Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інформаційних технологій та комп'ютерної інженерії
(повне найменування інституту, назва факультету (відділення))

Кафедра обчислювальної техніки
(повна назва їїкафедри (предметної, циклової комісії))

Пояснювальна записка

до бакалаврської дипломної роботи
на тему:

«ПРОГРАМНИЙ ЗАСІБ ІЗ ВЕБІНТЕРФЕЙСОМ ДЛЯ ВЕТЕРИНАРНОЇ КЛІНІКИ»

Виконав: студент 2 (4) курсу,
групи КІ-22мсз
спеціальності 123 — Комп'ютерна інженерія
Куліш Д. В.
(прізвище та ініціали)
Керівник: к.т.н., ст.викл. каф. ОТ
Дудник О. В.
(прізвище та ініціали)
«12» 06 2024 р.
Рецензент: д.т.н., професор, завідувач каф. ПЗ
Романюк О. Н.
(прізвище та ініціали)
«14» 06 2024 р.

Допущено до захисту
Завідувач кафедри ОТ
д.т.н., проф Азаров О.Д.
"11" 06 2024 р.

Вінниця ВНТУ — 2024 рік

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Освітньо-кваліфікаційний рівень бакалавр

Спеціальність - 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТ

д.т.н., проф. Азаров О.Д.

"23" квітня 2024 р.

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Кулішу Дмитру Володимировичу

1 Тема роботи: програмний засіб із вебінтерфейсом для ветеринарної клініки, керівник роботи к.т.н., ст. викл. каф. ОТ Дудник О. В. затверджені наказом вищого навчального закладу від 11.03.2024 р. №80.

2 Термін подання студентом роботи 14.06.2024

3 Вихідні дані до роботи: обґрунтування доцільності створення програмного засобу із вебінтерфейсом для ветеринарної клініки, технології реалізації вебінтерфейсів та роботи з базами даних.

4 Зміст розрахунково-пояснювальної записки: вступ, аналіз сучасних веб-технологій для реалізації вебінтерфейсів, розробка програмного засобу із вебінтерфейсом, тестування та розгортання програмного засобу.

5 Перелік графічного матеріалу — структурна схема програмного засобу із вебінтерфейсом, лістинг програми.

6 Консультанти розділів роботи наведені в таблиці 1.

Таблиця 1 — Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ 1-3	к.т.н., ст.викл. каф. ОТ Дудник О.В.	12.03.2024 дата	21.05.2024 дата
		підпис	підпис

7 Дата видачі завдання 12.03.2024 р.

8 Календарний план виконання роботи показаний в таблиці 2.

Таблиця 2 — Календарний план виконання роботи

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі роботи	09.03.2024	вик.
2	Аналіз сучасних технологій для розробки вебінтерфейсів	12.03.2024 — 16.03.2024	вик.
3	Огляд підходів проектування баз даних та програмування веб-додатків	17.03.2024 — 21.03.2024	вик.
4	Аналіз та вибір структури програмного засобу	22.03.2024 — 27.03.2024	вик.
5	Аналіз існуючих аналогів програмних засобів ветеринарних клінік	28.03.2024 — 05.04.2024	вик.
6	Створення бази даних програмного засобу	06.04.2024 — 11.04.2024	вик.
7	Програмна реалізація вебінтерфейсу	12.04.2024 — 06.05.2024	вик.
8	Проектування обробки форми запису на прийом та збереження введених даних	07.05.2024 — 14.05.2024	вик.
9	Тестування та розгортання програмного засобу	15.05.2024 — 17.05.2024	вик.
10	Оформлення пояснювальної записки та ілюстративного матеріалу	18.05.2024 — 30.05.2024	вик.
11	Перевірка якості виконання бакалаврської роботи	30.05.2024 — 08.06.2024	вик.

Студент _____ Куліш Д. В.

Керівник роботи _____ Дудник О. В.

АНОТАЦІЯ

УДК 004.4

Куліш Д. В. Програмнй засіб із використанням вебінтерфейсу для ветеринарної клініки. Бакалаврська дипломна робота зі спеціальності 123 «Комп'ютерна інженерія», освітня програма «Комп'ютерна інженерія». Вінниця: ВНТУ, 2024. 92 с.

На укр. мові. Бібліогр.: 23 назви; рис.: 41; табл.: 4.

У бакалаврській дипломній роботі було розроблено програмний засіб із використанням вебінтерфейсу для ветеринарної клініки. Він сприяє підвищенню ефективності роботи клініки та задоволенню потреб її клієнтів. У загальній частині розглянуто сучасні технології для реалізації вебінтерфейсів, а також методи для роботи з базами даних. У технологічній частині представлено деталі виконання розробки програмного засобу.

Ключові слова: вебінтерфейс, програмний засіб, ветеринарна клініка, онлайн-запис, JavaScript, PHP, база даних.

ABSTRACT

Kulish D. V. Software tool using a web interface for a veterinary clinic. Bachelor qualification work on specialty 123 «Computer engineering», educational program «Computer engineering». Vinnytsia: VNTU, 2024. 92 p.

In Ukrainian language. Bibliographer: 23 titles; Fig.: 41; table: 4.

In the bachelor's qualification work, a software tool using a web interface for a veterinary clinic was developed. It helps to improve the efficiency of the clinic and meet the needs of its clients. the general part of the qualification work discusses modern technologies for implementing web interfaces, as well as methods for working with databases. The technological part deals with the development of the software tool.

Key words: web interface, software tool, veterinary clinic, online appointment, JavaScript, PHP, database.

ЗМІСТ

ВСТУП

Сучасні технології все більше проникають в наше життя, змінюючи спосіб його організації. Інтернет став невід'ємною частиною нашого існування, і його використання постійно зростає. Це стосується і сфери ветеринарної медицини.

Ветеринарна клініка — це заклад, який надає послуги з лікування та догляду за тваринами. Вона повинна бути доступною для потенційних клієнтів, а також інформативною та зручною у використанні. Інтерактивний веб-сайт є ідеальним способом вирішення цих завдань.

Програмний засіб із вебінтерфейсом ветеринарної клініки дозволяє:

— надавати інформацію про послуги клініки, її лікарів і персонал, а також про ціни на послуги;

— забезпечувати взаємодію з клієнтами, наприклад, для запису на прийом або консультації;

— розміщувати рекламу клініки в Інтернеті.

Програмний засіб ветеринарної клініки може бути статичним або динамічним (інтерактивним). Статичний вебінтерфейс містить лише статичну інформацію, яка не змінюється з часом. Інтерактивний вебінтерфейс дозволяє користувачам взаємодіяти з ним, наприклад, заповнювати форми, замовляти товари або послуги.

Актуальність теми виправдовується тим, що програмні засоби із вебінтерфейсами стають невід'ємною частиною сучасного бізнесу, і ветеринарна клініка, яка надає можливість запису на прийом через веб-додаток, дає клієнтам можливість забезпечити собі необхідний сервіс у зручний для них час, без зайвих телефонних дзвінків та очікування в черзі. Тим самим, наявність програмного засобу у ветеринарної клініки стає важливою конкурентною перевагою, що дозволяє виділятись на фоні конкурентів та залучати нових клієнтів.

Мета роботи полягає у підвищені ефективності роботи клініки та задоволенню потреб її клієнтів за рахунок створення програмного засобу із використанням вебінтерфейсу, який враховує специфіку ветеринарної сфери.

Завдання роботи, які необхідно вирішити для досягнення поставленої мети:

— проаналізувати сучасні веб-технології для реалізації вебінтерфейсів;

— розробити структуру та дизайн програмного засобу із вебінтерфейсом;

— розробити функціонал програмного засобу;

— виконати тестування функціоналу та сумісності програмного засобу.

Об'єктом дослідження є процеси надання послуги з лікування та догляду за тваринами у ветклініках.

Предметом дослідження є технології та засоби, які використовуються для розробки програмних засобів із вебінтерфейсом для ветклінік.

Для досягнення поставленої мети будуть використовуватися такі **методи дослідження**:

— аналітичний метод використовується для аналізу потреб клієнтів ветклінік та того, яким чином існуючі програмні засоби ветеринарних клінік задовольняють ці потреби;

— аналітичний метод також для аналізу сучасних веб-технологій та засобів побудови систем із вебінтерфейсом;

— синтетичний метод використовується для синтезу нових рішень на основі отриманих даних, розробки структури та дизайну програмного засобу, розробки функціоналу веб-сайту;

— експериментальний метод використовується для тестування вебінтерфейсу.

Новизна дослідження полягає у створенні програмного засобу із вебінтерфейсом, який оптимізований для використання у сучасній ветеринарній сфері України. Він реалізує можливості особистого кабінету користувача, онлайн запису на прийом у клініку та забезпечує інтерактивну взаємодію з клієнтами, що підвищує ефективність роботи клініки та задоволенню потреб її клієнтів.

1 АНАЛІЗ СУЧАСНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ВЕБІНТЕРФЕЙСІВ

1.1 Огляд технологій реалізації розмітки та стилів веб-додатків

Існує два основних способи реалізації розмітки та стилів веб-додатків:

1) використання HTML та CSS;
2) використання фреймворків та бібліотек.

HTML — Це мова гіпертекстової розмітки, що використовується для визначення структури та вмісту веб-сторінки. Вона описує елементи сторінки, заголовки, абзаци, зображення та посилання.

CSS — Це каскадні таблиці стилів, які використовуються для стилізації зовнішнього вигляду веб-сторінки. CSS визначає колір, шрифт, розмір, розташування та інші візуальні властивості елементів гіпертекстової розмітки.

Ці технології дозволяють створювати динамічні веб-сторінки з використанням таких елементів, як анімація, відео, інтерактивні карти та ін.

Нижче наведено приклади того, як HTML5 та CSS можна використовувати для створення інтерактивності:

— мова гіпертекстової розмітки HTML5 додає до стандарту ряд нових елементів і атрибутів, які дозволяють створювати анімацію на веб-сторінках, елемент <animate> дозволяє створювати анімацію властивостей елементу;

— каскадні таблиці стилів CSS дозволяють відстежувати стан елементів на веб-сторінках, наприклад, чи є елемент видимим або активним, що можна використовувати при створенні таких інтерактивних елементів, як випадаючі меню або кнопки;

— CSS дозволяє реагувати на події на веб-сторінках, наприклад, натискання клавіш або миші, що можна використовувати для створення таких елементів, як наприклад форми;

— HTML5 підтримує вбудовані аудіо та відео елементи, що дозволяє легко додавати медіафайли на веб-сторінки.

Фреймворки та бібліотеки пропонують готові компоненти та функціональні можливості, які спрощують розробку веб-сайтів. Вони можуть включати в себе попередньо написаний код HTML, CSS та JavaScript, а також інструменти для створення динамічних веб-сторінок та інтерактивних елементів [1].

1.2 Аналіз технологій програмування функціоналу програмного засобу із вебінтерфейсом

Для створення динамічних та інтерактивних веб-сайтів, які отримують дані з баз даних, використовується комбінація трьох основних технологій — JavaScript, PHP та MySQL.

Мова програмування JavaScript. Дозволяє як на стороні клієнта, так і на стороні сервера, робити веб-сторінки більш інтерактивними, додаючи певні динамічні функції. При впровадженні інтерактивності на веб-сайт, дуже зручно використовувати JavaScript для валідації (перевірки правильності введених даних) форми, а також для відправлення даних на сервер [2].

PHP (гіпертекстовий препроцесор) — це мова програмування на стороні сервера, яка використовується для створення динамічного вмісту веб-сторінок. Вона може отримувати доступ до баз даних, обробляти дані користувачів, генерувати HTML-код та надсилати його до веб-браузера.

PHP часто використовується для створення веб-додатків, таких як форуми, блоги та інтернет-магазини.

1.3 Огляд підходів для роботи з базою даних

База даних — структурована система зберігання інформації, яка дозволяє зручно зберігати, керувати та отримувати доступ до даних. База даних**,** полегшує пошук, аналіз та оновлення інформації.

Існує декілька типів баз даних:

— реляційні бази даних;

— NoSQL бази даних;

— ієрархічні бази даних.

Реляційні бази даних зберігають дані в таблицях із чітко визначеними зв’язками. Цей тип баз даних є найбільш використовуваним у веб-додатках.

NoSQL системи зберігання інформації є більш гнучкими, ніж реляційні і добре підходять для неструктурованих даних, таких як JSON. Їх часто використовують у мобільних додатках.

Ієрархічні бази даних зберігають дані в деревоподібній структурі, де кожен елемент може мати декілька підпорядкованих елементів.

Для роботи з системою зберігання інформації, потрібно користуватись системами керування базами даних (СКБД) — це програмне забезпечення, що використовується для створення, керування та доступу до баз даних.

До найпопулярніших СКБД можна віднести наступні системи:

— MySQL;

— Oracle Database;

— Interbase;

— Microsoft SQL Server.

Розглянемо найбільш використовувану з них, MySQL — це система керування реляційними базами даних (СКБД), яка використовується для зберігання та керування даними веб-застосунків. Вона дозволяє створювати, читати, оновлювати та видаляти дані в базі. MySQL є популярною СКБД завдяки тому, що це безкоштовна програма, що має відкритий код. Вона проста в використанні та налаштуванні, легко масштабується при потребі та є надійною системою.

Одним з способів підключення до MySQL є використання PHP Data Objects (PDO) — розширення мови програмування PHP, яке забезпечує інтерфейс для роботи з різними базами даних, включаючи MySQL.

Підключитись та працювати з СКБД можливо ще за допомогою MySQLi, це також є розширенням для PHP, яке дозволяє взаємодіяти з базами даних. Проте, дане розширення може працювати лише з базами даних MySQL, в той час як PDO дозволяє підключатись до різних систем керування базами даних.

1.4 Аналіз існуючих аналогів програмних засобів ветеринарних клінік

Перед початком розробки програмного засобу, потрібно здійснити пошук аналогів вебінтерфейсів ветеринарних клінік, визначити їх переваги та недоліки, щоб не допустити такі ж помилки та використати переваги проаналізованих систем. Проаналізовано три веб-ресурси, які реалізують функції предметної області. Спершу обрано для аналізу програмний засіб ветеринарної клініки «Айболіт» (режим доступу: http://aybolit.superdovidka.ua/). Наступним — вебінтерфейс клініки «Зоополіс» (режим доступу: https://zoopolisvet.com/). Третім аналогом обрано веб-сторінку ветеринарної клініки «Улюбленець» (режим доступу: http://vetsait.com/). Заключною для аналізу було вирішено обрати веб-додаток ветеринарної клініки «VetHouse» (режим доступу: https://vethouse.com.ua).

Розглянемо детальніше представлені вище аналоги. Першим розглянемо програмний засіб клініки «Айболіт». На рисунку 1.1 зображено вигляд головної сторінки ветклініки. Дана сторінка має досить застарілий дизайн, весь текст відображається в кучі та має малі міжрядкові відступи, що робить читання тексту важким. Особистий кабінет користувача та онлайн запис на прийом відсутні у даному програмному засобі. Номер телефону клініки в шапці сторінки є не клікабельним.

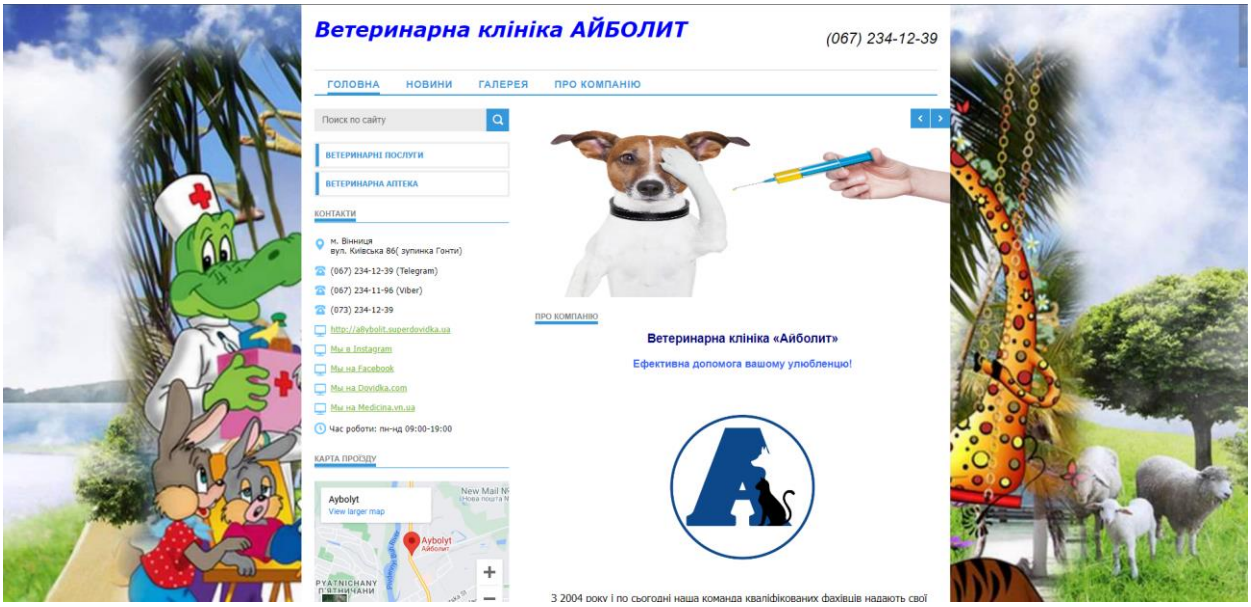

Рисунок 1.1 — Головна сторінка програмного засобу ветклініки «Айболіт»

Також, даний програмний засіб містить сторінку «Ветеринарна аптека» (рисунок 1.2), але при її відкритті, відображається лише картинка та немає детальної інформації про ветеринарні препарати, що знаходяться в клініці.

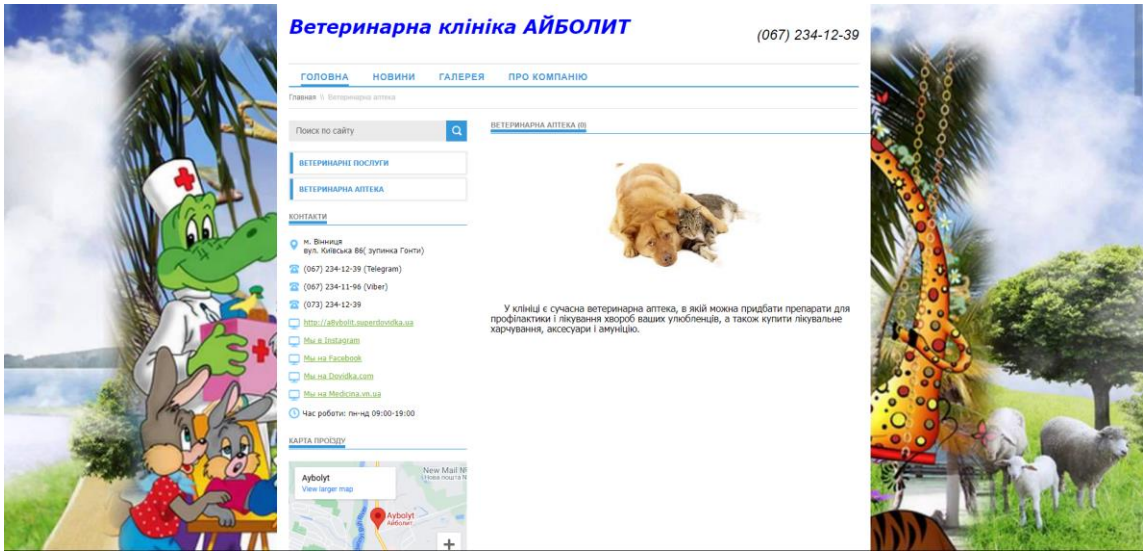

Рисунок 1.2 — Сторінка ветеринарної аптеки ветклініки «Айболіт»

Перейдемо до аналізу вебінтерфейсу клініки «Зоополіс». Вона має більш сучасний та комфортний для очей дизайн (рисунок 1.3), ніж попердній аналог. Є секція з послугами (рисунок 1.4), які надає ветклініка, контактні дані та мапа розташування. Якщо звернути увагу на функціонал сайту, можемо побачити що навігаційне меню та онлайн запис на прийом — відсутні.

Рисунок 1.3 — Головна сторінка клініки «Зоополіс»

В секції «Contact Us» відображений номер телефону, при натисканні на який, відкривається додаток «Телефон», що може бути зручним аспектом для клієнтів клініки. В даному програмного засобі, не пророблений дизайн в плані розділення секцій між собою. Таким чином, секція про послуги, частина з контактними даними та інші блоки сайту, зливаються в один, що робить сайт менш сегментованим та розсіює увагу потенційного клієнта. Програмний засіб клініки «Зоополіс» не має особистого кабінету. Проте, в нижньому блоці сайту, показані логотипи соціальних мереж, при натисканні на які, відкриваються вдповідні сторінки клініки в мережах «Facebook» та «Instagram».

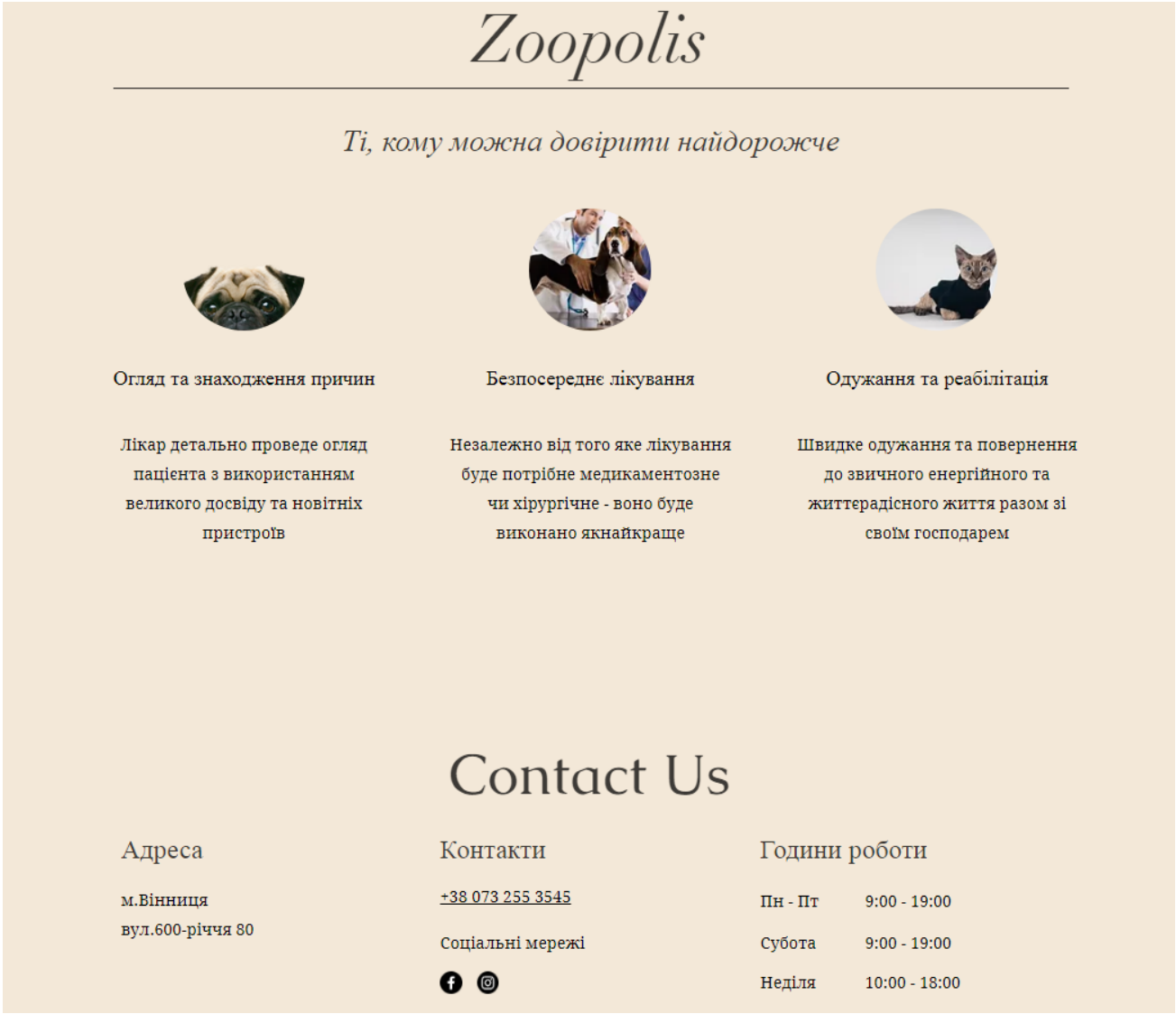

Рисунок 1.4 — Секція з інформацією про послуги та контактні дані

Третім аналогом, розглянемо сайт ветеринарної клініки «Улюбленець». Сторінка веб-сайту (рисунок 1.5) містить навігаційне меню, селектор мов, на якій може відображатись сайт. Шапка інтерфейсу потребує детальнішої пропрацьовки, над головним блоком відображається білий фон, на якому в суцільному рядку показується адреса клініки, контактні телефони, графік роботи, електронна пошта та соціальні мережі. Це не є найбільш вдалим рішенням з точки зору дизайну, тому що користувачу буде важко знайти потрібну інформацію.

