PET HEALTHCARE MANAGEMENT SYSTEM

A MINI-PROJECT REPORT

Submitted by

VARSHINI S-240701579

DEVISHREE J-240701702

in partial fulfillment of the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project "PET HEALTHCARE MANAGEMENT SYSTEM" is the bonafide work of “VARSHINI S”, “DEVISHREE J” who carried out the project work under my supervision.

SIGNATURE

Mrs.SATHYAVATHI S

ASSISTANT PROFESSOR SG

Dept. of Computer Science and Engg,

Rajalakshmi Engineering College Chennai

This mini project report is submitted for the viva voce examination to be held on

INTERNAL EXAMINER **EXTERNALEXAMINER**

ABSTRACT

In the modern pet care landscape, efficient management of pet health records and veterinary appointments is crucial for ensuring quality healthcare services. Pet owners often struggle with organizing appointment schedules, maintaining medical histories, and coordinating with veterinarians effectively. To address this challenge, our team has developed a

comprehensive Pet Healthcare Management System using Java Swing and MySQL database. The main objective of this project is to create a centralized platform where pet owners can register their pets, book appointments, manage schedules, and interact seamlessly with veterinary professionals. This system provides secure user authentication with SHA-256 password hashing, role-based access control for owners and veterinarians, automated email reminders, and appointment management functionality. The system features two distinct user interfaces: a Pet Owner Dashboard for managing pets and appointments, and a Vet Dashboard for handling scheduled consultations. By implementing this system, veterinary clinics and pet care centers can maintain well-organized digital records of pet health information, enabling efficient healthcare management and improving the overall pet care experience.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman MR. S. **MEGANATHAN** and the chairperson

DR. M.THANGAM MEGANATHAN for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head Of The Department **Dr.E.M. MALATHY** and our Deputy Head Of The Department **Dr. J. MANORANJINI** for being ever supporting force during our project work.

We also extend our sincere and hearty thanks to our internal guide Dr. V. JANANEE, for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

VARSHINI S

DEVISHREE J

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The Pet Healthcare Management System is a Java Swing and MySQL-based desktop application designed for efficient management of pet health records and veterinary appointments. It enables pet owners to register their pets, book appointments with veterinarians, and manage their schedules effectively. The system features secure user authentication with SHA-256 password hashing for enhanced data security. It supports role-based access control with separate interfaces for pet owners and veterinarians, ensuring appropriate functionality for each user type. The system includes an automated reminder service that sends email notifications for upcoming appointments, helping reduce no-shows and improving clinic efficiency. This comprehensive platform streamlines pet healthcare management within veterinary clinics and pet care centers.

1.2 SCOPE OF THE WORK

The project enables veterinary clinics and pet care centers to efficiently manage pet health records and appointment schedules by storing pet information, owner details, breed data, and appointment histories systematically. It offers quick access through intuitive interfaces for both pet owners and veterinarians, providing professional dashboards for pet registration, appointment booking, schedule management, and consultation tracking with secure access control and comprehensive validation. The system includes automated reminder functionality that notifies pet owners about upcoming appointments, reducing missed appointments and improving overall service quality.

1.3 PROBLEM STATEMENT

Many veterinary clinics and pet owners face challenges in organizing and managing pet health records and appointment schedules effectively. Traditional manual methods are time-consuming and error-prone, making it difficult to track appointments, maintain pet medical histories, or coordinate between owners and veterinarians. The lack of automated reminders leads to missed appointments, resulting in lost revenue for clinics and compromised pet health. Additionally, the absence of centralized access control and validation mechanisms can lead to scheduling conflicts, data inconsistencies, and security vulnerabilities.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The main objective of this project is to create a centralized pet healthcare management platform with secure authentication, role-based dashboards, and efficient appointment scheduling capabilities. This system helps to maintain the availability and details of pets, breeds, veterinarians, and appointments in an organized manner. The specific objectives include providing secure user registration and authentication with password hashing, enabling pet owners to register and manage multiple pets with breed information, facilitating appointment booking with veterinarian selection and date scheduling, allowing pet owners to reschedule or cancel appointments as needed, providing veterinarians with a dedicated dashboard to manage assigned appointments, implementing automated email reminder service for upcoming appointments, ensuring data integrity through comprehensive validation mechanisms, and supporting role-based permissions for appropriate access control.

