КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра прикладних інформаційних систем
122 «Комп'ютерні науки»
(шифр і назва спеціальності)

«Прикладні інформаційні системи»
(назва освітньої програми)

Кваліфікаційна робота бакалавра

на тему: «Програмна система обліку у ветеринарній клініці»

Виконала _____
(Підпис)
Ткаченко Анастасія Олексіївна
(прізвище, ім'я, по батькові)

Керівник Краснощок Віктор Миколайович
(прізвище, ім'я, по батькові)
_____
(Резолюція «До захисту»)

Попередній захист:

_____ _____
_____
(Висновок: "До захисту в екзаменаційній комісії ")
Завідувач кафедри _____ Плескач В.Л. _____
(Підпис) (Прізвище, ініціали) (Дата)

Засвідчую, що у цьому курсовому проекті немає запозичень з праць інших авторів без відповідних посилань. 80%
Студент _____
(підпис)

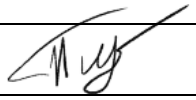
Київ – 2022

Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій
Кафедра прикладних інформаційних систем

Назва теми: «Програмна система обліку тварин у ветеринарній клініці»

Освітня програма: Прикладне програмування

Спеціальність: Комп'ютерні науки

ПІБ	Підпис
Ткаченко Анастасія Олексіївна	

Назва роботи українською та англійською мовами

Програмна система обліку у ветеринарній клініці Software accounting system in a veterinary clinic

Мета бакалаврської роботи, завдання

Мета бакалаврської роботи: Створення зручної програмної системи обліку тварин та супутніх йому процесів у ветеринарній клініці План роботи: 1. Сучасні підходи до розроблення і впровадження програмних систем 2. Аналіз архітектурних рішень і вибір програмних засобів для реалізації програмних систем 3. Програмна реалізація системи обліку тварин у ветеринарній клініці

ПІБ, ступінь, звання наукового керівника роботи: Краснощок В.М., к.т.н., доцент

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ БАКАЛАВРА

Номер	Назва етапів кваліфікаційної роботи бакалавра	Термін виконання етапів кваліфікаційної роботи бакалавра	Відмітка про виконання
1.	Вибір теми та наукового керівника кваліфікаційної роботи бакалавра	26.10.2020	Виконала
2.	Видача завдання кваліфікаційної роботи бакалавра	23.11.2020	Заява
3.	Настановча групова співбесіда з питань кваліфікаційної роботи бакалавра	01.12.2020	Виконала
4.	Затвердження плану кваліфікаційної роботи бакалавра	18.02.2021	Виконала
5.	Підбір та вивчення літературних та інших джерел з теми дослідження	25.02.2021	Виконала
6.	Підготовка і подання науковому керівнику першого варіанту І розділу роботи	05.03.2021	Виконала
7.	Підготовка і подання науковому керівнику першого варіанту ІІ розділу роботи	09.04.2021	Виконала
8.	Підготовка і подання науковому керівнику першого варіанту ІІІ розділу роботи	07.05.2021	Виконала
9.	Подання роботи у першому варіанті	11.05.2021	Виконала
10.	Оформлення пояснювальної записки кваліфікаційної роботи бакалавра	12.05.2021	Виконала
11.	Подання кваліфікаційної роботи бакалавра на попередній захист	**24.05.2021**	Виконала
12.	Врахування зауважень керівника і подання роботи в остаточному варіанті (з відповідним висновком про допуск) на кафедру	28.05.2021	виконала
13.	Затвердження роботи в цілому (підготовка письмового відгуку керівника, письмова рецензія на бакалаврської роботу)	11.06.2021	
14.	Захист кваліфікаційної роботи бакалавра	22.06.2021	
		23.16.2021	
		24.06.2021	
		25.06.2021	

Здобувач вищої освіти _____

Керівник ____ _____

ВІДОМІСТЬ ДИПЛОМНОЇ РОБОТИ

Зміст пояснювальної записки (перелік питань під час дослідження)

Складові частини дипломної роботи	Обсяг, арк. 100
Титульний аркуш	1 ст.
Завдання до дипломної роботи (календарний план проекту)	1 ст.
Відомість дипломної роботи	1 ст.
Пояснювальна записка до дипломної роботи	1 ст.
Анотація	1 ст.
Анотація (іноземною мовою-англійською)	1 ст.
Зміст	2 ст.
Словник термінів	2 ст.
Вступ	2 ст.
1. Сучасні підходи до розроблення і впровадження програмних систем. Аналіз рішень реалізації програмних систем	7 ст.
2. Вибір програмних засобів для реалізації програмної системи обліку тварин у ветклініці	13 ст.
3. Програмна реалізація програмної системи обліку тварин у ветклініці	20 ст.
Висновки	1 ст.
Перелік посилань	2 ст.
Додатки	44 ст.

				ДП ХХХХ 00.000.00		
	ПІБ	Підп.	Дат			
Розробн.	Ткаченко А.О.			Відомість дипломної	Лист	Листів
Керівн.	Краснощок В.М.					

Н/контр.	Базилюк А.М.			роботи	
Зав.каф.	Плескач В.Л.				

АНОТАЦІЯ (РЕФЕРАТ)

До бакалаврської дипломної роботи Ткаченко Анастасії Олексіївни на тему «Програмна система обліку у ветеринарній клініці»

Дана дипломна робота присвячена створенню зручної для використання лікарями і власниками тварин системи обліку та пов’язаних з обліком процесів домашніх улюбленців з використанням сучасних технологій та розробці бази даних для цієї системи.

Розроблене програмне забезпечення являє собою програму, яка надає змогу завантажувати в базу дані, зберігати ці дані в обліковому записі користувача та шукати потрібні дані.

У даній дипломній роботі було розроблено архітектуру, модуль авторизації та реєстрації, модуль завантаження даних та модуль зв’язку з БД, а також розроблено дизайн програмної системи.

Для розробки оптимізованої системи було проведено дослідження актуальних на даний час технологій, що дозволяють створювати програмні системи. Вибір технологій для реалізації дипломної роботи був зроблений по цим критеріям: наявні можливості, масштабованість, програмне забезпечення та вирішення проблем, що можуть виникнути під час розробки програмної системи.

Загальний обсяг роботи: 3 розділи загальною кількістю 100 сторінок, 19 рисунків, 1 таблиця, 13 посилань, 1 додаток.

Ключові слова: програмна система, програмне забезпечення, Java, Scene Builder, MySQL

ANNOTATION (ABSTRACT)

To Anastasia Tkachenko's bachelor's thesis on "Software accounting system in veterinary clinic" topic

This thesis is devoted to the creation of a user-friendly system of pet accounting with the use of modern technologies and the development of a database for this system.

Developed software is a program that allows you to download data to the database, save this data in the user account and search for the necessary data.

In this thesis, the architecture, authorization and registration module, data download module and database communication module were developed, as well as the software system design.

To develop an optimized system, a study of current technologies that allow you to create software systems was conducted. The choice of technologies for the thesis was made according to the following criteria: available capabilities, scalability, software and solutions to problems that may arise during software development.

Total volume of work: 3 sections with a total of 100 pages, 19 figures, 1 table, 13 links, 1 addition.

Keywords: Software system, software, Java, Scene Builder, MySQL

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

FXML – мова розмітки користувальницького інтерфейсу на основі XML

XML (*Extensible Markup Language*) - запропонований консорціумом World Wide Web Consortium (W3C) стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема, через Інтернет.

API (*Application Programming Interface, API*) — набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

СУБД – система управління базами даних

БД – база даних

ПЗ – програмне забезпечення

СКБД – система керування базою даних

ВСТУП

Програмна система - це група інтегрованих програмних засобів, які підтримують певний діловий процес споживача (або його частину) і спільно використовують бази даних.

Актуальність досліджень програмних систем можна показати на прикладі застосунку для додавання домашніх улюбленців у базу даних ветклініки. Усі діагнози та інструкції з лікування видаються у печатному вигляді, що в наш час являється незручно: для власника тварини ці папери являються єдиним зберігачем інформації і при їх пошкодженні перевірити потрібну інформацію буде можливо тільки за допомогою дзвінків у клініку, що забирає багато часу. Також при повторних походах у ветклініку оновлювана інформація діагнозів буде додавати кількість макулатури і пошук актуального діагнозу та рецепту тих же ліків буде займати ще більше часу. Тож створення додатку для обліку тварин у ветклініці буде економити час власникам, а також буде набагато зручнішим і ефективнішим для пошуку інформації про стан улюбленця і методи його лікування.

Теоретичне значення даної роботи представлено у дослідженні та аналізі питань, що стосуються розробки програмних систем, вибору актуальних технологій у створенні програмної системи, які будуть корисним у застосуванні у різних обставинах.

Практичне значення дипломної роботи можна охарактеризувати створенням програмної системи, яка дозволить вносити дані про тварину та її власника, отримувати інформацію про діагноз та лікування зареєстрованого улюбленця, розклад лікарів, наявних безхатніх тваринок а також загублених тваринок. Додатково реалізовано процес реєстрації, авторизації, додавання тварини та пошук її серед вже існуючих.

Метою дипломної роботи є розробка програмної системи обліку тварин та пов’язаних з ними процесів у ветеринарній клініці

Завдання дослідження:

- аналіз існуючих компонентів програмних систем, їх принципів роботи та взаємодії;
- опис програмно-технічних рішень, різних видів забезпечення програмної системи;
- ознайомлення з існуючими програмними системами;
- ознайомлення з процесом створення програмної системи від створення прототипу до проведення тестування;
- реалізація на практиці програмної системи обліку тварин у ветеринарній клініці

Об’єктом дослідження у цій роботі є процес обліку у ветеринарній клініці, а **предметом дослідження** – програмні засоби обліку у ветеринарній клініці

Методи дослідження полягають в системному аналізі та синтезі, теорії систем та застосуванні клієнт-серверної моделі архітектурного рішення досліджуваної системи, методах математичної статистики для збору даних веб-сервісу.

У процесі виконання дипломної роботи було використано: редактор коду Intellig IDEA, документацію з використання технологій програмної системи, мови програмування Java, базу даних MySQL.

Структура дипломної роботи складається із вступу, трьох розділів, висновку, додатків і списку використаних джерел.

РОЗДІЛ 1: СУЧАСНІ ПІДХОДИ ДО РОЗРОБЛЕННЯ І ВПРОВАДЖЕННЯ ПРОГРАМНИХ СИСТЕМ. АНАЛІЗ РІШЕНЬ РЕАЛІЗАЦІЇ ПРОГРАМНИХ СИСТЕМ

1.1 Основні поняття програмної системи

Програмною системою або програмним забезпеченням називається загальне поняття, яке є набором кодованих інструкцій для керування процесором (CPU) комп'ютера. Процесором CPU комп'ютера зчитуються кодовані інструкції, які потім цей же процесор і починає виконувати. Виконання комп'ютером програмної системи полягає у тому, що він керує апаратними компонентами комп'ютера та маніпулює інформацією. Для прикладу, прийом інформації з клавіатури та її відображення на екран є типовим для персональних комп'ютерів.

Для того щоб вирішити конкретні задачі комп’ютеру потрібно послідовно виконати чітко визначений набір операцій. Ці операції ж являють собою набір дій, які виконуються центральним процесором. Програмою же встановлюються необхідні дії та послідовність їх виконання. А набір програм, що забезпечують можливість використання комп'ютера для вирішення різноманітних завдань, створюють комп'ютерне програмне забезпечення.

Усе програмне забезпечення, використане комп'ютерами, можна поділити на 3 основні види:

Системне програмне забезпечення. Системою ПЗ являється сукупність програм, що призначуються для забезпечення роботи комп'ютера та його мереж і для організації створення взаємодії між користувачем і комп'ютером.

До системного ПЗ в основному входять:

- Оболонки цих операційних систем;

- драйвера;
- ОС (операційні системи);
- архіватори;
- антивіруси;
- програми що обслуговують диски,
- програми що обслуговують комп'ютерні мережі та інше.

Прикладне програмне забезпечення. Прикладним ПЗ називають сукупність програм, які призначені для виконання завдань за допомогою комп'ютера.

Пакетом прикладних програм являється система програм, яка забезпечує розв'язання задач певного типу.

Прикладами прикладного ПЗ можна вважати:

- текстові редактори і текстові процесори;
- електронні таблиці;
- системи управління базами дани;
- графічні пакети;
- комп'ютерні ігри і системи мультимедіа;
- навчаючі програми;
- програми створення презентацій;
- системи штучного інтелекту та експертні системи;
- програмне забезпечення для роботи з електронною поштою тощо.

Використання прикладного ПЗ вимагає певного набору апаратних засобів, відповідного обсягу пам’яті комп’ютера, певного системного програмного забезпечення. Тобто при виборі прикладної програми користувач повинен враховувати можливості свого комп’ютера.

Третій вид програмного забезпечення — системи програмування, які призначені для розробки інших програм.

Програмне забезпечення комп'ютера постійно вдосконалюється. Покращена програма може істотно відрізнятися від попередньої своєї версії або варіанту. Різні версії мають однакові назви, і щоб розрізнити їх, до назви програми додається відповідний номер версії. Однак бувають випадки, коли деякі функції попередніх версій відсутні в наступних версіях. Через це вам потрібно точно знати, з якою версією програми ви працюєте, оскільки різні версії мають різні можливості.

1.2 Технології реалізації програмних систем

Технологія розробки програмного забезпечення повинна відповідати наступному переліку вимог:

1. Стандартизація мов проектування програм, проектування та тестування програмних модулів, а також гарантії їх якості. Це дозволить значно зменшити тиражування розробок, запровадити програмування на складання та накопичити якісні програмні продукти на підприємствах і в країні для його повторного використання в якості стандартних компонентів.

2. Здійснення постійного контролю та забезпечення якості програми або програм.

3. Програма не повинна мати неперевірених способів і ситуацій функціонування, які в результаті призводять до несподіваних результатів.

4. Користувач програми повинен описати чітке уявлення про можливості програми та технологічні умови її роботи, які можуть гарантувати певні функції чи якості.

5. Технологія розробки програмного забезпечення повинна забезпечувати відмову від програмного продукту від його розробника, тобто людський фактор у програмуванні має бути зведений до мінімуму.

6. Технологія розробки програмного забезпечення та засоби його підтримки/автоматизації повинні забезпечувати цілеспрямовану роботу команди програмістів, а не окремих осіб. Вона повинна спонукати колектив працювати тільки правильно і злагоджено; має автоматично блокувати будь-які несанкціоновані дії.

7. Необхідно вести точну документацію всіх етапів розробки. Документація також має бути записана та збережена на магнітних носіях. Доступ до цієї інформації має бути відкритим, простим і автоматизованим.

8. Робота користувача має забезпечуватися розробленою інформаційно-довідковою системою.

9. Інструменти автоматизації технологій повинні охоплювати всі етапи роботи команди програмістів.

10. Технологія розробки програмного забезпечення повинна бути легкою для вивчення, з інструментами, які автоматично вмикаються.

11. Технологія розробки програмного забезпечення повинна мати засоби автоматичного запису в хронологічному порядку всіх дій, які виконуються в процесі колективного виробництва програмного забезпечення - повинні вестись і зберігатися в системних журналах (протоколах, щоденниках) розробки. Ці інструменти повинні дозволити відновити будь-який стан процесу розробки в будь-який момент виробництва програмного продукту.

1.3 Забезпечення якості ПЗ

Якість програмного забезпечення визначається асоціацією IEEE як ступінь відповідності програмного забезпечення встановленому набору властивостей. У Міжнародному стандарті якості ISO 9000: 2007 це поняття визначається як набір

характеристик програмного забезпечення, що відповідають встановленим і очікуваним вимогам. Якісні характеристики програмного забезпечення включають:

• **функціональність** - виконання заявлених функцій, відповідність стандартам, функціональна сумісність, безпека, точність;

• **надійність** - стійкість до збоїв, можливість відновлення, комплектність;

• **ефективність** - економія часу, ефективність використання;

• **простота використання** - ергономічність, інтуїтивна зрозумілість, повна документація;

• **зручність в обслуговуванні** - стабільність, придатність для управління та змін;

• **портативність** - простота монтажу, замінність, сумісність.

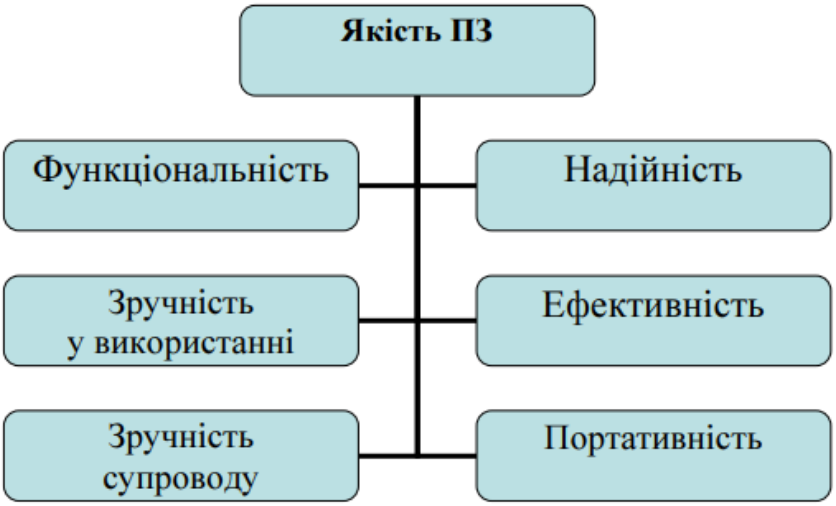


Рисунок 1.1 Характеристики якості програмного забезпечення

1.4 Постановка задачі

У зв’язку з ситуацією в країні та в світі в цілому складно відвідувати будь-які заклади, включаючи ветеринарні клініки. Основною задачею даної роботи являється створення зручної для користування програмної системи, що дає

можливість не тільки вносити домашнього улюбленця у базу даних ветклінік, а також переглядати його оновлювані дані, передивлятись інформацію про лікарів та їхній розклад для можливості записатись на консультацію або просто з ними зв’язатись, дивитись інформацію про розшукуваних улюбленців і за необхідності вносити в цей список свого загубленого улюбленця, передивлятись списки безпритульних тварин, а також змінити господаря свого улюбленця.

