PET ADOPTION CENTER

A MINI-PROJECT REPORT *Submitted by*

HEMASHREE R 240701190
HEMALATHA ST 240701189

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

CHENNAI

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **“PET ADOPTION CENTER”** is the bonafide work of **“HEMASHREE R and HEMALATHA ST”** who carried out the project work under my supervision.

SIGNATURE	
Dr. V. JANANEE	
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 **EXTERNAL EXAMINER**

ABSTRACT

The Pet Adoption Center Management System is designed to simplify the process of adopting pets and managing adoption requests. The project provides an organized platform that connects people who want to adopt with animals in need of a home. This database management system enables the admin to manage pet records, rescue locations, adoption requests, and user feedback efficiently. Users can browse available pets, send adoption requests, and submit feedback. The goal of the system is to make the pet adoption process more transparent, organized, and user-friendly

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The Pet Adoption Center Management System helps people find and adopt pets easily. It provides essential information about available pets, their breed, age, and rescue location.The system is designed to bridge the gap between pet shelters and potential adopters, making the adoption process smooth and efficient

1.2 SCOPE OF THE WORK

The system helps manage adoption data efficiently and ensures proper tracking of pets,users, and rescue locations. It offers quick access to pet details and simplifies adoption management for both admin and users.

1.3 PROBLEM STATEMENT

Many animal shelters face difficulties managing their data and adoption requests manually. The lack of a centralized system causes delays, mismanagement, and incomplete information sharing. This project aims to overcome those issues through a digital, database-driven solution.to people.

1.4 AIM AND OBJECTIVES OF THE PROJECT

- To create a system that manages pet adoption records efficiently.

- To maintain a database of pets, rescue locations, and adoption.

- To provide a user-friendly interface for adopters and administrators.l

- To encourage more adopt

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel i5
Memory Size	:	8GB (Minimum)
HDD	:	1 TB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 11
Front – End	:	Java Swing
Back - End	:	MySql
Language	:	Java,SQL

CHAPTER 3

MODULE DESCRIPTION

This application consists of two modules. When the program runs, it will ask for a confirmation to the login window. The person who interacts can login as an Administrator or as a User. The description of the modules are as follows:

1. Admin login

The admin logs in with a valid email and password. The admin can view and manage all adoption requests, accept or reject them, add new pets, view rescue locations, and check feedback from users.

2.User login

The user logs in using basic credentials to browse available pets, view rescue locations,contact the admin, send feedback, and submit requests for adopting a pet.

CHAPTER 4

4. 1 Database Connection Code

```
package com.petadoption.backend;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class DBConnection {
    private static final String URL = "jdbc:oracle:thin:@localhost:1521/FREEPDB1";
    private static final String USER = "system"; // Replace with your Oracle username
    private static final String PASSWORD = "Hema@29032007";
    public static Connection getConnection() {
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            return DriverManager.getConnection(URL, USER, PASSWORD);
        } catch (ClassNotFoundException e) {
            System.out.println("Oracle JDBC Driver not found.");
            e.printStackTrace();
        } catch (SQLException e) {
            System.out.println("Connection failed.");

            e.printStackTrace();

        }

        return null;

    }

}
```