CHAPTER 2
SYSTEM SPECIFICATIONS

2.1 SOFTWARE SPECIFICATIONS

Component	**Specification**
Operating System	WINDOWS 10
Front-End	Java Swing
Back-End	MySQL
Language	Java, SQL

CHAPTER 3
MODULE DESCRIPTION

This application consists of eight major modules. When the program runs, it displays the login window where users can authenticate themselves as either a Pet Owner or a Veterinarian. The description of the modules are as follows:

1. User Authentication Module

When a user tries to access the system, they need to either login with existing credentials or register as a new user. The authentication module handles user registration with username, email, password, and user type, password validation with minimum 6 characters requirement, SHA-256 password hashing for enhanced security, user login with username and password verification, session management for logged-in users, separate registration flow for veterinarians requiring additional details, and database integrity checks for unique usernames and emails.

2. Pet Owner Dashboard Module

After successful login, pet owners are directed to their personalized dashboard which displays welcome message with username, three main navigation options for pet and appointment management, quick access to Pet Registration and Management, quick access to Appointment Booking, quick access to Appointment Management for view, reschedule and cancel operations, logout functionality to return to login screen, and clean user-friendly interface with large clearly labeled buttons.

3. Pet Registration Module

This module allows authenticated pet owners to register and manage their pets with display of all registered pets in a table format showing ID, Pet Name, Owner Name and

Breed. It includes pet name and owner name input fields, breed selection from a dropdown populated with available breeds from database, Add Pet functionality with comprehensive validation, real-time table refresh to show newly added pets, non-editable table cells to prevent accidental data modification, back navigation to return to Owner Dashboard, and database integration linking pets to their owners through user_id foreign key.

4. Appointment Booking Module

The appointment booking module provides comprehensive scheduling capabilities including pet selection dropdown populated with user's registered pets, veterinarian selection dropdown showing vet name and specialization, appointment type selection for Health Issue or General Checkup, problem description field required for health issues and auto-filled for checkups, date picker for appointment scheduling in YYYY-MM-DD format, validation preventing past date bookings, automatic appointment confirmation with unique appointment ID, email notification sent to pet owner upon successful booking, display of appointment details including pet name, vet name and date, and integration with Appointments, Pets and Vets tables.

5. Appointment Management Module

This module displays and manages all appointments for the logged-in pet owner with table view showing appointment ID, pet name, vet name, date, problem description and status. It includes filter to display only scheduled and cancelled appointments, reschedule functionality allowing date modification for scheduled appointments, validation ensuring new appointment dates are not in the past, cancel appointment functionality with confirmation dialog, status update from Scheduled to Cancelled upon cancellation, email notifications sent for both reschedule and cancellation actions, refresh button to reload appointment data, owner-specific view showing only appointments for their pets, and prevention of modifications to completed appointments.

6. Veterinarian Dashboard Module

The Vet Dashboard provides veterinarians with tools to manage their appointments including display of all appointments assigned to the logged-in veterinarian, table showing appointment ID, pet name, owner name, date, problem description and status, view of both scheduled and cancelled appointments, Mark as Completed functionality for scheduled appointments, confirmation dialog before marking appointments complete, status update from Scheduled to Completed, automatic table refresh after status updates, logout functionality, and vet-specific filtering ensuring veterinarians only see their assigned appointments.

7. Database Utility Module

This module handles all database operations and utility functions with database connection management, JDBC driver initialization with error handling, SHA-256 password hashing for secure credential storage, simulated email service for appointment notifications, static utility methods accessible throughout the application, error handling and logging for database operations, and database connection URL configuration with timezone settings.

8. Automated Reminder Service Module

The background reminder service provides automated appointment notifications with ScheduledExecutorService running as a daemon thread, automatic execution every 12 hours to check for upcoming appointments, identification of appointments scheduled for the next day, email reminder generation with appointment details, integration with Users, Pets, Vets and Appointments tables, batch processing of multiple reminders in a single execution, console logging for monitoring reminder service activity, automatic startup when application launches, and system-level service requiring no user interaction.