Веб-сайт має функціонал відображення статей на сторінках «Новини ветклініки» та «Клінічні випадки». Список послуг, що надає клініка є клікабельними, але при натисканні на відповідну послуги, відкривається знову головна сторінка сайту. Також, реалізовано кнопку, що при натисканні повертає користувача на початок сторінки. Ще однією корисної функцією, доступною у вебінтерфейсі, є пошук по статтям (рисунок 1.6). Можна написати текст, який хочете знайти, у відповіднне поле, та після натискання на кнопку «Пошук», стаття що має необхідний текст відфільтрується, та відобразиться на екрані. Функціонал особистого кабінету, списку ветеринарних препаратів доступних в клініці та онлайн запису на прийом відсутній, що ускладнює взаємодію між потенційним клієнтом та самою ветеринарною клінікою.

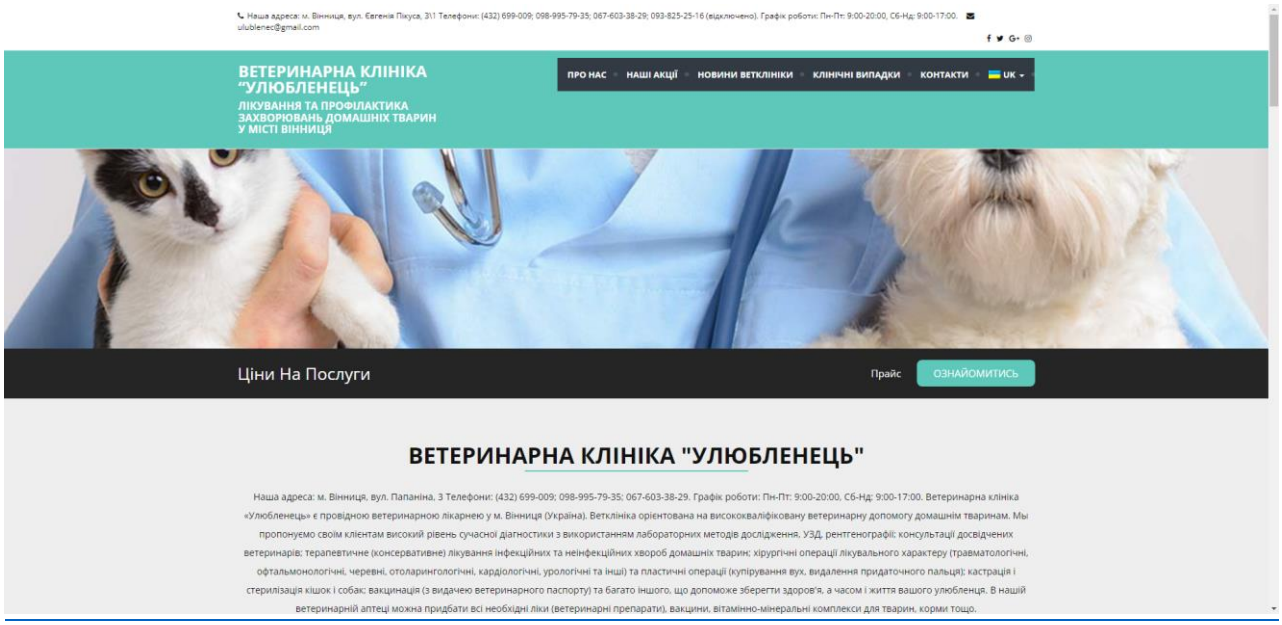

Рисунок 1.5 — Сторінка сайту ветеринарної клініки «Улюбленець»

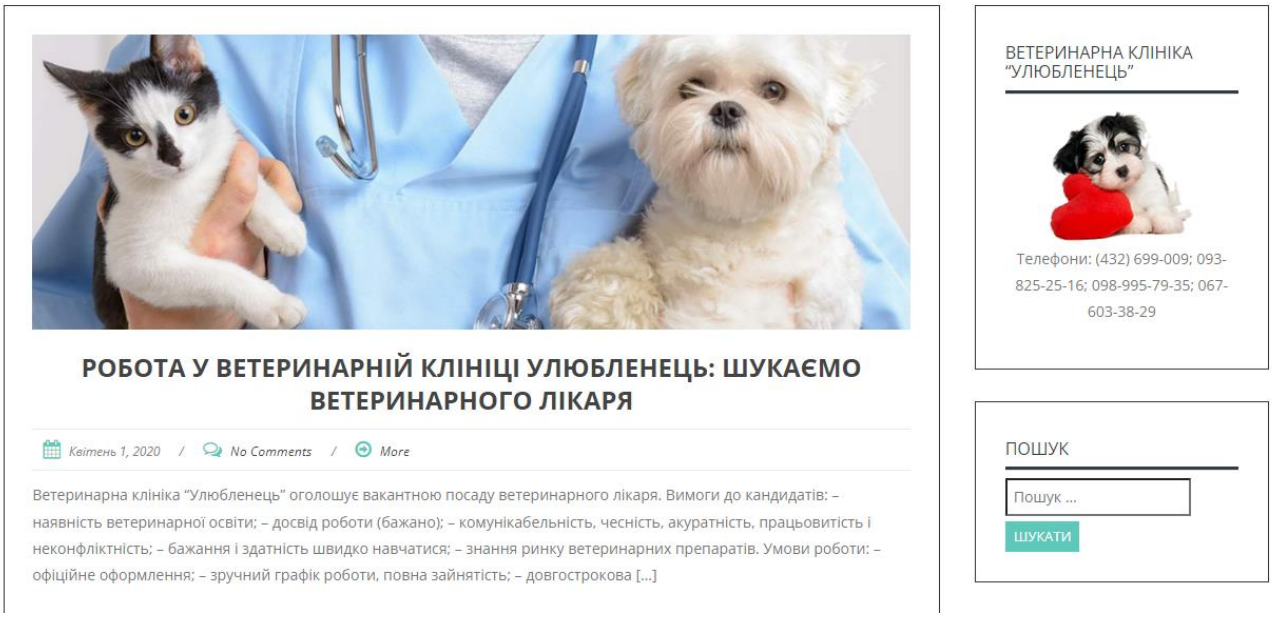

Рисунок 1.6 — Функціонал пошуку статей на сайті клініки «Улюбленець»

Останнім аналогом, розглянемо програмний засіб ветеринарної клініки «VetHouse». Даний вебінтерфейс має досить приємний дизайн, добре пропрацьоване меню, присутній функціонал онлайн запису на прийом, та є можливість залишити повідомлення керівнику клініки (рисунок 1.7). Меню сайту містить вкладку «Особистий кабінет», але при переході по ній, нічого не відображається.

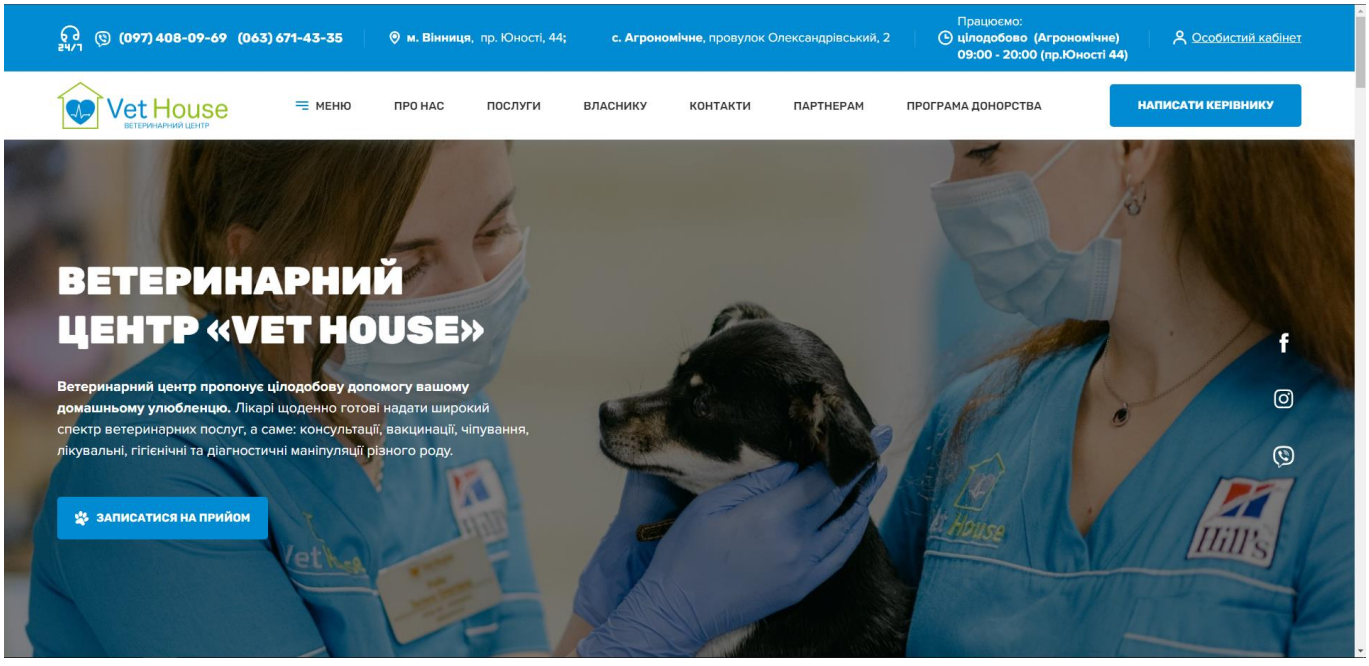

Рисунок 1.7 — Програмний засіб ветеринарної клініки «VetHouse»

Після огляду систем-аналогів було проаналізовано всі переваги та недоліки цих ресурсів. До переваг відносяться такі функції, як онлайн запис на прийом, пошук по сторінці та зручний користувацький інтерфейс. Основним недоліком є відсутність особистого кабінету користувача, а також не завжди інтуїтивний дизайн вебінтерфейсу, що ускладнює роботу з програмним засобом.

Порівняльна характеристика описаних аналогів наведена в таблиці 1.1.

Таблиця 1.1 — Порівняльна характеристика програмних засобів ветеринарних клінік

Назва ветеринарної клініки	Ветеринарна клініка «Айболіт»	Ветеринарна клініка «Зоополіс»	Ветеринарна клініка «Улюбленець»	Ветеринарна клініка «VetHouse»
Інтерфейс користувача	Користувацький інтерфейс є досить застарілим, малі відступи у тексті, через це його важко читати.	Має сучасний та комфортний для очей дизайн, відсутнє розділення блоків сайту.	Шапка інтерфейсу потребує детальнішої пропрацьовки, можливо покращити бічне меню.	Інтерфейс користувача є добре пропрацьованим, інтуїтивним, необхідну інформацію знайти легко.
Функціональність	Реалізовано слайдер із зображеннями тварин та пошук по сайту. Особистий кабінет та онлайн запис відсутні.	Розроблено блоки з послугами клініки, контактні дані, та блок з лікарями. Функціонал онлайн запису та особистий кабінет відстуні.	Створено пошук статей клініки, клікабельні послуги, категорії статей. Особистий кабінет та запис на прийом не реалізовані.	Реалізовано функціонал онлайн запису на прийом, повідомення керівнику, особистий кабінет не працює.

1.5 Структура вебінтерфейсу та системи навігації

1.5.1 Обгрунтування структури вебінтерфейсу

Згідно поставленого завдання від ветеринарної клініки, структура вебінтерфейсу повинна складатись з наступних елементів:

— шапка інтерфейсу є верхньою частиною сайту та має містити логотип ветклініки;

— навігаційне меню має бути списком посилань на різні розділи додатку;

— основний блок контенту є центральною частиною сайту та повинен містити найважливішу інформацію, наприклад форму для запису на прийом;

— додаткові блоки контенту повинні містити додаткову інформацію, яка може бути корисною для відвідувачів, в нашому випадку це може бути список ветеринарних препаратів, відгуки клієнтів або інформацію про лікарів клініки;

— футер вебінтерфейсу є нижньою частиною програмного засобу та мусить містити контакти клініки та інформацію про її розташування.

Розташування елементів на головній сторінці веб-додатку схематично показано на рисунку 1.8.

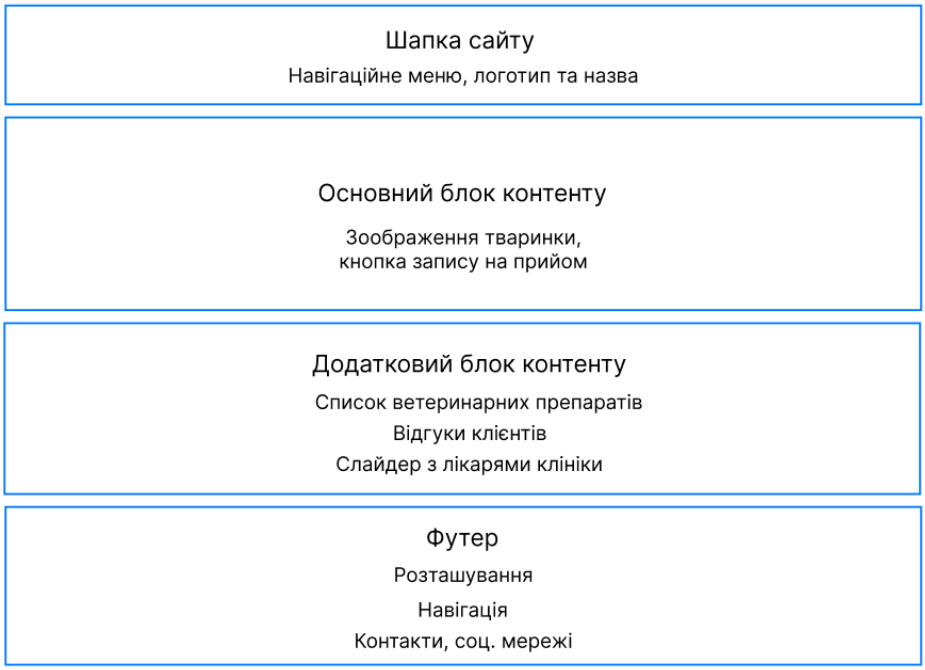

Рисунок 1.8 — Схема головної сторінки

1.5.2 Мапа вебінтерфейсу (система навігації)

Мапа навігації вебінтерфейсу — це візуальне представлення системи навігації сайту. Вона може бути представлена у вигляді меню, панелі інструментів або інших елементів інтерфейсу користувача.

Мапа навігації вебінтерфейсу повинна бути легкою для розуміння і використання [3]. Вона повинна відповідати цільовій аудиторії сайту та його цілям. Багато веб-сайтів мають видимі для користувачів схеми сайту, які представляють структуру сайту у вигляді дерева. Ці схеми допомагають користувачам легко знайти потрібні сторінки, а також можуть використовуватися пошуковими системами.

Система навігації вебінтерфейсу зображена на рисунку Б.1.

1.6 Вибір дизайну вебінтерфейсу та його колірної гами

Згідно поставлених ветеринарною клінікою вимог, програмний засіб із вебінтерфейсом повинен виглядати стильно, сучасно та приємно для очей [4]. У якості основних кольорів пропонувалось використовувати білі та світло-фіолетові відтінки. Основу додатку повинні складати фотографії високої якості.

При виборі дизайну програмного засобу було враховано наступні фактори:

— вебінтерфейс присвячений темі ветеринарії, тому дизайн повинен відображати цю тему;

— програмний засіб повинен забезпечувати користувачам доступ до всіх необхідних функцій, таких як запис на прийом, інформація про послуги, контактні дані, інформація про персонал, тощо;

— основним кольором веб-додатку є світло-фіолетовий колір, він створює атмосферу спокою і комфорту, що може бути важливим для власників домашніх тварин, які шукають ветеринарну допомогу для своїх улюбленців;

— фон програмного засобу має бути світлий, він робить сайт більш приємним для очей і полегшує читання тексту;

— текст на сайті повинен мати темний колір, такий текст на світлому фоні добре читається.

2 РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ ІЗ ВЕБІНТЕРФЕЙСОМ ДЛЯ ВЕТЕРИНАРНОЇ КЛІНІКИ

2.1 Створення макету сторінок програмного засобу в сервісі Figma

Figma — це векторний онлайн-сервіс для розробки інтерфейсів та прототипування, який дозволяє користувачам спільно працювати над проектами [5]. Сервіс доступний у двох форматах: у браузері та як клієнтський застосунок на ПК. Figma зберігає онлайн-версії всіх файлів, з якими працював користувач.

Макет програмного засобу було розроблено в браузерній версії застосунку. Створений макет зображено на рисунку 2.1.

Рисунок 2.1 — Макет програмного засобу

2.2 Розробка логотипу

Як ідею для логотипу я обрав логотип із текстом. Він буде складатись з картинки та назви ветеринарної клініки під нею. Такий логотип є простим та зрозумілим, що робить його легко впізнаваним [6]. Це може бути особливо важливо для ветеринарної клініки, адже клієнти будуть мати можливість легко знайти та запам’ятати логотип.

Після ідеї я перейшов до створення варіантів ескізів в застосунку Figma. Ескізи допомагають розвинути свою ідею та побачити, як вона виглядає на практиці. При створенні ескізу варто поекспериментувати з різними формами та ідеями логотипу.

Створенні ескізи зображено на рисунку 2.2.

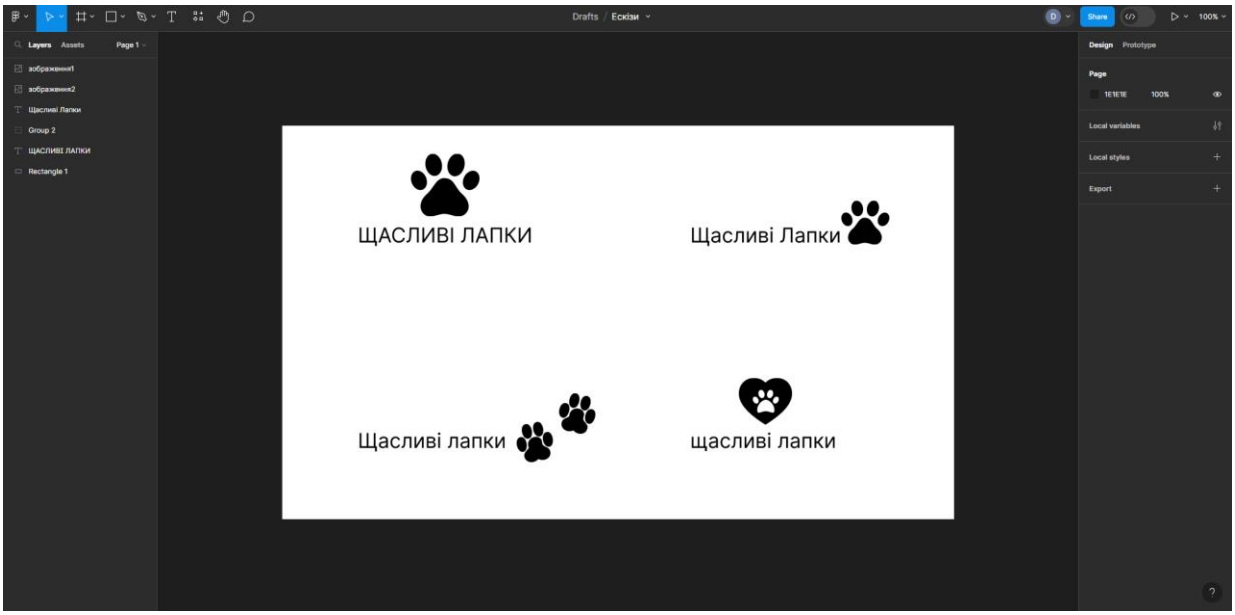

Рисунок 2.2 — Зображення створених ескізів в додатку Figma

Після створення та вибору варіанту ескізу, було проведено проектування векторної версії логотипу, так як векторні зображення завантажуються набагато швидше і мають кращу якість. Векторна графіка виглядає оригінально, створює ефект цифрового малюнка та може створювати складні зображення, які неможливо відтворити за допомогою растрової графіки, це дозволить створити логотип, що буде виділятись на фоні конкурентів. Процес створення векторного логотипу зображено на рисунку 2.3.

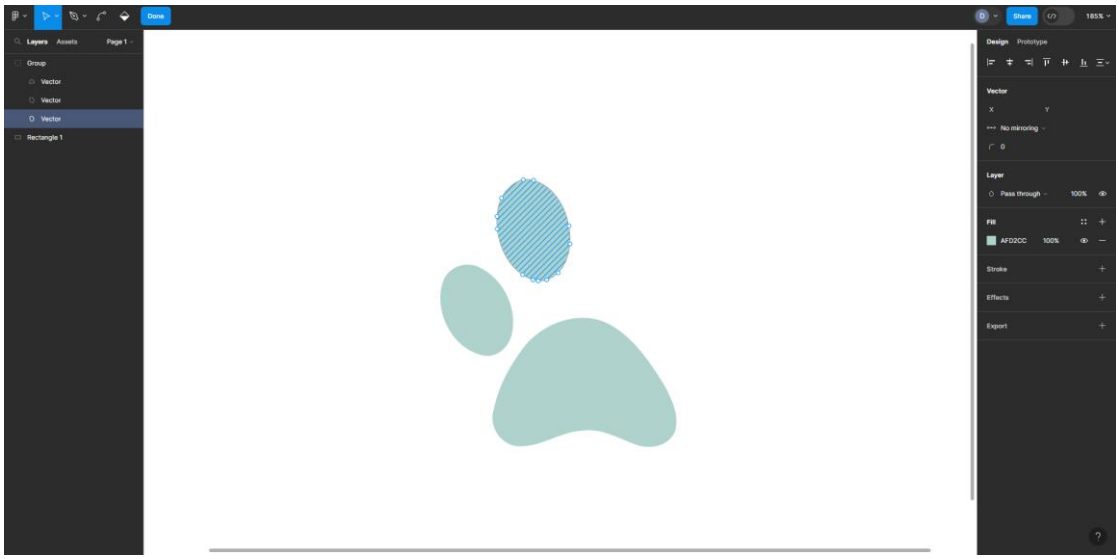

Рисунок 2.3 — Процес створення векторного логотипу за допомогою інструмента «Перо»

Наступним етапом після створення векторної версії логотипу, є додавання кольорів за необхідності та підбір шрифтів для назви клініки (рисунок 2.4).

