Sri Eshwar College of Engineering
An Autonomous Institution
Affiliated to Anna University, Chennai

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Academic Year

2023 – 2024 (ODD Semester)

MINI PROJECT REPORT

R19CS652 Database Technologies

R19CS203 Object Oriented Programming using Java

PET ADOPTION SYSTEM

Submitted by,

[22EC067]	**Logitha Luckshmi V J**
[22EC086]	**Nagalakshmi S**
[22EC120]	**Raksana Sri S**

Mentored by,

Mrs. E.Saranya,
Assistant Professor,
Department of CSE

Dr. Rajesha Narasimha Murthy ,
Associate Professor,
Department of ECE

ABSTRACT

The Pet Adoption System is a JavaFX-based desktop application aiming to streamline the pet adoption process for both pet sellers and seekers. This platform provides distinct functionalities for each user type. Pet sellers can securely log in, manage their listed pets by adding, removing, and viewing them through unique pet IDs, and receive real-time updates on adoption applications. On the other hand, pet seekers can create accounts, explore available pets categorized as dogs, cats, and others, submit adoption applications, and utilize the platform as a communication hub for interactions with sellers.

Implemented with JavaFX, Java, and MySQL for backend support, the system promotes a user-friendly experience and responsible pet ownership. Future enhancements may include advanced features such as detailed pet health records for sellers and integration with online payment gateways for seekers. This project report outlines the development process, key features, and potential improvements of the Pet Adoption System, emphasizing its contribution to simplifying the adoption journey for both pet sellers and seekers.

CHAPTER 1
INTRODUCTION

1.1 OBJECTIVES

- **Simplify Adoption Process:** Develop a user-friendly platform to streamline the pet adoption process, providing an intuitive interface for both pet sellers and seekers.
- **User Authentication and Profiles:** Implement secure user authentication, allowing pet sellers and seekers to create accounts, log in securely, and manage their profiles.
- **Efficient Pet Management:** Enable pet sellers to efficiently manage their listed pets by adding, removing, and viewing them using unique pet IDs.
- **Enhance Pet Exploration:** Provide pet seekers with categorized options for exploring pets, such as dogs, cats, and others, fostering a seamless and enjoyable browsing experience.

1.2 SCOPE OF THE PROJECT

The Pet Adoption System project aims to create a user-friendly platform for both pet sellers and seekers. Pet sellers can efficiently manage their listings, add, remove, and view pets with a unique identification system. Seekers, on the other hand, benefit from a categorized exploration experience, simplifying their search for pets like dogs, cats, and others. Seekers can utilize the platform to explore pets and initiate the adoption process seamlessly, fostering effective communication with sellers. Developed using JavaFX and integrated with MySQL, the system prioritizes visual appeal, scalability, and potential future enhancements, contributing to a streamlined and responsible pet adoption experience.

CHAPTER 2
SYSTEM ANALYSIS AND SPECIFICATION

2.1 PROBLEM DESCRIPTION

The Pet Adoption System project addresses prevailing challenges in the pet adoption ecosystem, characterized by fragmented processes, insufficient tools for pet sellers, ineffective communication, intricate pet exploration, and technological constraints. Through the development of a pioneering JavaFX desktop application seamlessly integrated with MySQL, the project aspires to transform the pet adoption landscape. Its goals encompass centralizing and streamlining the adoption process, providing robust tools for pet sellers, establishing real-time communication channels, simplifying pet exploration for seekers, and leveraging modern technologies to enhance the overall user experience. This initiative holds the promise of surmounting existing obstacles, presenting an efficient, user-centric platform poised to revolutionize and promote responsible pet adoption practices.

2.2 FUNCTIONAL REQUIREMENT –

SOFTWARE AND HARDWARE REQUIREMENT

Software Requirements:

1. **Java Development Kit (JDK):** The project, being developed in JavaFX, requires the latest JDK for Java application development.
2. **JavaFX SDK:** To utilize the JavaFX library for creating the graphical user interface.
3. **Integrated Development Environment (IDE):** A recommended IDE such as IntelliJ IDEA or Eclipse for Java development, providing code editing, debugging, and build tools.

4. **MySQL Database:** For storing user profiles, pet information, and adoption-related data.
5. **MySQL Connector/J:** JDBC driver for connecting the Java application to the MySQL database.

Hardware Requirements:

1. **Processor:** A multi-core processor (e.g., Intel Core i5 or AMD equivalent) for efficient application performance.
2. **RAM:** At least 8GB of RAM to handle the application's processing and memory requirements.
3. **Storage:** Adequate storage space for the application, development tools, and potential pet data storage.
4. **Graphics Card:** A dedicated graphics card is not mandatory but can enhance the performance of JavaFX-based graphical interfaces.
5. **Operating System:** The system should support the chosen Java development environment and be compatible with JavaFX.