1.5 Технічне завдання

Дана робота розрахована на зручне використання власниками тварин з наданням їм можливості реєструвати своїх домашніх улюбленців для подальшого перегляду їх діагнозів та методів лікування, оновленням яких займаються люди з доступом до баз з даними тварин. Також власнику тварини буде надаватись можливість віддати улюбленця під нагляд іншої людини, або помістити улюбленця у «розшук».

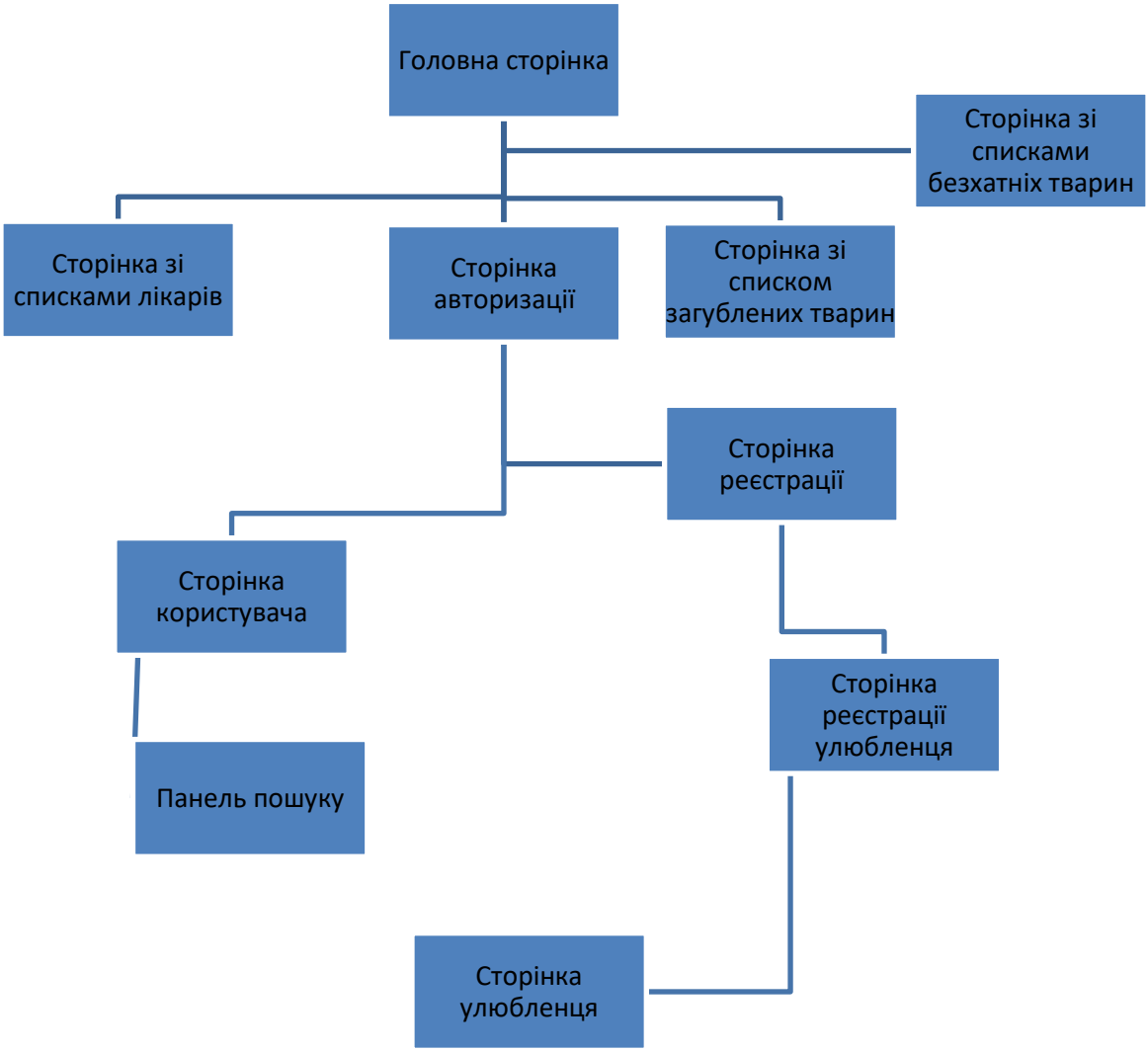


Рисунок 1.2 Карта сторінок проекту

База даних проекту поділяється на декілька таблиць, залежних і незалежних одна від одної. Залежними один від одної являються таблиці улюбленців і власників. Користувачу надається право тільки вносити дані у базу даних через графічний інтерфейс, перебирати дані таблиць для пошуку потрібних даних, а також переміщувати дані однієї таблиці в іншу (відмітка улюбленця як загубленого). Змінювати дані може тільки людина з доступом до самої бази даних

Висновки

В цьому розділі були розглянуті основні поняття програмних систем, технології їх реалізації та забезпечення якості. Тож можемо зробити висновок що якісна програмна система це:

- Система, яка виконує заявлені функції
- Завершена та стійка до відмов система з можливістю відновлення
- Ефективна у використанні система
- Система з інтуїтивною зрозумілістю, що робить її зручною у використанні
- Стабільна, придатна для контролю та внесенню змін система

РОЗДІЛ 2: ВИБІР ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ ОБЛІКУ ТВАРИН У ВЕТЕРИНАРНІЙ КЛІНІЦІ

В даному розділі проводиться опис програмних засобів по розробці програмної системи обліку тварин у ветклініці. Для якісної розробки будуть проведені наступні роботи:

- аналіз вихідних даних;
- опис технологій та додатків програмних систем;
- опис функціоналу.

2.1 Аналіз вихідних даних

Для створення програмної системи було виведено наступну реалізацію архітектури та компонентів, які повинні відповідати наступним умовам:

- Розробка системи авторизації та аутентифікації;
- забезпечення чистоти, читаності, простоти та повторного використання коду;
- рефакторинг та оптимізація контролерів;
- отримання актуальної інформації про діагноз та метод лікування улюбленця;
- пошук лікаря та отримання актуальної інформації про його розклад;
- можливість вносити інформацію в потрібні бази БД.

2.2 Опис технологій та допоміжних засобів програмної системи

Для реалізації проекту було використано технології та бібліотеки створення програмних систем: мову програмування Java, базу данних MySQL, інтсрументарій Java FX та інше. За допомогою даних технологій можна виконати

усі поставлені задачі практичного завдання роботи. Для ознайомлення з використаними технологіями у розробці застосунку, надається повний опис кожної з використаних технологій, їх функціонал та обґрунтування того з якою метою вони були залучені.

2.2.1 Мова програмування Java та її інструментарій Java FX

Java — це мова, на якій компілюються й інтерпретуються програми. В той же час вона має доволі просту структуру мови високого рівня. Написана програма компілюється у проміжну форму - байт-код. Потім програма виконується, що означає що вона інтерпретується середовищем виконання Java. Байт-код доволі сильно відрізняється від машинного коду, що являється послідовністю нулів і одиниць. Байт-кодом називається набір інструкцій, подібних до команд асемблера. Машина виконує машинний код безпосередньо, тож байт-код потрібно інтерпретувати. Виходячи з цього, машинний код може використовуватись лише на конкретній платформі, для якої він і скомпільований. В свою чергу байт-код може виконуватися на будь-якій платформі, у якій наявне середовище виконання Java. Ця функція і робить програми Java незалежними від архітектури. Оскільки байт-код є проміжною формою програми, для його інтерпретації потрібно небагато.

Байт-код створюється для машини, якої насправді не існує. Ця машина називається віртуальною машиною Java (JVM), вона існує лише в пам’яті комп’ютера. Створення компілятора байт-коду Java для неіснуючої машини — це лише половина процесу, який забезпечує незалежність архітектури. Друга частина виконується інтерпретатором Java, який виступає посередником між віртуальною машиною Java і реальним комп'ютером.

Розробники Java створювали мову з метою втілення таких базових принципів:

◊ простота;

◊ безпека;

◊ перенесення;

◊ об’єктно-орієнтована направленість;

◊ стійкість щодо помилок;

◊ багатопоточність;

◊ незалежність від архітектури;

◊ інтерпретація;

◊ висока продуктивність;

◊ розподіленість;

◊ динамічність.

JavaFX - це GUI інструментарій (Graphical user interface - графічний інтерфейс користувача - різновид користувальницького інтерфейсу, в якому всі елементи (кнопки, меню, значки, списки) представлені користувачеві на дисплеї, виконані у вигляді картинок, графіки .) для Java.

JavaFX спрямований на створення ігор та настільних додатків на Java. Фактично, він замінить Swing через запропонований новий інструмент GUI для Java. Це також дозволяє нам стилізувати файли макета GUI (XML) і зробити їх більш елегантними за допомогою CSS, так само, як ми звикли до мережевих програм.

JavaFX додатково працює з інтегрованою 3D-графікою, а також аудіо, відео та вбудованими мережними додатками в єдиний інструментарій GUI… Він простий у освоєнні та добре оптимізований. Він підтримує багато операційних систем, а також Windows, UNIX системи та Mac OS.

Особливості JavaFX:

- JavaFX спочатку поставляється з великим набором частин графічного інтерфейсу, таких як будь-які кнопки, текстові поля, таблиці, дерева, меню, діаграми і т.д., що в свою чергу заощадить нам вагон часу.

- JavaFX часто юзає стилі CSS, і ми зможемо використовувати спеціальний формат FXML для створення GUI, а не робити це в коді Java. Це полегшує швидке розміщення графічного інтерфейсу користувача або зміну зовнішнього вигляду чи композиції без необхідності довго грати у коді Java.

- JavaFX має готові до використання частини діаграми, тому нам не потрібно писати їх з нуля будь-коли, коли вам потрібна базова діаграма.

- JavaFX додатково поставляється за допомогою 3D графіки, яка часто корисна, якщо ми розробляємо якусь гру або подібні програми.

2.2.2 Програмна платформа Intellij IDEA

IntelliJ IDEA — це інтелектуальна, контекстно-залежна IDE для роботи з Java та іншими мовами JVM, такими як Kotlin, Scala та Groovy, у всіх типах програм. Крім того, IntelliJ IDEA Ultimate може допомогти вам розробити повноцінні веб-додатки зі своїми потужними інтегрованими інструментами, підтримкою JavaScript та пов’язаних технологій, а також розширеною підтримкою популярних фреймворків, таких як Spring, Spring Boot, Jakarta EE, Micronaut, Quarkus, Гелідон . Крім того, ви можете розширити IntelliJ IDEA за допомогою безкоштовних плагінів, розроблених JetBrains, які дозволять вам працювати з іншими мовами програмування, включаючи Go, Python, SQL, Ruby і PHP.

Кожен аспект IntelliJ IDEA розроблено щоб забезпечити безперебійну роботу з коробки. Він забезпечує швидкий доступ до всіх функцій і інтегрованих

інструментів, які важливі для вашої роботи, а також надає широкий спектр налаштувань. Ви можете налаштувати все для підтримки свого робочого процесу: встановити ярлики, встановити плагіни, налаштувати інтерфейс на свій смак тощо.

Незважаючи на те, що IntelliJ IDEA розроблена в основному для розробки на Java, вона також розуміє багато інших мов програмування, включаючи:

- Groovy,
- Kotlin,
- Scala,
- JavaScript,
- TypeScript,
- SQL,

і надає розумну допомогу в кодуванні для кожної з них. Початкова індексація вихідного коду дозволяє IDE створити віртуальну карту проекту. Використовуючи інформацію з віртуальної карти, він може виявляти помилки на льоту, пропонувати варіанти завершення коду з точним усвідомленням контексту, виконувати рефакторинг тощо.

2.2.3 Інструмент візуального компонування Scene Builder

JavaFX Scene Builder-ом називається інструмент візуального компонування, який може дозволяти своїм користувачам доволі швидко розробляти інтерфейси користувача JavaFX без потреби кодування. Користувачам надається змога перетягувати компоненти інтерфейсу користувача в робочу область, а також змінювати їх властивості, застосовувати таблиці стилів і тд. А сам код FXML для створеного ними макета буде автоматично генеруватись у фоновому режимі. Результатом буде файл FXML, який потім буде можливо об’єднати з проектом Java, зв’язавши інтерфейс користувача з логікою програми.

У користувача є можливість використовувати Scene Builder в поєднанні з будь-якою IDE Java, але більш тісно він все ж інтегрований з IDE NetBeans. Нам надається можливість прив’язати користувальницький інтерфейс до нашого вихідного коду, який, в свою чергу, оброблятиме події та дії, що виконуються з кожним елементом у простому процесі, запустити програму в NetBeans, і будь-які зміни FXML у NetBeans також будуть відображені у нашому проекті Scene Builder.

Scene Builder генерує FXML, мову розмітки на основі XML, яка дозволятиме користувачам визначати інтерфейс програми окремо від логіки програми. Ви також можете відкривати та редагувати наявні файли FXML, авторами яких є інші користувачі.

Scene Builder написаний як програма JavaFX, яка підтримується в Windows, Mac OS X і Linux. Це чудовий приклад повноцінного настільного додатка JavaFX. Scene Builder упакований як окрема програма, що в свою чергу означає, що він постачається разом із власною приватною копією JRE.

Ключові риси:

JavaFX Scene Builder включає такі ключові функції:

- **Інтерфейс WYSIWYG із перетягуванням дозволяє швидко створювати макет GUI без необхідності писати вихідний код.** Нам надається можливість додавати, комбінувати і під кінець редагувати елементи керування графічним інтерфейсом JavaFX до свого макета – в цьому допоможуть бібліотеки елементів керування GUI та панелі вмісту.

- **Тісна інтеграція з IDE NetBeans забезпечує оптимальний робочий процес розробки.**

- **Інтеграція з будь-якою IDE Java проста, оскільки це окремий інструмент розробки.**

- **Автоматичне генерування коду FXML відбувається під час створення та зміни макета GUI**. Згенерований код FXML зберігається в окремому файлі від джерела логіки програми та файлів таблиць стилів.

- **Функції редагування та попереднього перегляду в реальному часі дозволяють швидко візуалізувати зміни макета графічного інтерфейсу, які ви вносите без необхідності компіляції.** Ці функції допомагають мінімізувати час розробки вашої програми. Ви також можете призначити каскадні таблиці стилів (CSS) своєму макету графічного інтерфейсу та попередньо переглянути застосований вигляд і вигляд.

- **Надається доступ до повної бібліотеки елементів керування графічним інтерфейсом JavaFX.** Щоб переглянути повний список підтримуваних компонентів графічного інтерфейсу JavaFX 8, введіть FX8 у текстовому полі пошуку панелі «Бібліотека». Список включає компоненти TreeTableView, DatePicker і SwingNode.

- **Тепер доступна можливість додавати власні компоненти графічного інтерфейсу до бібліотеки.** Бібліотеку доступних компонентів графічного інтерфейсу можна розширити, імпортуючи налаштовані компоненти графічного інтерфейсу з файлів JAR сторонніх розробників, файлів FXML або додаючи їх із панелей «Ієрархія» або «Вміст». Додаткову інформацію див. у посібнику користувача Scene Builder.

- **Надається підтримка 3D.** Документи FXML, що містять тривимірні об’єкти, тепер можна завантажувати та зберігати в інструменті Scene Builder 2.0. Ви можете переглядати та редагувати властивості тривимірних об’єктів за допомогою панелі «Інспектор» (властивості комплексу «Матеріал і сітка» поки що не підтримуються). Однак ви не можете створювати нові тривимірні об’єкти за допомогою інструмента Scene Builder.

- **Додано підтримку форматованого тексту.** Новий контейнер TextFlow тепер доступний у бібліотеці компонентів графічного інтерфейсу. Ви можете перетягувати кілька текстових вузлів та інших типів вузлів у контейнер TextFlow. Ви також можете безпосередньо маніпулювати текстовими вузлами, щоб перевпорядкувати їх у контейнері. Вбудовані функції та функції редагування властивостей також доступні для кожного текстового вузла.

- **JavaFX Scene Builder Kit постачається разом із Scene Builder 2.0**. Набір являє собою API, який дозволяє інтегрувати панелі та функціональні можливості Scene Builder безпосередньо в графічний інтерфейс більшої програми або IDE Java, наприклад NetBeans, IntelliJ і Eclipse. Дивіться примітки до випуску JavaFX Scene Builder для отримання додаткової інформації.

- **Підтримка CSS забезпечує гнучке керування зовнішнім виглядом інтерфейсу вашого додатка.**

- **Підтримка різних платформ надається в операційних системах Windows, Linux і Mac OS X.**

2.2.4 База даних MySQL

MySQL — це безкоштовна система керування реляційними базами даних, що була розроблена компанією «TcX» для збільшення швидкості обробки великих баз даних. Ця система керування базами даних (СКБД) з відкритим вихідним кодом була створена як альтернатива комерційним системам. MySQL з самого початку була дуже схожою на mSQL, але з часом вона розширилась, і тепер MySQL є однією з найпоширеніших систем керування базами даних. Вона використовується в основному для створення динамічних веб-сторінок, оскільки має відмінну підтримку з різних мов програмування.

MySQL на даний момент є найпопулярнішою системою керування базами даних, яка використовується для управління реляційною базою даних. Це програмне забезпечення баз даних з відкритим вихідним кодом, яке підтримується Oracle. MySQL — це швидка, масштабована та проста у використанні система управління базами даних у порівнянні з Microsoft SQL Server та Oracle Database. Зазвичай MySQL використовується в поєднанні зі сценаріями PHP для створення потужних і динамічних корпоративних серверів або веб-додатків.

MySQL був розроблений шведською компанією MySQL AB і написаний на мові програмування C і C++. Це програмне забезпечення системи управління реляційною базою даних (RDBMS), яке забезпечує багато речей, а саме:

- дозволяє нам виконувати операції з базою даних над таблицями, рядками, стовпцями та індексами.

- визначає зв'язки з базою даних у вигляді таблиць (набір рядків і стовпців), також відомими як зв'язки.

- забезпечує цілісність зв'язків між рядками або стовпцями різних таблиць.

- дозволяє автоматично оновлювати індекси таблиць.

- використовує багато запитів SQL і поєднує корисну інформацію з кількох таблиць для кінцевих користувачів.

MySQL використовує архітектуру клієнт-сервер. Ця модель призначена для кінцевих користувачів, які називаються клієнтами, для доступу до ресурсів із центрального комп'ютера, відомого як сервер, через мережеві служби. Тут клієнти роблять запити через графічний інтерфейс користувача (GUI), і сервер повертає потрібний результат, як тільки інструкції збігаються. Процес MySQL такий же, як і в моделі клієнт-сервер.