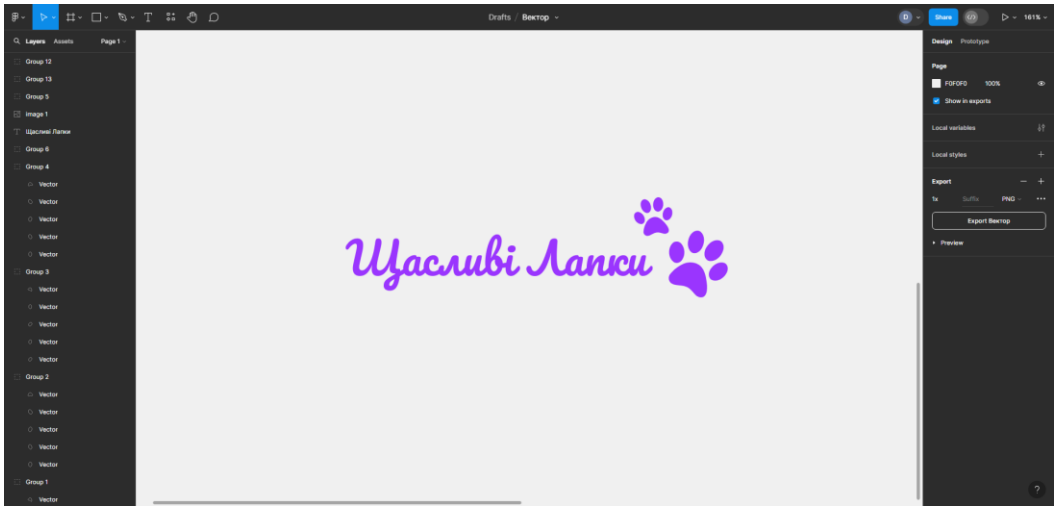

Рисунок 2.4 — Остаточна версія логотипу

Відображення готового логотипу на веб-сторінці, зображено на рисунку 2.5.

Рисунок 2.5 — Відображення логотипу на вебінтерфейсі

2.3 Архітектура програмного засобу

Створення архітектури додатку — це багатоетапний процес, спрямований на чітке визначення структури та компонентів майбутньої системи.

Загалом відокремлюють три типи верхньорівневої архітектури веб додатків, це:

— моноліт;

— мікросервіси;

— серверлес.

У монолітній архітектурі всі компоненти та модулі системи щільно інтегровані один з одним і нерозривно пов'язані між собою. Це означає, що зміна одного компонента неминуче впливає на функціонування інших.

Мікросервісна архітектура — це підхід до розробки програмного забезпечення, який розбиває систему на дрібні, незалежні модулі (мікросервіси). Ці модулі комунікують між собою за допомогою повідомлень, що робить систему гнучкою та масштабованою.

Серверлес архітектура — це рішення розробки програмного засобу, яке звільняє розробників від необхідності самостійно налаштовувати та обслуговувати сервери. Замість цього, серверлесс зосереджує увагу на розробці функціоналу, а завдання з розгортання та масштабування автоматично беруть на себе хмарні платформи.

В програмному засобі із вебінтерфейсом для ветеринарної клініки, мікросервіси використовуються лише для окремих функцій, і їх вплив на моноліт мінімальний. В такому випадку дану систему можна назвати монолітом з мікросервісними компонентами. Мікросервіси не змінюють загальну структуру моноліту, а лише доповнюють його функціонал.

Використання моноліту з мікросервісними компонентами є виправданим, оскільки:

— функціонал системи є відносно стабільний, без плану значних змін в найближчий час;

— розробка та підтримка монолітної системи зазвичай дешевші, ніж мікросервісної;

— використовування мікросервісів для розширення функціоналу системи можливе без необхідності переписувати весь моноліт.

Порівняльна характеристика архітектурних рішень наведена в таблиці 2.1.

Таблиця 2.1 — Порівняльна характеристика архітектурних рішень

Характеристика	Моноліт	Мікросервіси	Серверлес
Розгортання	Просте та швидке	Складніше, потребує налагодження взаємодії	Складніше, потребує налаштування хмарної інфраструктури
Розробка	Швидка, всі компоненти в одній кодовій базі	Складніша, потребує розробки декількох підсистем	Проста, кожен модуль - це окремий код
Масштабування	Можливе лише повністю	Можливе на рівні окремих модулів	Можливе на рівні окремих модулів
Надійність	При збої не працює вся система	При збої одного модуля система може працювати далі	При збої одного модуля система може працювати далі
Вартість	Дешевший в розробці	Дорожчий в розробці	Дешевший в розробці, але може бути дорожчий в експлуатації
Контроль	Розробник має повний контроль	Розробник має менший контроль	Розробник має дуже обмежений контроль

Архітектура програмного засобу із вебінтерфейсом показана на рисунку 2.6.

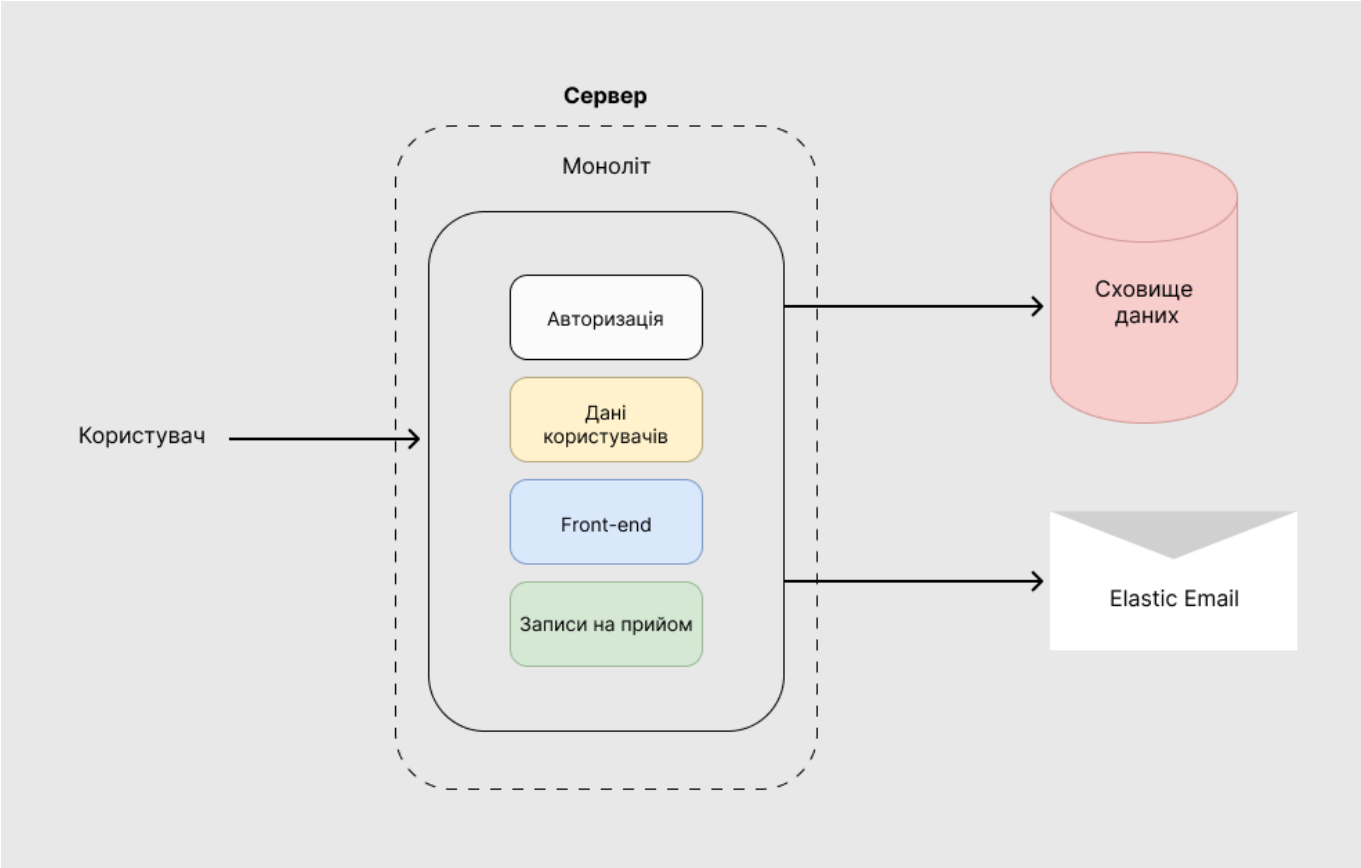

Рисунок 2.6 — Схема архітектури програмного засобу

2.4 Реалізація головної сторінки вебінтерфейсу

Відповідно до розробленого макету програмного засобу (рисунок 2.1), було розпочато розробку головної сторінки веб-додатку, починаючи з шапки (навігаційного меню). Згідно поставлених вимог, шапка сайту має складатись з блоку, де відображається логотип та навігаційного меню. Кожен пункт меню має містити посилання на відповідну секцію сторінки. Також, активний елемент навігіаційного меню, має відрізнятись від інших елементів списку. Це реалізовано за допомогою додавання класу «active» до назви відповідної сторінки у файлі коду. Логотип клініки має бути клікабельним та перекидати користувача на головну сторінку сайту.

Наступним етапом було створення основного блоку головної сторінки, де має розміщуватись назва ветеринарної клініки, її слоган, кнопка онлайн запису на прийом у ветклініку. Було також вирішено, додати зображення собаки в основний блок, щоб привернути увагу користуваічв та підкреслити що веб-додаток має саме тематику

ветеринарної клініки. Фон основного блоку головної сторінки має світло-фіолетовий колір, основний текст на сторінці відображається чорним кольором. Текст кнопки «Запис на прийом» було вирішно зробити світлого кольору, так як колір фону кнопки темно-фіолетовий, для того щоб кнопка виділялась поміж інших елементів вебінтерфейсу. Відповідно, темний текст на темному фоні буде не читабельним.

Основна секція веб-додатку складається з трьох параграфів. Перший параграф відображає назву ветеринарної клініки. Другий — акцентує увагу користувачів на слогані клініки. Привертання уваги здійснено за рахунок збільшення розміру шрифту девізу. Встановлено розмір тексту — 2.6 rem, тобто 41.6 пікселів. Одиниці Rem (кореневий EM) встановлюють розмір шрифту елемента відносно розміру кореневого елемента, та за замовчуванням 1 rem = 16 пікселів. Такий тип одиниць розміру використовується для зручності глобального масштабування, тому елементи, які планується масштабувати, зазвичай задаються в rem. Третій параграф основного блоку, відображає інформацію про те, що клініка надає широкий спектр ветеринарних послуг цілодобово. Фрагмент реалізації основої секції наведено в лістингу 2.1.

Лістинг 2.1 — Фрагмент реалізації основного блоку вебінтерфейсу

```
    <div class="container">
<div class="content">
<span>ВЕТЕРИНАРНА КЛІНІКА <b>«ЩАСЛИВІ ЛАПКИ»</b></span>
        </div> <h1>Ваші домашні улюбленці важливі для нас</h1>
        <p>Клініка надає широкий спектр послуг для вашого вихованця цілодобово. </p> <button class="modal-open"><a href="#ex1" class="btn" id="showForm"  rel="modal:open">Запис на прийом</a>
</button>
        <img src="./images/dog-png-22638.png" aria-label="puppy" alt="puppy" />
```

Відображення розробленого основного блоку вебінтерфейсу показано на рисунку 2.7.

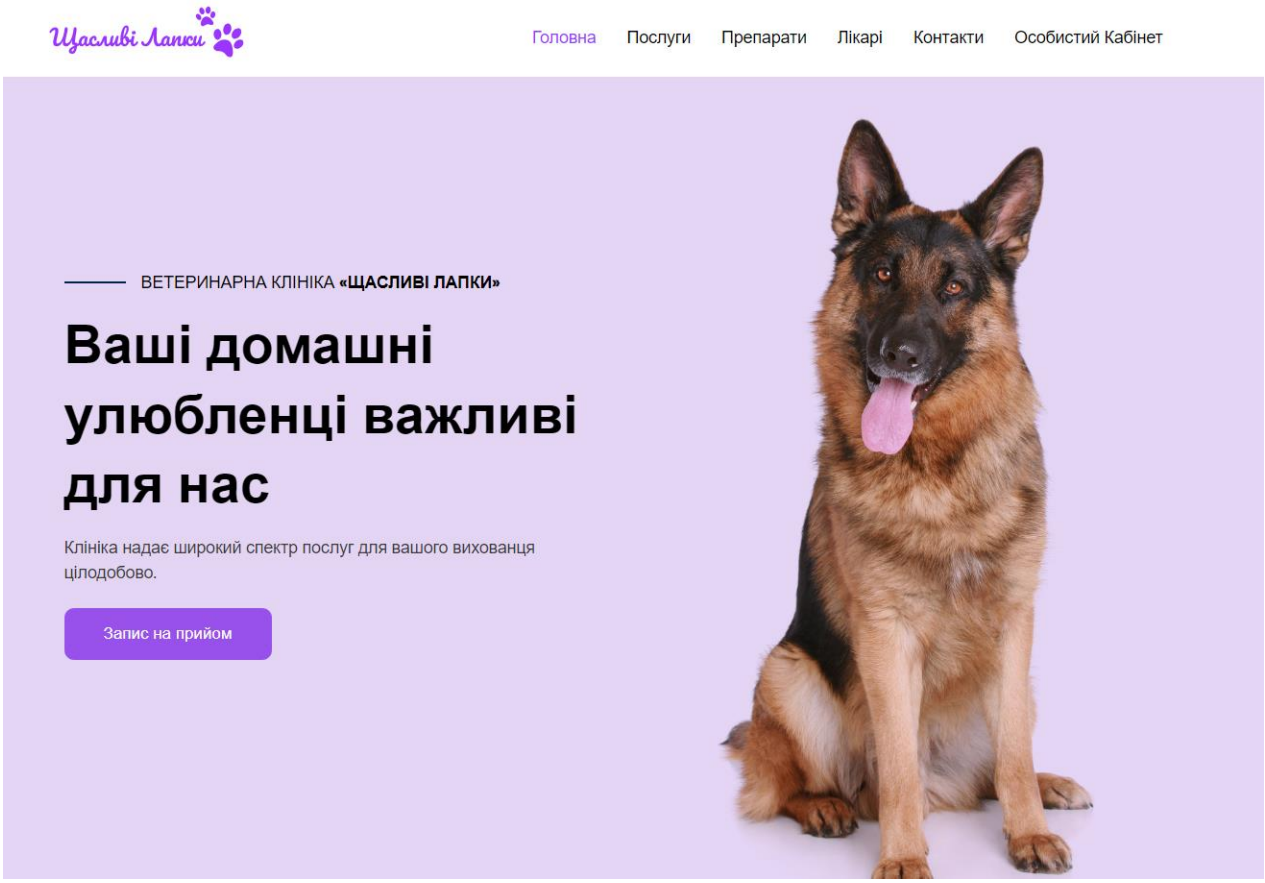

Рисунок 2.7 — Основний блок вебінтерфейсу

2.4 Імплементація модального вікна з формою запису на прийом у ветклініку

2.4.1 Розробка інтерфейсу модального вікна

Наступним етапом створення програмного засобу, була розробка модального вікна, що має форму запису на прийом у клініку. Для реалізації розмітки модального вікна було використано HTML тег — form, який було загорнуто в блок з класом modal. В секцію form було додано усі необхідні поля форми, такі як:

— ім’я користувача;

— номер телефону;

— електронна пошта;

— ім’я улюбленця;

— тип прийому;

— дата прийому;

— проблема.

Полю тип прийому було присвоєно тип — випадаючий список, з встановленими варіантами відповіді. Полю дата прийому було присвоєно тип — datepicker. При натисканні на поле «Дата прийому» відображається календар, де користувач може обрати потрібну йому дату для запису на прийом у ветеринарну клініку.

При проєктуванні логіки модального вікна, було використано мову програмування JavaScript [7]. Блок з вікном повинен мати стиль display: none при завантаженні сторінки, та змінювати свій стиль на display: block при натисканні на кнопку «Запис на прийом». Щоб це реалізувати, спочатку було проініціалізовано функцію, яка за допомогою селекторів знаходила саме модальне вікно та змінювала йому стиль, з display: none на display: block та навпаки, при закриванні вікна, що дозволить модальному вікну з’являтись на екрані та зникати з екрану.

Всередині блоку було реалізовано форму для зворотнього зв’язку, за допомогою тегу <form>. Додані відповідні поля до форми. Також, було додано CSS стилі до форми.

Відображення спроєктованої форми запису на прийом показано на рисунку 2.8.

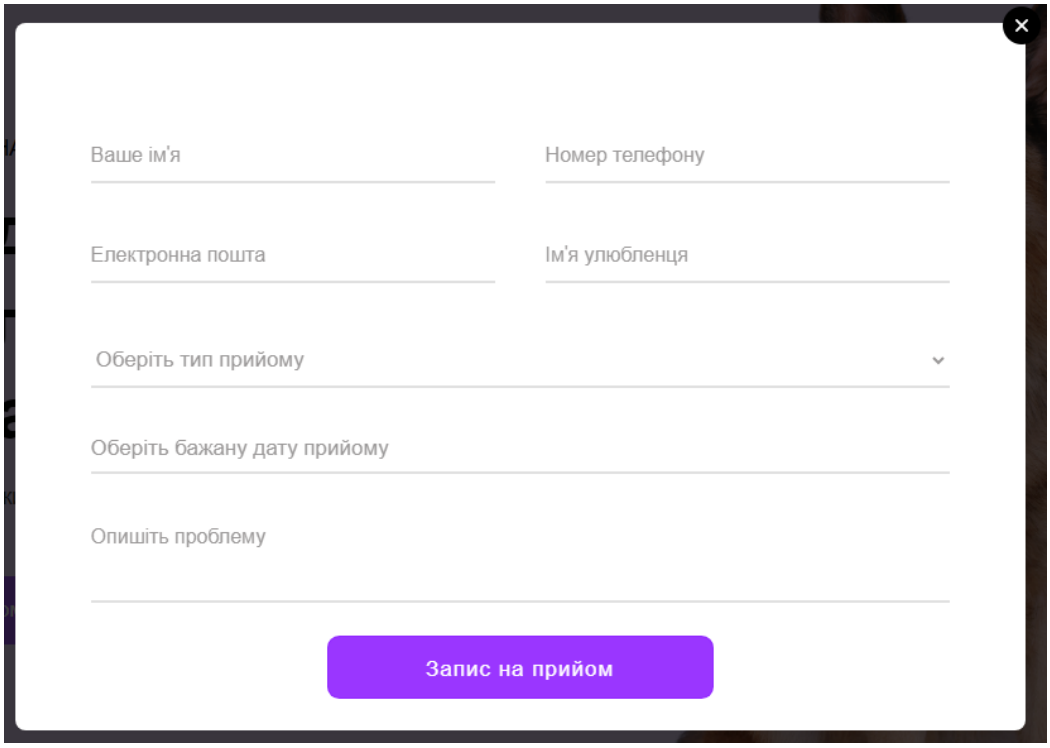

Рисунок 2.8 — Модальне вікно з формою запису на прийом у ветеринарну клініку

2.4.2 Реалізація валідації форми

Після створення форми, було вирішено додати до неї валідацію полів. Було додано до полів форми атрибуи <pattern>, де вказується регулярний вираз [8], згідно з яким перевіряються дані в полі форми. Якщо присутній цей атрибут, то форма не буде відправлятися, поки поле не буде заповнено правильно.

Реалізовано JavaScript-код, за домогою якого показується повідомлення про неправильне введення даних, наприклад: «Неправильний формат телефону.». Щоб створити валідацію, було створенно змінні, та за допомогою методу document.getElementById, знайдено відповідні елементи на сайті та призначено їх цим змінним.

Властивість nextElementSibling є властивістю тільки для читання, та повертає наступний елемент на тому ж рівні дерева, в даному випадку це label.

За допомогою змінних було реалізовано валідацію поля. Якщо, поле є пустим — показується помилка, наприклад: «Введіть ваш номер телефону», та якщо поле не відповідає паттерну який вказаний в HTML-коді, показується повідомлення: «Неправильний формат телефону.» (рисунок 2.9). Якщо, після неправильного заповнення поля, користувач вводить коректне значення, повідомлення про помилку пропадає. Реалізація валідації поля «Номер телефону» наведена в лістингу 2.2.

Лістинг 2.2 — Додавання атрибутів <pattern> до полів форми

```
<input type="text" required id="userName" pattern="[А-яІі][а-яІі]+">
 <input type="tel" required id="phoneNumber" pattern="[0-9]{10}">
<input type="email" required id="userMail" pattern=".+@gmail\.com">
```

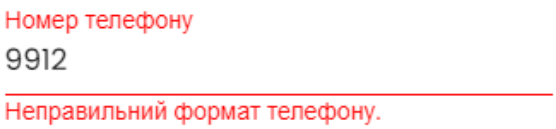

Рисунок 2.9 — Відображення валідації поля «Номер телефону» в формі запису на прийом

Валідація усіх наступних полів реалізована аналогічним чином, окрім валідації поля «Бажана дата прийому» оскільки це поле має тип «datepicker» та є обов’язковим для заповнення, для нього було зроблено валідацію вже при відправці форми. Якщо дата прийому не вибрана, за допомогою функції alert(); з’являється повідомлення «Оберіть бажану дату прийому.»

Результат відображення модального вікна, при натисканні на кнопку «Запис на прийом» із пустим полем «Бажана дата прийому» наведено на рисунку 2.10.

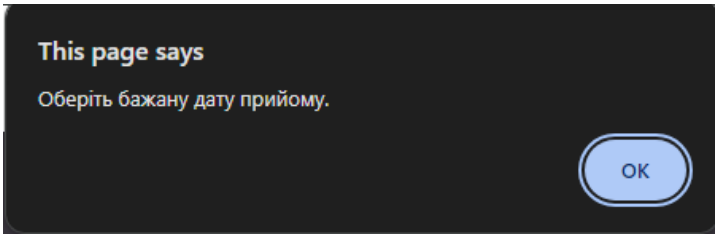

Рисунок 2.10 — Відображення модального вікна із повідомленням «Оберіть бажану дату прийому.»

2.5 Реалізація відправки форми на пошту

Відправку форми на пошту було реалізовано за допомогою бібліотеки SMTP.js.

SMTP.js — це бібліотека JavaScript, яка дозволяє надсилати електронні листи безпосередньо з веб-браузера без необхідності в серверному обладнанні. Вона фактично діє як клієнтська обгортка для Simple Mail Transfer Protocol (SMTP), який є стандартним протоколом для надсилання електронних листів через Інтернет.

SMTP.js настільки ж безпечний, як і сам SMTP [9]. Всі з'єднання використовують шифрування TLS, а облікові дані не розкриваються на стороні клієнта. Вміст листа та дані залишаються локальними для браузера.