CHAPTER 4
CODING

Sample 1: Database Connection and Utility Methods

```
import java.sql.*;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.HexFormat;

public class DbManager {

    private        static        final        String        URL        =
"jdbc:mysql://localhost:3306/pet?useSSL=false&serverTimezone=UTC";
    private static final String USER = "root";
    private static final String PASSWORD = "root";

    static {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (Exception e) {
            throw new RuntimeException("MySQL JDBC Driver not found!", e);
        }
    }

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }

    // Hash passwords using SHA-256
    public static String hashPassword(String password) {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            byte[] hash = md.digest(password.getBytes());
            return HexFormat.of().formatHex(hash);
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException("SHA-256 not available", e);
        }
    }
}
```

Sample 2: User Authentication

```
import java.sql.*;

public class AuthService {

    // LOGIN
    public static User login(String username, String password, String userType) throws SQLException {

        String sql = "SELECT user_id, email FROM Users WHERE username = ? AND password = ? AND user_type = ?";

        try (Connection conn = DbManager.getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {

            stmt.setString(1, username);
            stmt.setString(2, DbManager.hashPassword(password));
            stmt.setString(3, userType.toLowerCase());

            ResultSet rs = stmt.executeQuery();
            if (rs.next()) {
                return new User(
                        rs.getInt("user_id"),
                        username,
                        rs.getString("email"),
                        userType
                );
            }
        }
        return null; // invalid credentials
    }
}
```

Sample 3: User Registration with Validation

```
import java.sql.*;

public class RegistrationService {

    public static boolean registerUser(String username, String email, String password,
String userType)
            throws SQLException {

        if (username.isEmpty() || email.isEmpty() || password.isEmpty())
            throw new IllegalArgumentException("All fields are required!");

        if (password.length() < 6)
            throw new IllegalArgumentException("Password must be at least 6
characters!");

        String sql = "INSERT INTO Users (username, email, password, user_type)
VALUES (?, ?, ?, ?)";

        try (Connection conn = DbManager.getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {

            stmt.setString(1, username);
            stmt.setString(2, email);
            stmt.setString(3, DbManager.hashPassword(password));
            stmt.setString(4, userType.toLowerCase());

            stmt.executeUpdate();
            return true;
        }
    }

    // Optional: Register Vet Details
    public static void registerVet(int userId, String vetName, String specialization)
throws SQLException {
        String sql = "INSERT INTO Vets (vet_name, specialization, user_id) VALUES
(?, ?, ?)";

        try (Connection conn = DbManager.getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql)) {
```

```
            stmt.setString(1, vetName);
            stmt.setString(2, specialization);
            stmt.setInt(3, userId);

            stmt.executeUpdate();
        }
    }
}
```

Sample 4: Appointment Booking

```
import java.sql.*;
import java.time.LocalDate;

public class AppointmentService {

    public static int bookAppointment(int petId, int vetId, LocalDate date, String
problemDescription)
            throws Exception {

        if (date.isBefore(LocalDate.now()))
            throw new IllegalArgumentException("Cannot book past appointment!");

        String sql = """
                INSERT INTO Appointments
                (pet_id, vet_id, appointment_date, problem_description, status)
                VALUES (?, ?, ?, ?, 'Scheduled')
                """;

        try (Connection conn = DbManager.getConnection();
             PreparedStatement stmt = conn.prepareStatement(sql,
Statement.RETURN_GENERATED_KEYS)) {

            stmt.setInt(1, petId);
            stmt.setInt(2, vetId);
            stmt.setDate(3, Date.valueOf(date));
            stmt.setString(4, problemDescription);

            stmt.executeUpdate();

            ResultSet rs = stmt.getGeneratedKeys();
            if (rs.next())
                return rs.getInt(1);

            return -1;
        }
    }
}
```