Sample 1

Sample 2 depicts the AdoptionRequestDAO class, which handles all operations related to pet adoption requests in the Pet Adoption Center Management System.It allows users to submit new adoption requests and enables the administrator to view, approve, or reject pending requests.This class ensures proper data integrity by updating the request status in the database and recording each request with the current date.

```
package com.petadoption.backend;

import java.sql.*;

import java.util.*;

public class AdoptionRequestDAO {
// Submit a new adoption request
    public static boolean submitRequest(int userId, int petId, String message) {

        try (Connection conn = DBConnection.getConnection();

             PreparedStatement ps = conn.prepareStatement(

                "INSERT INTO AdoptionRequests (user_id, pet_id, message, status, request_date)
VALUES (?, ?, ?, ?, ?)")) {
```

```
            ps.setInt(1, userId);

            ps.setInt(2, petId);

            ps.setString(3, message);

            ps.setString(4, "Pending");

            ps.setDate(5, new java.sql.Date(System.currentTimeMillis())); // current date

            ps.executeUpdate();

           return true;           }
catch (Exception e) {
            e.printStackTrace();

            return false;

        }

    }

    // Get all pending adoption requests

    public static List<Map<String, String>> getPendingRequests() {
List<Map<String, String>> requests = new ArrayList<>();
        try (Connection conn = DBConnection.getConnection();

             Statement stmt = conn.createStatement();

            ResultSet rs = stmt.executeQuery("SELECT * FROM AdoptionRequests  WHERE status =
'Pending'")) {

            while (rs.next()) {
                Map<String, String> req = new HashMap<>();

                req.put("id", String.valueOf(rs.getInt("id")));

                req.put("user_id", String.valueOf(rs.getInt("user_id")));

                req.put("pet_id", String.valueOf(rs.getInt("pet_id")));

                req.put("message", rs.getString("message"));

                req.put("request_date", String.valueOf(rs.getDate("request_date")));

                requests.add(req);

            }

        } catch (Exception e) {

            e.printStackTrace();

        }

        return requests;

    }

    // Approve a request

    public static void approveRequest(int requestId) {

        updateStatus(requestId, "Approved");

    }

    // Reject a request
    public static void rejectRequest(int requestId) {
        updateStatus(requestId, "Rejected");
    }
    // Internal method to update request status
    private static void updateStatus(int requestId, String status) {
        try (Connection conn = DBConnection.getConnection();                PreparedStatement ps =
conn.prepareStatement(
                "UPDATE AdoptionRequests SET status = ? WHERE id = ?")) {
            ps.setString(1, status);
            ps.setInt(2, requestId);
            ps.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Sample 2

Sample 3 depicts the Adoption model class, which acts as a data structure for representing each pet adoption request in the Pet Adoption Center system.

```
}
```

This class holds the necessary details such as the pet ID, user ID, date of request, adoption status, and any message from the user.

It helps in encapsulating adoption-related data and transferring it efficiently between different layers of the application**.**

```java
package com.petadoption.model;


import java.util.Date;


public class Adoption {


    private int petId;


    private int userId;


    private Date requestDate;


    private String status;


    private String message;


    public Adoption(int petId, int userId, Date requestDate, String status, String message) {


        this.petId = petId;


        this.userId = userId;


        this.requestDate = requestDate;


        this.status = status;


        this.message = message;


    }
    public int getPetId() { return petId; }

    public int getUserId() { return userId; }

    public Date getRequestDate() { return requestDate; }

    public String getStatus() { return status; }

    public String getMessage() { return message; }

}
```

Sample 3

Sample 4 depicts the AdminDashboard class, which provides the graphical user interface (GUI) for the administrator in the Pet Adoption Center Management System.

It allows the admin to easily navigate through key functionalities such as adding new pets and reviewing adoption requests.

This interface is implemented using Java Swing, ensuring a simple and user-friendly desktop layout for efficient management.l package com.petadoption.ui;

```
import javax.swing.*;

public class AdminDashboard {

    public void show() {

        JFrame frame = new JFrame("Admin Dashboard");

        frame.setSize(400, 300);

        frame.setLayout(null);
        JButton addPetBtn = new JButton("Add Pet");

        addPetBtn.setBounds(120, 40, 150, 40);

        addPetBtn.addActionListener(e -> new PetForm().show());
JButton viewRequestsBtn = new JButton("Adoption Requests");
        viewRequestsBtn.setBounds(120, 100, 150, 40);

        viewRequestsBtn.addActionListener(e -> new AdoptionReviewPage().show());

        frame.add(addPetBtn);

        frame.add(viewRequestsBtn);

        frame.setVisible(true);

    }

}
```

Sample 4

- **Main class**

Sample 6 depicts the **Main class**, which serves as the entry point for the Pet Adoption Center Management System.

It initializes the application by launching the RoleSelectionPage, where users can choose to log in either as an Administrator or a User.

This class acts as the central starting module that connects all UI and backend components of the system.