Для виконання дипломної роботи було створено наступні колекції – Користувачі (users), Тваринки(pets), Лікарі(doctors), Загублені тваринки(lost pets), Безпритульні тваринки(homeless pets). Графічне зображення усіх колекцій можна побачити на рисунках 2.9 і 2.10.

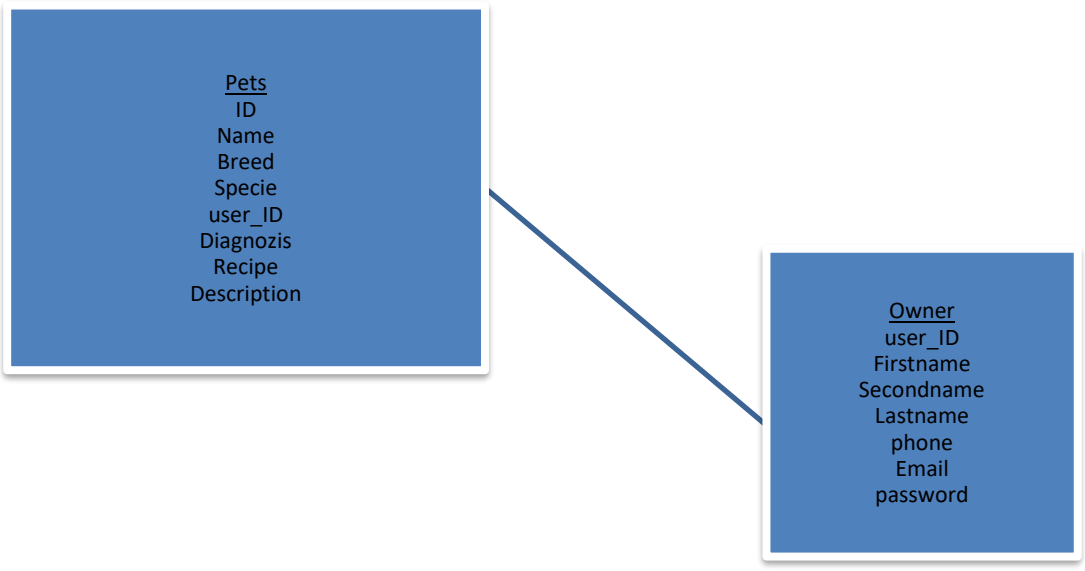


Рисунок 2.1 – Структура БД(1)

Рисунок 2.2 – Структура БД(2)

2.3 Функціональні можливості програмної системи

Функціонал програмної системи обліку тварин у ветклініці

Назва функціоналу	Опис функціоналу
Реєстрація користувача	Функція, яка дозволяє створювати облікові записи для кожного користувача.
Реєстрація тваринки	Функція, яка дозволяє створювати акаунти для кожної тваринки.
Пошук тваринки	Функція, яка дозволяє користувачу знайти потрібного улюбленця
Пошук лікаря	Функція, яка дозволяє віднайти за заданими параметрами потрібного лікаря.

Перегляд загублених тваринок	Функція, яка дозволяє переглянути загублених тварин.

Перегляд безпритульних тваринок	Функція, яка дозволяє переглянути безпритульних тваринок

Авторизація користувача	Для можливості ідентифікування користувачів та збереженням їх даних на сервері кожен користувач повинен пройти авторизації.

Таблиця 2.1

Це є головний функціонал, що вимагається від програмної системи для обліку домашніх тварин у ветклініці. Він дозволить передивлятися інформацію про діагнози та лікування улюбленця, яка може оновлюватися тільки користувачами що мають прямий доступ до баз даних з усією інформацією

Висновки

Було описано усі використані програмні засоби при розробці програмної системи. Так, було обрано мову програмування Java, інструментарій Java FX, СУБД MySQl та інструмент візуального компонування Scene Builder

Для створення динамічного ресурсу, який можливо оновлювати та додавати нову інформацію було використано додаткові бібліотеки та пакети, які сприяють полегшенню розробці.

Розробка програмної системи з потужним функціоналом потребує якісної підготовки та чіткого плану по виконанню всіх робіт, які стосуються створення прототипу застосунку.

РОЗДІЛ 3: ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОГРАМНОЇ СИСТЕМИ ОБЛІКУ ТВАРИН У ВЕТКЛІНІЦІ

3.1 Функціональні вимоги до системи

Для користування веб-сервісом користувач спочатку повинен пройти аутентифікацію. У користувачів, які не пройшли аутентифікацію, є можливість зареєструватися у системі, або користуватися можливостями, які не потребують аутентифікації.

Зареєстровані користувачі мають доступ до такої функціональності та таких сторінок:

-Реєстрація улюбленця;

-Перегляд сторінки улюбленця;

-Пошук серед улюбленців.

Перед аутентифікаією в користувачів також є такі можливості як:

-Перегляд загублених тварин;

-Перегляд безпритульних тварин для можливості їх отримання;

-Пошук лікарів за потрібною спеціалізацією для можливості записатися на консультацію/медогляд/та інше.

Різниця між зареєстрованими і незареєстрованими користувачами тільки в можливості створення облікових записів зі своїми улюбленцями.

3.2 Загальний опис архітектури

У розробці програмної системи була вибрана клієнт-серверна архітектура.

Архітектура клієнт-сервер (client-server architecture) — це концепція інформаційної мережі, в якій основна частина її ресурсів зосереджена на серверах, що обслуговують її клієнтів. Архітектура клієнт-сервер заснована на двох компонентах: клієнті і сервері.

Клієнт – комп’ютер користувача, який відправляє запит до сервера для надання інформації або виконання певних дій

Сервер – більш потужний комп’ютер або обладнання, призначене для вирішення певних завдань, надання клієнтам доступу до певних ресурсів, зберігання інформації і баз даних

Модель такої системи полягає в тому, що клієнт відправляє запит на сервер, де він обробляється, а готовий результат надсилається клієнту. Сервер може обслуговувати декілька клієнтів одночасно. Якщо одночасно отримано більше одного запиту, вони стають у чергу і виконуються сервером послідовно. Іноді запити можуть мати пріоритети. Запити з вищим пріоритетом повинні виконуватися раніше.

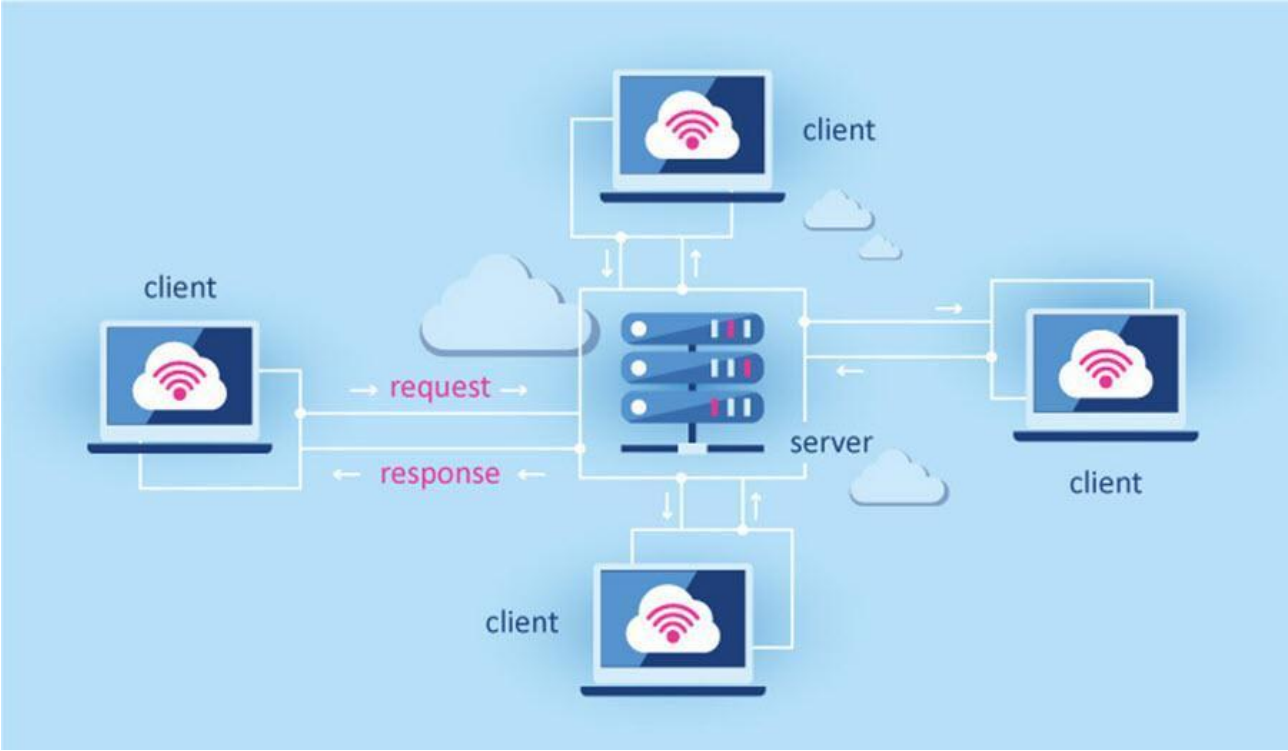


Рисунок 3.1 – Схема клієнт-серверної архітектури

Функції, які реалізуються на стороні клієнта:

- надання користувальницького інтерфейсу;

- формулювання запиту до сервера і його відправка;
- отримання результатів запиту і відправка додаткових команд (запитів на додавання, оновлення або видалення даних).

Функції, які реалізуються на сервері:

- зберігання, доступ, захист і резервне копіювання даних;
- обробка запиту клієнта;
- відправлення результату клієнтові.

Також у розробці програмної системи використовувався архітектурний шаблон MVC. MVC — це шаблон програмування, який дозволяє розділити логіку програми на три частини:

- **Модель –** контролер надає дані моделі, а вона вже виконує необхідні операції над даними і передає їх у View.
- **View –** виводить дані, отримані від моделі, користувачу.
- **Controller -** Обробляє дії користувача, перевіряє отримані дані та передає їх моделі.

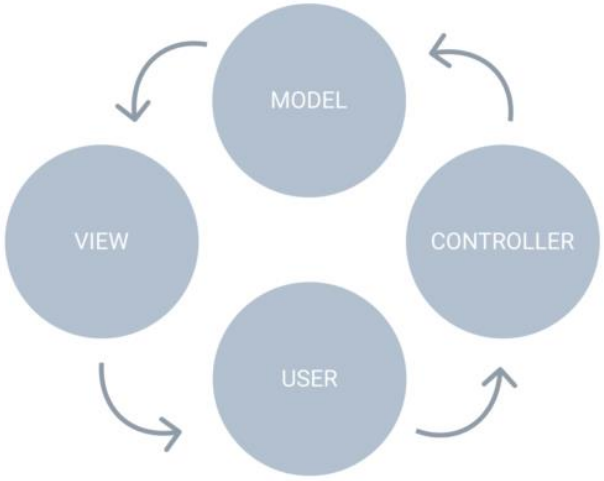


Рисунок 3.2 Схема взаємодії з шаблоном MVC

Переваги використання шаблону MVC:

- Висока згуртованість. Шаблон MVC дозволяє логічно групувати відповідні дії разом на контролері.
- Глобальна архітектура програми.
- Низький зв'язок між моделлю, виглядом і контролером.
- Легкість модифікації шляхом розподілу обов'язків.
— Спрощений механізм налагодження програми.
- Для однієї моделі може бути декілька переглядів.

Недоліки використання шаблону MVC:

- Необхідність використання великої кількості ресурсів.
- Механізм поділу програми на модулі є складним, оскільки кожен функціональний модуль необхідно розділити на три блоки, що ускладнює архітектуру функціональних модулів програми.
- Складний процес розширення функціональності.

3.3 Розробка графічного інтерфейсу

Для полегшення розробки графічного інтерфейсу було використано інструмент візуального компонування Scene Builder

Scene Builder генерує FXML, мову розмітки на основі XML, яка дозволяє користувачам визначати інтерфейс користувача програми окремо від логіки програми:

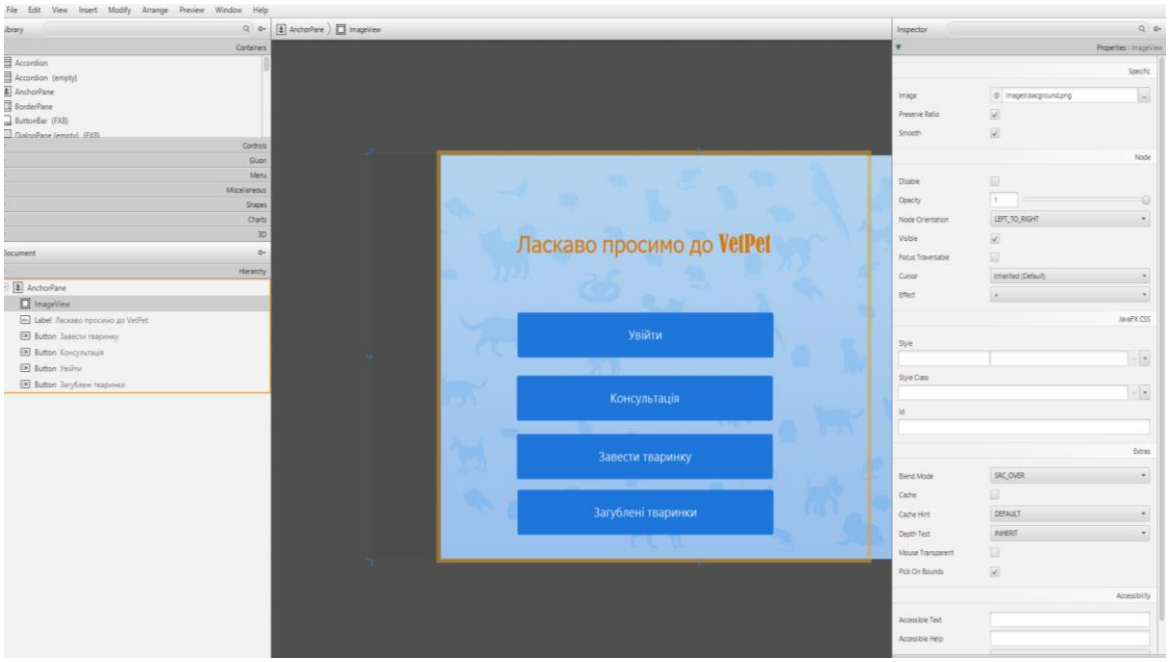


Рисунок 3.3 Розробка графічного інтерфейсу у Scene Builder

Scene Builder дозволяє відкрити FXML файл, створений в Intellij IDEA, та надає змогу редагувати його графічно, самостійно змінюючи його код:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
```

```
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.firstPageController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Label contentDisplay="CENTER" layoutX="125.0" layoutY="91.0"
prefHeight="79.0" prefWidth="466.0" style="-fx-background-color: null;"
text="Ласкаво просимо до VetPet" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="35.0" />
            </font>
        </Label>
        <Button fx:id="SignButton" layoutX="125.0" layoutY="415.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Завести тваринку" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
      <Button fx:id="SignButton1" layoutX="125.0" layoutY="328.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Консультація" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Button fx:id="SignButton2" layoutX="126.0" layoutY="234.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Увійти" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Button fx:id="SignButton21" layoutX="126.0" layoutY="498.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Загублені тваринки" textFill="WHITE">
         <font>
            <Font size="19.0" />
```

```
        </font>
      </Button>
    </children>
</AnchorPane>
```

Також Scene Builder дозволяє підв’язати контролери до редагуємих файлів:

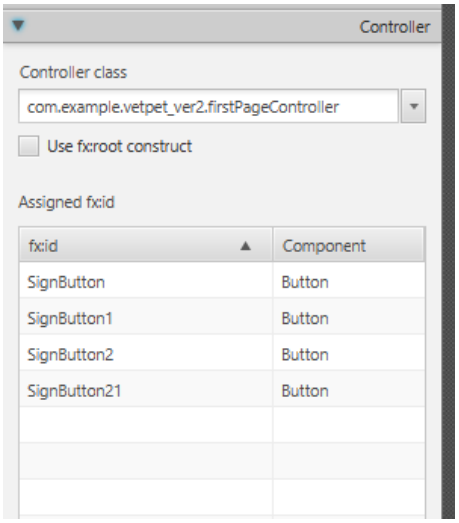


Рисунок 3.4 Підв’язка конролеру та компонентів графічного інтерфейсу

Графічний інтерфейс цієї програмної системи включає в себе:

- Головну сторінку;
- Сторінку авторизації;
- Сторінку реєстрації;
- Панель пошуку;
- Сторінки зі списками лікарів/безпритульних тварин/загублених тварин;
- Сторінку користувача/улюбленця;
- Вікно помилки

Інструментом для переходу між створеними вікнами являються кнопки – кожна кнопка розташована в контролерах вікон і, залежно від вікна, може

оперувати від одної до декількох діями: якщо при переході з головної сторінки на сторінку авторизації кнопка виконує тільки функцію переходу, то кнопка на сторінці авторизації відповідає одразу за функцію перевірки введених даних і тільки після цього вже за функцію переходу між вікнами.

За введення даних відповідають об’єкти TextField – з них дані зчитуються і вносяться в бази даних. За виведення відповідають два типи контейнерів в залежності від вікна, в якому виведення даних потрібне – на сторінках користувача і улюбленця за виведення даних відповідають Label, на сторінках же зі списками за виведення даних відповідають VBox – вони дають можливість заповнити дані стовпцями, тож таким чином у вікнах виводяться списки з усіма знайденими по запитам даними.

3.4 Підключення баз даних

Для роботи з базами даних в Java було використано бібліотеку java.sql – вона надає API для доступу та обробки даних, що зберігаються в джерелі даних (зазвичай у реляційній базі даних), за допомогою мови програмування JavaTM.