Дана бібліотека допомагає в обробці зв’язку із реальним зовнішнім сервером протоколу листування. Для коректного налаштування, потрібно надати бібліотеці конфігураційні дані, такі як адреса сервера, порт, ім’я користувача та пароль для автентифікації.

SMTP.js дозволяє вказати відправника, одержувача, тему, текст листа та навіть вкладення для електронних листів.

Для роботи бібліотеки, її потрібно підключити в проєкт. Це було зроблено за допомогою додавання скрипту CDN в файл коду.

Щоб покращити та спростити процес відправки електронних листів, було прийняте рішення використовувати хмарний SMTP-сервіс — Elastice Email.

Elastic Email — це універсальна платформа для доставки електронних листів. Вони надають SMTP та HTTP API, а також користувацький інтерфейс для надсилання листів, переглядання статистики по листам та інше [10].

Переваги сервісу Elastice Email перед іншими платформами доставки електронних листів:

— платформа пропонує безкоштовний план, який дозволяє надсилати до 4000 листів на місяць, що є значною економією витрат порівняно з іншими SMTP-сервісами;

— сервіс надає детальну аналітику про відправлені листи, включаючи кількість відкритих листів, що дозволить використовувати ці дані ветеринарній клініці для оцінки ефективності процесу онлайн-запису та виявлення потенційних проблем.

Вигляд головної сторінки платформи зображено на рисунку 2.11.

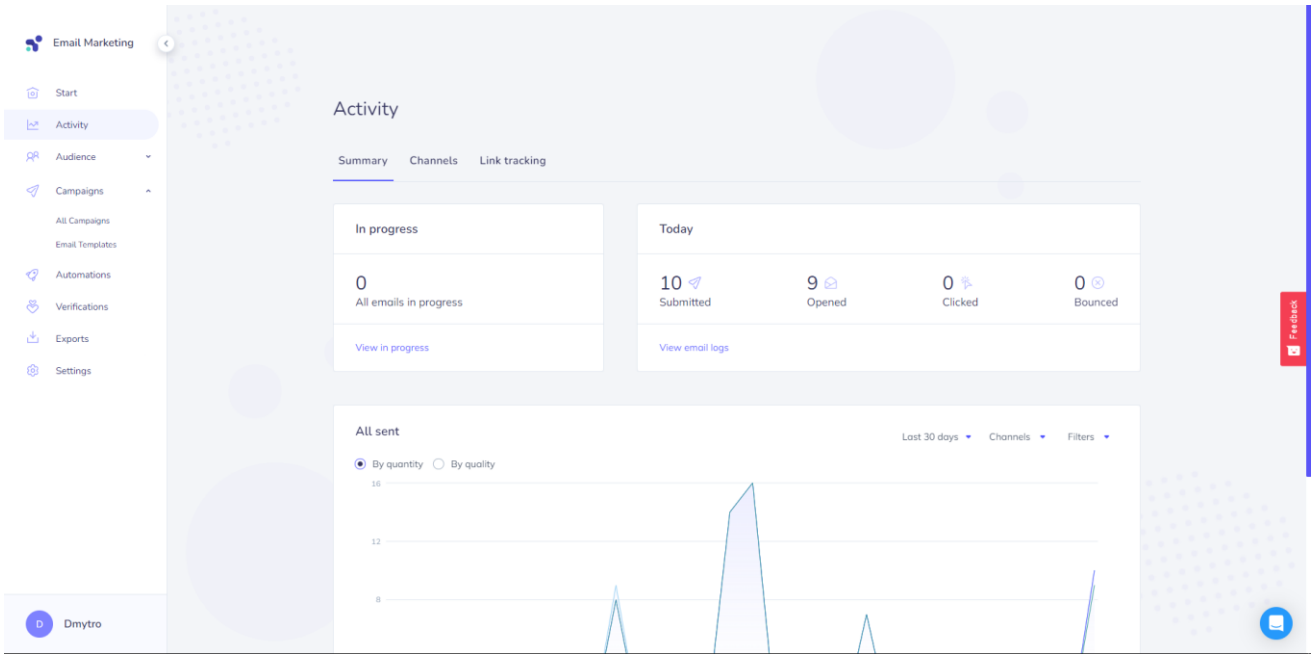

Рисунок 2.11 — Головна сторінка платформи Elastic Email

Після реєстрації на платформі, потрібно перейти в Налаштування та створити обліковий запис SMTP, вказавши ім’я користувача, пароль та електронну адресу. Нам було надано SecureToken який ми будемо в подальшому використовувати в коді для відправлення форму на пошту.

Після налаштування Elastice Email, було створено функцію emailSend() (додаток Г), де додано змінні для кожного поля форми та присвоєно їм значення, які вводить користувач, за допомогою методу document.getElementById.value. В функцію додано, захищений ключ — властивість SecureToken, яка є необхідною для використання SMTP.js. Ключ формується в налаштуваннях профілю на веб-сайті платформи, що надає доступ до SMTP.

Після відправки листа, виконується валідація, та якщо запит на відправку листа, має код стану — 200 (успішно), то відправляється форма в електронному листі та показується повідомлення «"Успішно!", "Ваша заявка на прийом надіслана, з вами зв'яжеться наш менеджер."» В іншому разі, якщо код стану запиту не дорівнює успішному, відображається повідомлення «Ой-ой, сталась помилка. Спробуйте будь ласка ще раз».

Для гарного відображення повідомлень про успіх чи помилку, було використано JavaScript бібліотеку — SweetAlert [11]. Дана бібліотека — це заміна функції window.alert() у JavaScript, яка показує гарні модальні вікна. Це окрема бібліотека, яка не має жодних залежностей і складається з JavaScript-файлу та CSS-файлу.

Sweet Alert використовується для того, щоб зробити вікно сповіщення більш привабливим і легким у дизайні. Дана бібліотека також дозволяє відображати користувачам повідомлення про помилки або успіх після виконання дій на веб-додатку.

Відображення отрманого листа на електронній пошті Gmail показано на рисунку 2.12.

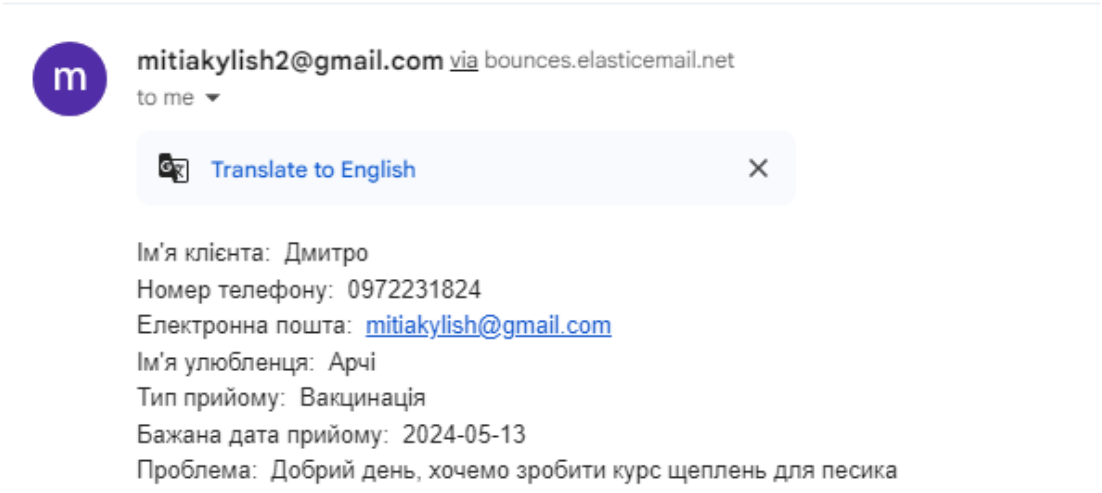

Рисунок 2.12 — Відображення листа на електронній пошті Gmail

Як відображено на рисунку 2.12, в листі приходять усі поля, що вказані в функції emailSend(). Усі поля електронного листа відображаються з нового рядку, завдяки використанню тегу
 при формуванні змінної тіла листа — messageBody.

2.6 Створення динамічного списку ветеринарних препаратів

2.6.1 Реалізація пагінації на сторінці

Пагінація, або посторінкова навігація — це метод розбивки великих обсягів інформації на менші, більш зручні для сприйняття користувачем. Ця практика широко використовується на у програмних засобах, де потрібно відображати багато даних, не перевантажуючи користувача та систему. Пагінація дозволяє робити запити на завантаження даних, лише в тому випадку, якщо в цьому є необхідність користувача, тим самим, зменшується час завантежння сторінки та використання ресурсів систем клієнтів.

Існує декілька поширених видів пагінації, кожен з яких має свої особливості:

— числова пагінація є найпоширенішим типом пагінації, де використовуються номери сторінок для навігації;

— пагінація з кнопками, де замість номерів сторінок використовуються кнопки «Назад», «Вперед» або іконки відповідних кнопок;

— динамічна пагінація підлаштовується під кількість даних, що відображаються, автоматично розбиваючи їх на сторінки.

Згідно поставлених клінікою вимог, було прийняте рішення імплементувати саме посторінкову навігацію з кнопками для таблиці ветеринарних препаратів, оскільки вона є найбільш інтуїтивно зрозумілою для більшості користувачів.

Для реалізації пагінації у таблиці з ветеринарними препаратами було створено необіхдний блок з таблицею в HTML.

Створенно дві змінні — rowsPerPage, яка відповідає за кількість рядків на кожній сторінці, та змінну — currentPage, яка за замовчуванню дорівнює одиниці, та зберігає поточний номер сторінки. Створення змінних наведено в лістингу 2.3.

Лістинг 2.3 — Створення змінних для зберігання даних про сторінки

```
const rowsPerPage = 5;
let currentPage = 1;
```

Тіло таблиці було створене пустим, тому що елементи таблиці зберігаються в JavaScript масиві «data, звідки вони витягуються та додаються в основне тіло таблиці. Масив було заповнено інформацією про реальні ветеринарні ліки [12].

Функція додавання даних з масиву data[] в таблицю реалізована наступним чином: створена змінна, яка знаходить елемент з ID «data-body» в HTML-документі. Видаляє всі існуючі рядки з таблиці, щоб підготувати її до нового вмісту. За допомогою циклу for, створюються рядки та витягуються для них дані.

Код програми, створює функцію generateTable, яка генерує HTML-таблицю з даними, переданими у вигляді масиву об'єктів.

Перед генерацією нового вмісту таблиці, функція очищає HTML-вміст елемента з ідентифікатором «data-body». Це гарантує, що при кожному виклику функції створюється нова таблиця, а не додаються нові рядки до існуючої.

Для кожного елемента в масиві data (об'єкт row), функція додає новий рядок <tr> до таблиці, використовуючи властивості об'єкта row.name, row.description, row.value. Цей рядок додається до внутрішнього HTML tableBody через innerHTML +=, що означає додавання нового рядка до вже існуючого вмісту. Функція

generateTable викликається з параметром data, який повинен бути масивом об'єктів, кожен з яких має властивості name, description та value.

Відображення створеної таблиці з ветеринарними препаратми зображена на рисунку 2.13.

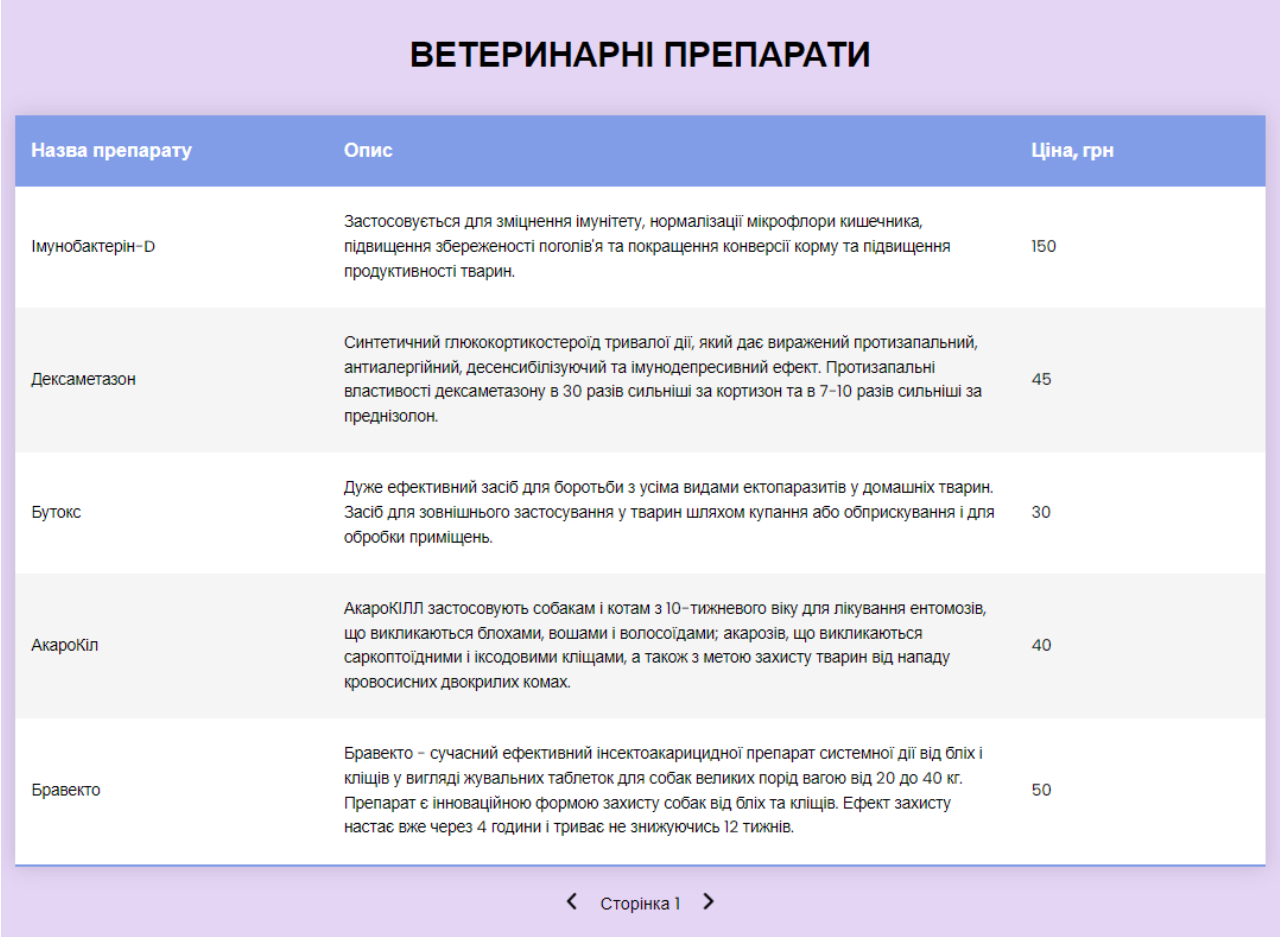

Рисунок 2.13 — Таблиця з ветеринарними препаратами

Як зображено на рисунку 2.13, відображається 5 рядків препаратів на кожній сторінці, відповідно до значення змінної rowsPerPage. На веб-сторінці розміщені стрілки навігації, які дозволяють користувачам переходити між різними сторінками таблиці ветеринарних препаратів. При натисканні на стрілку вправо відображається наступна сторінка, а при натисканні на стрілку вліво — попередня. При переході на наступну чи попередню сторінку, відповідно змінюється значення змінної currentPage. Ця змінна зберігає номер поточної сторінки, яка відображається між

кнопками навігації. Це допомагає чітко бачити, на якій сторінці даних знаходяться клієнти.

Такий підхід до розробки пагінації дозволяє добре масштабуватись для великих наборів даних, адже користувачі не будуть зазнавати проблем з продуктивністю при роботі з великими обсягами інформації.

Друга сторінка списку ветеринарних препаратів зображена на рисунку 2.14.

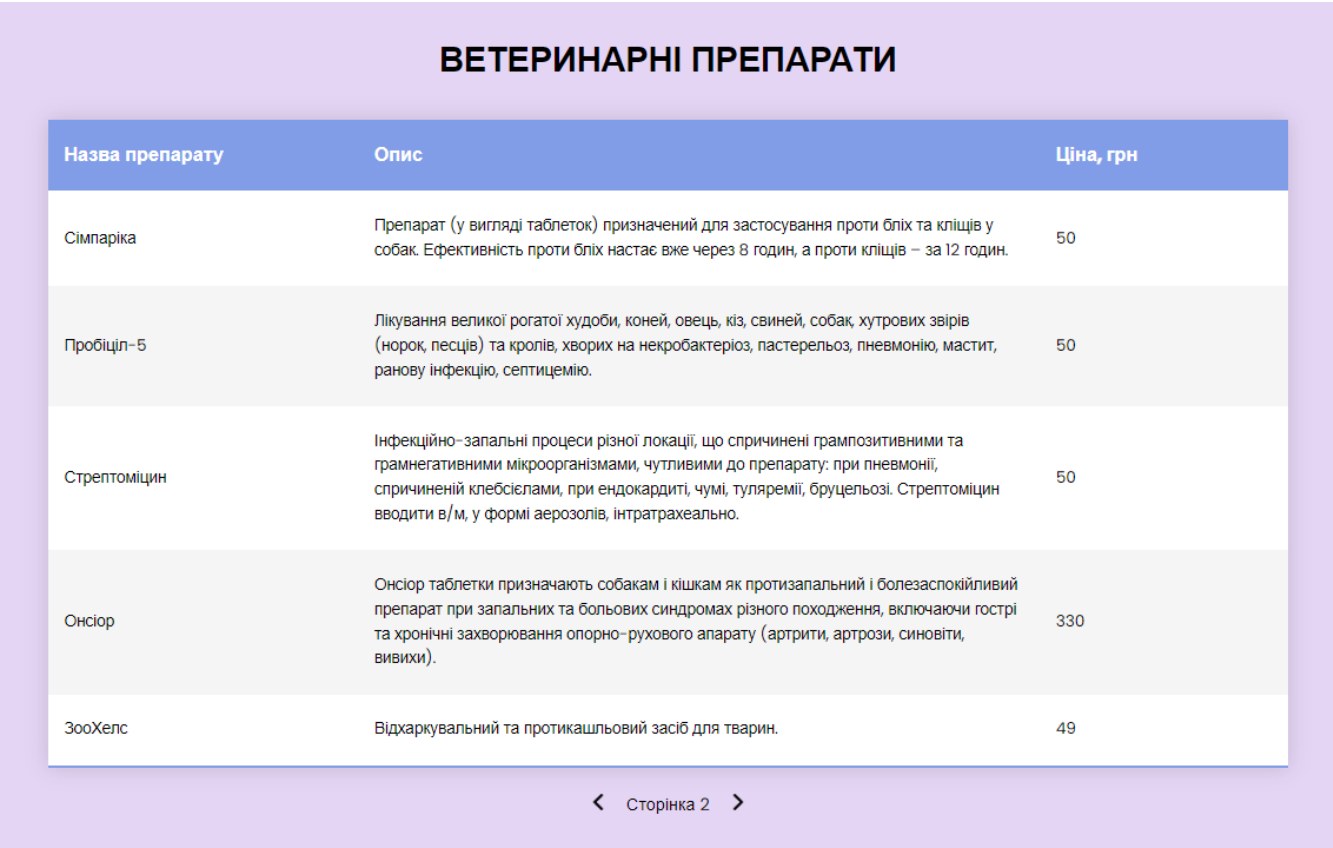

Рисунок 2.14 — Друга сторінка списку ветеринарних препаратів

2.6.2 Реалізація пошуку на сторінці

Пошук на сторінці — це функція, що дозволить користувачам швидко знаходити потрібно ветеринарні препарати та перевіряти їх ціну у даній ветеринарній клініці.

Перед створенням JavaScript коду, було додано поле для пошуку на сторінці, та кнопка для очищення поля від введених даних.

Пошук реалізовано за назвою препарату або за ціною. Для цього, створено функцію searchData (лістинг 2.4), яке приймає як параметр текст, який користувач вводить в поле пошуку та перевіряє чи відповідає якийсь елемент масиву введеному значенню.

Потім, за допомогою функції generateTable (filteredData) та updatePagination() відображаються результати пошуку.

Лістинг 2.4 — Реалізація пошуку препаратів мовою JavaScript

```
function searchData(searchTerm) {
const filteredData = data.filter(item =>
 { const nameMatch =
item.name.toLowerCase().includes(searchTerm.toLowerCase());
    const priceMatch = String(item.value).includes(searchTerm);
    return nameMatch || priceMatch;  });
  generateTable(filteredData);
  currentPage = 1;
document.getElementById("search").addEventListener("keyup", () => { const searchTerm = document.getElementById("search").value; searchData(searchTerm); });
```

Елементи таблиці, що не підпадають під шукане значення, не відображаються на сторінці. Це реалізовано шляхом фільтрування ветеринарних препаратів з використанням методу filter(), який обробляє введене користувачем значення та оновлює таблицю, відображаючи лише відфільтровані методом дані.

Очищення поля пошуку реалізовано за допомогою обробника подій, що викликається при натисканні користувачем на кнопку очищення поля. Після натискання, значення поля пошуку змінюється на пустий рядок, також викликається функція пошуку, яка приймає пустий рядок як параметр. Що дозволяє повторно відобразити усі елементи таблиці.

Перевірка роботи функціонування пошуку показано на рисунках 2.15 та 2.16.

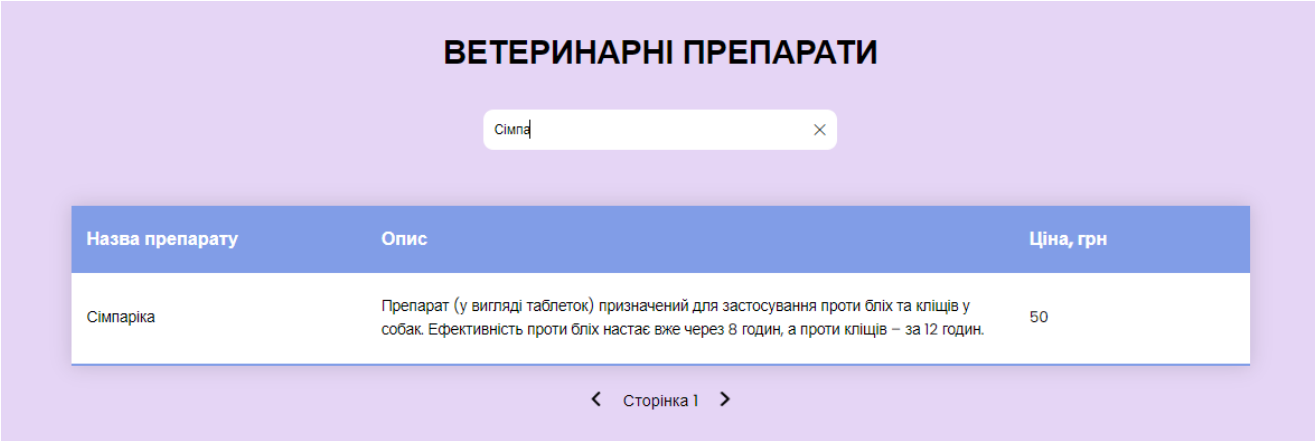

Рисунок 2.15 — Пошук по назві препарату

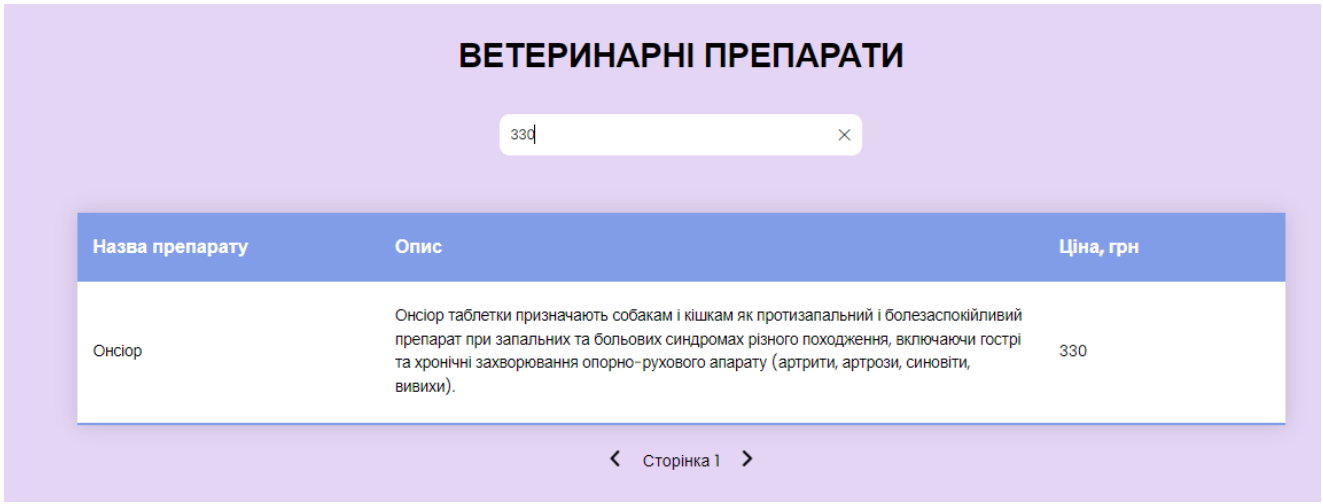

Рисунок 2.16 — Пошук по ціні препарату

2.7 Реалізація динамічного списку відгуків клієнтів на сторінці

Динамічний список відгуків клієнтів ветеринарної клініки повинен бути розташований під блоком з ветеринарними препаратами. Кожен відгук складається з заголовку, тексту відгуку, зображення профілю клієнта та оцінки його відвідування клініки по системі від 1 до 5.