```
package com.petadoption;

import com.petadoption.ui.RoleSelectionPage;
```

```
public class Main {

    public static void main(String[] args) {

        new RoleSelectionPage().show();

    }

}
```

CHAPTER 5

SCREEN SHOTS

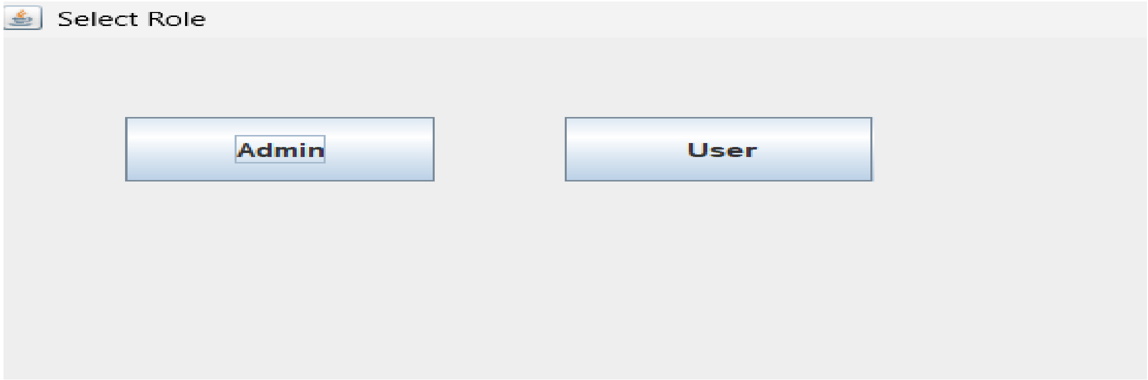
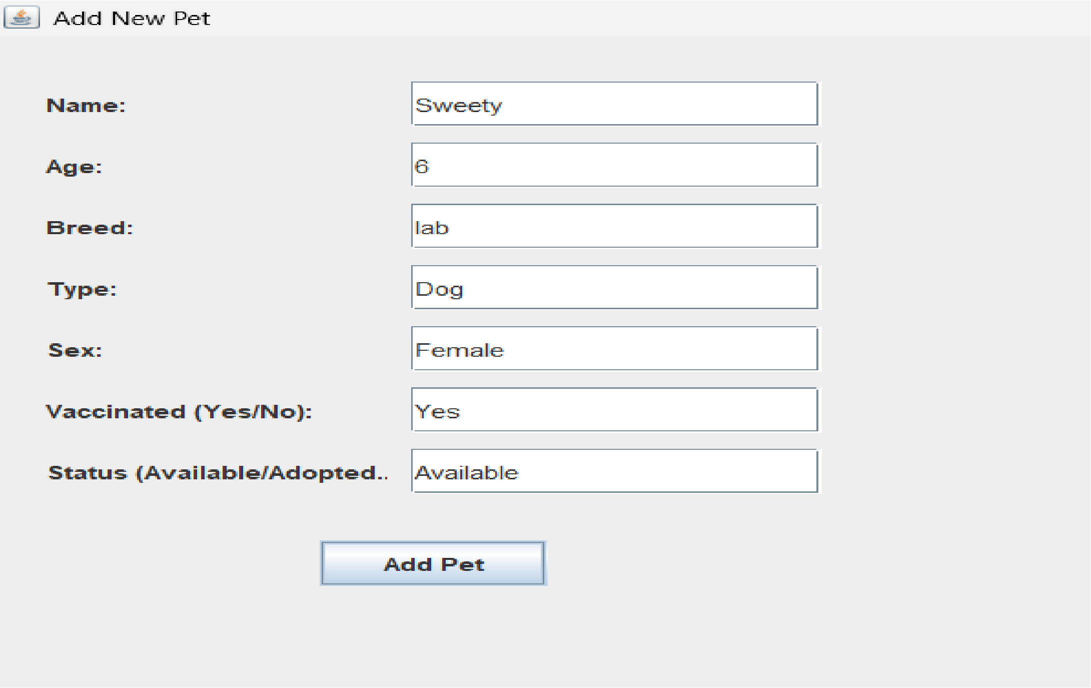


Fig 5.1 Role Selection Page



Name: Sweety

Age: 6

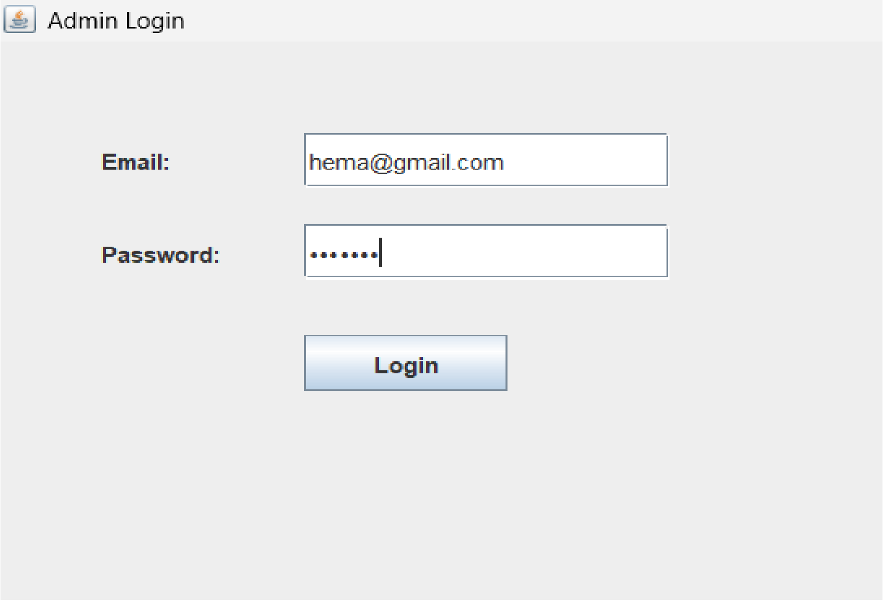