```
import java.sql.*;

public class DatabaseHandler extends Configs {

    Connection dbConnection;

    public Connection getDbConnection() throws ClassNotFoundException,
SQLException{
        String connection = "jdbc:mysql://"+ dbHost+":" + dbPort +"/"+bdName;

        Class.forName("com.mysql.cj.jdbc.Driver");

        dbConnection = DriverManager.getConnection(connection, dbUser, bdPass);
```

```
        return dbConnection;
    }
```

Створений клас DatabaseHandler тепер буде викликатись кожен раз коли програмі потрібно буде скористатись базою даних. База даних цієї програми має декілька таблиць:

- Лікарі;
- Користувачі;
- Улюбленці;
- Загублені улюбленці;
- Безпритульні тварини

З цих таблиць пов’язані між собою тільки таблиця користувачів і улюбленців – таблиця улюбленців має в собі ключі таблиці користувачів. Це дозволяє одразу побачити якому користувачу належить певна тваринка і при потребі змінити власника.

3.5 Інструкція користувача

При запуску програми нас переносить на головну сторінку:

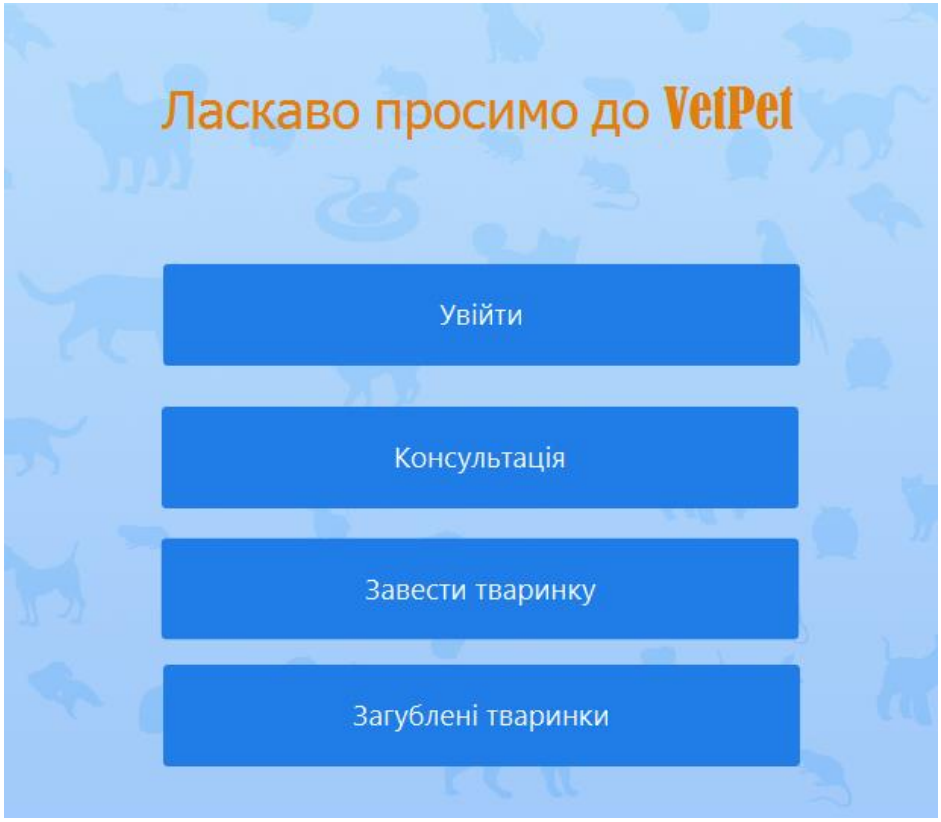


Рисунок 3.6 Вікно головної сторінки

Це вікно надає користувачу можливість авторизуватися, а також ще три можливості, для використання яких авторизація не потрібна. Цими трьома можливостями являються:

- Запис на консультацію до лікаря:

  Натиснувши на кнопку «Консультація» ми переходимо на вікно з панеллю для пошуку, що просить ввести спеціалізацію потрібного лікаря:

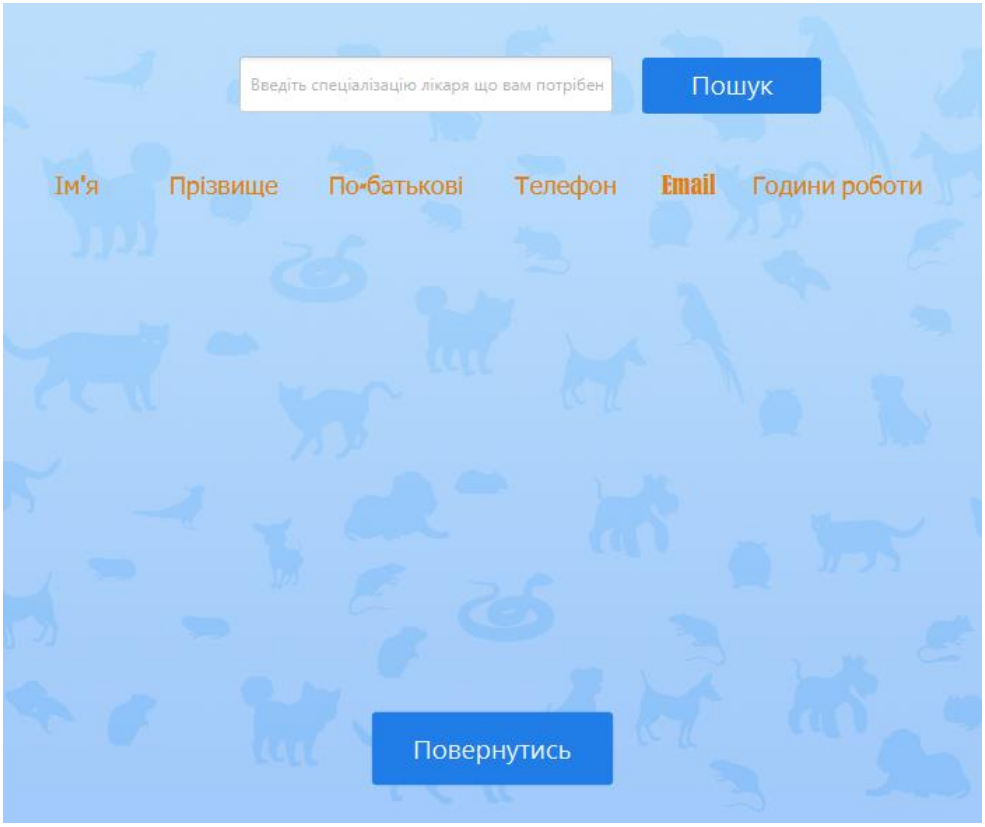


Рисунок 3.7 Вікно пошуку лікаря

Вводимо потрібну спеціалізацію і програма показує зареєстрованих лікарів з інформацією про їхні години роботи, а також показує їхні телефон і пошту для подальшої можливості зв’язатися з ними:

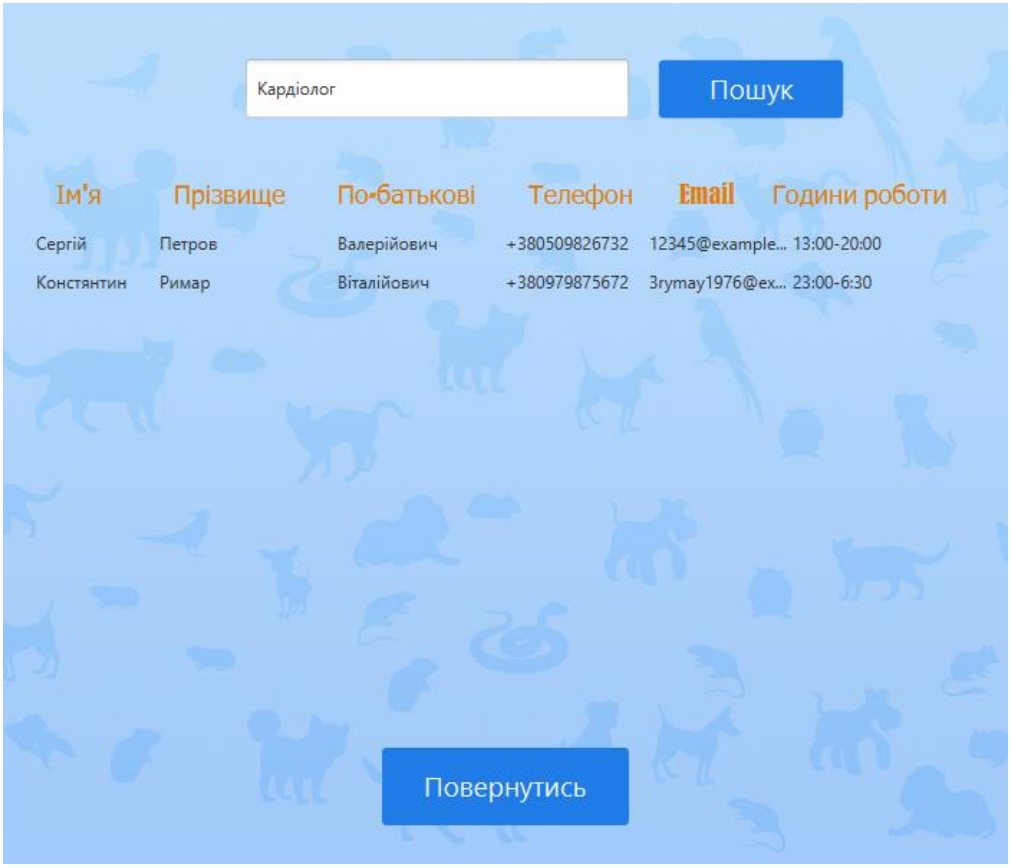


Рисунок 3.8 Результати пошуку лікаря

Для повернення на головну сторінку натискаємо кнопку «Повернутись»;

- Перегляд безпритульних тварин для подальшого «всиновлення»: Натиснувши на кнопку «Завести тваринку» ми потрапляємо на вікно з панеллю пошуку, що просить ввести вид бажаної для «всиновлення» тваринки:

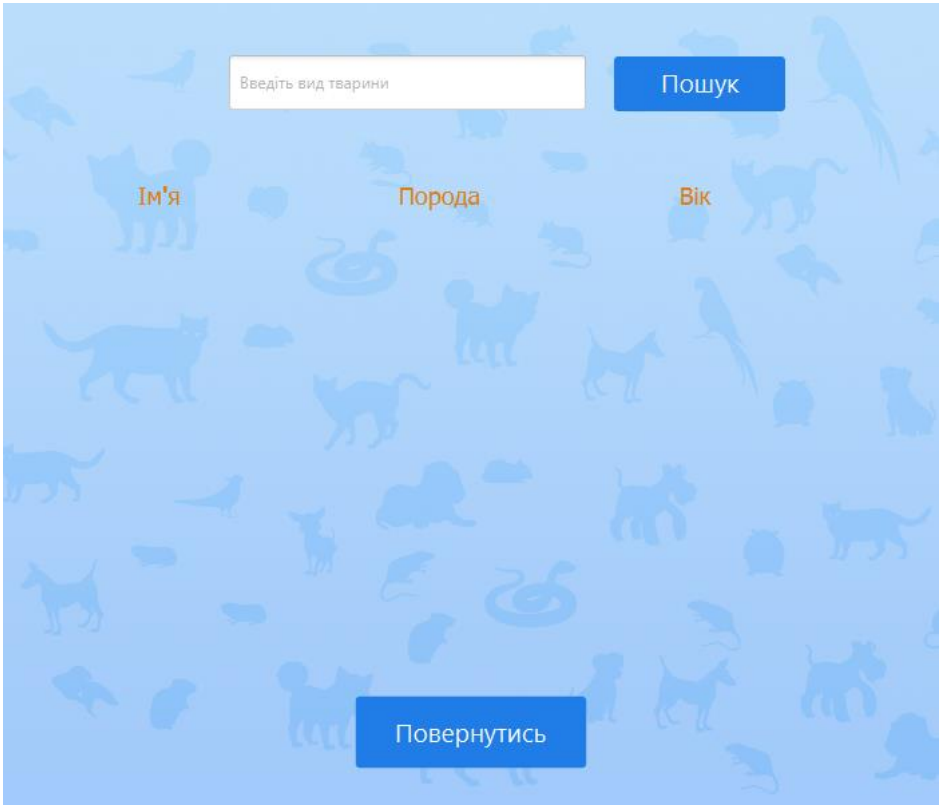


Рисунок 3.9 Панель пошуку безпритульних тварин по їхньому виду

Ввівши дані натискаємо на «пошук» і нам висвічуються дані наявних безпритульних тварин з інформацією про них:

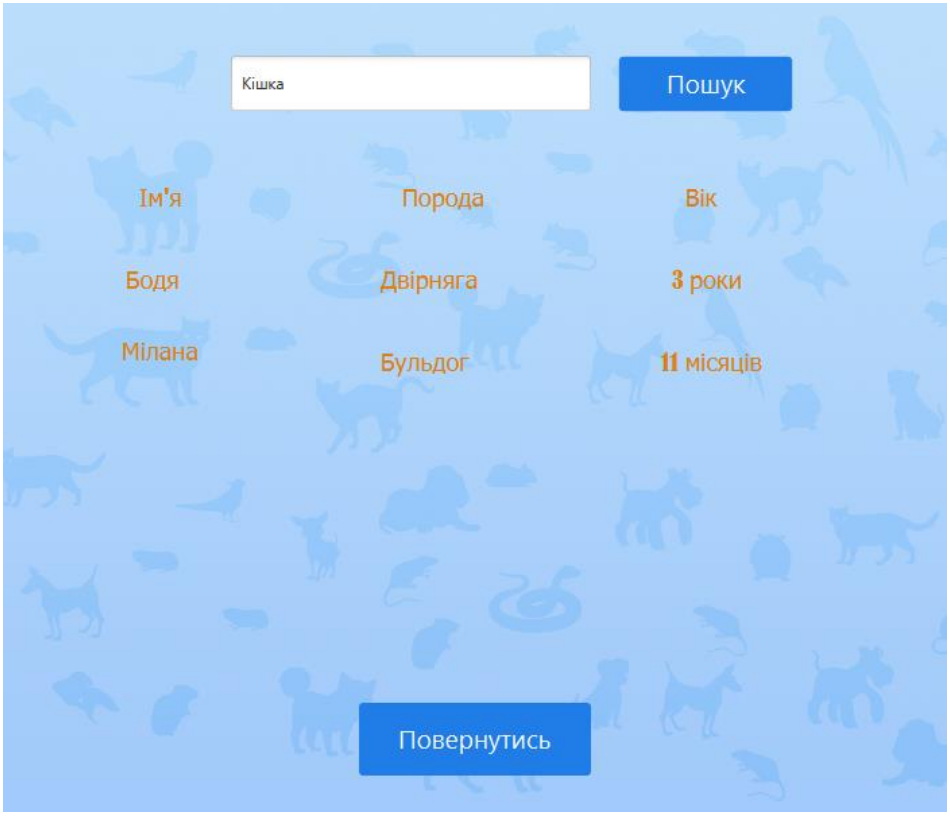


Рисунок 3.10 Результати пошуку серед безпритульних тварин

Натискаємо на кнопку «Повернутись» щоб перейти на головну сторінку;

- Перегляд загублених тваринок для можливості допомогти з пошуком:
  При натисканні на кнопку «Загублені тваринки» ми потрапляємо на вікно з інформацією про загублених тварин:

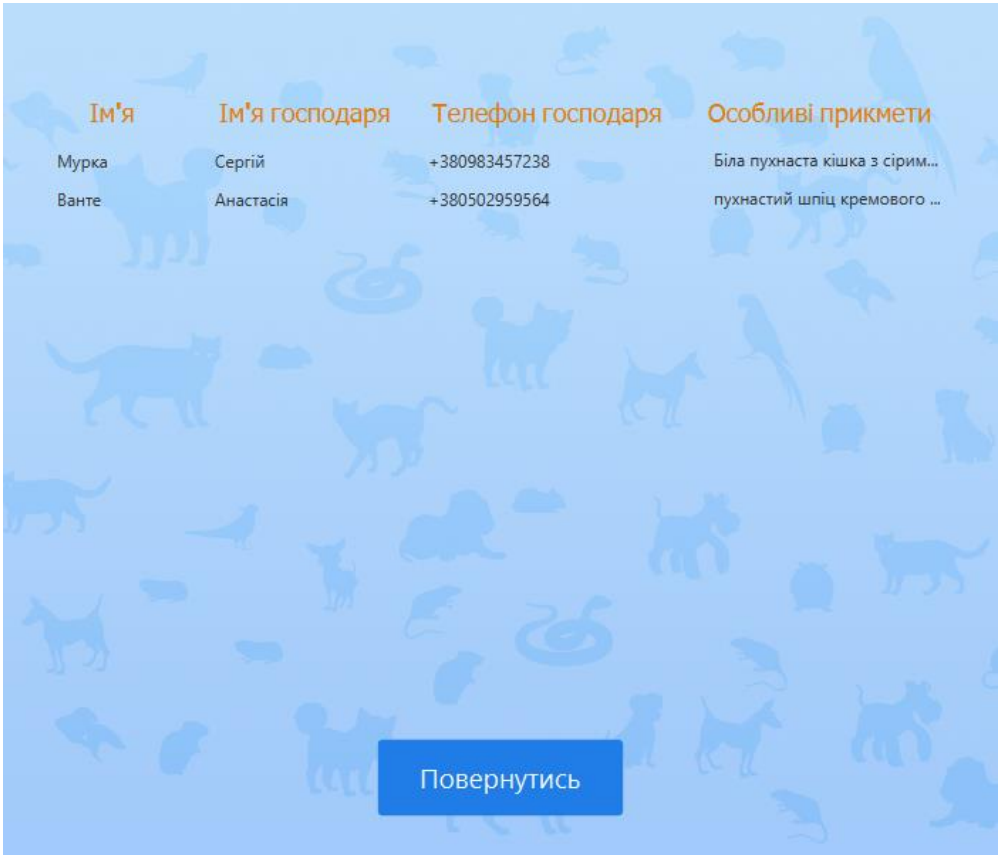


Рисунок 3.11 Вікно з інформацією про загублених тварин

В цьому вікні наявні така інформація як:

-Ім’я улюбленця;

-Ім’я господаря тваринки;

-Телефон господаря для можливості зв’язатись при знаходженні тваринки;

-Опис тваринки.

Для авторизації/реєстрації натискаємо на кнопку «Увійти» на головній сторінці – ця кнопка відповідає за перехід з головної сторінки на сторінку авторизації:

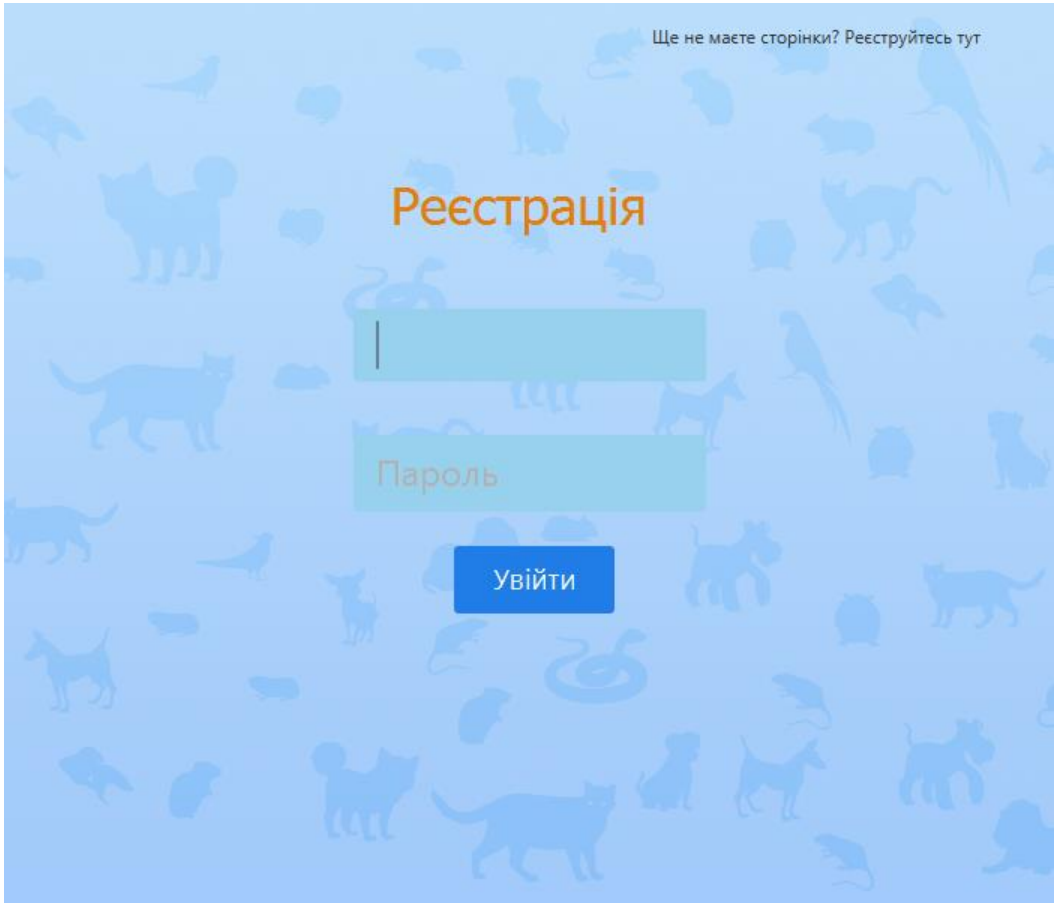


Рисунок 3.12 Сторінка авторизації

На сторінці авторизації наявні дві панелі для вводу даних (пошти і паролю), після вводу яких при натисканні кнопки «увійти» програма перевірить введені дані і, якщо все правильно введено, перейде на сторінку користувача:

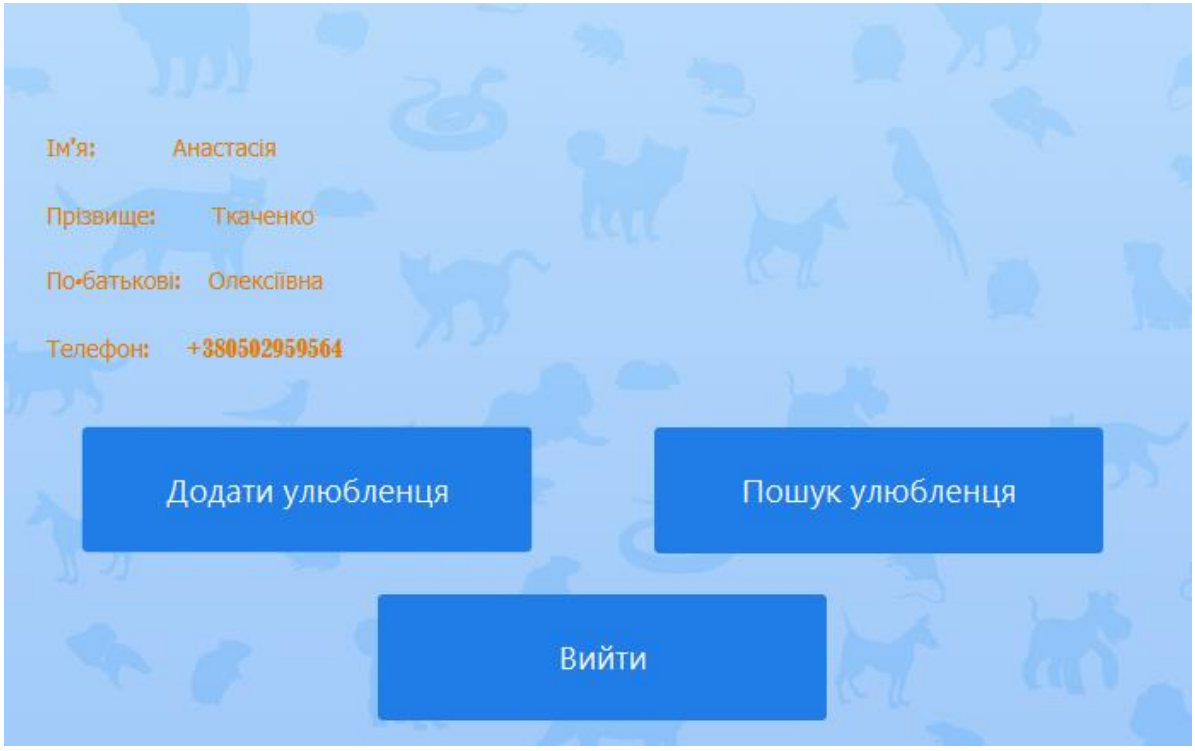


Рисунок 3.13 Сторінка користувача

На сторінці користувача містяться введені при реєстрації дані, а також наявні три кнопки:

- **Додати улюбленця:**

  Ця кнопка відповідає за перехід на вікно реєстрації домашнього улюбленця:

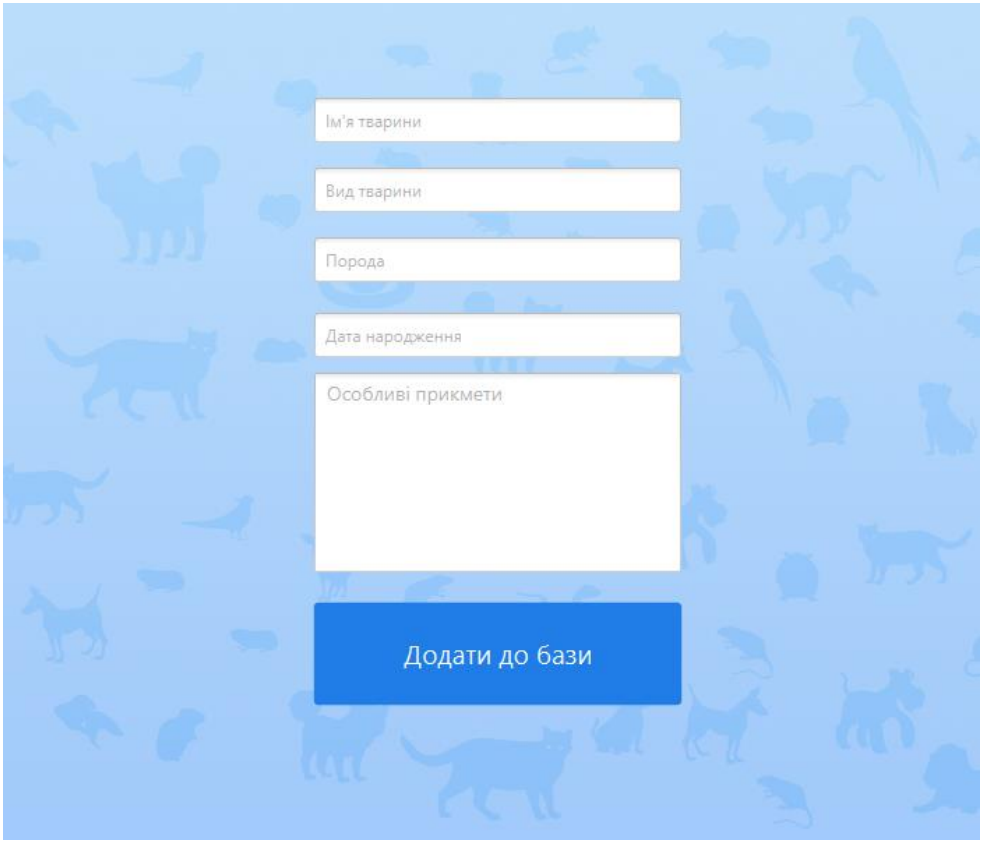


Рисунок 3.14 Сторінка реєстрації улюбленця

Вводимо потрібні дані і натискаємо на «Додати до бази» - при натисканні кнопки програма відкриває вікно сторінки улюбленця, яке містить введені при реєстрації дані та дві кнопки: кнопку «позначити як загубленого», що додає улюбленця у списки загублених тварин і «вийти», що повертає нас на сторінку користувача:

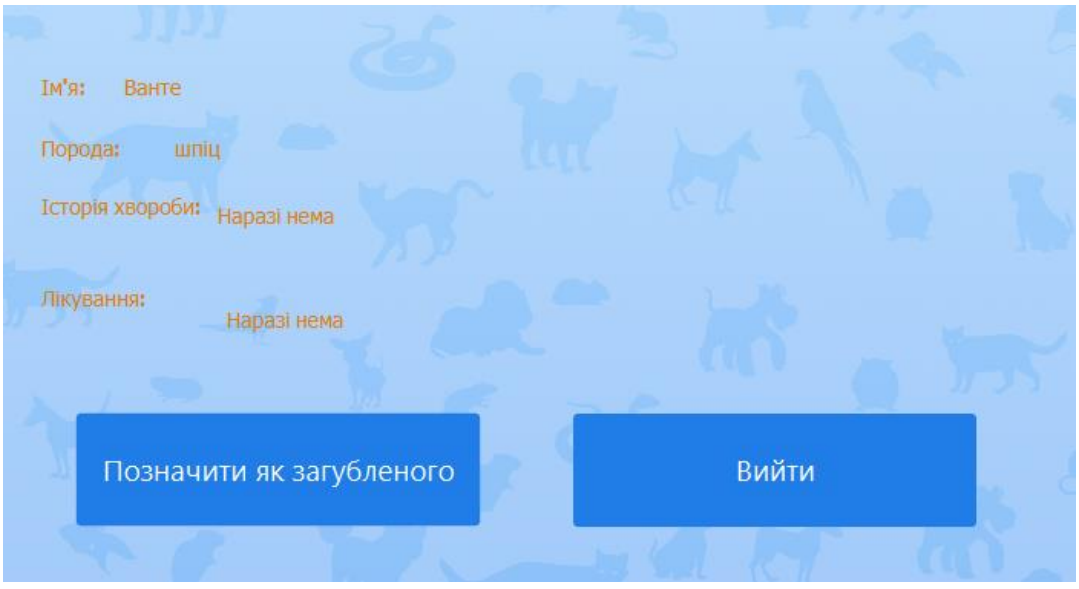


Рисунок 3.15 Сторінка улюбленця

- **Пошук улюбленця:**

  При натисканні на цю кнопку відкривається панель пошуку серед улюбленців користувача – при введені потрібного імені користувача переносить на сторінку улюбленця (рисунок 3.15)

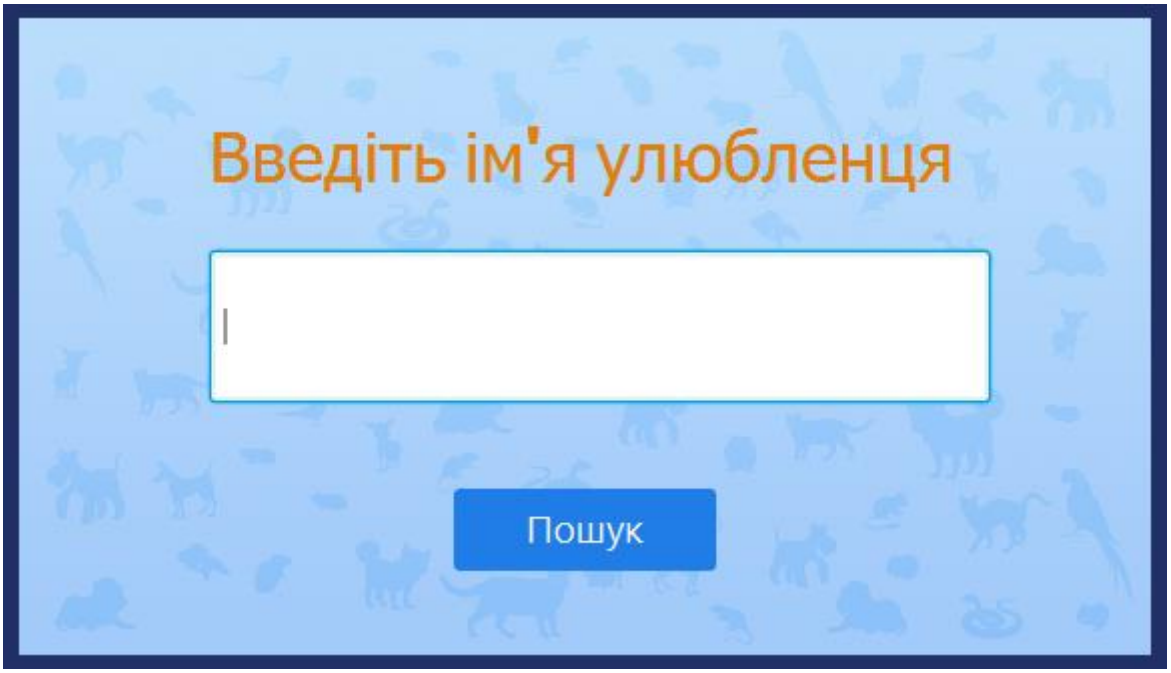


Рисунок 3.16 Панель пошуку

- **Вийти:**

  Ця кнопка повертає користувача на головну сторінку

Для незареєстрованих користувачів програма надає можливість зареєструватись – для цього потрібно натиснути на текст у правому верхньому куті сторінки авторизації (рисунок 3.12) і програма відкриє вікно реєстрації:

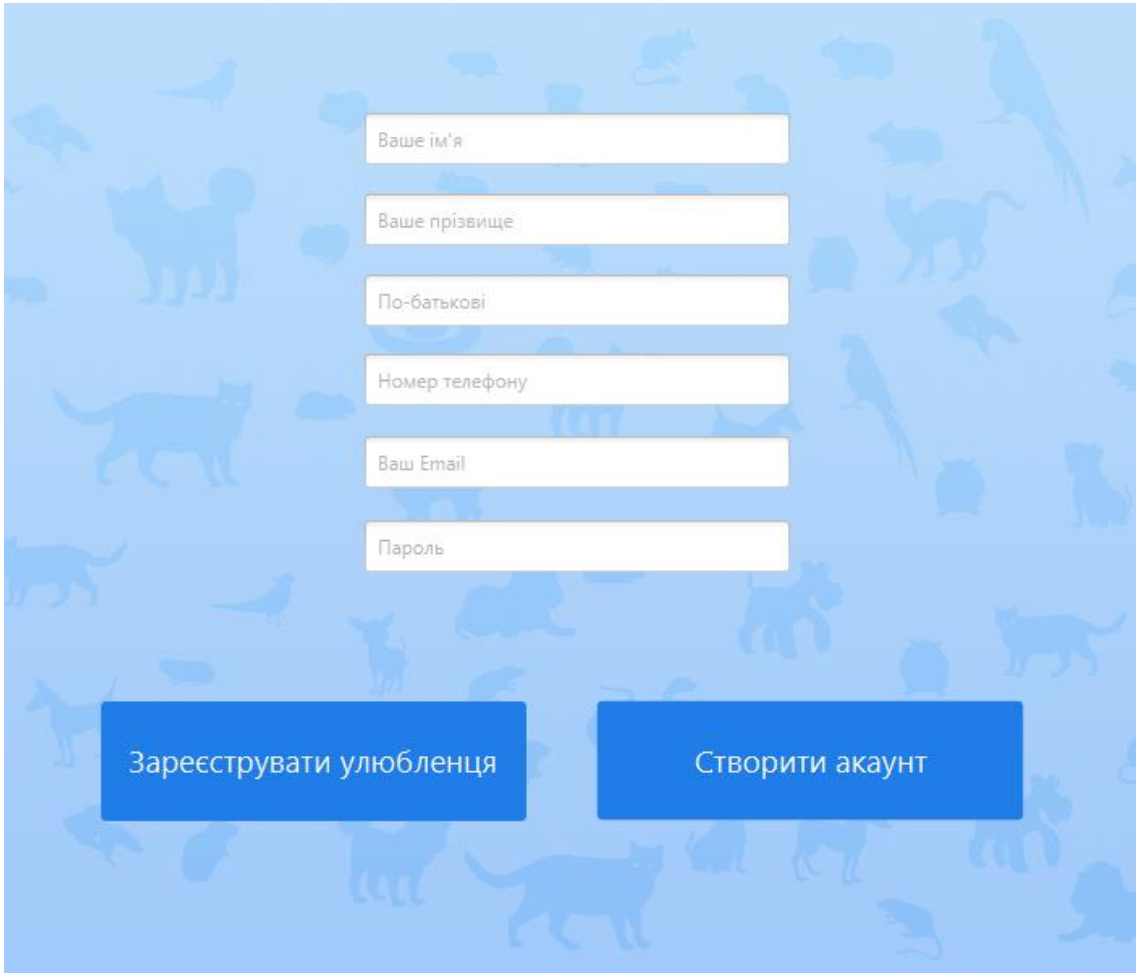


Рисунок 3.17 Сторінка реєстрації