CHAPTER 5
SCREENSHOTS

Fig 5.1 Login Page

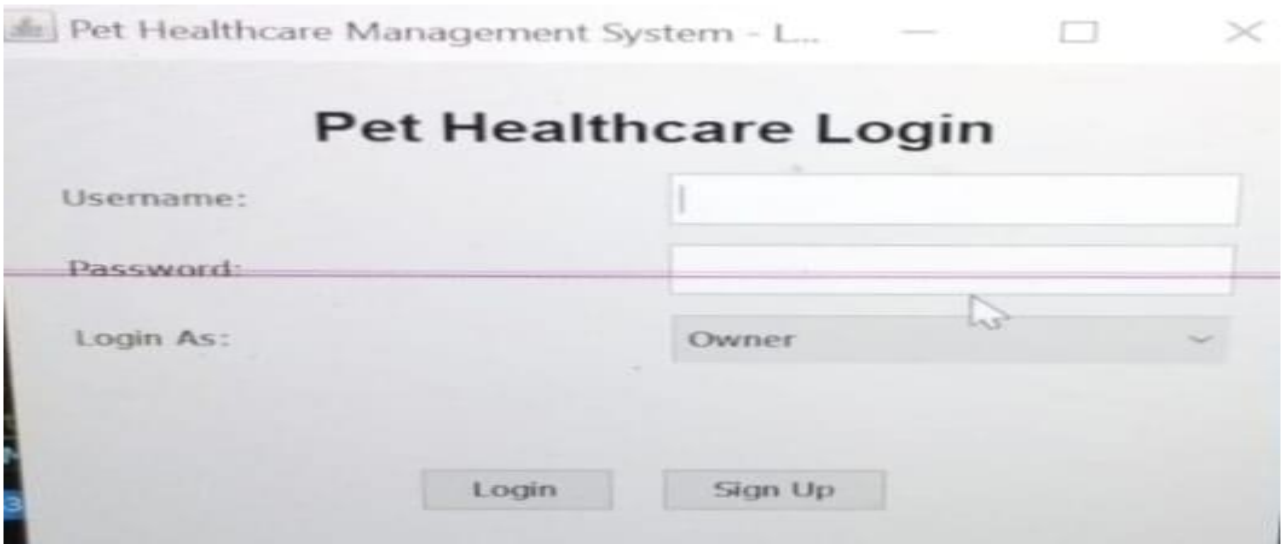


Fig 5.2 Signup Page

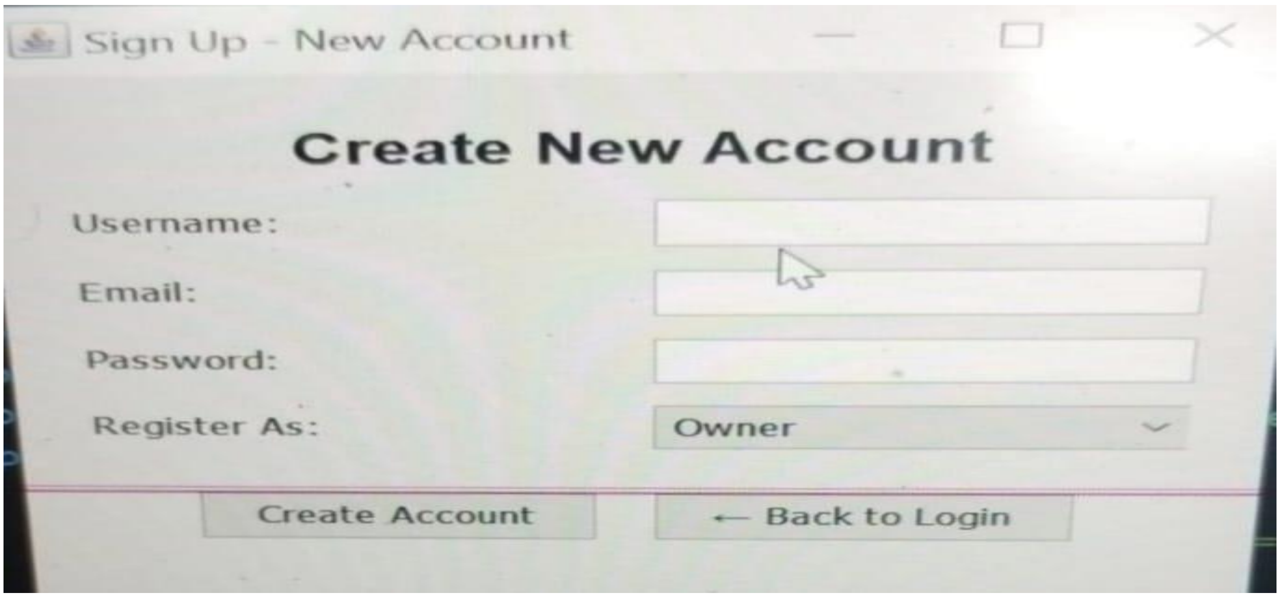
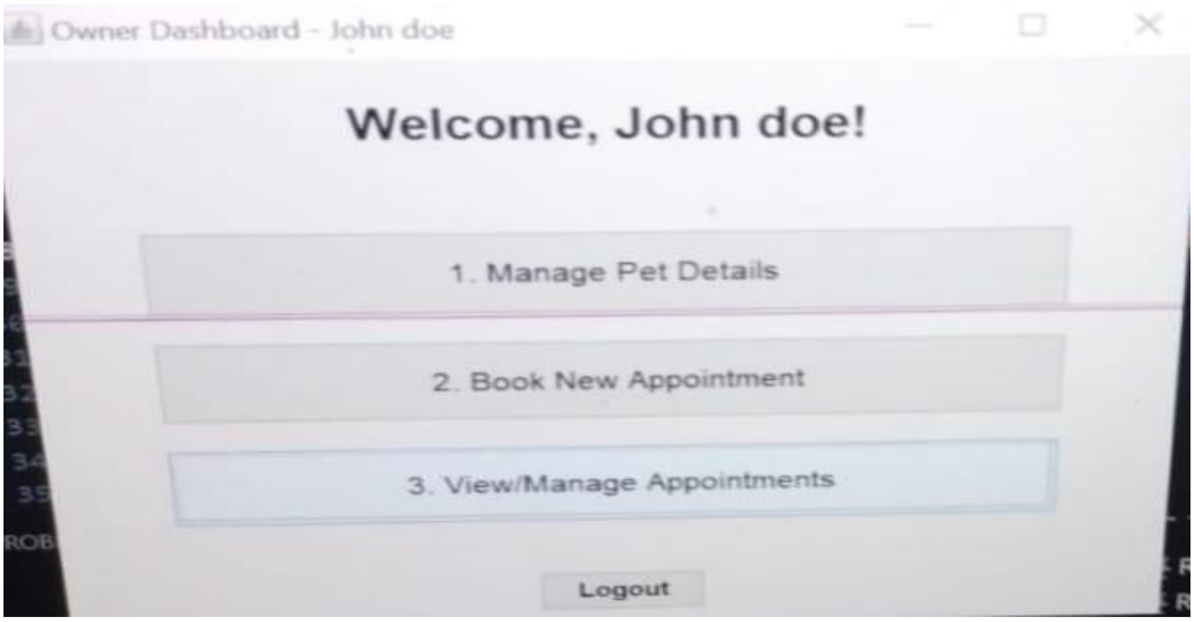