Breed: lab

Type: Dog

Sex: Female

Vaccinated (Yes/No): Yes

Status (Available/Adopted.. Available

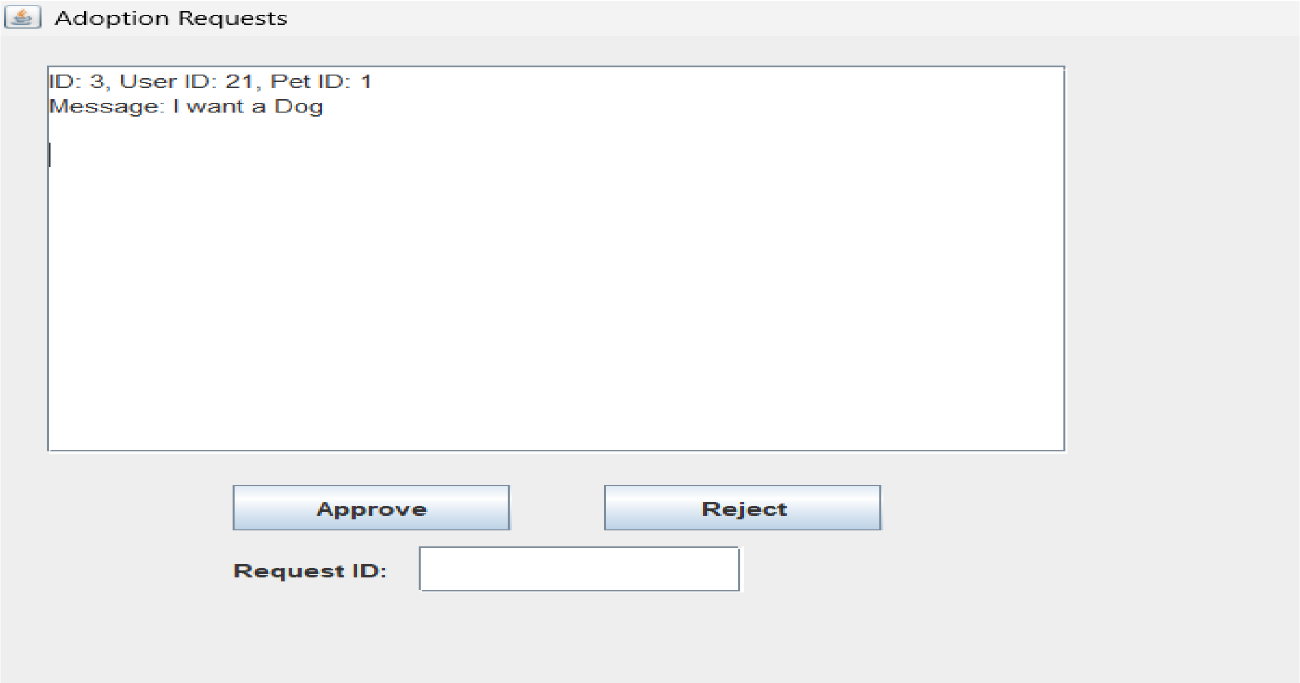


Email: hema@gmail.com

Password: •••••••