Після вводу потрібних даних ми можемо перейти на сторінку користувача (рисунок 3.13), або одразу зареєструвати улюбленця (рисунок 3.14)

ВИСНОВКИ

В даній дипломній роботі було проаналізовано та розглянуто різноманітні способи та технології для створення програмної системи.

Досліджено які технології є більш зручними для створення програмної системи під різні потреби та функціонал. Також було наведено актуальні приклади найпопулярніших та розповсюджених інструментів для створення програмних систем.

Здійснено аналіз основних засобів, що входить до оптимізації програмної системи.

Були вивчені технологій побудови програмних систем.

Для проектування та розробки системи кваліфікаційної роботи була застосована мова програмування Java, інструментарій Java FX, СУБД MySQl та інструмент візуального компонування Scene Builder, а також використано пакети та бібліотеки, які було описано раніше.

Після проведення тестування швидкості роботи та оптимізації програмної системи було розроблено повноцінну документацію з користування програмними інтерфейсами. Всі компоненти програмної системи працюють коректно, швидко та безперебійно.

СПИСОК ЛІТЕРАТУРИ

1. Тимур Машнін «JavaFX 2.0. Розробка RIA-додатків», https://books.google.de/books?id=Dm2x2ELZnesC&printsec=copyright&hl=uk#v=onepage&q&f=false

2. “Learn Javafx Building User Experience” by Kishori Sharan, https://books.google.com.ua/books?id=Wb8ICAAAQBAJ&printsec=frontcover&redir_esc=y#v=onepage&q&f=false

3. “Learning php mysql & javascript 5th edition” by robin nixon (o'reilly 2018) pdfhttps://sd.blackball.lv/library/learning_php_mysql_and_javascript_(2018).pdf

4. «high performance mysql optimization backups and replication» by baron schwartzhttps://books.google.com.ua/books?id=JXFuCQAAQBAJ&printsec=frontcover&redir_esc=y#v=onepage&q&f=false

5. “Learning MySQL: Get a Handle on Your Data” 2nd Edition by Vinicius M. Grippa https://ebin.pub/learning-mysql-get-a-handle-on-your-data-2nbsped-9781492085928.html

6. “Effective java” by Joshua Bloch https://github.com/muhdkhokhar/test/blob/master/Joshua%20Bloch%20-%20Effective%20Java%20(3rd)%20-%202018.pdf

7. КОНСПЕКТ ЛЕКЦІЙ з дисципліни «ТЕХНОЛОГІЯ СТВОРЕННЯ ПРОГРАМНИХ ПРОДУКТІВ» ГНАТОВСЬКА Г.А. http://eprints.library.odeku.edu.ua/id/eprint/291/1/HnatovskayaHA_Texnologii_stvorennya_program_produktov_KL_2015.pdf

8. Основи програмування мовою Java М.Ф. Копитко, К.С. Іванків https://ami.lnu.edu.ua/wp-content/uploads/2017/05/Java.pdf

9. Mastering Javafx 10 - Build Advanced And Visually Stunning Java Applications Sergey Grinev

https://vdoc.pub/documents/mastering-javafx-10-build-advanced-and-visually-stunning-java-applications-true-pdf-7ddag7fmgmh0

10. JavaFX 9 by Example by Carl Dea, Gerrit Grunwald, José Pereda, Ph.D Sean Phillips, Mark Heckler https://link.springer.com/content/pdf/bfm%3A978-1-4842-1961-4%2F1.pdf

11. MySQL 8 Query Performance Tuning: A Systematic Method for Improving Execution Speeds by Jesper Wisborg Krogh https://pdfcoffee.com/mysql-8-query-performance-tuningpdf-pdf-free.html

12. The Self-taught Programmer: The Definitive Guide to Programming Professionally by Cory Althoff

https://www.knowledgeisle.com/wp-content/uploads/2019/12/Cory-Althoff-The-Self-Taught-Programmer_-The-Definitive-Guide-to-Programming-Professionally-Self-Taught-Media-2016.pdf

13.Кэти Сьерра, Берт Бейтс, Изучаем Java. Мировой компьютерный бестселлер https://library-it.com/programming/java/izuchaem-java-mirovoj-kompyuternyj-bestseller-2012/

ДОДАТКИ

ДОДАТОК А

Код оформлення головної сторінки:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.firstPageController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
```

```
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Label contentDisplay="CENTER" layoutX="125.0" layoutY="91.0"
prefHeight="79.0" prefWidth="466.0" style="-fx-background-color: null;"
text="Ласкаво просимо до VetPet" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="35.0" />
            </font>
        </Label>
        <Button fx:id="SignButton" layoutX="125.0" layoutY="415.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Завести тваринку" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
      <Button fx:id="SignButton1" layoutX="125.0" layoutY="328.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Консультація" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Button fx:id="SignButton2" layoutX="126.0" layoutY="234.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Увійти" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Button fx:id="SignButton21" layoutX="126.0" layoutY="498.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Загублені тваринки" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
    </children>
</AnchorPane>
```