Fig 5.3 Owner Dashboard



Owner Dashboard - John doe

Welcome, John doe!

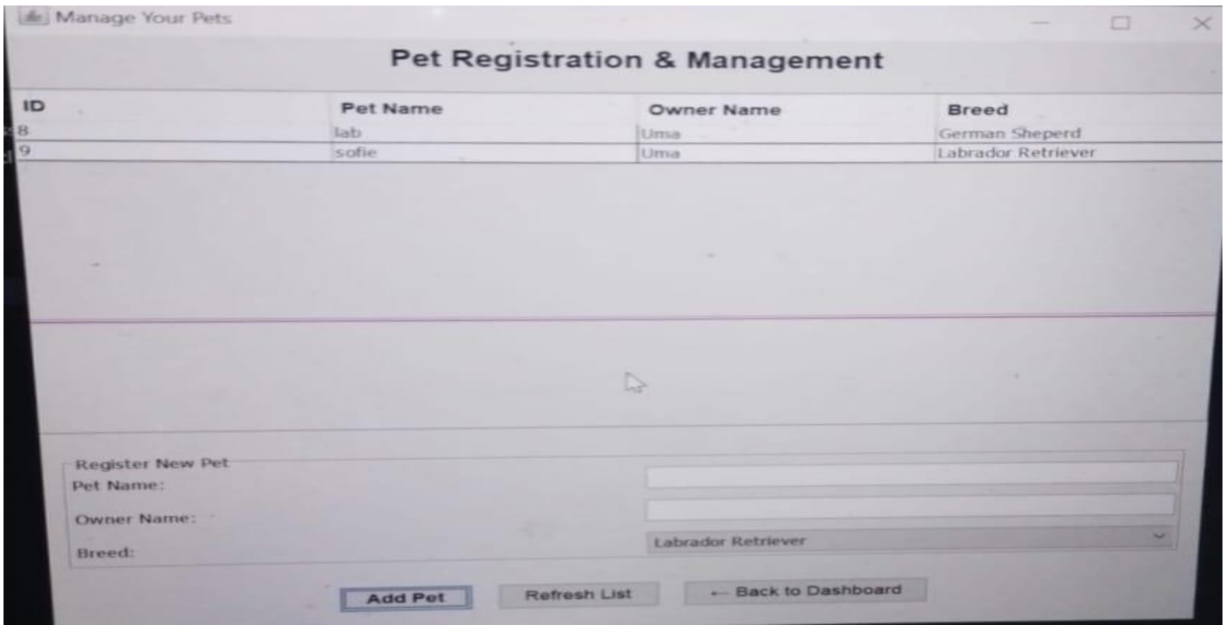
1. Manage Pet Details

2. Book New Appointment

3. View/Manage Appointments

Logout

Fig 5.4 Pet Registration Interface



Manage Your Pets

Pet Registration & Management

ID	Pet Name	Owner Name	Breed
8	lab	Uma	German Sheperd
9	sofie	Uma	Labrador Retriever