Fig 5.2 Administrator Login Page

Fig 5.3 Add New Pet Interface



ID: 3, User ID: 21, Pet ID: 1
Message: I want a Dog

Fig 5.4 Adoption Request Management Page

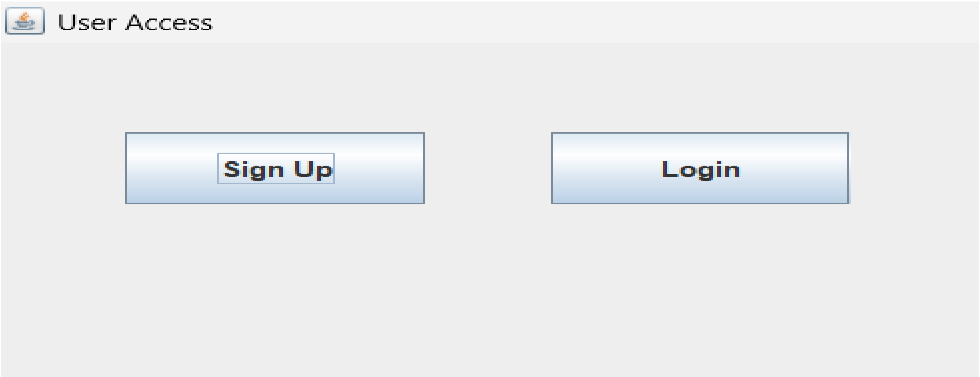