Контролер головної сторінки:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class firstPageController {
```

```
    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button SignButton;

    @FXML
    private Button SignButton1;

    @FXML
    private Button SignButton2;

    @FXML
    private Button SignButton21;

    @FXML
    void initialize() {
        SignButton2.setOnAction(event ->{
            SignButton2.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("hello-view.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

        SignButton21.setOnAction(event ->{
            SignButton21.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("LostPets.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });
        SignButton1.setOnAction(event ->{
            SignButton1.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("SearchDoctor.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
```

```
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });
        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("ChoosePet.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

    }

}
```

Код оформлення сторінки пошуку лікарів:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.ChooseDoctorController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-148.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="264.0" layoutY="519.0"
```

```
mnemonicParsing="false" prefHeight="53.0" prefWidth="172.0" style="-fx-
background-color: #207ce6;" text="Повернутись" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
          </Button>
      <TextField fx:id="searchText" layoutX="169.0" layoutY="39.0"
prefHeight="41.0" prefWidth="267.0" promptText="Введіть спеціалізацію лікаря що
вам потрібен" />
      <Button fx:id="Search" layoutX="457.0" layoutY="40.0"
mnemonicParsing="false" prefHeight="30.0" prefWidth="128.0" style="-fx-
background-color: #207ce6;" text="Пошук" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Label contentDisplay="CENTER" layoutX="37.0" layoutY="106.0"
prefHeight="53.0" prefWidth="51.0" style="-fx-background-color: null;"
text="Ім'я" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label contentDisplay="CENTER" layoutX="119.0" layoutY="106.0"
prefHeight="53.0" prefWidth="108.0" style="-fx-background-color: null;"
text="Прізвище" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label contentDisplay="CENTER" layoutX="233.0" layoutY="106.0"
prefHeight="53.0" prefWidth="108.0" style="-fx-background-color: null;"
text="По-батькові" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label contentDisplay="CENTER" layoutX="366.0" layoutY="106.0"
prefHeight="53.0" prefWidth="108.0" style="-fx-background-color: null;"
text="Телефон" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label contentDisplay="CENTER" layoutX="471.0" layoutY="106.0"
prefHeight="53.0" prefWidth="108.0" style="-fx-background-color: null;"
text="Email" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <VBox fx:id="nameBox" layoutX="23.0" layoutY="159.0" prefHeight="349.0"
prefWidth="100.0" />
      <VBox fx:id="lastnameBox" layoutX="109.0" layoutY="159.0"
prefHeight="349.0" prefWidth="100.0" />
      <VBox fx:id="secondnameBox" layoutX="233.0" layoutY="159.0"
prefHeight="349.0" prefWidth="108.0" />
      <VBox fx:id="phoneBox" layoutX="350.0" layoutY="159.0" prefHeight="349.0"
prefWidth="100.0" />
      <VBox fx:id="emailBox" layoutX="450.0" layoutY="159.0" prefHeight="349.0"
prefWidth="100.0" />
      <Label fx:id="T" contentDisplay="CENTER" layoutX="536.0" layoutY="106.0"
prefHeight="53.0" prefWidth="128.0" style="-fx-background-color: null;"
```

```
text="Години роботи" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <VBox fx:id="TimeBox" layoutX="550.0" layoutY="159.0" prefHeight="349.0"
prefWidth="100.0" />
    </children>
</AnchorPane>
```

Контролер сторінки пошуку лікарів:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class ChooseDoctorController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Label labl2;

    @FXML
    private Label labl3;

    @FXML
    private Label labl4;

    @FXML
    private Label labl5;

    @FXML
    private Label labl1;

    @FXML
    private Label lbl1;

    @FXML
    private Label lbl2;
```

```
    @FXML
    private Label lbl3;

    @FXML
    private Label lbl4;

    @FXML
    private Label lbl5;


    @FXML
    private Button Search;

    @FXML
    private Button SignButton;

    @FXML
    private TextField searchText;

    @FXML
    private VBox emailBox;

    @FXML
    private VBox lastnameBox;

    @FXML
    private VBox nameBox;

    @FXML
    private VBox phoneBox;

    @FXML
    private VBox secondnameBox;

    @FXML
    private VBox TimeBox;




    @FXML
    void initialize() {
        Search.setOnAction(event ->{
            String loginText = searchText.getText().trim();

            if (loginText.equals("")) {
                System.out.println("Error");
                try {
                    FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("Error.fxml"));
                    Parent root2 = (Parent) fxmlLoader.load();
                    Stage stage1 = new Stage();
                    stage1.initModality(Modality.APPLICATION_MODAL);
                    stage1.initStyle(StageStyle.UNDECORATED);
                    stage1.setTitle("ABC");
                    stage1.setScene(new Scene(root2));
                    stage1.show();

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
```

```
            else {
                try {
                    getdoc(loginText);
                } catch (SQLException e) {
                    e.printStackTrace();
                } catch (ClassNotFoundException e) {
                    e.printStackTrace();
                }
            }


        });

        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
            FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("firstPage.fxml"));
            Parent root2 = null;
            try {
                root2 = (Parent) fxmlLoader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }
            Stage stage1 = new Stage();
            stage1.initModality(Modality.APPLICATION_MODAL);
            stage1.initStyle(StageStyle.UNDECORATED);
            stage1.setTitle("ABC");
            stage1.setScene(new Scene(root2));
            stage1.show();


        });

    }


    public void getdoc(String spec) throws SQLException, ClassNotFoundException
{
        DatabaseHandler handler = new DatabaseHandler();

        Doctor doctor = new Doctor();
        doctor.setSpecialization(spec);
        ResultSet resultSet = handler.getDoctor(doctor);

        nameBox.setSpacing(10);
        lastnameBox.setSpacing(10);
        secondnameBox.setSpacing(10);
        phoneBox.setSpacing(10);
        emailBox.setSpacing(10);
        TimeBox.setSpacing(10);

        Label label = new Label();
        label.setTextFill(Color.ORANGE);
        label.setFont(Font.font("Bernard MT Condensed",18));

        Label label1 = new Label();
        label1.setTextFill(Color.ORANGE);
        label1.setFont(Font.font("Bernard MT Condensed",18));

        Label label2 = new Label();
        label2.setTextFill(Color.ORANGE);
        label2.setFont(Font.font("Bernard MT Condensed",18));
```

```
        Label label3 = new Label();
        label3.setTextFill(Color.ORANGE);
        label3.setFont(Font.font("Bernard MT Condensed",18));


        while (resultSet.next()) {
            label.setText(resultSet.getString("DOCTOR_FIRSTNAME"));
            nameBox.getChildren().add(new
Label(resultSet.getString("DOCTOR_FIRSTNAME")));
            lastnameBox.getChildren().add(new
Label(resultSet.getString("DOCTOR_LASTNAME")));
            secondnameBox.getChildren().add(new
Label(resultSet.getString("DOCTOR_SECONDNAME")));
            phoneBox.getChildren().add(new
Label(resultSet.getString("DOCTOR_PHONE")));
            emailBox.getChildren().add(new
Label(resultSet.getString("DOCTOR_EMAIL")));
            TimeBox.getChildren().add(new
Label(resultSet.getString("JOBTIME")));
            /*label1.setText(resultSet.getString("DOCTOR_LASTNAME"));
            lastnameBox.getChildren().add(label1);
            label2.setText(resultSet.getString("DOCTOR_SECONDNAME"));
            secondnameBox.getChildren().add(label2);
            label3.setText(resultSet.getString("DOCTOR_PHONE"));
            phoneBox.getChildren().add(label3);*/

        }


    }
    private void SearchPet(String phoneText) throws SQLException,
ClassNotFoundException {

        if (phoneText.equals("2")) {
            labl1.setText("2");
            labl2.setText("2");
            labl3.setText("2");
            labl4.setText("2");
            labl5.setText("2");
            lbl1.setVisible(true);
            lbl2.setVisible(true);
            lbl3.setVisible(true);
            lbl4.setVisible(true);
            lbl5.setVisible(true);

        }
        else {
            labl1.setVisible(true);
            labl2.setVisible(true);
            labl3.setVisible(true);
            labl4.setVisible(true);
            labl5.setVisible(true);
        }

        DatabaseHandler handler = new DatabaseHandler();
        Doctor doctor = new Doctor();
        doctor.setSpecialization(phoneText);

        ResultSet resultSet = handler.getDoctor(doctor);



        int counter = 0;
```

```
        while (resultSet.next()) {
            counter++;

        }

        if (counter >= 1 ) {


        }

    }

}
```

Код оформлення сторінки з безпритульними тваринками:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.PetChooseController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="264.0" layoutY="519.0"
mnemonicParsing="false" prefHeight="53.0" prefWidth="172.0" style="-fx-
background-color: #207ce6;" text="Повернутись" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
        <TextField fx:id="searchText" layoutX="169.0" layoutY="39.0"
prefHeight="41.0" prefWidth="267.0" promptText="Введіть вид тварини" />
        <Button fx:id="Search" layoutX="457.0" layoutY="40.0"
mnemonicParsing="false" prefHeight="30.0" prefWidth="128.0" style="-fx-
background-color: #207ce6;" text="Пошук" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
        <Label contentDisplay="CENTER" layoutX="101.0" layoutY="117.0"
prefHeight="53.0" prefWidth="61.0" style="-fx-background-color: null;"
text="Ім'я" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="18.0" />
            </font>
        </Label>
```

```
        <Label contentDisplay="CENTER" layoutX="296.0" layoutY="117.0"
prefHeight="53.0" prefWidth="108.0" style="-fx-background-color: null;"
text="Порода" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="18.0" />
            </font>
        </Label>
        <Label contentDisplay="CENTER" layoutX="506.0" layoutY="117.0"
prefHeight="53.0" prefWidth="108.0" style="-fx-background-color: null;"
text="Вік" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="18.0" />
            </font>
        </Label>
      <Label fx:id="labl1" contentDisplay="CENTER" layoutX="91.0"
layoutY="178.0" prefHeight="53.0" prefWidth="61.0" style="-fx-background-color:
null; visibility: false;" text="Бодя" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label fx:id="labl2" contentDisplay="CENTER" layoutX="280.0"
layoutY="178.0" prefHeight="53.0" prefWidth="108.0" style="-fx-background-color:
null; visibility: false;" text="Двірняга" textAlignment="CENTER"
textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label fx:id="labl3" contentDisplay="CENTER" layoutX="496.0"
layoutY="178.0" prefHeight="53.0" prefWidth="108.0" style="-fx-background-color:
null; visibility: false;" text="3 роки" textAlignment="CENTER"
textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label fx:id="lbl1" contentDisplay="CENTER" layoutX="88.0" layoutY="231.0"
prefHeight="53.0" prefWidth="61.0" style="-fx-background-color: null;
visibility: false;" text="Мілана" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label fx:id="lbl2" contentDisplay="CENTER" layoutX="280.0"
layoutY="239.0" prefHeight="53.0" prefWidth="108.0" style="-fx-background-color:
null; visibility: false;" text="Бульдог" textAlignment="CENTER"
textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Label fx:id="lbl3" contentDisplay="CENTER" layoutX="488.0"
layoutY="239.0" prefHeight="53.0" prefWidth="108.0" style="-fx-background-color:
null; visibility: false;" text="11 місяців" textAlignment="CENTER"
textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
    </children>
</AnchorPane>
```

Контролер сторінки з безпритульними тваринами:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class PetChooseController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;


    @FXML
    private Button Search;

    @FXML
    private Button SignButton;

    @FXML
    private TextField searchText;

    @FXML
    private VBox NameBox;

    @FXML
    private VBox BreedBox;

    @FXML
    private VBox AgeBox;



    @FXML
    void initialize() {
        Search.setOnAction(event ->{
            String loginText = searchText.getText().trim();

            if (loginText.equals("")) {
                System.out.println("Error");
                try {
```

```
                    FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("Error.fxml"));
                    Parent root2 = (Parent) fxmlLoader.load();
                    Stage stage1 = new Stage();
                    stage1.initModality(Modality.APPLICATION_MODAL);
                    stage1.initStyle(StageStyle.UNDECORATED);
                    stage1.setTitle("ABC");
                    stage1.setScene(new Scene(root2));
                    stage1.show();

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
            else {
                try {
                    SearchPet(loginText);
                } catch (SQLException e) {
                    e.printStackTrace();
                } catch (ClassNotFoundException e) {
                    e.printStackTrace();
                }
            }


        });

        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
            FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("firstPage.fxml"));
            Parent root2 = null;
            try {
                root2 = (Parent) fxmlLoader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }
            Stage stage1 = new Stage();
            stage1.initModality(Modality.APPLICATION_MODAL);
            stage1.initStyle(StageStyle.UNDECORATED);
            stage1.setTitle("ABC");
            stage1.setScene(new Scene(root2));
            stage1.show();


        });

    }


    private void SearchPet(String specie) throws SQLException,
ClassNotFoundException {

        DatabaseHandler handler = new DatabaseHandler();

        HomelessPet homelessPet = new HomelessPet();
        homelessPet.setSpecies(specie);
        ResultSet resultSet = handler.getHomelessPet(homelessPet);

        NameBox.setSpacing(10);
        BreedBox.setSpacing(10);
        AgeBox.setSpacing(10);
```

```
        Label label = new Label();
        label.setTextFill(Color.ORANGE);
        label.setFont(Font.font("Bernard MT Condensed",18));



        while (resultSet.next()) {
            NameBox.getChildren().add(new Label(resultSet.getString("NAME")));
            BreedBox.getChildren().add(new Label(resultSet.getString("BREED")));
            AgeBox.getChildren().add(new Label(resultSet.getString("AGE")));

