи можна повернути корм, якщо він не підійшов?",
    ]
```

Наступний код створює простий сервер на Node.js з використанням Express, який отримує повідомлення від користувача, передає їх у Dialogflow, отримує відповідь від чат-бота і відправляє її назад у веб-застосунок:

Імпорт бібліотек, після чого створюємо екземпляр Express та задаємо порт, на якому буде працювати сервер. Налаштування middleware для обробки JSON, що дозволяє розбирати JSON-запити, які надходять у POST-запитах:

```
const express = require("express");
const { SessionsClient } = require("@google-
cloud/dialogflow");
const app = express();
const port = 3000;
app.use(express.json());
```

Ініціалізація Dialogflow API:

```
const projectId = " pet-food-assistant ";
const sessionClient = new SessionsClient();
```

Маршрут для обробки повідомлень бота (це створює API-ендпойнт /chat, який приймає POST-запити від фронтенду). Оскільки запит до Dialogflow асинхронний, використовуємо async/await.

```
app.post("/chat", async (req, res) => {
```

Отримання повідомлення від клієнта використовує унікальний ідентифікатор сесії користувача:

```
const sessionId = " b25f6e0a-2dcb-4b3a-95a4-2c4313b77c9e";
const text = req.body.message;
```

Формування запиту до Dialogflow виконується через шлях до конкретної сесії бота у Google Cloud. Цей параметр важливий, бо він дозволяє боту запам’ятовувати контекст розмови.

```
    const sessionPath =
sessionClient.projectAgentSessionPath(projectId, sessionId);
```

Створення запиту до бота: передаємо текст повідомлення користувача та вказуємо, що запит українською мовою (а не англійською):

```
    const request = {
        session: sessionPath,
        queryInput: {
            text: { text, languageCode: "uk" }
        }
    };
```

Відправкою запиту до Dialogflow займається функція detectIntent(), яка надсилає текст боту і отримує його відповідь.

```
    const responses = await sessionClient.detectIntent(request);
```

Наступний крок відповідно до алгоритму роботи бота, це формування відповіді. Тут fulfillmentText – це текст відповіді, який бот надіслав у відповідь.

```
res.send({ reply: responses[0].queryResult.fulfillmentText });
```

Фронтенд отримає об'єкт:

```
{ "reply": "Рекомендую корм для котів Royal Canin" }
```

Отже, послідовність наступна: фронтенд (React) відправляє POST-запит на /chat, передаючи текст повідомлення. Бекенд (Express) отримує запит і пересилає його в Dialogflow API. Бот аналізує повідомлення та повертає відповідь. Бекенд отримує відповідь від бота і надсилає її назад на фронтенд. Фронтенд відображає відповідь бота у чаті.

Розглянемо приклад запуску чатбота (рис. 3.6). Користувач просить допомогти консультанту у виборі їжі для свого питомця з урахуванням що він має алергію.

Рисунок 3.6 – Приклад діалогу користувача з онлайн-консультантом

Якщо у базі даних є товар, який відповідає потребам покупця, бот відображає повідомлення з його назваю на пропонує кнопки для швидкого переходу до каталогу для ознайомлення з повною інформацією щодо корму

або перехід відразу до корзини користувача, де цей товар вже буде автоматично додано (рис. 3.7).

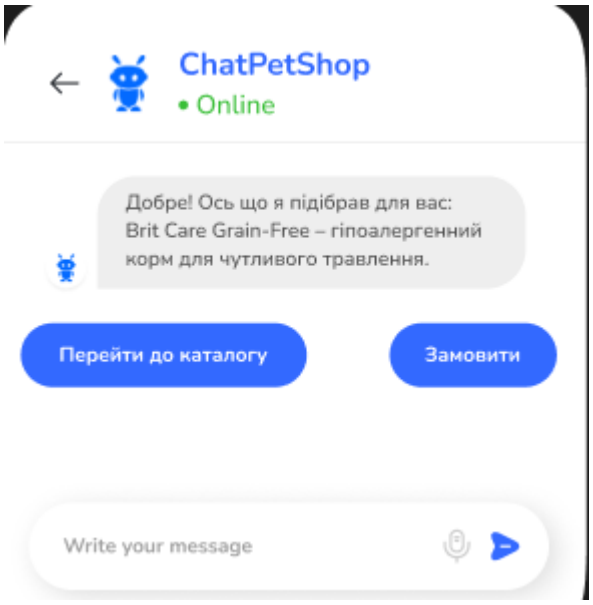

Рисунок 3.7 – Відображення гарячих кнопок у чаті з онлайн-консультантом

Якщо не знайдено готового рішення щодо запиту користувача, бот пропонує звернутись до консультанта (рис. 3.7):

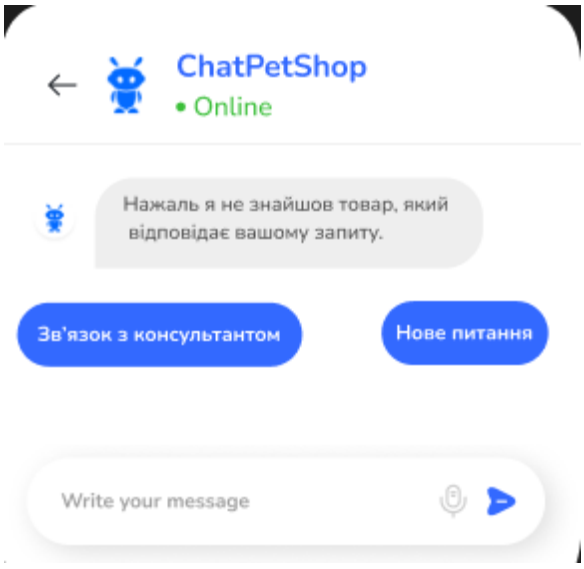

Рисунок 3.7 – Варіант відсутності готового рішення

Персоналізовані рекомендації корму на основі історії покупок – це одне з ключових покращень чат-бота, а саме, інтелектуальні рекомендації корму для домашніх тварин. Це дозволяє зробити обслуговування більш персоналізованим і покращити досвід користувачів.

Основна ідея – аналізувати історію покупок користувача та на її основі пропонувати найкращі корми. Алгоритм роботи:

1. Бот отримує дані про останні замовлення користувача.
2. Знаходить закономірності у виборі корму (тип тварини, клас корму, особливості – гіпоалергенний, беззерновий тощо).
3. Формує персоналізовану рекомендацію на основі цих даних.
4. Видає користувачеві пропозицію та можливість одразу замовити корм.

Якщо користувач ще не робив замовлень, бот запитає про:

- тип тварини (кіт, собака та інші);
- вік тварини (дитинча, дорослий, літній);
- особливі потреби (гіпоалергенний, беззерновий, дієтичний корм).

Отримання останніх куплених товарів користувача виконує наступний SQL-запит. Він отримує 5 останніх куплених кормів користувачем та визначає категорію корму, його тип (для котів, собак тощо), особливості:

```
    SELECT p.product_id, p.name, p.category, p.animal_type,
p.age_group, p.special_features
    FROM orders o
    JOIN order_items oi ON o.order_id = oi.order_id
    JOIN products p ON oi.product_id = p.product_id
    WHERE o.user_id = 1
    ORDER BY o.order_date DESC
    LIMIT 5;
```

Наступний запит шукає товари для того ж типу тварини, віку та з подібними характеристиками, видає 3 випадкові варіанти, щоб урізноманітнити рекомендації.