Fig 5.5 User Access Page

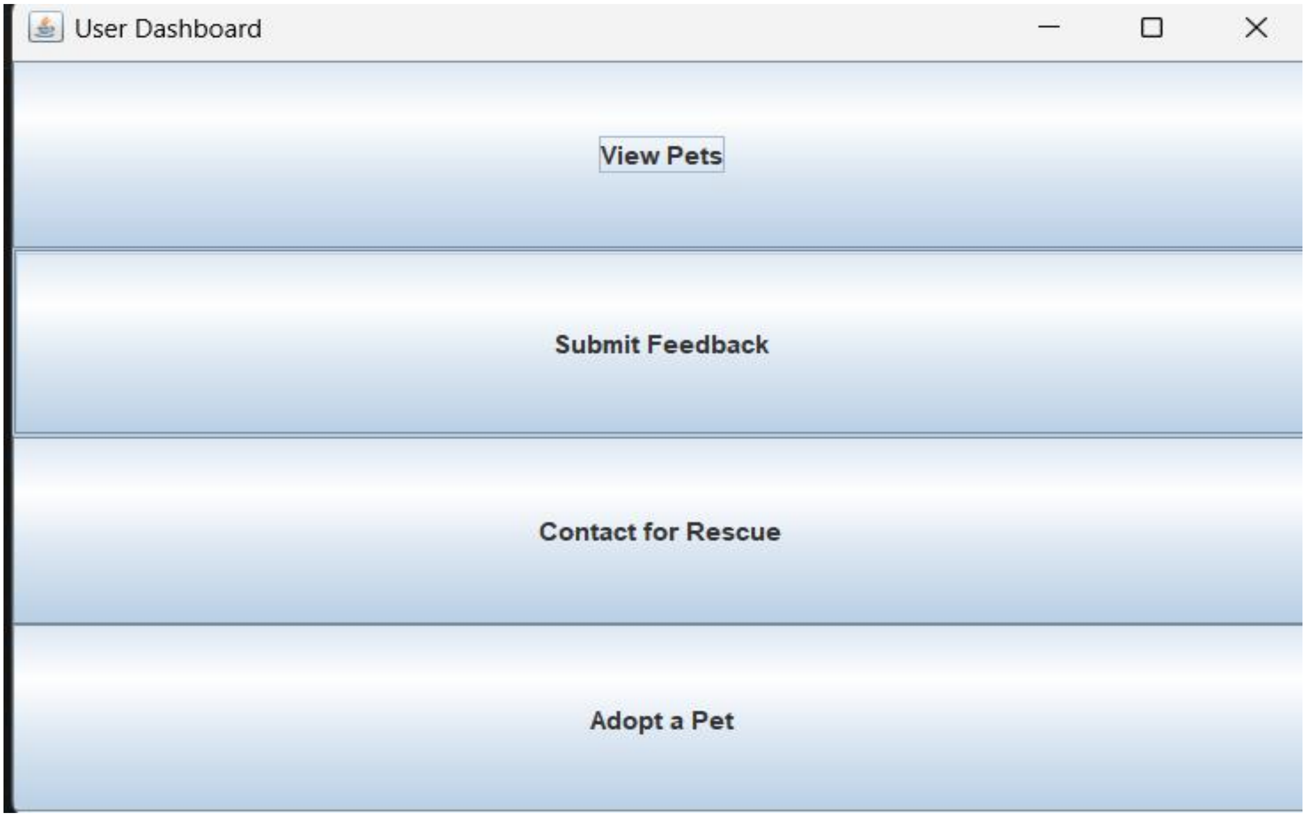
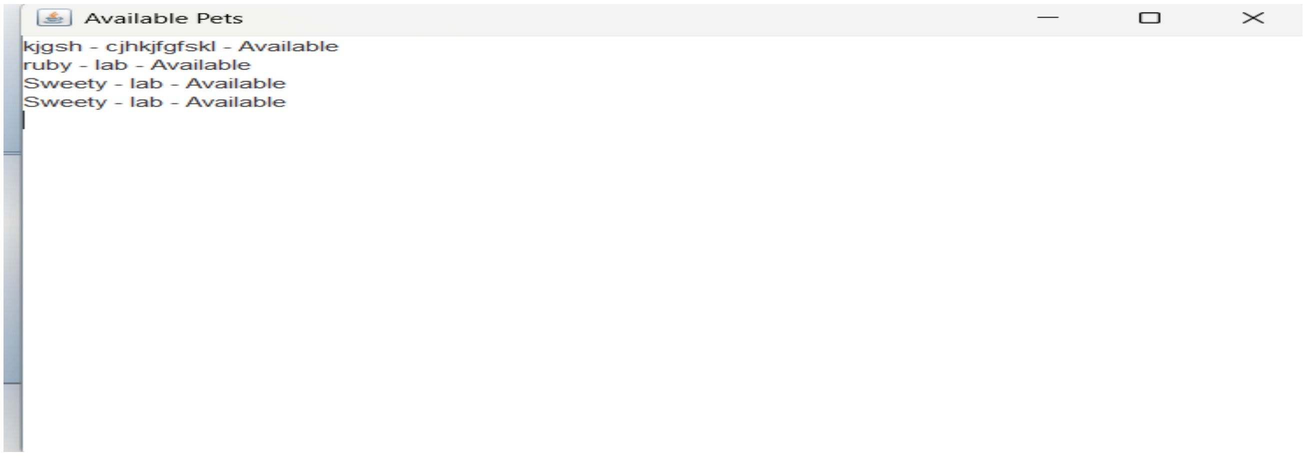


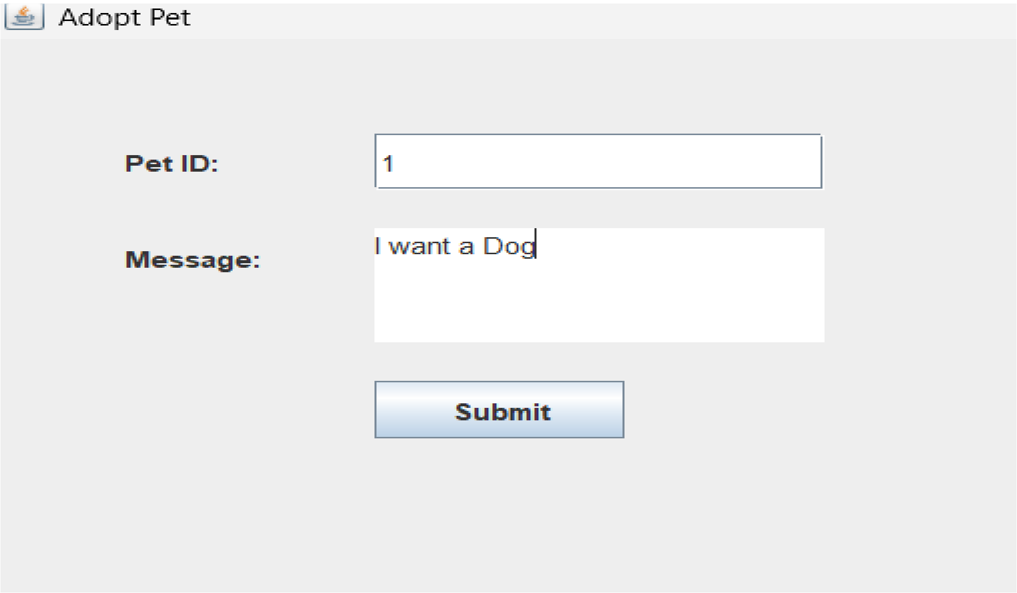
Fig 5.6 User Dashboard Interface

Fig 5.6 Database creation