Було створено наступні масиви з даними:

1) imageUrls[] — масив, який містить зображення профілів користувачів;
2) texts[] — масив, що містить самі тексти відгуків;
3) titles[] — масив, який містить назви заголовків відгуків;

4) usernames[] — масив, який містить ім’я та прізвище користувача (автора відгуку);

5) rating_points[] — масив, що містить оцінку від користувача.

Оцінка клієнта в форматі від 1 до 5, повинна відображатись у вигляді зірочок. При цьому, якщо оцінка клієнта є дробовою, наприклад — 3.5, іконки зірок мають це відображати. Це імплементовано шляхом перевірки оцінки на дробовість. Якщо оцінка є цілим значенням, в масив з загальною оцінкою додається зображення, яке відображає цілу зірку. У випадку, якщо оцінка є дробовою, в масив з загальною оцінкою додається картинка, де зображено зірку поділену навпіл.

Також, було реалізовано функціонал, додавання та зміни кількості зірочок, що будуть відображатись у відгуку, відповідно до значення елемента масиву rating_points [13]. Функція оновлення зірочок рейтингу називається «updateTestimonial», приймає як параметри індекс та номер відгуку, для того щоб при наступних викликах функцію оновити відгуки використовуючи встановлені елементи масиву JavaScript.

Оновлення заголовків та текстів відгуків відбувається шляхом отримання тексту відповідних елментів з використання textContent та призначення нового вмісту тексту. TextContent — це властивість, яка використовується для отримання або встановлення текстового вмісту. Властивість textContent зазвичай використовується для отримання текстового вмісту елемента для подальшої обробки або фільтрування тексту з веб-сторінки.

Для зміни відгуків при натисканні на кнопку «Наступні відгуки», було додано метод addEventListener() (лістинг 2.5), який має оновлювати кожен блок з відгуками, запам’ятовувати індекс відгуку та змінювати відгуки по колу з масиву даних. Також, щоб не виникало помилок при запуску сторінки, функцію updateTestimonial() та метод для оновлення блоку з відгуками було об’єднано та додано у функцію window.onload.

Функція window.onload виконується після повного завантаження сторінки, що дозволяє уникнути помилок, пов'язаних із спробою доступу до DOM елементів до початку їх створення.

Переваги використання функції window.onload при розробці програмного засобу:

— головна перевага window.onload полягає в тому, що код, розміщений всередині неї, гарантовано виконається лише після того, як вся сторінка та всі її ресурси будуть повністю завантажені;

— використання window.onload гарантує, що ваш код виконається лише після того, як всі елементи DOM стануть доступними.

Перевірка роботи функціоналу блоку з відгуками. При відкритті сайту, відображаються перший (рисунок 2.17) та другий (рисунок 2.18) елементи масивів відповідно.

Лістинг 2.5 — Реалізація методу для оновлення відгуків при натисканні на кнопку

```
const buttonToChange = document.querySelector('.next-comment');
let currentIndex = 0;
buttonToChange.addEventListener("click", function() {
updateTestimonial(currentIndex + 1, 1);
currentIndex = (currentIndex + 2) % texts.length;  });
```

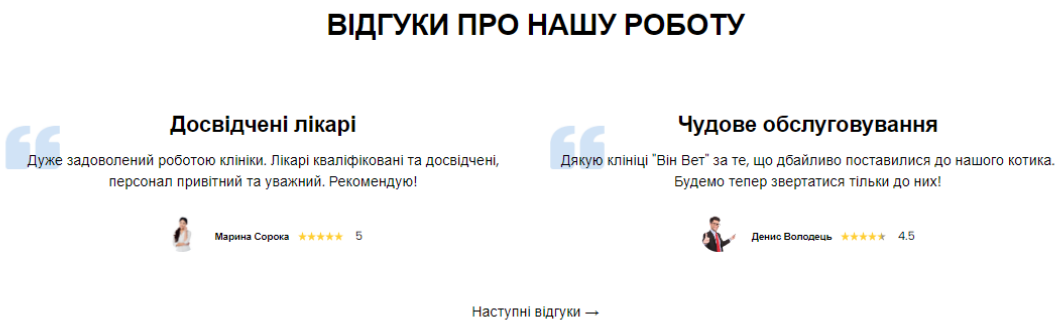

Рисунок 2.17 — Відображення блоку з відгуками при запуску сайту

При натисканні на кнопку «Наступні відгуки», блоки з відгуками оновлюються, та підтягуються наступні елементи масивів.

Рисунок 2.18 — Відображення інших відгуків при настиканні на кнопку «Наступні відгуки»

2.8 Реалізація слайдеру з інформацією про лікарів

Слайдер — це динамічний веб-елемент, який використовується для представлення декількох зображень або текстових блоків одним за одним. Він зазвичай складається з контейнера, що містить елементи контенту, та елементів керування, які дозволяють користувачеві переключатись між ними. Користувачі можуть вручну перемикатися між елементами за допомогою кнопок або автоматично, за допомогою функції автопрокручування.

Переваги використання слайдерів:

— слайдери дозволяють розміщувати багато різного контенту на невеликій площі екрана;

— слайдери додають динамічності до вебінтерфейсу та роблять його більш цікавим для користувачів.

Незважаючи на переваги, потірбно розуміти, що слайдери не завжди є найкращим рішенням для вебінтерфейсу. Їх слід використовувати з обережністю, щоб не ускладнити навігацію для користувачів та не перевантажити дизайн.

Існує безліч безкоштовних та платних слайдерів, доступних на ринку. Для реалізації слайдеру з інформацією про лікарів, було використано JavaScript бібліотеку «Swiper.js».

Swiper.js — це потужна бібліотека JavaScript, яка дозволяє швидко додавати адаптивні слайдери на веб-сайт [14]. Вона оптимізована для високої продуктивності, забезпечуючи плавну та безперебійну роботу слайдерів на різних пристроях, оскільки використовує апаратне прискорення браузера для забезпечення плавної анімації переходів між слайдами. Бібліотека має високу гнучкість в налаштуванні, а також доступна в різних фреймворках Javascript, таких як акими як React, Angular та Vue.js. Swiper.js було підключено до проєкту шляхом додавання скрипту CDN.

Після підключення бібліотеки, було створено HTML-блоки для слайдеру. Згідно документації, основний блок слайдеру має містити клас «swiper», далі по ієрархії мають йти блоки «swiper-wrapper» та «swiper-slide». Реалізація слайдеру Реалізація логіки слайдеру є досить простою, адже бібліотеки надає усі можливості для налаштувань слайдеру. В JavaScript коді (лістинг 2.6) прописуються наступні параметри: slidesPerView — кількість слайдів на кожній сторінці, spaceBetween — відстань між слайдами в пікселях та navigation, де визначаються HTML класи кнопок, які відповідають за навігацію по слайдеру, navigation — визначає елементи для кнопок навігації. Цей параметр містить два властивості: «nextEl» та «prevEl».

Лістинг 2.6 — Реалізація слайдеру мовою програмування JavaScript

```
var swiper = new Swiper(".mySwiper",
{
slidesPerView: 3,
spaceBetween: 30,
navigation: {
nextEl: ".swiper-button-next",
prevEl: ".swiper-button-prev",
},
});
```

Результат відображення створеного слайдеру з інформацією про лікарів наведено на рисунку 2.19.

Рисунок 2.19 — Слайдер з інформацією про лікарів

Як показано на рисунку 2.19, з двох боків секції з інформацією про лікарів, відображаються кнопки керування слайдером, що перемикають дані персоналу ветеринарної клініки.

2.9 Налаштування бази даних та таблиць

Для коректної роботи програмного засобу, було обрано систему керуваня базами даних — MySQL [15] та панель керування phpMyAdmin.

Було обрано саме MySQL, оскільки це є чудовим вибіром для веб-додатків та невеликих проектів завдяки простоті використання та високій продуктивності.

MySQL підтримує різні типи таблиць та сховищ даних, що дозволяє адаптувати її до різних потреб зберігання даних. Вона також підтримує широкий спектр типів даних, таких як цілі числа, рядки, дати та час.

PhpMyAdmin — це безкоштовний веб-інтерфейс керування базами даних, який використовується для адміністрування баз даних MySQL. Він пропонує зручний графічний інтерфейс, що робить його доступним як для досвідчених користувачів MySQL, так і для початківців.

До основних функій phpmyadmin можна віднести:

— створення, редагування та видалення баз даних, таблиць та записів;

— виконання SQL-запитів для прямого керування даними в базі даних;

— імпорт та експорт даних у різних форматах, таких як CSV, SQL та XML;

— резервне копіювання та відновлення баз даних.

Відповідно до функціоналу та зручності використання, було прийняте рішення на користь використання саме даного веб-інтерфейсу для керування базами даних.

Було створено базу даних, використовуючи запити мови SQL [16].

SQL — це мова програмування, спеціально розроблена для роботи з реляційними базами даних. Вона використовується для створення, читання, оновлення та видалення даних, а також для керування структурами бази даних, такими як таблиці та індекси. SQL підтримується практично всіма системами керування реляційними базами даних (СКБД), такими як MySQL, PostgreSQL, Oracle, Microsoft SQL Server та інші.

При проєктуванні бази даних, було використано метод оптимізації бази даних, такий як нормалізація відношень у БД.

Нормалізація відношень забезпечує максимально чітке відображення сутностей предметної області. Це досягається шляхом усунення надмірності даних. Але, варто звернути увагу на те, що при роботі з великими обсягами даних, нормалізація може призвести до зниження швидкодії. Це пов'язано з тим, що при зверненні до даних система змушена виконувати більше операцій, а саме з’єднування декількох дрібніших таблиць, замість однієї великої.

Також, для оптимізації роботи бази даних, можна впровадити індекси таблиць. Індекс в базі даних — це спеціальна структура даних, розроблена для покращення швидкості пошуку та вибірки записів з таблиці на основі значень одного або кількох полів. Використання індексації значно прискорює виконання певних запитів до бази даних. Проте, важливо пам'ятати, що при кожній зміні в базовій таблиці, індекси потребують оновлення системою, що в свою чергу, створює додаткове навантаження.

При розробці таблиць бази даних, спочатку було створено таблицю users, де будуть зберігатись дані користувачів.

Далі, було створено таблицю appointments, для зберігання даних про записи користувачів на прийом у ветклініку.

Загальна схеми бази даних показана на рисунку 2.20.

Таблиці бази даних мають відношення один до багатьох, оскільки один користувач з таблиці users може мати багато записів в таблиці appointments.

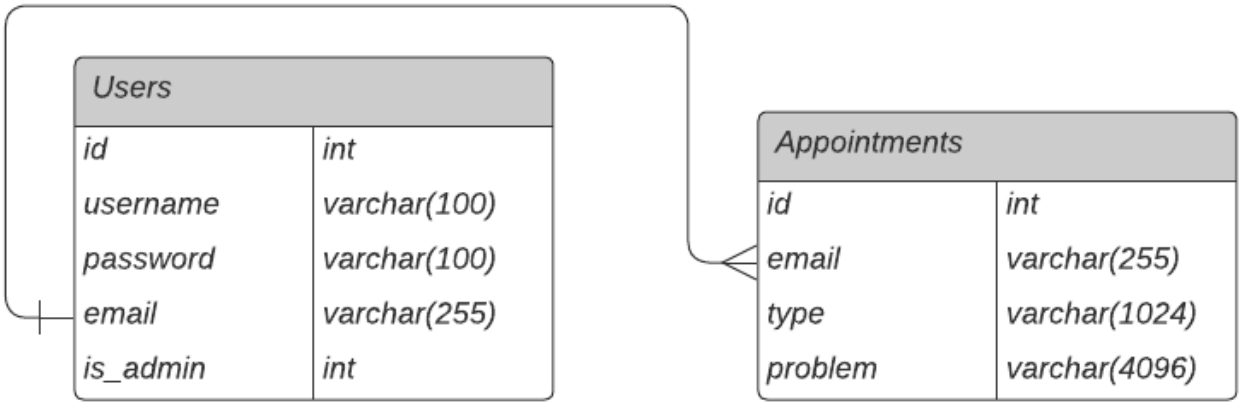

Рисунок 2.20 — Схема бази даних

Опис полів таблиць:

1) Id — унікальний ідентифікатор запису;
2) email — електронна пошта клієнта;
3) date — дата запису на прийом у клініку;
4) type — тип прийому;
5) problem — проблема клієнта;
6) is_admin — перевірка на наявність прав адміністратора у користувача;
7) username — ім’я користувача.

Для усіх полів таблиць було застосовано кодування utf8mb4_general_ci.

utf8mb4_general_ci — це комбінація двох окремих понять, що використовуються в MySQL для роботи з текстами. utf8mb4 — це кодування символів, яке визначає, як символи різних мов перетворюються в послідовності байтів для зберігання та передачі. general_ci — це колація, яка визначає правила сортування та порівняння символів один до одного на основі їх бітових кодів.

Так як, utf8mb4 може кодувати практично всі символи Unicode, це робить його придатним для зберігання текстів на різних мовах. Як альтернативу, можна використовувати кодування utf8mb4_unicode_ci, але ця колація може бути трохи повільнішою в порівнянні з general_ci.

Відображення створеної бази даних та таблиць в дереві phpMyAdmin наведено на рисунку 2.21.

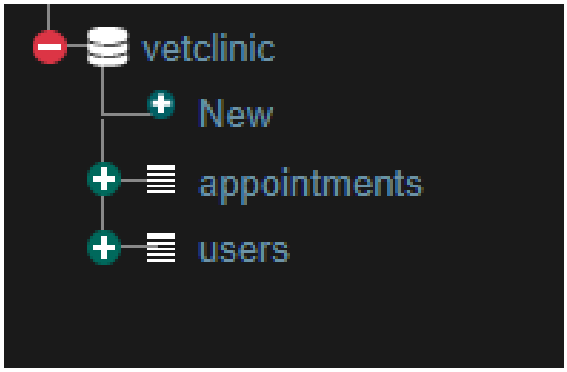

Рисунок 2.21— Відображення створеної бази даних та таблиць

Підключення до бази даних було імплементовано за допомогою функції PHP PDO [17]. Для підключення до бази, потрібно призначити дані, що відповідають назві хоста, імені користувача, паролю та назві бази даних до відповідних змінних. Потім, ці змінні необхідно додати в клас PDO та відправити запит на підключення до бази з використанням функуції mysqli_connect (лістинг 2.7).

Лістинг 2.7 — Підключення до бази даних за допомогою функції PHP PDO

```
<?php $host = "db1.ho.ua";
  $user = "vinvetclinic";
  $password = "Aowemv213Pas";
  $db_name = "vinvetclinic";
  $pdo = new PDO("mysql:host=$host; dbname=$db_name; charset=utf8mb4", "$user", "$password",  array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8"));
  $con = mysqli_connect($host, $user, $password, $db_name);
$con->set_charset("utf8mb4");  ?>
```

PDO дозволяє використовувати один і той самий код для роботи з різними базами даних. Це робить програмний код більш універсальним та полегшує перехід на іншу СКБД у майбутньому.

Пароль користувачів було вирішено хешувати, задля підвищення безпеки за допомогою 128-бітного криптографічного алгоритму хешування — MD5.

MD5 працює шляхом обробки вхідного повідомлення поблочно, використовуючи складні математичні операції, при цьому, дана функція є досить швидкою хеш-функцією, що робить її придатною для широкого спектру застосувань. Кожен блок даних проходить через чотири етапи:

— функція основної обробки застосовує ряд математичних операцій до блоку даних, створюючи нове 32-бітове значення;

— нове значення об'єднується з поточним значенням хешу;

— повторення попередніх двох кроків для кожного блоку вхідного повідомлення;

— після обробки всіх блоків даних, отримується 128-бітний хеш-код, який представляє собою дайджест вхідного повідомлення.

При авторизації користувача, введений в поле пароль конвертується в 128-бітний хеш-код, який вже порівнюється з хешом, що зберігається в базі даних.

Приклад роботи шифру MD5 — пароль «testpassword» після хешування буде відображатись як «e16b2ab8d12314bf4efbd6203906ea6c».

Відображення захешованих паролів в базі даних показане на рисунку 2.22.

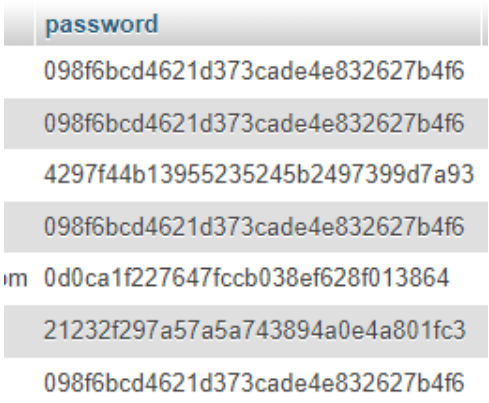

Рисунок 2.22 — Відображення захешованих паролів в базі даних

Реалізація авторизації користувачів відбувається за допомогою запиту в базу даних з введеними логіном та паролем. Якщо повертається результат, що користувач з даними параметрами існує, відбувається перевірка на те, чи є людина адміністратором чи ні. Якщо так, відкривається сторінка admin.php, якщо ні — відкривається сторінка cabinet.php. Якщо логін і пароль не відповідають жодному користувачу, показується повідомлення що логін чи пароль не вірний.

2.10 Проєктування особистого кабінету та панелі адміністратора

При розробці програмного засобу, було прийняте рішення розробити особистий кабінет користувача. Увійти в нього можна буде натисканням на відповідне посилання у навігаційному меню. При натисканні на «Особистий кабінет» відбувається перевірка на те, чи є зараз користувач авторизованим. Це реалізовано з використанням перевірки значення масиву $_SESSION['login']. Якщо користувач на даний момент не авторизований — відкривається сторінка авторизації/реєстрації користувачів. Авторизація — це надання і перевірка прав на вчинення будь-яких дій в системі. Після авторизації користувача відбувається перевірка, на те чи є він адміністратором чи звичайним клієнтом. Це реалізовано за допомогою вибірки інформації про користувача з бази даних, перевевіряється значення поля “is_admin”, яке може мати два значення: 1 (адміністратор) та 0 (звичайний користувач). Якщо “is_admin” = 1, користувача після авторизації перенаправляє на панель адміністратора (рисунок 2.23), де відображаються уся інформація про записи на прийом у ветклініку. Якщо “is_admin” = 0, після авторизації користувач буде перенаправлений в особистий кабінет (рисунок 2.24), де показуються записи на прийом у ветклініку лише цього користувача. Користувач може скасувати свій запис на прийом, якщо він ще не відбувся, це відбувається шляхом SQL-запиту в базу даних, з методом DELETE, передаючи ідентифікатор запису в запиті.

Записи на прийом в особистому кабінеті реалізовані за допомогою карток. Також розроблено логіку відображення різних іконок у карток, в залежності від типу прийому у ветклініці. Наприклад, якщо тип прийому — грумінг, в цій картці в особистому кабінеті відображається іконка ножиць та гребінця. При типу прийому —

вакцинація, показується іконка шприца. Якщо прийом — повторний огляд чи консультація — відображається зображення лікаря. При типу — аналізи, на сторінці особистого кабінету зображається мікроскоп. Реалізація відображення записів у особистому кабінеті показана в лістингу 2.8.