Register New Pet

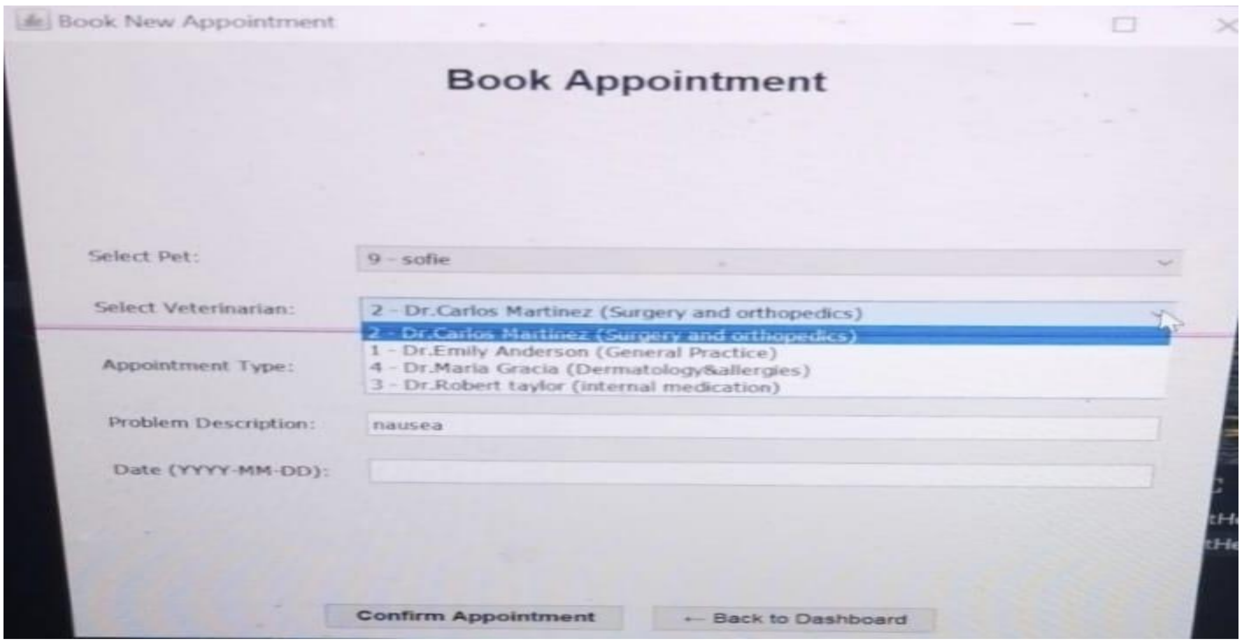
Pet Name:

Owner Name:

Breed: Labrador Retriever

Add Pet | Refresh List | ← Back to Dashboard

Fig 5.5 Appointment Booking


Book New Appointment

Book Appointment

Select Pet: 9 - sofie

Select Veterinarian: 2 - Dr.Carlos Martinez (Surgery and orthopedics)