```
     SELECT p.product_id, p.name, p.category, p.animal_type,
p.age_group, p.special_features
     FROM products p
     WHERE p.animal_type = 'собака'
     AND p.age_group = 'дорослий'
     AND 'гіпоалергенний' =
ANY(string_to_array(p.special_features, ', '))
     ORDER BY RANDOM()
     LIMIT 3;
```

ВИСНОВКИ

Сьогодні онлайн-торгівля кормами для домашніх тварин стрімко зростає, і все більше покупців надають перевагу цифровим сервісам для швидкого та зручного вибору продукції. Основні фактори, які підтверджують актуальність такого сервісу:

– автоматизація обслуговування клієнтів – чат-бот забезпечує швидкі відповіді на запити користувачів без залучення менеджерів;

– персоналізовані рекомендації – система аналізує історію покупок та пропонує найбільш релевантні товари;

– економія часу – замість тривалого пошуку на сайті користувач отримує відповідний корм за кілька секунд;

– гнучкість використання – бот може працювати як у веб-застосунку, так і в месенджерах (Telegram, Viber, WhatsApp);

– інтеграція зі складськими залишками – система може повідомляти про наявність товару та рекомендувати аналоги.

Таким чином, такий сервіс є затребуваним, особливо серед зайнятих власників тварин, які хочуть швидко підібрати корм без необхідності детального вивчення асортименту.

В рамках розробки веб-застосунку було виконано наступне:

1. Розробка модульної архітектури сервісу.

Визначено основні компоненти: чат-бот, система рекомендацій, база знань, база даних покупців та замовлень. Створено схему модульної архітектури, що включає взаємодію фронтенду, бекенду та NLP-системи.

2. Обрані інструментальні засоби розробки фронтенду та бекенду, а також систему керування БД.

3. Спроектовано структуру БД, що містить таблиці користувачів, замовлень, товарів, категорій корму та історії покупок. Реалізовано SQL-запити для аналізу покупок та генерації персоналізованих рекомендацій.

4. Чат-бот з NLP.

Використано Google Dialogflow для побудови діалогової моделі. Навчено бота розуміти ключові запити користувачів та відповідати відповідно до їх потреб. Налаштовано інтеграцію з базою знань для відповідей на часті питання.

5. Рекомендаційна система, що аналізує попередні покупки користувачів та пропонує найкращі варіанти корму. Вона використовує SQL-запити для вибору схожих товарів за параметрами: тип тварини, вік, особливі потреби (беззерновий, гіпоалергенний тощо).

Розроблений сервіс є інноваційним рішенням для автоматизації підбору корму для домашніх тварин. Він поєднує сучасні технології NLP, персоналізовані рекомендації, інтеграцію з базою знань та зручний веб-інтерфейс. Реалізація цього проекту дає змогу значно покращити досвід покупців, підвищити продажі магазину та мінімізувати навантаження на менеджерів.

Майбутнє цього сервісу – розширення можливостей та впровадження штучного інтелекту для ще більш точного прогнозування потреб клієнтів. Веб-сервіс має всі шанси стати успішним у сфері онлайн-продажу зоотоварів та забезпечити якісний сервіс для власників домашніх тварин.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Selling Dog Food Online: A Beginner's Guide. URL: https://debutify.com/blog/selling-dog-food-online (дата звернення 03.03.2025).
2. Top 5 Pet Food Directory Websites for 2024 URL: https://directorist.com/blog/pet-food-directory-websites/ (дата звернення 04.03.2025).
3. Pet Food URL: https://www.eworldtrade.com/food-beverage/pet-food/ (дата звернення 04.03.2025).
4. Top 30 Marketplaces for Pet Supplies. URL: https://e-tailize.com/blog/top-20-marketplaces-for-pet-supplies/ (дата звернення 11.03.2025).
5. Функціональні та нефункціональні вимоги. URL: https://www.guru99.com/uk/functional-vs-non-functional-requirements.html (дата звернення 24.03.2025).
6. Create A Gantt Chart. URL: https://software.fish/project-management-software/gantt-chart-template (дата звернення 25.03.2025).
7. Мікросервісна архітектура. URL: https://foxminded.ua/mikroservisna-arkhitektura/ (дата звернення 28.03.2025).
8. Різниця між фронт і бекенд розробкою. URL: https://foxminded.ua/ru/front-end-back-end-raznica/ (дата звернення 02.04.2025).
9. React Introduction. URL: https://www.geeksforgeeks.org/reactjs-introduction/ (дата звернення 02.04.2025).
10. PostgreSQL: The World's Most Advanced Open Source Relational Database. URL: https://www.postgresql.org/ (дата звернення 11.04.2025).
11. Що таке обробка природної мови (NLP). URL: https://metinvest.digital/ru/page/1052 (дата звернення 13.04.2025).