Лістинг 2.8 — Реалізація відображення записів у особистому кабінеті

```
<?php $currentDate = date("Y-m-d"); $columnCount = 0;
    while ($row = $result->fetch_assoc()) {
      echo "<b>Дата запису:</b> " . $row["date"] . "<br>";
      echo "<b>Тип прийому:</b> " . $row["type"] . "<br>";
      echo "<b>Проблема:</b> " . $row["problem"] . "<br>";
      if ($row["type"] === "Консультація" || $row["type"] === "Повторний огляд") { echo "<img src ='./images/physician.png'" . "<br>";
      } else if ($row["type"] === "Вакцинація") { echo "<img src ='./images/needle.png'" . "<br>";
      } else if ($row["type"] === "Аналізи") { echo "<img src ='./images/microscope2.png'" . "<br>"; ?>
```

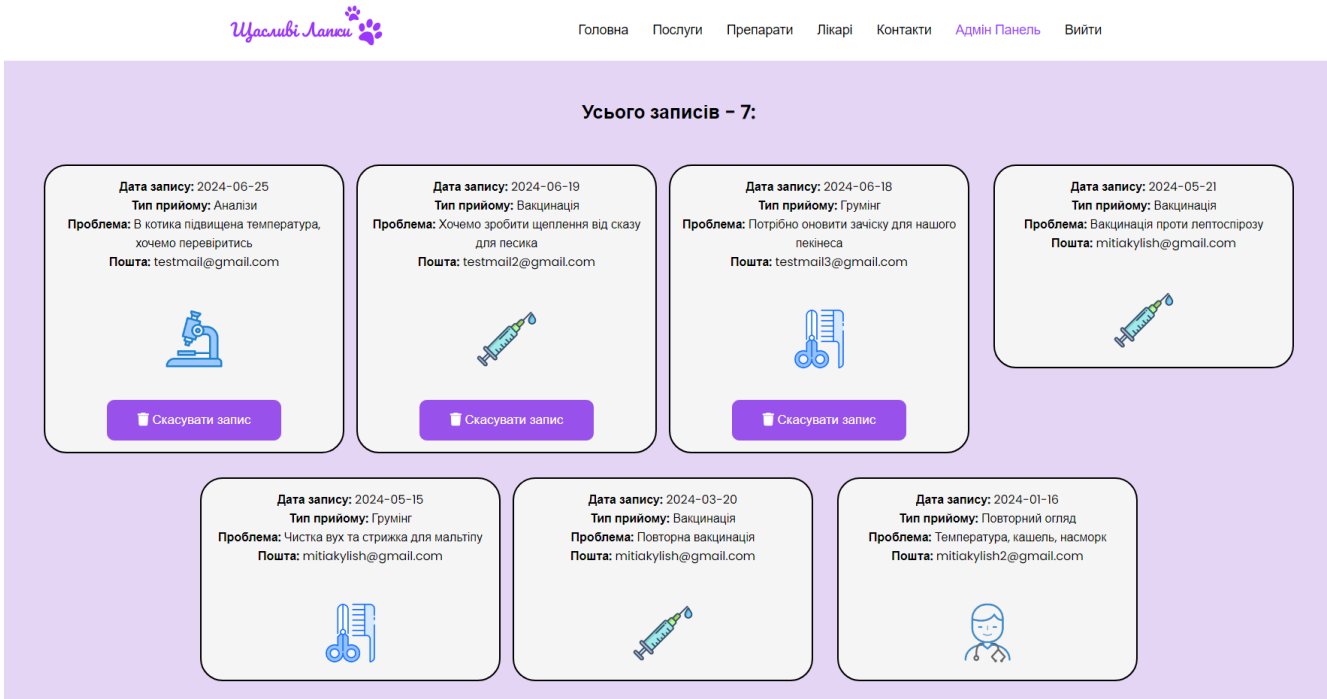

Рисунок 2.23 — Адмін панель при значенні “is_admin” = 1

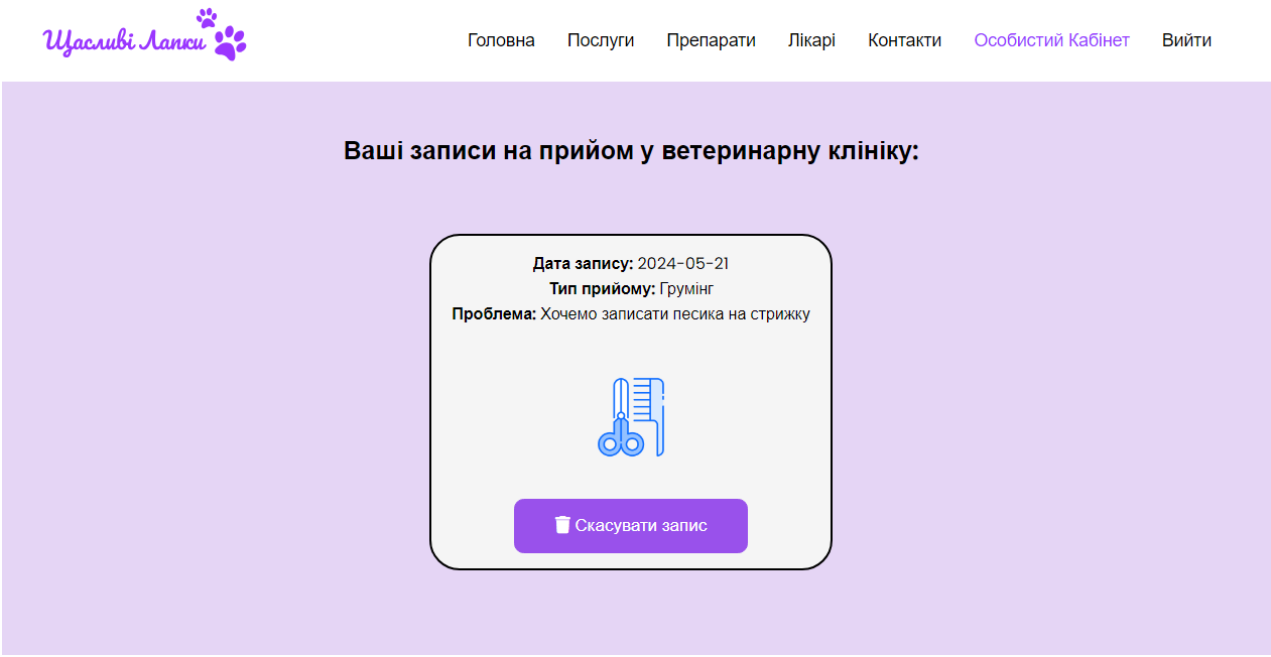

Рисунок 2.24 — Особистий кабінет при значенні “is_admin” = 0

3 ТЕСТУВАННЯ ТА РОЗГОРТАННЯ ПРОГРАМНОГО ЗАСОБУ

3.1 Функціональне тестування

Тестування програмного забезпечення — це процес технічного дослідження, який виконується на вимогу замовників, і призначений для вияву інформації про якість продукту відносно контексту, в якому він має використовуватись. До цього процесу входить виконання програми з метою знайдення помилок.

Функціональне тестування — тип тестування програмного забезпечення, який оцінює функціональність програмного засобу шляхом перевірки його на відповідність заданим вимогам [18]. Мета функціонального тестування полягає в переконанні, що додаток правильно виконує свої функції. Простіше кажучи, воно відповідає на питання: «Чи робить програмне забезпечення те, що воно повинно робити?».

Для проведення тестування веб-сайту, було створено тест-кейси. Тест-кейс це тестовий артефакт, суть якого полягає у виконанні деякої кількості дій та/або умов, необхідних для перевірки певної функціональності програмної системи, що розробляється.

Щоб тестовий сценарій був зрозумілим для всіх**,** його слід писати простою мовою, без спеціальних термінів та складних речень. Це пов'язано з тим, що до тестування можуть бути залучені фахівці з різним рівнем підготовки та досвіду [19].

Структура тестового випадку зазвичай складається з: назви перевірки, яку потрібно виконати тестувальнику, кроків, небхідних для виконання перевірки, очікуваного результату, та статусу тест-кейсу. Фактичний або актуальний результат проходженян тестового випадку, зазвичай додається вже після виконання тестування.

До тест-кейсу можуть бути додані наступні параметри: передумови, які потрібно виконати перед перевіркою тест-кейсу, післяумови, які потрібно виконати після проходження тест-кейсу.

Для забезпечення ефективності тестування, тест-кейси повинні регулярно оновлюватись у випадку змін в вимогах або функціональності системи.

Створені тест-кейси та результат їх виконання наведено в таблиці 3.1.

Таблиця 3.1 — Тест-кейси для тестування функціоналу сайту

Дія	Кроки	Очікуваний результат	Фактичний результат	Статус
Відкриття форми запису на прийом	1. Перейти на головну сторінку сайту 2. Натиснути на кнопку «Запис на прийом»	Форма для запису на прийом з'явилась.	Форма для запису на прийом з'явилась.	Успішно
Перевірка валідації полів форми	1. Відкрити форму 2. Заповнити поля «Ім'я», «Номер телефону», «Електронна пошта» невалідними значеннями.	Колір плейсхолдеру поля стає червоним, повідомленя про введене значення.	Колір плейсхолдеру поля стає червоним, повідомленя про введене значення.	Успішно
Перевірка роботи навігації по сайту	1. Натиснути на кнопки навігаційного меню: «Послуги», «Препарати», «Лікарі», «Контакти»	Користувачу відображається секцію сайту, яка була обрана у навігаційному меню.	Користувачу відображається секцію сайту, яка була обрана у навігаційному меню.	Успішно
Перевірка роботи пошуку ветеринарних препаратів	1. Перейти до секції «Ветеринарні препарати» 2. Ввести в поле пошуку назву препарату «Бравекто»	При введенні назви препарату, відображається інформація про цей препарат.	При введенні назви препарату, відображається інформація про цей препарат.	Успішно
Успішна відправка форми	1. Заповнити всі поля 2. Натиснути кнопку «Запис на прийом»	Дані з форми приходять на пошту та інформація відображається коректно.	Дані з форми приходять на пошту та інформація відображається коректно	Успішно

Результати виконання тест-кейсів зображено на рисунках 3.1, 3.2 та 3.3.

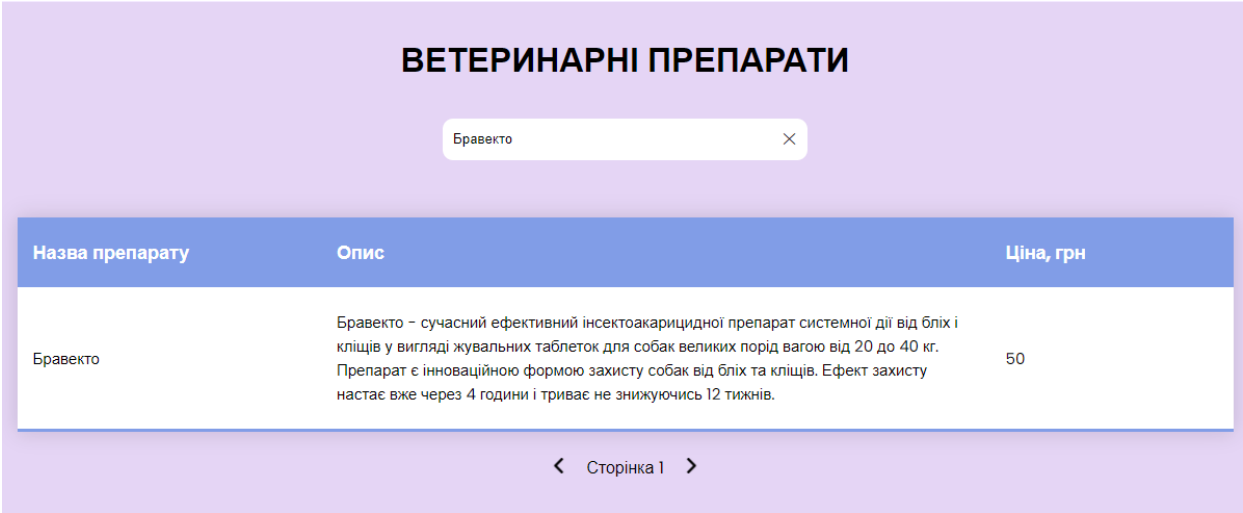

Рисунок 3.1 — Результат перевірки тест-кейсу з пошуку ветеринарних препаратів

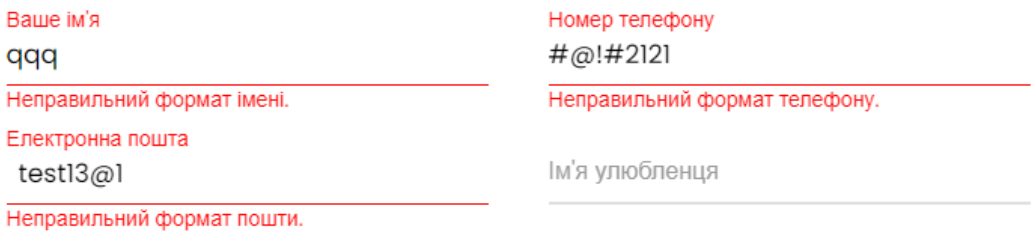

Рисунок 3.2 — Результат перевірки тест-кейсу з валідації полів форми

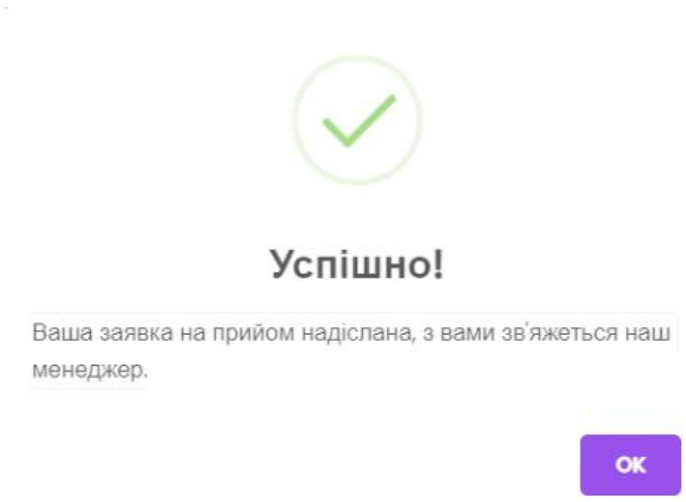

Рисунок 3.3 — Результат перевірки тест-кейсу з успішної відправки форми

3.2 Тестування сумісності

Тестування сумісності — це тип нефункціонального тестування, спрямований на забезпечення безперебійної роботи програмного забезпечення в різних середовищах, іншими словами, сумісності з певними пристроями та встановленими на них додатками. Існує декілька категорій тестування сумісності програмного забезпечення:

— тестування сумісності з різними операційними системами;

— кросбраузерне тестування;

— тестування сумісності між пристроями.

Кросбраузерне тестування має на меті перевірити, чи правильно відображається вебінтерфейс у всіх браузерах та їх версіях.

Тестування сумісності з пристроями — це перевірка того, чи працює програмне забезпечення належним чином на пристроях різних виробників та з різними технічними характеристиками (версія ОС, розмір ОЗП, діагональ екрана і т.д.).

Для коректного проведення тестування сумісності програмного забезпечення потрібно обрати пристрої та браузери, на яких ви хочете перевірити роботу програми. Далі, необхідно вирішити що буде об’єктом тестування, це може бути увесь програмний засіб, або тільки частина функціоналу [20]. Якщо мова йде про тестування сумісності на смартфонах, то щоб отримати точні результати тестування потрібно по можливості використовувати реальні пристрої.

Для того, щоб користуватись сайтом потрібно мати браузер. Програмний засіб було перевірено на сумісність у таких браузерах:

— Google Chrome, Version 125.0.6422.77 (Official Build) (64-bit);

— Microsoft Edge, Version 125.0.2535.67 (Official build) (64-bit).

Розроблений сайт повністю сумісний з цими браузерами, і вони забезпечують бездоганну роботу без будь-яких проблем чи дефектів. Відображення головної сторінки веб-додатку у браузері Microsoft Edge зображено на рисунку 3.4. Робота веб-додатку в браузері Google Chrome показана на рисунку 3.5.

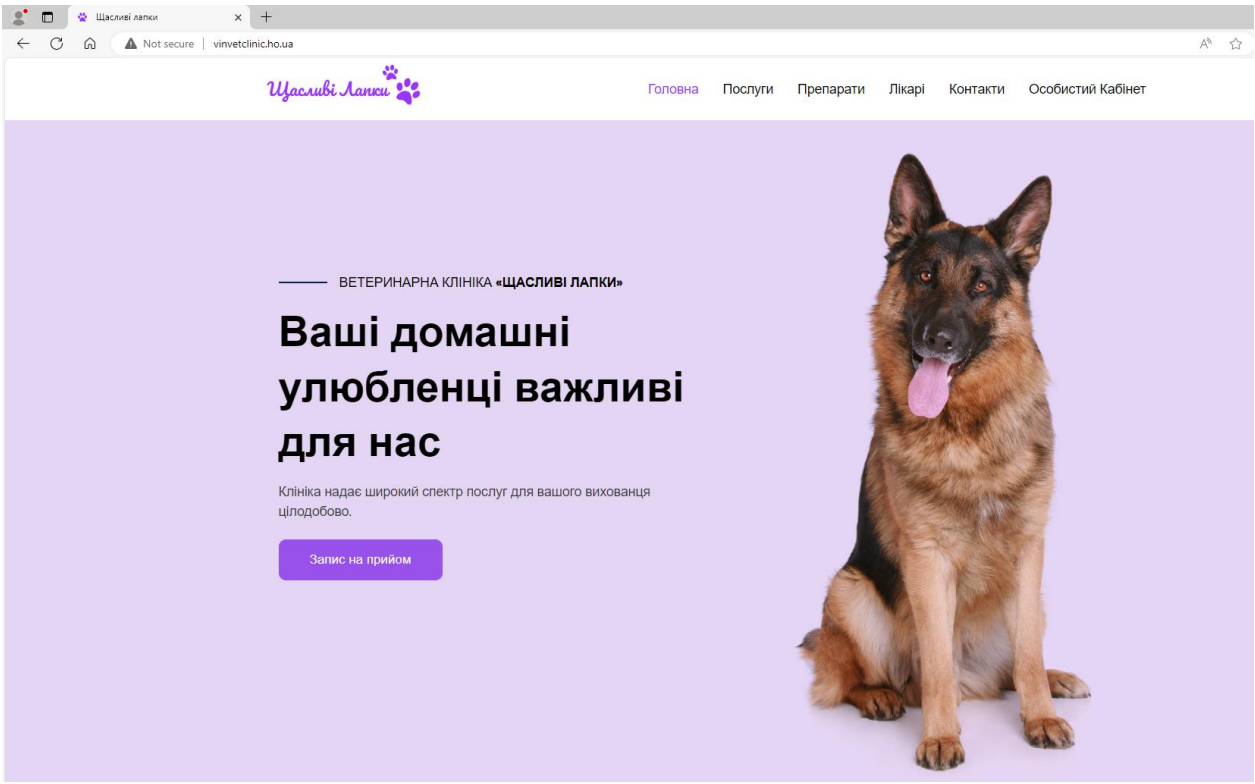

Рисунок 3.4 — Головна сторінка веб-додатку в браузері Microsoft Edge

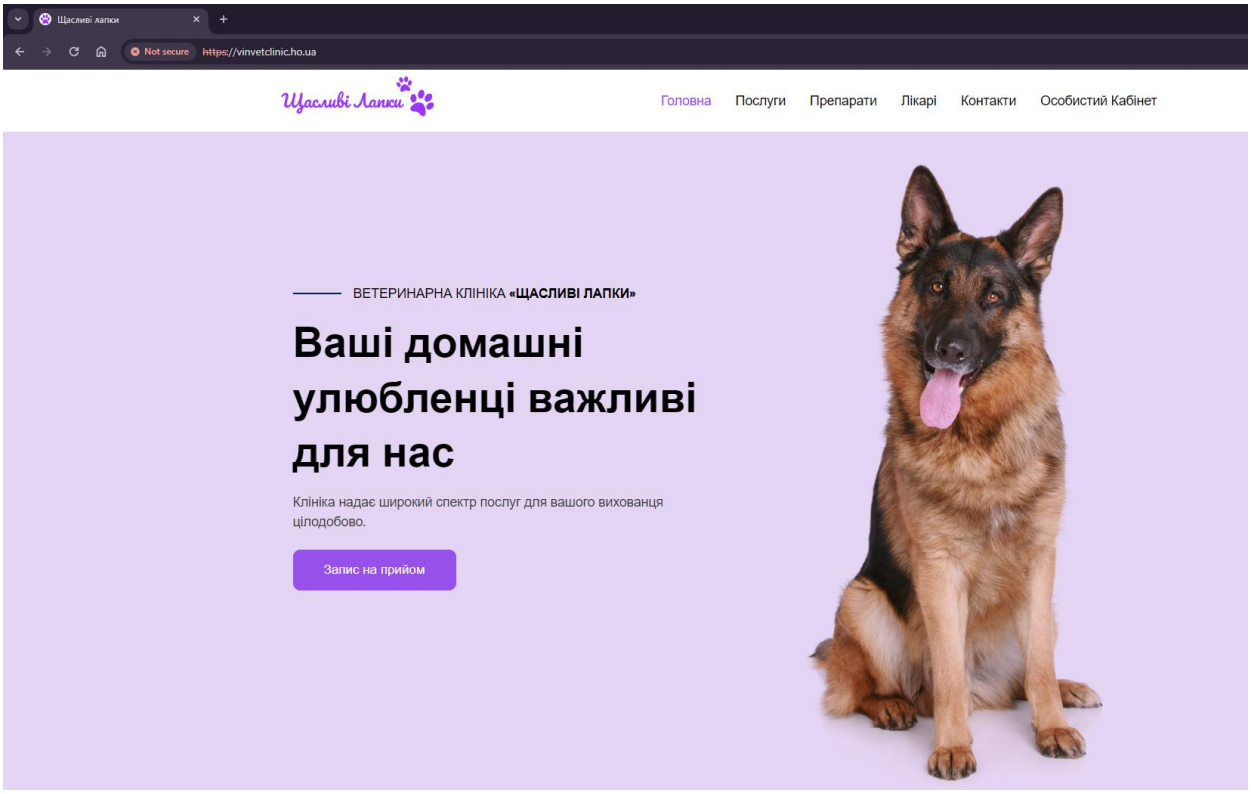

Рисунок 3.5 — Головна сторінка вебінтерфейсу в браузері Google Chrome

Програмний засіб було протестовано на двох реальних мобільних девайсах — Apple iPhone 12 Pro та Xiaomi Redmi Note 10 Pro.

Пристрій Apple iPhone 12 Pro має операційну систему iOS 15.4, розмір екрану: 6,1 дюйма та роздільну здатність екрану: 2532x1170 пікселів.

Девайс Xiaomi Redmi Note 10 Pro має операційну систему Android 11, розмір екрану: 6,67 дюйма та роздільну здатність екрану: 2400x1080 пікселів.

Відображення головних компонентів програмного засобу (основний блок головної сторінки та особистий кабінет) на цих двох пристроях відображено на рисунках 3.6 — 3.8.

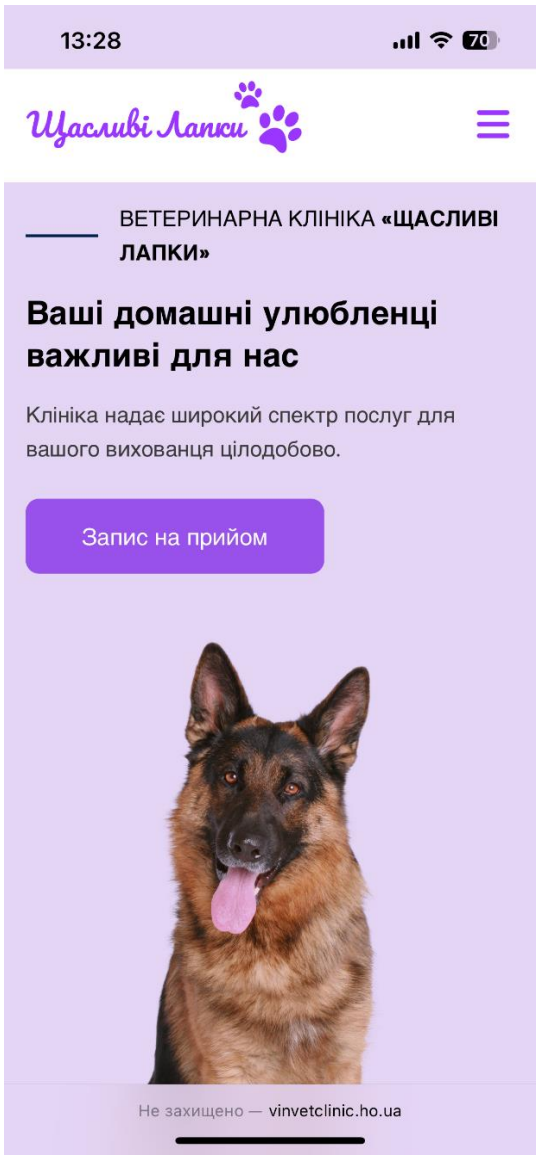

Рисунок 3.6 — Відображення основого блоку головної сторінки на iPhone 12 Pro

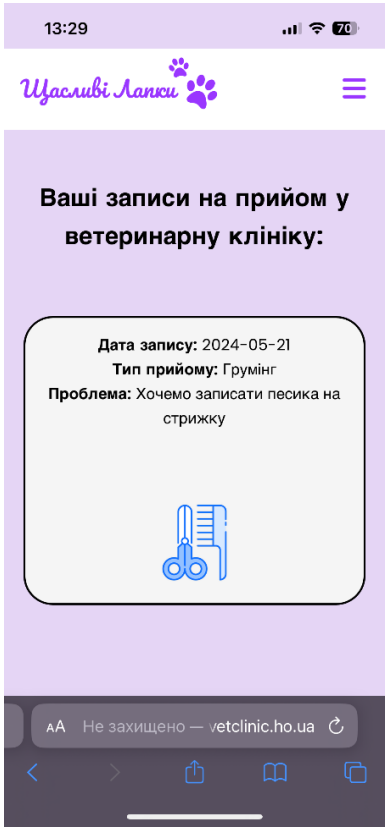

Рисунок 3.7 — Відображення особистого кабінету на iPhone 12 Pro

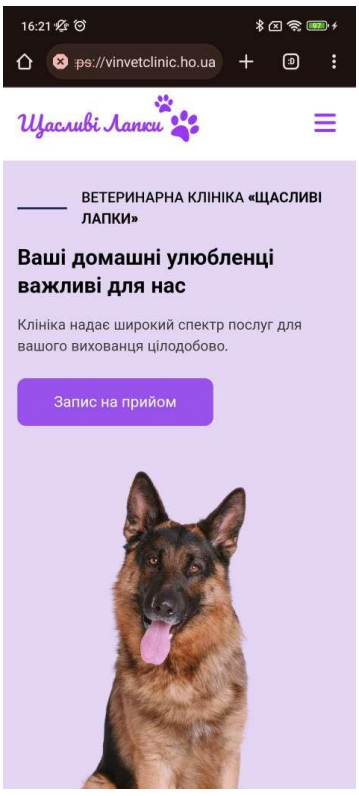

Рисунок 3.8 — Відображення основого блоку головної сторінки на Xiaomi Redmi Note 10 Pro

Відображення особистого кабінету на смартфоні Xiaomi Redmi Note 10 Pro показано на рисунку 3.9.