        }


    }
}
```

Код оформлення сторінки з загубленими тваринками:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Accordion?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.LostPetController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="264.0" layoutY="519.0"
mnemonicParsing="false" prefHeight="53.0" prefWidth="172.0" style="-fx-
background-color: #207ce6;" text="Повернутись" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
        <Label contentDisplay="CENTER" layoutX="61.0" layoutY="50.0"
prefHeight="53.0" prefWidth="61.0" style="-fx-background-color: null;"
text="Ім'я" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="18.0" />
            </font>
        </Label>
        <Label contentDisplay="CENTER" layoutX="302.0" layoutY="50.0"
prefHeight="53.0" prefWidth="165.0" style="-fx-background-color: null;"
text="Телефон господаря" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="18.0" />
```

```
            </font>
        </Label>
        <Label contentDisplay="CENTER" layoutX="496.0" layoutY="50.0"
prefHeight="53.0" prefWidth="172.0" style="-fx-background-color: null;"
text="Особливі прикмети" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="18.0" />
            </font>
        </Label>
      <Label contentDisplay="CENTER" layoutX="152.0" layoutY="50.0"
prefHeight="53.0" prefWidth="165.0" style="-fx-background-color: null;"
text="Ім'я господаря" textAlignment="CENTER" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="18.0" />
         </font>
      </Label>
      <Accordion layoutX="46.0" layoutY="103.0" />
      <VBox fx:id="nameBox" layoutX="38.0" layoutY="103.0" prefHeight="397.0"
prefWidth="100.0" />
      <VBox fx:id="OwnerNameBox" layoutX="149.0" layoutY="103.0"
prefHeight="397.0" prefWidth="133.0" />
      <VBox fx:id="PhoneBox" layoutX="299.0" layoutY="103.0" prefHeight="397.0"
prefWidth="165.0" />
      <VBox fx:id="DescrBox" layoutX="500.0" layoutY="102.0" prefHeight="397.0"
prefWidth="165.0" />
    </children>
</AnchorPane>
```

Контролер з загубленими тваринками:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class LostPetController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button SignButton;
```

```
    @FXML
    private VBox DescrBox;

    @FXML
    private VBox OwnerNameBox;

    @FXML
    private VBox PhoneBox;

    @FXML
    private VBox nameBox;


    @FXML
    void initialize() throws SQLException, ClassNotFoundException {

        getpet();
        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
            FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("firstPage.fxml"));
            Parent root2 = null;
            try {
                root2 = (Parent) fxmlLoader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }
            Stage stage1 = new Stage();
            stage1.initModality(Modality.APPLICATION_MODAL);
            stage1.initStyle(StageStyle.UNDECORATED);
            stage1.setTitle("ABC");
            stage1.setScene(new Scene(root2));
            stage1.show();


        });



    }
    public void getpet() throws SQLException, ClassNotFoundException {
        DatabaseHandler handler = new DatabaseHandler();

        LostPet lostPet = new LostPet();
        ResultSet resultSet = handler.getLostPet(lostPet);

        nameBox.setSpacing(10);
        OwnerNameBox.setSpacing(10);
        PhoneBox.setSpacing(10);
        DescrBox.setSpacing(10);


        while (resultSet.next()) {
            nameBox.getChildren().add(new
Label(resultSet.getString("pets_name")));
            OwnerNameBox.getChildren().add(new
Label(resultSet.getString("firstname")));
            PhoneBox.getChildren().add(new
Label(resultSet.getString("phone_number")));
            DescrBox.getChildren().add(new
Label(resultSet.getString("description")));
```

```
        }


    }


}
```

Код сторінки авторизації:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.firstPageController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Label contentDisplay="CENTER" layoutX="125.0" layoutY="91.0"
prefHeight="79.0" prefWidth="466.0" style="-fx-background-color: null;"
text="Ласкаво просимо до VetPet" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="35.0" />
            </font>
        </Label>
        <Button fx:id="SignButton" layoutX="125.0" layoutY="415.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Завести тваринку" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
      <Button fx:id="SignButton1" layoutX="125.0" layoutY="328.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Консультація" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Button fx:id="SignButton2" layoutX="126.0" layoutY="234.0"
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Увійти" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Button fx:id="SignButton21" layoutX="126.0" layoutY="498.0"
```

```
mnemonicParsing="false" prefHeight="66.0" prefWidth="420.0" style="-fx-
background-color: #207ce6;" text="Загублені тваринки" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
   </children>
</AnchorPane>
```

Контролер сторінки авторизації:

```
private void loginUser(String loginText, String passText) throws SQLException,
ClassNotFoundException {

    DatabaseHandler handler = new DatabaseHandler();
    User user = new User();
    user.setUsername(loginText);
    user.setPassword(passText);
    ResultSet resultSet = handler.getUser(user);


    int counter = 0;
    while (resultSet.next()){
        counter++;
        OwnerController.name = resultSet.getString("firstname");
        CheckUser.lastname = resultSet.getString("lastname");
        CheckUser.secname = resultSet.getString("po_batkovi");
        CheckUser.phonenumber = resultSet.getString("phone_number");
        CheckUser.id = resultSet.getInt("id_users");

    }
    if (counter>=1){
        try {

            FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("OwnerPage.fxml"));
            //OwnerController.name = CheckUser.firstname;
            OwnerController.lstname = CheckUser.lastname;
            OwnerController.secndname = CheckUser.secname;
            OwnerController.phne = CheckUser.phonenumber;
            OwnerController.id=CheckUser.id;

            Parent root1 = (Parent) fxmlLoader.load();
            Stage stage = new Stage();
            stage.initModality(Modality.APPLICATION_MODAL);
            stage.initStyle(StageStyle.UNDECORATED);
            stage.setTitle("ABC");

            stage.setScene(new Scene(root1));
            stage.show();

        } catch (IOException e) {
            e.printStackTrace();

        }
    }
```

Код сторінки реєстрації:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.PasswordField?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.OwnerRegistController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="367.0" layoutY="432.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Створити акаунт" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
      <TextField fx:id="NameField" layoutX="223.0" layoutY="68.0"
prefHeight="32.0" prefWidth="263.0" promptText="Ваше ім'я" />
      <TextField fx:id="LastNameField" layoutX="223.0" layoutY="118.0"
prefHeight="32.0" prefWidth="263.0" promptText="Ваше прізвище" />
      <TextField fx:id="SecondNameField" layoutX="223.0" layoutY="168.0"
prefHeight="32.0" prefWidth="263.0" promptText="По-батькові" />
      <TextField fx:id="EmailField" layoutX="223.0" layoutY="268.0"
prefHeight="32.0" prefWidth="263.0" promptText="Ваш Email" />
      <TextField fx:id="PhoneField" layoutX="223.0" layoutY="217.0"
prefHeight="32.0" prefWidth="263.0" promptText="Номер телефону" />
      <Button fx:id="SignButton1" layoutX="60.0" layoutY="432.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Зареєструвати улюбленця" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <PasswordField fx:id="PasswordField" layoutX="223.0" layoutY="320.0"
prefHeight="32.0" prefWidth="263.0" promptText="Пароль" />
    </children>
</AnchorPane>
```

Контролер сторінки реєстрації:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.sql.SQLException;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
```

```
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class OwnerRegistController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private TextField EmailField;

    @FXML
    private TextField LastNameField;

    @FXML
    private TextField NameField;

    @FXML
    private javafx.scene.control.PasswordField PasswordField;

    @FXML
    private TextField PhoneField;

    @FXML
    private TextField SecondNameField;

    @FXML
    private URL location;

    @FXML
    private Button SignButton;

    @FXML
    private Button SignButton1;

    @FXML
    void initialize() {



        SignButton.setOnAction(event ->{

            signUpNewUser();




            SignButton.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("OwnerPage.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
```

```
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

        SignButton1.setOnAction(event ->{
            SignButton1.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("PetRegistration.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });
    }

    private void signUpNewUser() {
        String firstName = NameField.getText();
        String lastName = LastNameField.getText();
        String po_batkovi = SecondNameField.getText();
        String email = EmailField.getText();
        String password = PasswordField.getText();
        String phone = PhoneField.getText();

        User user = new User (firstName, lastName, po_batkovi, email, password,
phone);

        DatabaseHandler dbHandler = new DatabaseHandler();

        try {
            dbHandler.signUpUser(user);
        } catch (SQLException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }


}
```

Код оформлення сторінки користувача:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
```

```
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.OwnerController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="46.0" layoutY="381.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Додати улюбленця" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
        <ImageView fitHeight="164.0" fitWidth="142.0" layoutX="25.0"
layoutY="25.0" pickOnBounds="true" preserveRatio="true" />
        <Label layoutX="25.0" layoutY="202.0" prefHeight="28.0" prefWidth="46.0"
text="Ім'я:" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="14.0" />
            </font>
        </Label>
        <Label layoutX="25.0" layoutY="242.0" prefHeight="28.0" prefWidth="67.0"
text="Прізвище:" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="14.0" />
            </font>
        </Label>
        <Label layoutX="25.0" layoutY="280.0" prefHeight="28.0"
prefWidth="120.0" text="По-батькові:" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="14.0" />
            </font>
        </Label>
      <Label layoutX="25.0" layoutY="320.0" prefHeight="28.0" prefWidth="120.0"
text="Телефон:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Button fx:id="SignButton1" layoutX="219.0" layoutY="479.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Вийти" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <Label fx:id="NameLabel" layoutX="99.0" layoutY="202.0" prefHeight="28.0"
prefWidth="156.0" text="Анастасія" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label fx:id="LastNameLabel" layoutX="122.0" layoutY="242.0"
prefHeight="28.0" prefWidth="184.0" text="Ткаченко" textFill="#e47e10">
```

```
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label fx:id="SecondNameLabel" layoutX="120.0" layoutY="280.0"
prefHeight="28.0" prefWidth="120.0" text="Олексіївна" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label fx:id="PhoneLabel" layoutX="107.0" layoutY="320.0"
prefHeight="28.0" prefWidth="238.0" text="+380502959564" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Button fx:id="SearchButton" layoutX="381.0" layoutY="381.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Пошук улюбленця" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
      <ImageView fx:id="image" fitHeight="150.0" fitWidth="142.0" layoutX="22.0"
layoutY="32.0" pickOnBounds="true" preserveRatio="true" />
    </children>
</AnchorPane>
```

Контролер сторінки користувача:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class OwnerController {

    public static String name;
    public static String lstname;
    public static String secndname;
    public static String phne;
    public static int id;

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;
```

```
    @FXML
    private Label NameLabel;

    @FXML
    private Label LastNameLabel;

    @FXML
    private Label SecondNameLabel;

    @FXML
    private Label PhoneLabel;

    @FXML

    private Button SearchButton;

    @FXML
    private Button SignButton;

    @FXML
    private Button SignButton1;

    @FXML
    private ImageView image;

    @FXML
    public void init(){
        NameLabel.setText(name);
        LastNameLabel.setText(lstname);
        SecondNameLabel.setText(secndname);
        PhoneLabel.setText(phne);
    }

    @FXML
    void initialize() {

        //image.setImage(new Image("avatar.png"));

        init();
        SignButton.setOnMouseClicked(value -> {
            SignButton.getScene().getWindow().hide();

            try {

                System.out.println(CheckUser.firstname);

                PetRegistController.ownphone=phne;
                PetRegistController.id = id;
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("PetRegistration.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
```

```
        });

        SignButton1.setOnMouseClicked(event -> {
            SignButton1.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("hello-view.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

        SearchButton.setOnAction(event ->{
            try {
                SearchController.phonesearch=id;
                PetController.ownername=name;
                PetController.phone=phne;
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("Search.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

    }

}
```

Код оформлення сторінки реєстрації улюбленця:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.TextArea?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
```

```
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.PetRegistController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="223.0" layoutY="430.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Додати до бази" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
        <TextField fx:id="NameField" layoutX="223.0" layoutY="68.0"
prefHeight="32.0" prefWidth="263.0" promptText="Ім'я тварини" />
        <TextField fx:id="SpeciesField" layoutX="223.0" layoutY="118.0"
prefHeight="32.0" prefWidth="263.0" promptText="Вид тварини" />
        <TextField fx:id="BreedField" layoutX="223.0" layoutY="168.0"
prefHeight="32.0" prefWidth="263.0" promptText="Порода" />
      <TextField fx:id="DateBirthField" layoutX="223.0" layoutY="222.0"
prefHeight="32.0" prefWidth="263.0" promptText="Дата народження" />
      <TextArea fx:id="DescriptionField" layoutX="223.0" layoutY="265.0"
prefHeight="143.0" prefWidth="263.0" promptText="Особливі прикмети">
         <font>
            <Font size="14.0" />
         </font>
      </TextArea>
    </children>
</AnchorPane>
```

Контролер сторінки реєстрації улюбленця:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class PetRegistController {

    public static String ownphone;
    public static int id;
    @FXML
    private ResourceBundle resources;
```

```
    @FXML
    private TextField DateBirthField;

    @FXML
    private TextArea DescriptionField;

    @FXML
    private URL location;

    @FXML
    private TextField NameField;

    @FXML
    private TextField SpeciesField;

    @FXML
    private TextField BreedField;

    @FXML
    private Button SignButton;

    @FXML
    void initialize() {
        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
            DatabaseHandler dbHandler = new DatabaseHandler();

                try {
                    dbHandler.signUpPetByUser(NameField.getText(),
SpeciesField.getText(),
                            BreedField.getText(), id, DateBirthField.getText(),
DescriptionField.getText() );

                } catch (SQLException e) {
                    e.printStackTrace();
                } catch (ClassNotFoundException e) {
                    e.printStackTrace();
                }

            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("OwnerPage.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

    }

}
```

Код оформлення сторінки улюбленця:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="600.0" prefWidth="700.0" style="-fx-background-
color: #1c489e;" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.PetController">
    <children>
        <ImageView fitHeight="670.0" fitWidth="1071.0" layoutX="-113.0"
layoutY="-1.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Button fx:id="SignButton" layoutX="48.0" layoutY="430.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Позначити як загубленого" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
      <ImageView fitHeight="164.0" fitWidth="142.0" layoutX="25.0"
layoutY="25.0" pickOnBounds="true" preserveRatio="true" />
      <Label layoutX="25.0" layoutY="202.0" prefHeight="28.0" prefWidth="46.0"
text="Ім'я:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label layoutX="25.0" layoutY="242.0" prefHeight="28.0" prefWidth="67.0"
text="Порода:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label layoutX="25.0" layoutY="280.0" prefHeight="28.0" prefWidth="120.0"
text="Історія хвороби:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label layoutX="25.0" layoutY="340.0" prefHeight="28.0" prefWidth="120.0"
text="Лікування:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label fx:id="DiagnozisField" layoutX="140.0" layoutY="273.0"
prefHeight="54.0" prefWidth="517.0" text="Наразі нема" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label fx:id="RecipeField" layoutX="146.0" layoutY="335.0"
prefHeight="66.0" prefWidth="506.0" text="Наразі нема" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
```

```
         </font>
      </Label>
      <Label fx:id="BreedField" layoutX="112.0" layoutY="242.0"
prefHeight="28.0" prefWidth="303.0" text="Порода:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Label fx:id="NameField" layoutX="79.0" layoutY="202.0" prefHeight="28.0"
prefWidth="440.0" text="Ім'я:" textFill="#e47e10">
         <font>
            <Font name="Bernard MT Condensed" size="14.0" />
         </font>
      </Label>
      <Button fx:id="SignButton1" layoutX="372.0" layoutY="430.0"
mnemonicParsing="false" prefHeight="73.0" prefWidth="263.0" style="-fx-
background-color: #207ce6;" text="Вийти" textFill="WHITE">
         <font>
            <Font size="19.0" />
         </font>
      </Button>
   </children>
</AnchorPane>
```

Контролер сторінки улюбленця:

```
package com.example.vetpet_ver2;

import java.io.IOException;
import java.net.URL;
import java.sql.SQLException;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

public class PetController {

    public static String petname;
    public static String breed;
    public static String phone;
    public static String diagnozis;
    public static String recipe;
    public static String ownername;
    public static String descr;

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Label BreedField;
```

```
    @FXML
    private Label DiagnozisField;

    @FXML
    private Label NameField;

    @FXML
    private Label RecipeField;

    @FXML
    private Button SignButton;

    @FXML
    private Button SignButton1;

    @FXML
    void initialize() {
        NameField.setText(petname);
        BreedField.setText(breed);

        SignButton1.setOnMouseClicked(event -> {
            SignButton1.getScene().getWindow().hide();
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("OwnerPage.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        });

        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
            DatabaseHandler dbHandler = new DatabaseHandler();
            try {
                dbHandler.SetLostPet(petname, ownername, phone, descr);
            } catch (SQLException e) {
                e.printStackTrace();
            } catch (ClassNotFoundException e) {
                e.printStackTrace();
            }

            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("OwnerPage.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");
                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
```

```
                e.printStackTrace();

            }
        });
    }

}
```

Код оформлення вікна для пошуку улюбленця:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="332.0" prefWidth="581.0" style="-fx-background-
color: #1c489e; -fx-border-color: #203066; -fx-border-width: 10;"
xmlns="http://javafx.com/javafx/18" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.SearchController">
    <children>
        <ImageView fitHeight="318.0" fitWidth="570.0" layoutX="8.0"
layoutY="7.0" pickOnBounds="true" preserveRatio="true">
            <image>
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Label contentDisplay="CENTER" layoutX="103.0" layoutY="27.0"
prefHeight="96.0" prefWidth="390.0" style="-fx-background-color: null;"
text="Введіть ім'я улюбленця" textAlignment="CENTER" textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="35.0" />
            </font>
        </Label>
        <Button fx:id="SignButton" layoutX="225.0" layoutY="242.0"
mnemonicParsing="false" prefHeight="34.0" prefWidth="131.0" style="-fx-
background-color: #207ce6;" text="Пошук" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
      <TextField fx:id="phoneField" layoutX="103.0" layoutY="123.0"
prefHeight="76.0" prefWidth="390.0" />
    </children>
</AnchorPane>
```