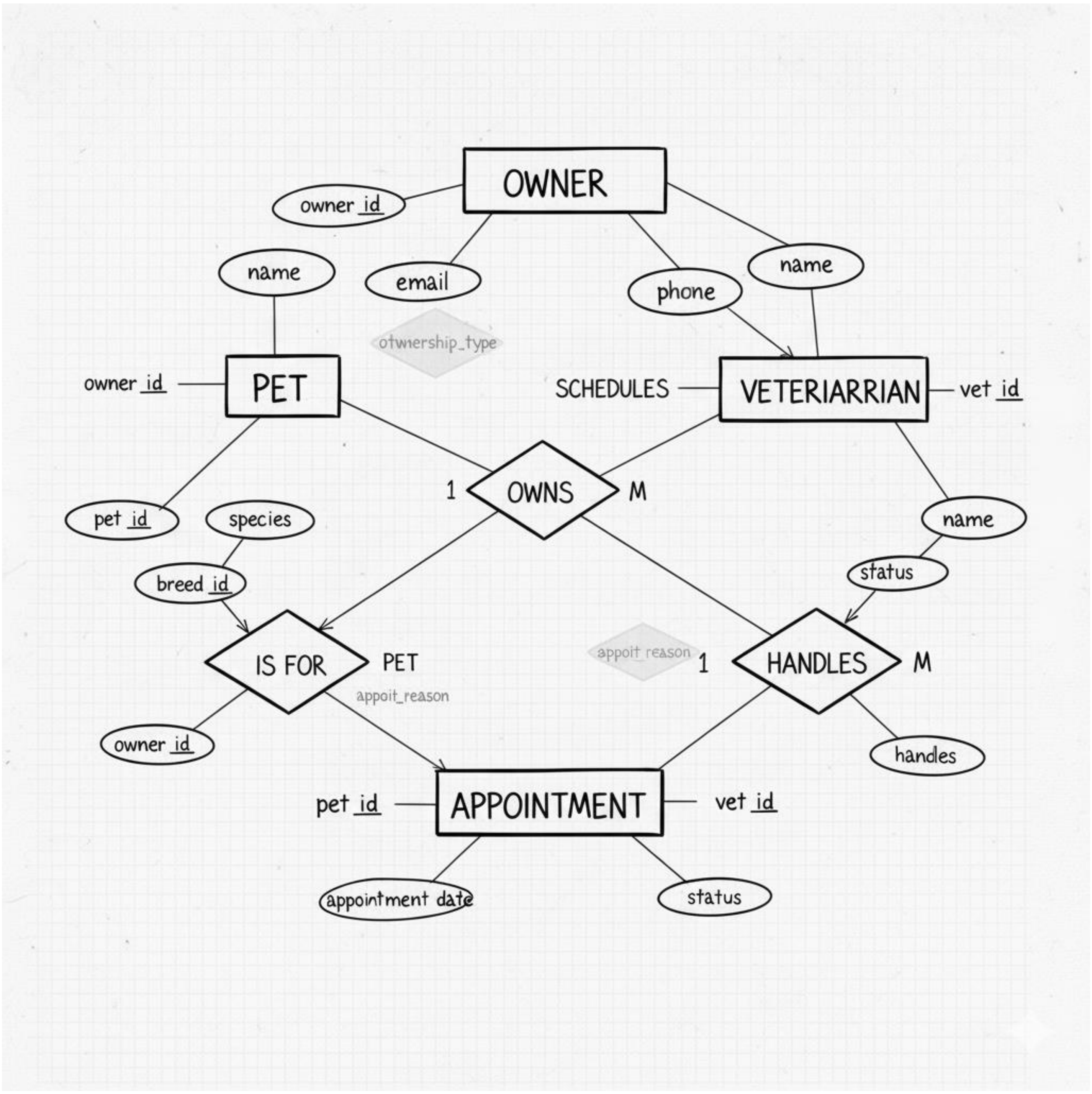
2 - Dr.Carlos Martinez (Surgery and orthopedics)
1 - Dr.Emily Anderson (General Practice)
4 - Dr.Maria Gracia (Dermatology&allergies)
3 - Dr.Robert taylor (internal medication)

Appointment Type:

Problem Description: nausea

Date (YYYY-MM-DD):

7.ER DIAGRAM


OWNER
owner id
email
phone
name
otwnership_type
PET
name
owner id
pet id
species
breed id
SCHEDULES
VETERIARRIAN
vet id
name
status
1
OWNS
M
IS FOR
PET
appoit_reason
owner id
appoit reason
1
HANDLES
M
handles
APPOINTMENT
pet id
vet id
appointment date
status

. REFERENCE

1. Oracle Corporation. Java Swing Documentation - Building Graphical User Interfaces. Oracle Java SE Documentation.
2. MySQL Documentation Team. MySQL 8.0 Reference Manual - SQL Statements and Database Management. MySQL Official Documentation.
3. Bloch, J. (2018). Effective Java (3rd ed.). Addison-Wesley Professional.
4. Freeman, E., Robson, E., Bates, B., & Sierra, K. (2020). Head First Design Patterns (2nd ed.). O'Reilly Media.