Рисунок 3.9 — Відображення особистого кабінету
на Xiaomi Redmi Note 10 Pro

Після тестування на реальних пристроях, додатково було перевірено роботу у емуляторі браузера Google Chrome з використанням Chrome DevTools. Chrome DevTools — це набір інструментів для веб-розробників, вбудованих безпосередньо. DevTools дозволяє редагувати сторінки «на льоту» та має влаштований емулятор з широким списком девайсів доступних для перевірки.

На основі описаної вище інформації можна зробити висновок, що сайт коректно працює та відображається на двох протестованих девайсах: iPhone 12 Pro та Xiaomi Redmi Note 10 Pro — клієнти цих пристроїв можуть без проблем користуватися можливостями розробленого програмного засобу.

3.3 Розгортання програмного засобу

Щоб розмістити продукт у вільний доступ, його потрібно завантажити на хостинг та вибрати домен. При виборі хостинг-провайдера важливо, щоб сайт працював без перебоїв, а підтримка відповідала швидко, якщо раптом щось трапиться. Тестовий період або гарантія повернення коштів в перший час після замовлення допоможуть не втратити гроші. Було прийнято рішення на користь хостинга ho.ua. Це український хостинг, який надає безкоштовний домен. Ціна хостингу достатньо низька — 75 грн/міс. Також присутній тестовий період на 3 місяці за 6 гривень, який я і обрав.

Переваги хостинга ho.ua [21]:

— вбудований файловий менеджер;

— підтримка нових версій PHP, HTML, CSS;

— вбудований phpMyAdmin.

Щоб завантажити веб-ресурс на хостинг, скористаємось файловим менеджером в панелі керування. Для цього, потрібно зайти на панель хостингу, та в навігаційному меню, обрати «Управління файлами». Потім, потрібно завантажити потрібні файли в каталог htdocs, попередньо видаливши присутні тестові файли (рисунок 3.10). Важливо переконатися, що файли завантажено у правильні підкаталоги, щоб структура сайту залишилася незмінною.

Потім, за допомогою phpMyAdmin [22] імпортуємо базу даних (рисунок 3.11). Для цього, потрібно перейти в панелі керування та в навігаційному меню натиснути на кнопку «База даних». Після натискання, з’явиться посилання на систему управління phpMyAdmin. Потрібно перейти по посиланню та увійти в систему, використовуючи логін та пароль. Після входу в phpMyAdmin, потрібно обрати базу даних, натиснути «Import», та завантажити вже підготовлену базу даних, обравши файл БД з диску персонального комп’ютера. Після завершення імпорту, необхідно перевірити, чи всі таблиці та дані імпортовані коректно, а також протестувати з'єднання бази даних з веб-ресурсом, щоб переконатися, що все працює без помилок.

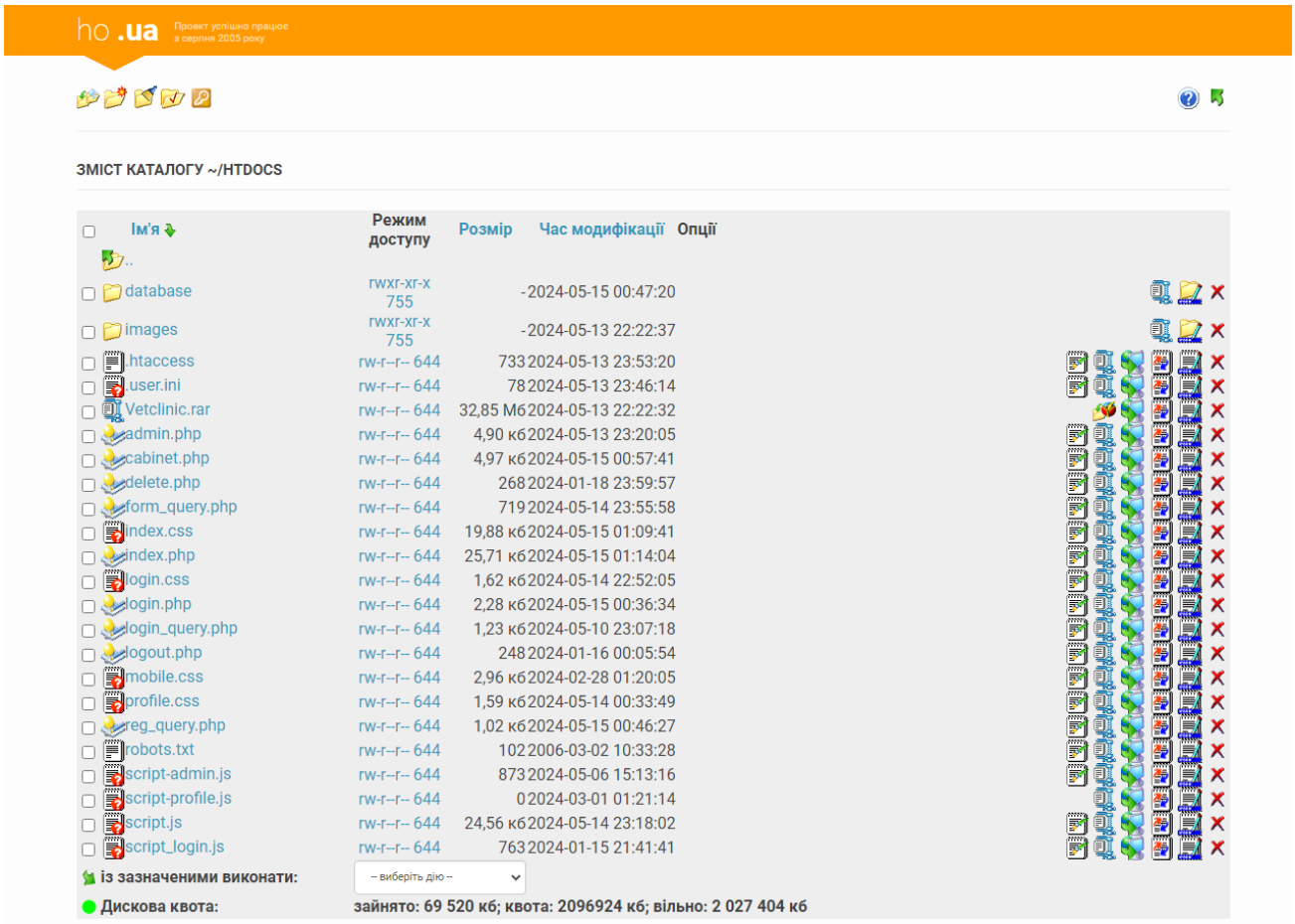

Рисунок 3.10 — Каталог htdocs файлового менеджеру

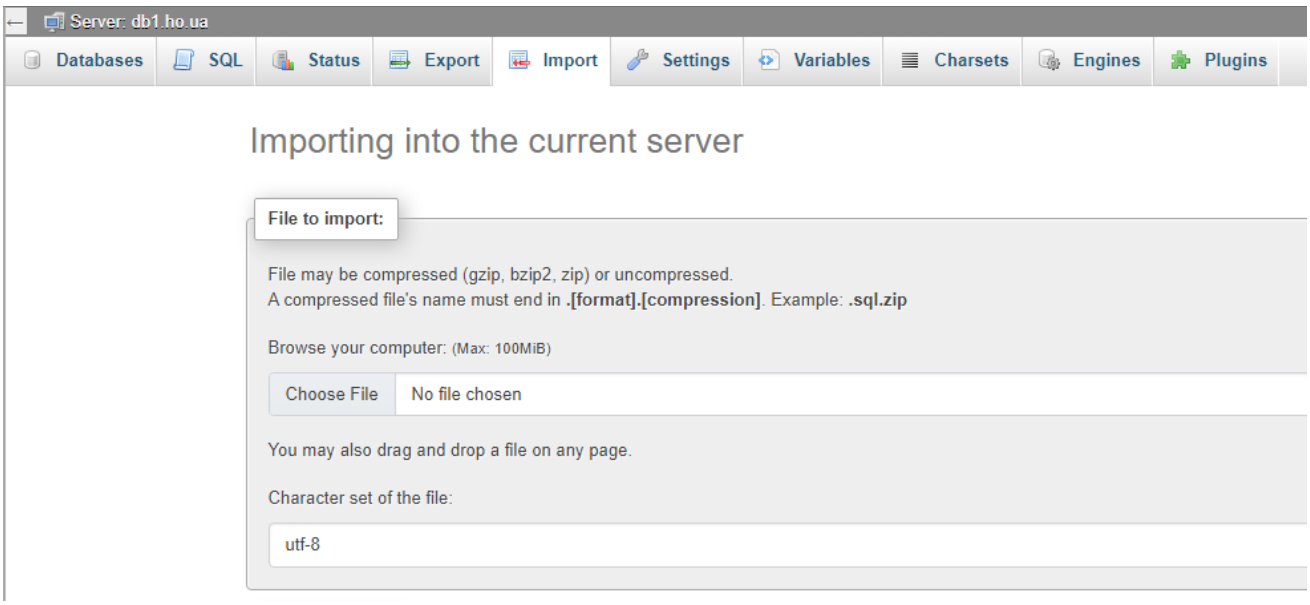

Рисунок 3.11 — Імпортування бази даних в phpMyAdmin

Важливо обирати кодування символів UTF-8 при імпортуванні бази даних. Це дозволить збільшити сумісність бази даних з різними операційними системами, покращить безпеку, так як UTF-8 може допомогти запобігти деяким типам атак, таким як SQL-ін'єкції, оскільки це кодування добре підтримується більшістю бібліотек обробки SQL-запитів і має менше ймовірностей для неправильного тлумачення символів.

Варто також звернути увагу на те, чи надає хостинг SSL-сертифікат для сайту.

SSL-сертифікат — це цифровий сертифікат, який підтверджує автентичність веб-додатку та забезпечує зашифроване з'єднання [23]. SSL розшифровується як Secure Sockets Layer — протокол безпеки, що створює зашифроване з'єднання між веб-сервером і веб-браузером.

Веб-сайти потребують SSL-сертифікат, щоб захистити дані користувачів, підтвердити право власності на веб-сайт, запобігти створенню шахрями підроблених версій сайту та викликати довіру у користувачів, тому це є досить вагомою причиною для використанням даного сертифікату.

Для бізнесу більш важливим є той факт, що для веб-адреси HTTPS потрібен SSL-сертифікат. HTTPS — це захищена форма HTTP, що означає, що HTTPS-сайти шифрують свій трафік за допомогою SSL. Усі браузери позначають HTTP сайти як «незахищені» (рисунок 3.12) в адресному рядку браузеру. Це надсилає користувачам чіткий сигнал про те, що сайт не заслуговує на довіру.

Рисунок 3.12 — Відображення статусу «Not secure» в адресному рядку сайту HTTP

Деякі хостинги надають SSL-сертифікат безкоштовно, але хостинг що було використано при розробці програмного засобу із вебінтерфейсом для ветеринарної клініки, потребує додаткової плати за сертифікат. Додаткова плата становить 100 грн на місяць.

ВИСНОВКИ

В бакалаврській дипломній роботі було проаналізовано особливості роботи сучасних ветеринарних клініки та потреби її клієнтів. В процесі аналізу було виявлено що існує можливість у підвищені ефективності роботи клініки та задоволенню потреб її клієнтів за рахунок створення програмного засобу із використанням вебінтерфейсу. Результатом роботи став обгрунтований, спроектований та реалізований програмний засіб із вебінтерфейсом для ветеринарної клініки мовами JavaScript та PHP. При цьому верстка вебінтерфейсу виконана з використанням мови розмітки HTML, таблиці стилів CSS та СКБД MySQL для роботи з базою даних.

В дипломній роботі було розроблено структуру та макет вебінтерфейсу, який включає обґрунтування вибору дизайну програмного засобу та його колірної гами. Використовуючи сервіс Figma, створено макети сторінок вебінтерфейсу та розроблено логотип. Було детально проаналізовано технології реалізації розмітки та стилів вебсайтів, а також технології для програмування та роботи з базами даних.

Також був розроблений функціонал, який дозволяє додавати, видаляти та отримувати інформацію з бази даних. Для демонстрації функціональних можливостей програмного засобу було проведене тестування, під час якого відправлялись форми запису на прийом у ветклініку на сервер та створювались користувачі з різним доступом прав. Тестування підтвердило коректну роботу програмного засобу та його здатність задовольняти вимоги.

Дипломна робота продемонструвала ефективність використання PHP, JavaScript, SQL, HTML/CSS для розробки сучасних програмних засобів із вебінтерфейсом, що забезпечують широкий набір функціональних можливостей і зручність у користуванні.

У підсумку, розроблений програмний засіб підвищує ефективність роботи клініки та рівень задоволеня потреб її клієнтів, демонструє високу продуктивність, універсальність і можливість подальшого розвитку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Веб-технології — що це таке та які найпопулярніші? [Електронний ресурс] // FutureNow — Режим доступу до ресурсу: https://futurenow.com.ua/veb-tehnologiyi-shho-tse-take-ta-yaki-najpopulyarnishi/.
2. JavaScript Tutorial [Електронний ресурс] // W3Schools — Режим доступу до ресурсу: https://www.w3schools.com/js/.
3. Site Mapping How To: What Website Page Mapping Is & Why You Need It [Електронний ресурс] // Slickplan — Режим доступу до ресурсу: https://slickplan.com/blog/site-mapping-a-comprehensive-guide
4. О. Цеслів. Основи програмування та веб-дизайн / Ольга Цеслів — Київ : Київський політехнічний інститут імені Ігоря Сікорського, 2020. — 149 с.
5. Figma for Beginners tutorial [Електронний ресурс] // Figma Learn — Режим доступу до ресурсу: https://help.figma.com/hc/en-us/sections/4405269443991-Figma-for-Beginners-tutorial-4-parts.
6. 7 принципів створення вдалого логотипу [Електронний ресурс] // Komarov Design — Режим доступу до ресурсу: https://www.komarov.design/printsipi-stvoriennia-vdalogho-loghotipa/.
7. Даккет Д. JavaScript та JQuery: Інтерактивна веб-розробка / Джон Дакетт. — Київ: Видавництво Кодекс, 2022. — 640 с.
8. Проста валідація форм з HTML5 [Електронний ресурс] // DevZone — Режим доступу до ресурсу: https://devzone.org.ua/post/prosta-validaciia-form-z-html5.
9. SMTP.js — Send Email without a Server from the Browser [Електронний ресурс] // Medium — Режим доступу до ресурсу: https://medium.com/@Scofield_Idehen/smtp-js-send-email-without-a-server-from-the-browser-9c6babfa80d9.
10. How to get started [Електронний ресурс] // ElasticEmail — Режим доступу до ресурсу: https://help.elasticemail.com/en/articles/2361393-how-to-get-started.
11. SweetAlert Guides [Електронний ресурс] // SweetAlert — Режим доступу до ресурсу: https://sweetalert.js.org/guides/.

12. Каталог ветеринарних препаратів [Електронний ресурс] // Бровафарма — Режим доступу до ресурсу: https://brovapharma.ua/katalog-tovarov.

13. How TO — Testimonials [Електронний ресурс] // W3Schools — Режим доступу до ресурсу: https://www.w3schools.com/howto/howto_css_testimonials.asp.

14. Getting Started With Swiper [Електронний ресурс] // Swiperjs — Режим доступу до ресурсу: https://swiperjs.com/get-started.

15. Getting Started with MySQL [Електронний ресурс] // MySQL — Режим доступу: https://dev.mysql.com/doc/mysql-getting-started/en/.

16. Мулеса О.Ю. Основи мови запитів SQL. — Ужгород, 2015. — 48 с

17. PHP Data Objects [Електронний ресурс] // php — Режим доступу до ресурсу: https://www.php.net/manual/en/book.pdo.php.

18. Functional Testing: What it is [Електронний ресурс] // testsigma — Режим доступу до ресурсу: https://testsigma.com/guides/functional-testing/.

19. База знань. Test case [Електронний ресурс] // QALight — Режим доступу до ресурсу: https://qalight.ua/baza-znaniy/test-case-2/

20. What is Compatibility Testing and How to Run It? [Електронний ресурс] // Qamadness — Режим доступу до ресурсу: https://www.qamadness.com/knowledge-base/what-is-compatibility-testing-and-how-to-run-it/.

21. Звичайні питання та відповіді по хостингу [Електронний ресурс] // ho.ua — Режим доступу до ресурсу: https://www.ho.ua/uk/faq.html.

22. PhpMyAdmin’s documentation [Електронний ресурс] // PhpMyAdmin — Режим доступу до ресурсу: https://docs.phpmyadmin.net/en/latest/.

23. What is SSL? | SSL definition [Електронний ресурс] // Cloudflare — Режим доступу до ресурсу: https://www.cloudflare.com/learning/ssl/what-is-ssl/

ДОДАТОК А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ
д.т.н., проф Азаров О.Д.
«06» травня 2024 року

ТЕХНІЧНЕ ЗАВДАННЯ
на виконання бакалаврської дипломної роботи
«Програмний засіб із вебінтерфейсом для ветеринарної клініки»
08-54.БДР.001.00.000.ТЗ

Керівник роботи: к.т.н., ст. викл. каф.ОТ
_____ Дудник О. В.

Виконав: студент гр. КІ-22мсз
_____Куліш Д. В.

ВНТУ 2024

1 Підстави для виконання бакалаврської дипломної роботи (БДР)

1.1 Важливою є актуальність дослідження у напрямку бакалаврської дипломної роботи, яка виправдовується тим, що онлайн сервіси стають невід'ємною частиною життя, і ветеринарна клініка, яка надає можливість запису на прийом через веб-додаток, дає клієнтам можливість забезпечити собі необхідний сервіс у зручний для них час. Тим самим, наявність програмного засобу у ветеринарної клініки стає важливою конкурентною перевагою.

1.2 Наказ про затвердження теми БДР

2 Мета БДР і призначення розробки

2.1 Мета роботи — розробка програмного засобу із вебінтерфейсом для ветеринарної клініки, який буде складатись із бази даних для зберігання даних записів на прийом, вебінтерфейсу з основної інформацію про ветклініку та програмної частини для відправки форми запису на сервер.

2.2 Призначення розробки — виконання бакалаврської дипломної роботи із подальшою підтримкою та розвитком.

3 Вихідні дані для виконання БДР

3.1 Обґрунтування доцільності створення програмного засобу із вебінтерфейсом для ветеринарної клініки.

3.2 Проведення аналізу сучасних технологій для реалізації вебінтерфейсів.

3.3 Огляд існуючих аналогів вебінтерфейсів ветеринарних клінік.

3.4 Розробка структурної схеми програмного засобу.

3.5 Розробка функціоналу програмного засобу.

4 Вимоги до виконання БДР

Програмний засіб із вебінтерфейсом має коректно відповідати на запити користувачів, відправляти форму запису на прийом на електронну пошту, обробляти помилки при відправці форми та авторизації користувачів.

5 Етапи БДР та очікувані результати

Етапи роботи та очікувані результати приведено в таблиці А.1.

Таблиця А.1 — Етапи БДР

№ з/п	Назва етапів дипломної роботи	Термін виконання		Очікувані результати
		початок	закінчення	
1	Аналіз завдання	09.03.2024	11.03.2024	Задачі дослідження
2	Огляд існуючих технологій для розробки вебінтерфейсів	12.03.2024	16.03.2024	Аналітичний огляд літературних джерел
3	Огляд підходів проектування баз даних та програмування веб-додатків	17.03.2024	21.03.2024	Розділ 1
4	Аналіз та вибір структури програмного засобу	22.03.2024	27.03.2024	Розділ 2
5	Аналіз та вибір дизайну вебінтерфейсу	28.03.2024	05.04.2024	Розділ 2
6	Створення бази даних програмного засобу	06.04.2024	11.04.2024	Розділ 2
7	Програмна реалізація вебінтерфейсу	12.04.2024	06.05.2024	Розділ 2
8	Проектування обробки форми запису на прийом та збереження введених даних	07.05.2024	14.05.2024	Розділ 2
9	Тестування та розгортання програмного засобу	15.05.2024	17.05.2024	Розділ 3
10	Оформлення пояснювальної записки та ілюстративного матеріалу	18.05.2024	30.05.2024	ПЗ, додатки, презентація

6 Матеріали, що подаються до захисту БДР

До захисту подаються: пояснювальна записка БДР, графічні матеріали, протокол попереднього захисту роботи на кафедрі, відзив наукового керівника, рецензія опонента, анотації до БДР українською та іноземною мовами, нормоконтроль про відповідність оформлення БДР діючим вимогам.

7 Порядок контролю виконання та захисту БДР

Виконання етапів графічної та розрахункової документації БДР контролюється

науковим керівником згідно зі встановленими термінами. Захист БДР відбувається на засіданні Державної екзаменаційної комісії, затвердженою наказом ректора.

8 Вимоги до оформлювання та порядок виконання БДР

8.1 При оформлювання БДР використовуються:

— ДСТУ 3008: 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302: 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— міждержавний ГОСТ 2.104-2006 «Єдина система конструкторської документації. Основні написи»;

— методичні вказівки до виконання бакалаврських кваліфікаційних робіт зі спеціальності 123 – «Комп'ютерна інженерія» (освітня програма «Комп'ютерна інженерія»). Кафедра обчислювальної техніки ВНТУ 2022.

8.2 Порядок виконання БДР викладено в «Положення про кваліфікаційні роботи на першому (бакалаврському) рівні вищої освіти СУЯ ВНТУ-03.02.02-П.001.01:21».