```
kjgsh - cjhkjfgfskl - Available
ruby - lab - Available
Sweety - lab - Available
Sweety - lab - Available
```

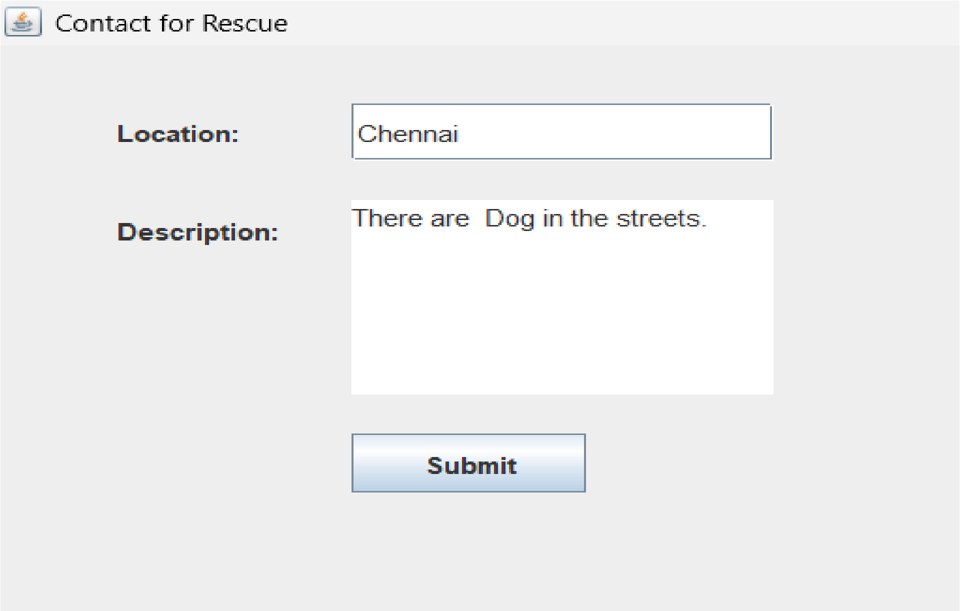
Fig 5.7 View Available Pets Page



Pet ID: 1

Message: I want a Dog

Fig 5.8 Adopt Pet Request Form

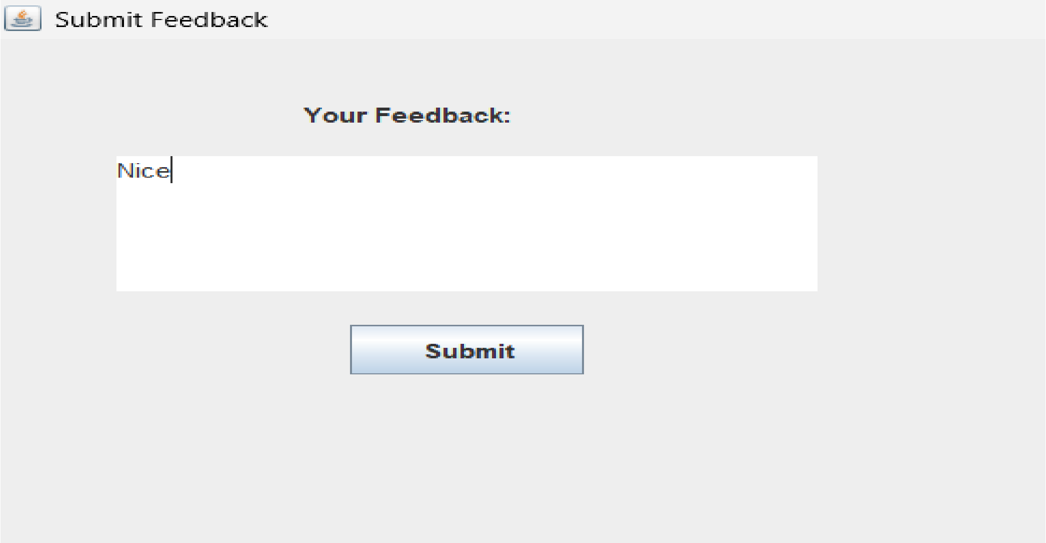


Contact for Rescue

Location: Chennai

Description: There are Dog in the streets.

Fig 5.9 Rescue Request Submission Form



Submit Feedback

Your Feedback:

Nice

Fig 5.10 Feedback Submission Form

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

In such a way, with the help of our project, customers will be able to check the list of bookings and can register themselves to avail a cab. The booking system clearly represents the available data of the customers for booking and the number of bookings using a booking log and management becomes easier. In future people will be able to book cabs according to the data available in the system and with respect to the availability. Hence this project makes the user and other advantages to be benefitted in all possible ways.