Контролер вікна для пошуку улюбленця:

```
package com.example.vetpet_ver2;

import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
```

```
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.TextField;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.stage.StageStyle;

import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;


public class SearchController {

    public static int phonesearch;
    public static  int petid;
    @FXML
    private TextField phoneField;

    @FXML
    private Button SignButton;

    @FXML
    void initialize(){
        SignButton.setOnAction(event -> {
            String loginText = phoneField.getText().trim();

            if (loginText.equals("")) {
                System.out.println("Error");
                try {
                    FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("Error.fxml"));
                    Parent root2 = (Parent) fxmlLoader.load();
                    Stage stage1 = new Stage();
                    stage1.initModality(Modality.APPLICATION_MODAL);
                    stage1.initStyle(StageStyle.UNDECORATED);
                    stage1.setTitle("ABC");
                    stage1.setScene(new Scene(root2));
                    stage1.show();

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
            else {
                SignButton.getScene().getWindow().hide();
                try {
                    SearchPet(loginText);
                } catch (SQLException e) {
                    e.printStackTrace();
                } catch (ClassNotFoundException e) {
                    e.printStackTrace();
                }
            }


        });

    }

    private void SearchPet(String phoneText) throws SQLException,
ClassNotFoundException {
```

```
        DatabaseHandler handler = new DatabaseHandler();
        Pet pet = new Pet();
        pet.setName(phoneText);
        pet.setPhone(phonesearch);

        ResultSet resultSet = handler.getPet(pet);


        int counter = 0;
        while (resultSet.next()) {
            counter++;
            PetController.petname=resultSet.getString("pets_name");
            PetController.breed=resultSet.getString("pets_breed");
            petid = resultSet.getInt("pets_id");
            PetController.descr = resultSet.getString("description");

        }

        if (counter >= 1 ) {
            try {
                FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("PetPage.fxml"));
                Parent root1 = (Parent) fxmlLoader.load();
                Stage stage = new Stage();
                stage.initModality(Modality.APPLICATION_MODAL);
                stage.initStyle(StageStyle.UNDECORATED);
                stage.setTitle("ABC");

                stage.setScene(new Scene(root1));
                stage.show();

            } catch (IOException e) {
                e.printStackTrace();

            }
        }

    }
}
```

Код оформлення вікна помилки:

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.text.Font?>

<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity"
minWidth="-Infinity" prefHeight="332.0" prefWidth="581.0" style="-fx-background-
color: #1c489e; -fx-border-color: #203066; -fx-border-width: 10;"
xmlns="http://javafx.com/javafx/18" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.example.vetpet_ver2.ErrorController">
    <children>
        <ImageView fitHeight="318.0" fitWidth="570.0" layoutX="8.0"
layoutY="7.0" pickOnBounds="true" preserveRatio="true">
            <image>
```

```
                <Image url="@images/bacground.png" />
            </image></ImageView>
        <Label contentDisplay="CENTER" layoutX="87.0" layoutY="62.0"
prefHeight="96.0" prefWidth="408.0" style="-fx-background-color: null;"
text="Акаунту з такими даними&#10; не існує" textAlignment="CENTER"
textFill="#e47e10">
            <font>
                <Font name="Bernard MT Condensed" size="35.0" />
            </font>
        </Label>
        <Button fx:id="SignButton" layoutX="208.0" layoutY="189.0"
mnemonicParsing="false" prefHeight="79.0" prefWidth="167.0" style="-fx-
background-color: #207ce6;" text="OK" textFill="WHITE">
            <font>
                <Font size="19.0" />
            </font>
        </Button>
    </children>
</AnchorPane>
```

Контролер вікна помилки:

```
package com.example.vetpet_ver2;

import java.net.URL;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.scene.control.Button;

public class ErrorController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button SignButton;

    @FXML
    void initialize() {
        SignButton.setOnAction(event ->{
            SignButton.getScene().getWindow().hide();
        });

    }

}
```

Код роботи з базами даних з усіх вікон:

```
package com.example.vetpet_ver2;

import java.sql.*;

public class DatabaseHandler extends Configs {
```

```
    Connection dbConnection;

    public Connection getDbConnection() throws ClassNotFoundException,
SQLException{
        String connection = "jdbc:mysql://"+ dbHost+":" + dbPort +"/"+bdName;

        Class.forName("com.mysql.cj.jdbc.Driver");

        dbConnection = DriverManager.getConnection(connection, dbUser, bdPass);

        return dbConnection;
    }

    public void signUpUser(User user) throws SQLException,
ClassNotFoundException {

        String insert = "INSERT INTO " + Const.USER_TABLE + "("
                + Const.USER_FIRSTNAME + "," + Const.USER_LASTNAME + ","
                + Const.USER_POBATKOVI + "," + Const.USER_USERNAME + ","
                + Const.USER_PASSWORD + "," + Const.USER_PHONE + ")"
                + "VALUES (?,?,?,?,?,?)";

        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(insert);
        preparedStatement.setString(1, user.getFirstName());
        preparedStatement.setString(2, user.getLastname());
        preparedStatement.setString(3, user.getPo_batkovi());
        preparedStatement.setString(4, user.getUsername());
        preparedStatement.setString(5, user.getPassword());
        preparedStatement.setString(6, user.getPhone_number());


        preparedStatement.executeUpdate();
    }

    public void signUpDoctor(String firstname, String lastname, String
pobatkovi, String workplace, String username,
                             String specialization, String password, String
phone)
            throws SQLException, ClassNotFoundException {

        String insert = "INSERT INTO " + Const.DOCTOR_TABLE + "("
                + Const.DOCTOR_FIRSTNAME + "," + Const.DOCTOR_LASTNAME + ","
                + Const.DOCTOR_POBATKOVI + "," + Const.DOCTOR_PHONE + ","
                + Const.DOCTOR_WORKPLACE + "," + Const.DOCTOR_SPECIALIZATION +
"," + Const.DOCTOR_USERNAME + ","
                + Const.DOCTOR_PASSWORD + ")"
                + "VALUES (?,?,?,?,?,?,?,?)";

        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(insert);
        preparedStatement.setString(1, firstname);
        preparedStatement.setString(2, lastname);
        preparedStatement.setString(3, pobatkovi);
        preparedStatement.setString(4, phone);
        preparedStatement.setString(5, workplace);
        preparedStatement.setString(6, specialization);
        preparedStatement.setString(7, username);
        preparedStatement.setString(8, password);
```

```
        preparedStatement.executeUpdate();
    }


    public void signup(int userid, int petis) throws SQLException,
ClassNotFoundException {
        String insert = "INSERT INTO " + Const.PET_AND_USER_TABLE +
                "(" + Const.USER_ID + "," + Const.PET_ID + ")" +
                "VALUES (?,?)";

        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(insert);
        preparedStatement.setString(1, String.valueOf(userid));
        preparedStatement.setString(2, String.valueOf(petis));

        preparedStatement.executeUpdate();
    }

    public void signUpPetByUser(String name, String species, String breed, int
phone, String databirth,
                                String description) throws SQLException,
ClassNotFoundException {

        String insert = "INSERT INTO " + Const.PET_TABLE + "("
                + Const.PET_NAME + "," + Const.PETS_SPECIES + ","
                + Const.PETS_BREED + "," + Const.USER_ID + "," +
Const.PET_DATABIRTH
                + "," + Const.LOST_DESCRIPTION + ")"
                + "VALUES (?,?,?,?,?,?)";

        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(insert);
        preparedStatement.setString(1, name);
        preparedStatement.setString(2, species);
        preparedStatement.setString(3, breed);
        preparedStatement.setString(4, String.valueOf(phone));
        preparedStatement.setString(5, databirth);
        preparedStatement.setString(6, description);


        preparedStatement.executeUpdate();
    }

    public void SetLostPet(String name, String ownername, String ownerphone,
String description) throws SQLException, ClassNotFoundException {

        String insert = "INSERT INTO " + Const.LOST_TABLE + "("
                + Const.PET_NAME + "," + Const.USER_FIRSTNAME + ","
                + Const.USER_PHONE + "," + Const.LOST_DESCRIPTION + ")"
                + "VALUES (?,?,?,?)";

        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(insert);
        preparedStatement.setString(1, name);
        preparedStatement.setString(2, ownername);
        preparedStatement.setString(3, ownerphone);
        preparedStatement.setString(4, description);


        preparedStatement.executeUpdate();
    }

    public ResultSet getDoctor(Doctor doctor) throws SQLException,
```

```
ClassNotFoundException {

        ResultSet resultSet = null;

        String select = "SELECT * FROM " + Const.DOCTOR_TABLE + " WHERE "
                + Const.DOCTOR_SPECIALIZATION +  "=?";
        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(select);
        preparedStatement.setString(1, doctor.getSpecialization());
        resultSet = preparedStatement.executeQuery();
        return resultSet;

    };

    public ResultSet getHomelessPet(HomelessPet homelessPet) throws
SQLException, ClassNotFoundException {

        ResultSet resultSet = null;

        String select = "SELECT * FROM " + Const.HOMELESS_PETS_TABLE + " WHERE "
                + Const.HOMELESS_SPECIES +  "=?";
        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(select);
        preparedStatement.setString(1, homelessPet.getSpecies());
        resultSet = preparedStatement.executeQuery();
        return resultSet;

    };
    public ResultSet getUser(User user) throws SQLException,
ClassNotFoundException {

        ResultSet resultSet = null;

        String select = "SELECT * FROM " + Const.USER_TABLE + " WHERE "
                + Const.USER_USERNAME + " =? AND " + Const.USER_PASSWORD + "
=?";
        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(select);
        preparedStatement.setString(1, user.getUsername());
        preparedStatement.setString(2, user.getPassword());
        resultSet = preparedStatement.executeQuery();
        return resultSet;

    };

    public ResultSet getLostPet (LostPet lostPet) throws SQLException,
ClassNotFoundException {
        ResultSet resultSet = null;

        String select = "SELECT * FROM " + Const.LOST_TABLE;
        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(select);
        resultSet = preparedStatement.executeQuery();
        return resultSet;
    }
    public ResultSet getPet(Pet pet) throws SQLException, ClassNotFoundException
{

        ResultSet resultSet = null;

        String select = "SELECT * FROM " + Const.PET_TABLE + " WHERE "
                + Const.PET_NAME + " =? AND " + Const.USER_ID + "=?";
        PreparedStatement preparedStatement =
```

```
getDbConnection().prepareStatement(select);
        preparedStatement.setString(1, pet.getName());
        preparedStatement.setInt(2, pet.getPhone());
        resultSet = preparedStatement.executeQuery();
        return resultSet;

    };
    public String getUserName(User user, String firstname){
        return firstname;
    }

    /*public ResultSet checkUser(User user) throws SQLException,
ClassNotFoundException {
        ResultSet resultSet = null;

        String select = "SELECT * FROM " + Const.USER_TABLE + " WHERE "
                + Const.USER_ID + " =?";
        PreparedStatement preparedStatement =
getDbConnection().prepareStatement(select);
        preparedStatement.setString(1, user.getId());
        resultSet = preparedStatement.executeQuery();
        return resultSet;
    }*/

}
```

Конфігурації:

```
package com.example.vetpet_ver2;

public class Configs {

    protected String dbHost = "localhost";
    protected String dbPort = "3306";
    protected String dbUser = "root";
    protected String bdPass = "Uko547896321";
    protected String bdName = "users";
}
```

Сторінка з усіма даними для роботи з базами даних:

```
package com.example.vetpet_ver2;

public class Const {

    public static final String USER_TABLE = "users";
    public static final String USER_ID = "id_users";
    public static final String USER_FIRSTNAME = "firstname";
    public static final String USER_LASTNAME = "lastname";
    public static final String USER_POBATKOVI = "po_batkovi";
    public static final String USER_USERNAME = "username";
    public static final String USER_PASSWORD = "password";
    public static final String USER_PHONE = "phone_number";

    public static final String DOCTOR_TABLE = "doctors";
    public static final String DOCTOR_ID = "DOCTOR_ID";
    public static final String DOCTOR_FIRSTNAME = "doctor_firstname";
```

```
    public static final String DOCTOR_LASTNAME = "doctor_lastname";
    public static final String DOCTOR_POBATKOVI = "doctor_secondname";
    public static final String DOCTOR_PHONE = "doctor_phone";
    public static final String DOCTOR_WORKPLACE = "doctor_workplace";
    public static final String DOCTOR_SPECIALIZATION = "doctor_specialization";
    public static final String DOCTOR_USERNAME = "doctor_email";
    public static final String DOCTOR_PASSWORD = "doctor_password";

    public static final String PET_TABLE = "pets";
    public static final String PET_ID = "PETS_ID";
    public static final String PET_NAME = "PETS_name";
    public static final String PETS_SPECIES = "PETS_SPECIES";
    public static final String PETS_BREED = "PETS_BREED";
    public static final String PET_DATABIRTH = "PET_BIRTHDATA";
    public static final String PETS_DIAGNOZIS = "PETS_DIAGNOZIS";
    public static final String PETS_RECIPE = "PETS_RECIPE";
    public static final String PET_DATEBIRTH = "PET_BIRTHDATA";

    public static final String LOST_TABLE = "lost_pets";
    public static final String LOST_ID = "idlostpets";
    public static final String LOST_DESCRIPTION = "description";

    public static final String PET_AND_USER_TABLE = "users_and_pets";
    public static final String HOMELESS_PETS_TABLE = "homeless_pets";
    public static final String ID_HOMELESS_PETS = "idHomeless_pets";
    public static final String HOMELESS_BREED = "Breed";
    public static final String HOMELESS_NAME = "Name";
    public static final String HOMELESS_SPECIES = "Species";
    public static final String HOMELESS_AGE = "Age";

    public static int ID;
}
```

Клас користувача:

```
package com.example.vetpet_ver2;

public class User {
    private int id;
    private String firstName;
    private String  lastname;
    private String  po_batkovi;
    private String  username;
    private String  password;
    private String  phone_number;

    public User(String firstName, String lastname, String po_batkovi, String
username, String password, String phone_number) {
        this.id = id;
        this.firstName = firstName;
        this.lastname = lastname;
        this.po_batkovi = po_batkovi;
        this.username = username;
        this.password = password;
        this.phone_number = phone_number;
    }

    public User() {

    }
```

```
    public void setId(){this.id = id;}

    public int getId(){return id;}

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastname() {
        return lastname;
    }

    public void setLastname(String lastname) {
        this.lastname = lastname;
    }

    public String getPo_batkovi() {
        return po_batkovi;
    }

    public void setPo_batkovi(String po_batkovi) {
        this.po_batkovi = po_batkovi;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getPhone_number() {
        return phone_number;
    }

    public void setPhone_number(String phone_number) {
        this.phone_number = phone_number;
    }
}
```

Клас лікаря:

```
package com.example.vetpet_ver2;

public class Doctor {
```

```
    private String id;
    private String firstName;
    private String  lastname;
    private String  po_batkovi;
    private String  phone_number;
    private String workplace;
    private String specialization;
    private String  username;
    private String  password;

    public Doctor(String id, String firstName, String lastname, String
po_batkovi, String phone_number, String workplace, String specialization, String
username, String password) {
        this.id = id;
        this.firstName = firstName;
        this.lastname = lastname;
        this.po_batkovi = po_batkovi;
        this.phone_number = phone_number;
        this.workplace = workplace;
        this.specialization = specialization;
        this.username = username;
        this.password = password;
    }

    public Doctor() {

    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastname() {
        return lastname;
    }

    public void setLastname(String lastname) {
        this.lastname = lastname;
    }

    public String getPo_batkovi() {
        return po_batkovi;
    }

    public void setPo_batkovi(String po_batkovi) {
        this.po_batkovi = po_batkovi;
    }

    public String getPhone_number() {
        return phone_number;
    }
```

```
    public void setPhone_number(String phone_number) {
        this.phone_number = phone_number;
    }

    public String getWorkplace() {
        return workplace;
    }

    public void setWorkplace(String workplace) {
        this.workplace = workplace;
    }

    public String getSpecialization() {
        return specialization;
    }

    public void setSpecialization(String specialization) {
        this.specialization = specialization;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

Клас улюбленця:

```
package com.example.vetpet_ver2;

public class Pet {
    private String id;
    private String name;
    private String species;
    private String breed;
    private int phone;
    private String diagnozis;
    private String recipe;


    public Pet(String id, String name, String species, String breed, int phone,
String diagnozis, String recipe) {
        this.id = id;
        this.name = name;
        this.species = species;
        this.breed = breed;
        this.phone = phone;
        this.diagnozis = diagnozis;
```

```
        this.recipe = recipe;
    }

    public Pet() {

    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSpecies() {
        return species;
    }

    public void setSpecies(String species) {
        this.species = species;
    }

    public String getBreed() {
        return breed;
    }

    public void setBreed(String breed) {
        this.breed = breed;
    }

    public int getPhone() {
        return phone;
    }

    public void setPhone(int phone) {
        this.phone = phone;
    }

    public String getDiagnozis() {
        return diagnozis;
    }

    public void setDiagnozis(String diagnozis) {
        this.diagnozis = diagnozis;
    }

    public String getRecipe() {
        return recipe;
    }

    public void setRecipe(String recipe) {
        this.recipe = recipe;
    }
}
```

Клас загубленого улюбленця:

```
package com.example.vetpet_ver2;

public class LostPet {
    private String id;
    private String name;
    private String ownername;
    private String ownerphone;
    private String decription;


    public LostPet(String id, String name, String ownername, String ownerphone,
String decription) {
        this.id = id;
        this.name = name;
        this.ownername = ownername;
        this.ownerphone = ownerphone;
        this.decription = decription;
    }

    public LostPet() {

    }
}
```

Клас безпритульних тварин:

```
package com.example.vetpet_ver2;

public class HomelessPet {
    private String idHomeless_pets;
    private String Name;
    private String Breed;
    private String Species;
    private String Age;

    public HomelessPet(String idHomeless_pets, String name, String breed, String
species, String age) {
        this.idHomeless_pets = idHomeless_pets;
        Name = name;
        Breed = breed;
        Species = species;
        Age = age;
    }

    public HomelessPet() {

    }

    public String getIdHomeless_pets() {
        return idHomeless_pets;
    }

    public void setIdHomeless_pets(String idHomeless_pets) {
        this.idHomeless_pets = idHomeless_pets;
```

```
    }

    public String getName() {
        return Name;
    }

    public void setName(String name) {
        Name = name;
    }

    public String getBreed() {
        return Breed;
    }

    public void setBreed(String breed) {
        Breed = breed;
    }

    public String getSpecies() {
        return Species;
    }

    public void setSpecies(String species) {
        Species = species;
    }

    public String getAge() {
        return Age;
    }

    public void setAge(String age) {
        Age = age;
    }
}
```

Сторінка з кодом для початку роботи програмної системи:

```
package com.example.vetpet_ver2;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

import java.io.IOException;

public class HelloApplication extends Application {
    @Override
    public void start(Stage stage) throws IOException {
        FXMLLoader fxmlLoader = new
FXMLLoader(HelloApplication.class.getResource("firstPage.fxml"));
        Scene scene = new Scene(fxmlLoader.load(), 700, 600);
        stage.setTitle("Hello!");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] args) {
        launch();
    }
}
```