ДОДАТОК Б

Структурна схема програмного засобу

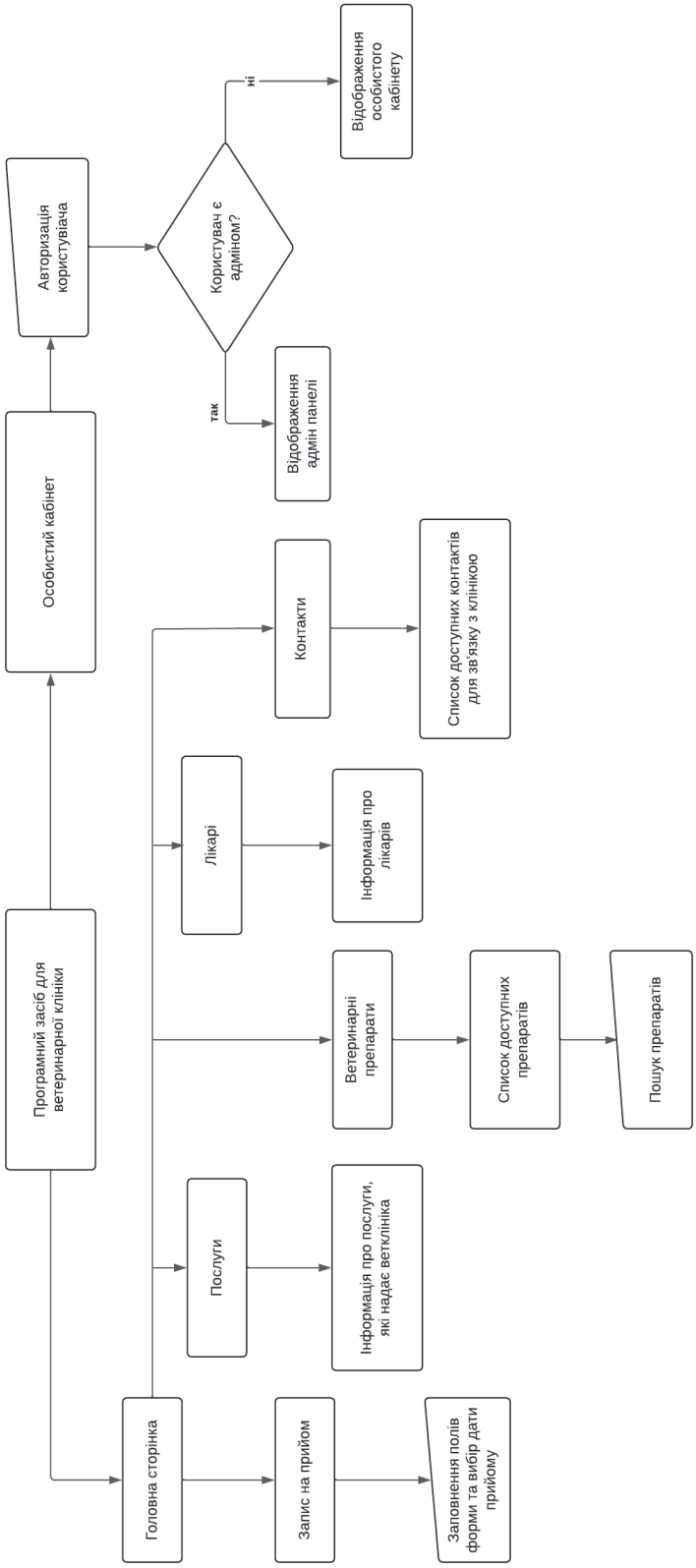

Рисунок Б.1 — Структурна схема програмного засобу

ДОДАТОК В

Лістинг головної сторінки

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta content="width=device-width, initial-scale=1" name="viewport" />
    <script src="https://kit.fontawesome.com/9acea6f878.js"
crossorigin="anonymous"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.0.0/jquery.min.js"></script>
    <script src="https://smtpjs.com/v3/smtp.js"></script>
    <script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
    <link rel="icon" type="image/x-icon" href="/images/favicon.svg">
    <title>Щасливі лапки</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/swiper@11/swiper-
bundle.min.css"/>
    <script src="https://cdn.jsdelivr.net/npm/swiper@11/swiper-bundle.min.js"></script>
    <script
     type="text/javascript"
     src="https://cdn.jsdelivr.net/npm/@emailjs/browser@3/dist/email.min.js"
   ></script>
   <script type="text/javascript">
     (function () {
       emailjs.init("ZBLMptE6vc7jL0HDf");
     })();
    </script>
  <!-- jQuery Modal -->
```

```
  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-modal/0.9.1/jquery.modal.min.js"></script>
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/jquery-modal/0.9.1/jquery.modal.min.css" />
    <title>Our Pet Vetinary Clinic</title>
    <link href='https://fonts.googleapis.com/css?family=Roboto' rel='stylesheet'/>
    <link rel="stylesheet" href="./index.css" />
  </head>
  <body>
    <button onclick="topFunction()" id="myBtn" title="Go to top"><i class="fas fa-angle-up"></i></button>
    <!-- HEADER SECTION -->
    <header class="header-section">
      <div class="container">
        <div class="content">
          <a href="#" class="logo"><img src="images/Group 13.svg" style="width: 10em;"></a>
          <!-- toggle menu -->
          <div class="toggle-menu">
            <i class="fas fa-bars"></i>
          </div>
          <!-- Nav menu -->
          <nav class="nav">
            <ul class="menu">
              <li><a class="active" href="#hero">Головна</a></li>
              <li><a href="#service">Послуги</a></li>
              <li><a href="#about">Препарати</a></li>
              <li><a href="#staff">Лікарі</a></li>
              <li><a href="#footer">Контакти</a></li>
              <? //CHECK FOR ADMIN ROLE
```

```
        require_once __DIR__ . "/database/pdo.php";
        session_start();
        if (isset($_SESSION['username'])) {
          $user = $_SESSION['username'];
          // Proceed with checking the admin role
          $sql = "SELECT is_admin FROM users WHERE username = '$user'";
          $currentRole = $con->query($sql);
          if ($currentRole) {
              $row = $currentRole->fetch_assoc();
              $role = $row['is_admin'];
              if ($role == 1) {
                  echo '<li><a href="admin.php">Адмін панель</a></li>';
              }
              else {
                echo '<li><a href="cabinet.php">Особистий кабінет</a></li>';
            }}}
            else {
              echo '<li><a href="cabinet.php">Особистий кабінет</a></li>';
          }
          ?>
        </ul> </nav>
      </div>
    </div>
  </header>
  <!-- HERO SECTION -->
  <section id="hero">
    <div class="container">
      <div class="content">
```

```
        <div class="text">
          <div class="subtitle">
            <div class="line"></div>
            <span>ВЕТЕРИНАРНА КЛІНІКА <b>«ЩАСЛИВІ ЛАПКИ»</b></span>
          </div>
          <h1>Ваші домашні улюбленці важливі для нас</h1>
          <p>Клініка надає широкий спектр послуг для вашого вихованця цілодобово.
</p>
          <!-- Button trigger modal -->
         <button class="modal-open"><a href="#ex1" class="btn" id="showForm"
rel="modal:open">Запис на прийом</a></button></div>
        <img src="./images/dog-png-22638.png" aria-label="puppy" alt="puppy" />
       </div>
     </div>
   </section>   <!-- Modal -->
  <div id="ex1" class="modal">
    <form id="myForm" action="form_query.php" method="post">
      <div class="form-row">
        <div class="input-data">
          <input type="text" required id="userName" pattern="[А-яІі][а-яІі]+">
          <div class="underline"></div>
          <label>Ваше ім'я</label>
        </div>
        <div class="input-data">
          <input type="tel" required id="phoneNumber" pattern="[0-9]{10}">
          <div class="underline" id="underline"></div>
          <label>Номер телефону</label>
        </div>
      </div>
```

```
    <div class="form-row" style="margin: 39px 0;">
      <div class="input-data">
        <input type="email" required id="userMail" pattern=".+@gmail\.com" name="email">
        <div class="underline" id="underline"></div>
        <label>Електронна пошта</label>
      </div>
      <div class="input-data">
        <input type="text" required id="petName">
        <div class="underline"></div>
        <label>Ім'я улюбленця </label>
      </div>
    </div>
    <div class="form-row">
    <div class="input-data">
      <div class="underline"></div>
      <select  class="dropdown-values" name="type" id="typeConsultation" style="border: none; color: #9b9999; border-bottom: 2px solid rgba(0,0,0, 0.12); font-size: 17px; width: 100%; padding-block: 8px; ">
        <option value="" disabled selected hidden>Оберіть тип прийому</option>
        <option value="Вакцинація">Вакцинація</option>
          <option value="Грумінг">Грумінг</option>
          <option value="Консультація">Консультація</option>
          <option value="Аналізи">Аналізи</option>
          <option value="Повторний огляд">Повторний огляд</option>
      </select>
    </div>
  </div>
    <div class="form-row">
```

```
        <div class="input-data">
          <div class="underline"></div>
          <input placeholder="Оберіть бажану дату прийому" class="date-input"
name="date" type="text"
            onfocus="(this.type='date'),(this.showPicker())" onblur="(this.type='text')"
id="date" />
        </div>
      </div>
      <div class="form-row">
        <div class="input-data textarea">
          <textarea rows="8" cols="80" required  id="problemDescription"
name="problem"></textarea>
          <br />
          <div class="underline"></div>
          <label>Опишіть проблему</label>
          <br />
          <div class="form-row submit-btn" style="width: 100%;
            margin: -24px 19px 150px 0px; justify-content: center;">
            <div class="input-data">
              <div class="inner"></div>
              <input type="submit" value="Запис на прийом" style="background-color: var(--
pry-clr);
              border: 1px solid var(--pry-clr);
              border-radius: 0.6rem;
              padding: 1rem 2.5rem;
              font-size: 17px;
              color: #fff;
              font-family: var(--sec-font);
              transition: all 0.3s ease-in-out;">
            </div>
```

```
        </div>
    </form>
  </div>
</div>
  </div>
    <!-- SERVICE SECTION -->
    <section id="service">
      <div class="container">
        <div class="content">
          <div class="title">
            <h2>ПОСЛУГИ</h2>
            <p>
              Ми пропонуємо широкий спектр послуг, який включає в себе наступні
            </p>
          </div>
          <div class="cards">
            <div class="card">
              <div class="circle">
                <img src="./images/physician.png" alt="radiodroid" />
              </div>
              <div class="text">
                <h5>Терапія</h5>
                <p> Заходи кваліфікованної ветеринарної допомоги </p>
              </div>
            </div>
            <div class="card">
              <div class="circle">
                <img src="./images/needle.png" alt="ambulance-icon" />
```

```
</div>
<div class="text">
  <h5>Вакцинація</h5>
  <p> Планове введення вакцини задля профілактики інфеційних захворювань
  </p>
</div> </div>
```

ДОДАТОК Г

Лістинг функції для відправки даних форми на пошту

```
function emailSend() {
  // User data from form inputs
  const userName = document.getElementById('userName').value;
  const phoneNumber = document.getElementById('phoneNumber').value;
  const mail = document.getElementById('userMail').value;
  const typeConsultation = document.getElementById('typeConsultation').value;
  const petName = document.getElementById('petName').value;
  const date = document.getElementById('date').value;
  const problemDescription = document.getElementById('problemDescription').value;
  function validateRequired(field) {
    return field.value.trim() !== '';
  }
  // Validate each field and display errors
  let isValid = true;
  if (!validateRequired(document.getElementById('date'))) {
    alert('Оберіть бажану дату прийому.');
    isValid = false;
  }
  if (!isValid) return; // Prevent form submission if errors
  // Prepare message body
  const messageBody = "Ім'я клієнта:  " + userName +
    "<br/> Номер телефону:  " + phoneNumber +
    "<br/> Електронна пошта:  " + mail +
  "<br/> Ім'я улюбленця:  " + petName +
    "<br/> Тип прийому:  " + typeConsultation +
    "<br/> Бажана дата прийому:  " + date +
```

```
        "<br/> Проблема:  " + problemDescription;
       // Send email using Email.js
       Email.send({
        SecureToken : "39a51fea-944c-437e-a254-192a2df36fa4",
        To : "mitiakylish2@gmail.com",
        From : "mitiakylish@gmail.com",
        Subject : "Запис на прийом у ветклініку",
        Body : messageBody
      }).then(
       message => {
        if (message == 'OK'){
      swal("Успішно!", "Ваша заявка на прийом надіслана, з вами зв'яжеться наш менеджер.", "success");
        }
        else {
          swal("Помилка", "Ой-ой, сталась помилка. Спробуйте будь ласка ще раз.", "error");
        }
       }
      );
      }
      const form = document.getElementById('myForm');
      form.addEventListener('submit', function(event) {
       event.preventDefault(); // Prevent default form submission
       emailSend();
       const formData = new FormData(form);
      fetch('form_query.php', {
        method: 'POST',
        body: formData  })
```

ДОДАТОК Д

Лістинг логіки сторінки реєстрації

```
<?php
session_start();
   require_once __DIR__ . "/database/pdo.php";
   $username = $_POST['username'];
   $password = $_POST['password'];
   $email = $_POST['email'];
   $password = md5($password);
$sql = "SELECT username FROM `users` WHERE username='{$username}'";
$result = mysqli_query($con,$sql) or die("Query unsuccessful") ;
     if (mysqli_num_rows($result) > 0) {
       header("Location: login.php?msg=failed");
     } else {
$sql = "INSERT INTO `users` (`username`, `password`,`email`) VALUES (:username, :password, :email)";
   $params = [
      "username" => $username,
      "password" => $password,
      "email" => $email,
   ];
   $prepare = $pdo->prepare($sql);
   $prepare->execute($params);
   $pdo->lastInsertId();
   $lastInsertId=$pdo->lastInsertId();
    $_SESSION['login'] = true;
    $_SESSION['username']=$username;
    $_SESSION['password']=$password;
   header("Location: /cabinet.php");
}?>
```

ДОДАТОК Е

Лістинг особистого кабінету

```
<?php
require_once __DIR__ . "/database/pdo.php";
session_start();
$_SESSION['status'] = 1;//if logged in
$_SESSION['status'] = 0;//if logged out
if (!$_SESSION['login']) {
  header("Location:login.php");
  die;
}
$user = $_SESSION['username'];
?>
<!DOCTYPE html>
<html lang="en">
<head>
  <script src="https://kit.fontawesome.com/a55eddfd22.js" crossorigin="anonymous"></script>
  <title>Особистий кабінет</title>
  <meta charset="UTF-8" />
  <meta content="width=device-width, initial-scale=1" name="viewport" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <link rel="stylesheet" href="./index.css" />
  <link rel="stylesheet" href="./profile.css" />
  <link rel="icon" type="image/x-icon" href="/images/favicon.svg">
  <script src="./script-admin.js"></script>
</head>
<body>
  <header class="header-section" style="position:relative;">
    <div class="container">
```

```
<div class="content">
  <a href="#" class="logo"><img src="images/Group 13.svg" style="width: 10em;"></a>
  <!-- toggle menu -->
  <div class="toggle-menu">
    <i class="fas fa-bars"></i>
  </div>
  <!-- Nav menu -->
  <nav class="nav">
    <ul class="menu">
      <li><a href="/#hero">Головна</a></li>
      <li><a href="/#service">Послуги</a></li>
      <li><a href="/#about">Препарати</a></li>
      <li><a href="/#staff">Лікарі</a></li>
      <li><a href="/#footer">Контакти</a></li>
      <li><a class="active" href="cabinet.php">Особистий кабінет</a></li>
      <li><a href="logout.php">Вийти</a></li>
      <?
      //CHECK FOR ADMIN ROLE
      $sql = "SELECT is_admin FROM users WHERE username = '$user'";
      $currentRole = $con->query($sql);
      $row = $currentRole->fetch_assoc();
      $role = $row['is_admin'];
      if ($role == 1) {
        echo '<li><a href = "admin.php">Адмін панель</a></li>';
      }
      ?>
    </ul>
  </nav>
</div>
```

```
  </div>
</header>
<?php
$sql = "SELECT email FROM users WHERE username = '$user'";
$currentEmail = $con->query($sql);
if ($currentEmail === false) {
  echo "Query error: " . $con->error;  // Handle query errors
} else {
  $row = $currentEmail->fetch_assoc();  // Fetch the first row as an associative array
  $email = $row['email'];              // Access the 'email' column
  // echo $email;                      // Output the email string
}
// Prepare the SQL query
$sql = "SELECT id, date, type, problem
      FROM appointments WHERE email = '$email'
      ORDER BY date DESC";
// Execute the query
$result = $con->query($sql);
// Close the connection
mysqli_close($con);
?>
<div class="row">
  <?php
  if ($result->num_rows > 0) {
    echo "<div style='text-align: center; padding-bottom: 2.5em; padding-top: 2em; font-size:25px;'><b>Ваші записи на прийом у ветеринарну клініку: </b> <br></div>";
    } else {
    echo "<div style='text-align: center; padding-bottom: 4em; padding-top: 2em; font-size:25px;'><b>У вас ще немає записів на прийом.</b><br><br><a href='http://vetclinic/#hero'> Записатись на прийом </a></div>";
```

```
        }
        ?>
      </div>
      <div class="column">
        <?php
        $currentDate = date("Y-m-d");
        $columnCount = 0;
        while ($row = $result->fetch_assoc()) {
          echo "<div class='card2'>";
          echo "<b>Дата запису:</b> " . $row["date"] . "<br>";
          echo "<b>Тип прийому:</b> " . $row["type"] . "<br>";
          echo "<b>Проблема:</b> " . $row["problem"] . "<br>";
          echo "<b>АЙДІ:</b> " . $row["id"] . "<br>";
          if ($row["type"] === "Консультація" || $row["type"] === "Повторний огляд")
{
            echo "<img src ='./images/physician.png'" . "<br>";
          } else if ($row["type"] === "Вакцинація") {
            echo "<img src ='./images/needle.png'" . "<br>";
          } else if ($row["type"] === "Аналізи") {
            echo "<img src ='./images/microscope2.png'" . "<br>";
          } else if ($row["type"] === "Грумінг") {
            echo "<img src ='./images/grooming.png'" . "<br>";
          }
          if ($row["date"] > $currentDate) {
            ?>
          <a href="/delete.php?id=<?=$row["id"]?>"class = "button" id="deleteRecord"
onclick="return confirm('Ви дійсно хочете скасувати запис на <?= $row['date']?>?');">
              <i class="fa-solid fa-trash"></i> Скасувати запис
          </a>
            </button>
```

```
            <?
            }
            echo "</div>";
            $columnCount++;
            if ($columnCount % 3 == 0) {
              echo "<br>"; /* Add a line break after every 3 columns */
            }
          }
          ?>
        </div>
      </body>
</html>
```

ДОДАТОК Ж

Лістинг панелі адміністратора

```
<?php
 require_once __DIR__ . "/database/pdo.php";
session_start();
$_SESSION['status'] = 1;//if logged in
$_SESSION['status'] = 0;//if logged out
if(!$_SESSION['login']){
   header("Location:login.php");
   die;
}
$user = $_SESSION['username'];
?>
<!DOCTYPE html>
<html lang="en">
  <head>
  <script                                src="https://kit.fontawesome.com/a55eddfd22.js"
crossorigin="anonymous"></script>
     <meta charset="UTF-8" />
     <meta content="width=device-width, initial-scale=1" name="viewport" />
     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
     <link rel="stylesheet" href="./index.css" />
     <link rel="stylesheet" href="./profile.css" />
     <link rel="icon" type="image/x-icon" href="/images/favicon.svg">
     <title>Адмін панель</title>
     <script src="./script-admin.js"></script>
</head>
<body>
<header class="header-section" style="position:relative;">
      <div class="container">
```

```
<div class="content">
 <a href="#" class="logo"><img src="images/Group 13.svg" style="width: 10em;"></a>
 <!-- toggle menu -->
 <div class="toggle-menu">
  <i class="fas fa-bars"></i>
 </div>
 <!-- Nav menu -->
 <nav class="nav">
  <ul class="menu">
   <li><a href="/#hero">Головна</a></li>
   <li><a href="/#service">Послуги</a></li>
   <li><a href="/#about">Препарати</a></li>
   <li><a href="/#staff">Лікарі</a></li>
   <li><a href="/#footer">Контакти</a></li>
   <?
   //CHECK FOR ADMIN ROLE
   $sql = "SELECT is_admin FROM users WHERE username = '$user'";
   $currentRole = $con->query($sql);
   $row = $currentRole->fetch_assoc();
   $role = $row['is_admin'];
   if ($role == 1) {
    echo '<li><a class="active">Адмін панель</a></li>';
        }
   ?>
   <li><a href="logout.php">Вийти</a></li>
  </ul>
 </nav>
</div>
</div>
```

```
    </header>
    <?php
$sql = "SELECT email FROM users";
$AllUsers = $con->query($sql); // Execute the query and store the result
if ($AllUsers === false) {
    echo "Query error: " . $con->error; // Handle query errors
} else {
    // Loop through each row in the result set
    while ($row = $AllUsers->fetch_assoc()) {
        $email = $row['email'];
       // echo $email . "<br>";  // Display each email on a new line
    }
}
// Close the result set (optional, but recommended for resource management)
$AllUsers->close();
// Prepare the SQL query (modified)
$sql = "SELECT id, date, type, problem, email
        FROM appointments
        ORDER BY date DESC";
// Execute the query
$result = $con->query($sql);
// Close the connection
mysqli_close($con);
?>
<div class="row">
  <?php
  if ($result->num_rows > 0) {
    echo "<div style='text-align: center; padding: 2em; font-size:25px;'><b>Усього
записів - $result->num_rows:</b><br></div>";
  }
```

```
        else{
          echo "<div style='text-align: center;  padding: 3em 4em; font-size:20px;'>У вас ще немає записів на прийом.<br><br><a href='http://vetclinic/#hero'> Записатись </a></div>";
        }
        ?>
      </div>
      <div class="column">
        <?php
        $currentDate = date("Y-m-d");
        $columnCount = 0;
        while ($row = $result->fetch_assoc()) {
          echo "<div class='card2'>";
          echo "<b>Дата запису:</b> " . $row["date"] . "<br>";
          echo "<b>Тип прийому:</b> " . $row["type"] . "<br>";
          echo "<b>Проблема:</b> " . $row["problem"] . "<br>";
          echo "<b>Пошта:</b> " . $row["email"] . "<br>";
          if($row["type"] === "Консультація" || $row["type"] === "Повторний огляд"){
          echo "<img src ='./images/physician.png'" . "<br>";
            }
            else if ($row["type"] === "Вакцинація"){
              echo "<img src ='./images/needle.png'" . "<br>";
            }
            else if ($row["type"] === "Аналізи"){
              echo "<img src ='./images/microscope2.png'" . "<br>";
            }
            else if ($row["type"] === "Грумінг"){
              echo "<img src ='./images/grooming.png'" . "<br>";
            }
    echo "</div>";
```

```
            $columnCount++;
            if ($columnCount % 3 == 0) {
              echo "<br>"; /* Add a line break after every 3 columns */
            }
          }
          ?>
        </div>
        </body>
</html>
```

ДОДАТОК И

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: "Програмний засіб із вебінтерфейсом для ветеринарної клініки"

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

Підрозділ кафедра обчислювальної техніки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 93,7% Схожість 6,3%

Аналіз звіту подібності (відмітити потрібне):

☐ Запозичення, виявлення у роботі, оформлені коректно і не містять ознак плагіату.

☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Захарченко С.М.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Куліш Д. В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Дудник О. В.
(підпис) (прізвище, ініціали)