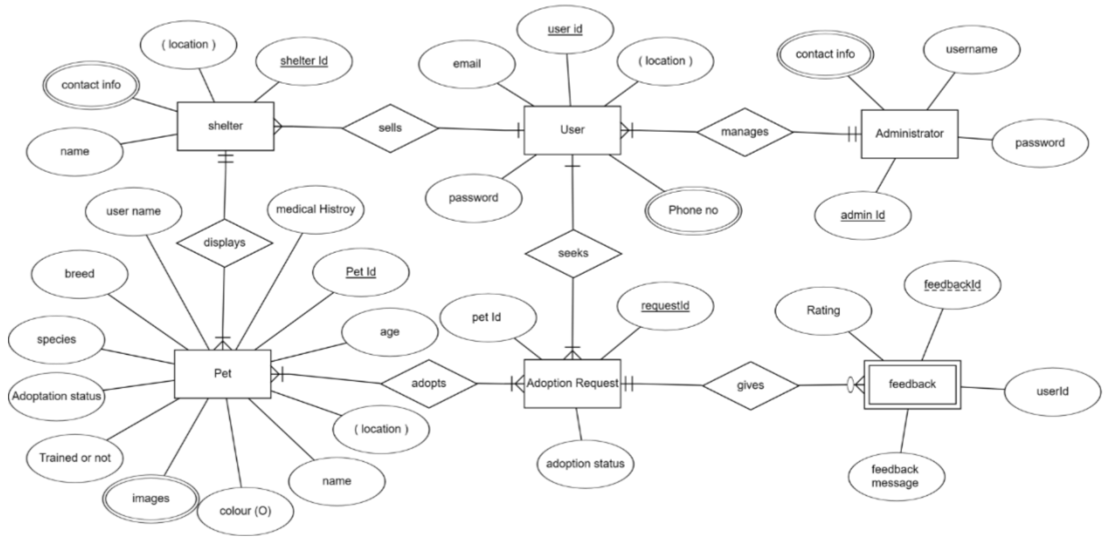
2.3 NON FUNCTIONAL REQUIREMENT

1. **Usability and Accessibility:**
    - Ensure an intuitive and visually appealing user interface using JavaFX for both pet sellers and seekers.
    - Guarantee cross-device compatibility and accessibility across various screen sizes.
2. **Performance and Scalability:**
    - Achieve low-latency response times for critical actions like pet listing updates and communication.
    - Design the system to scale efficiently, accommodating a growing user base and pet data.
3. **Reliability and Security:**
    - Maintain 24/7 availability with minimal downtime for maintenance.
    - Implement robust user authentication and data encryption to safeguard sensitive information.

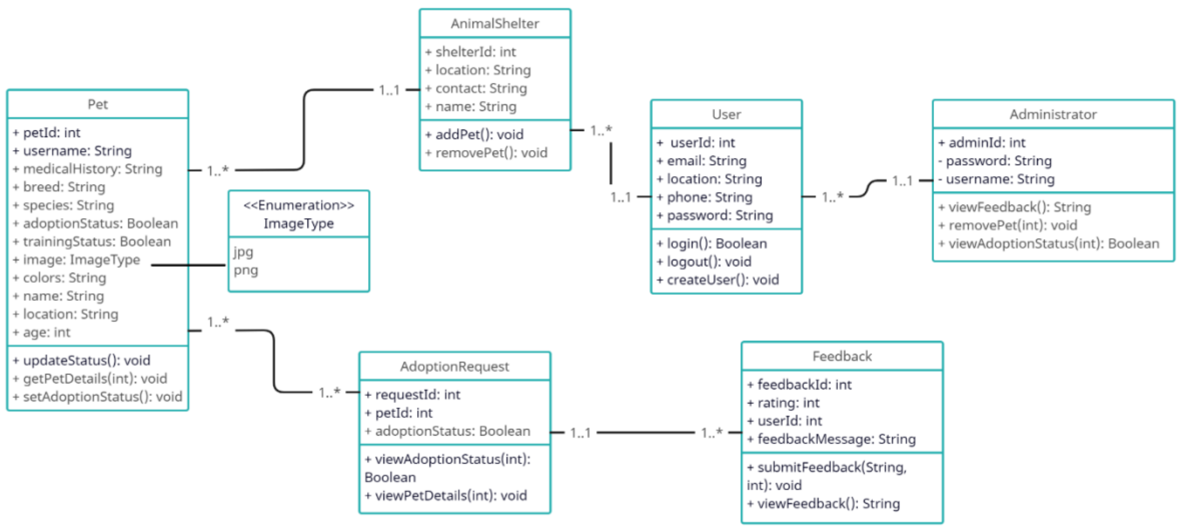
4. **Maintainability and Compatibility:**
    - Ensure well-documented and readable code for ease of maintenance.
    - Design the system for compatibility with major web browsers and various operating systems.
5. **Documentation and Fault Tolerance:**
    - Provide comprehensive documentation, including installation guides and user manuals.
    - Build fault-tolerant mechanisms to handle errors gracefully and maintain data integrity.

CHAPTER 3
SYSTEM DESIGN

3.1 ER DIAGRAM



3.2 SCHEMA DIAGRAM



CHAPTER 4
PROPOSED SOLUTION

4.1 USER INTERFACE DESIGN

The user interface for the Pet Adoption System was meticulously crafted using JavaFX and Scene Builder. This powerful visual layout tool facilitated an efficient and intuitive design process, allowing for the seamless arrangement of UI components. Leveraging the drag-and-drop interface design, the resulting desktop application boasts an intuitive dashboard for pet sellers, categorized pet listings for seekers, and a real-time communication hub. The use of Scene Builder not only expedited development but also ensured a visually appealing and user-friendly experience, enhancing the overall pet adoption journey for both sellers and seekers.

4.2 CLASS CONSTRUCTION

The Pet Adoption System employs a structured class construction featuring key entities such as "Seller," "Seeker," "AddPet," "RemovePet," and "ViewPet." The "Seller" and "Seeker" classes manage user profiles, authentication, and interaction. The "AddPet" class facilitates the addition of pets to the system, ensuring detailed information is captured. Conversely, the "RemovePet" class enables secure removal of listed pets. The "ViewPet" class orchestrates the presentation of pet details, implementing a streamlined exploration experience. This modular class design enhances code organization, scalability, and maintainability, contributing to the system's efficiency in connecting pet seekers with suitable pets.

4.3 DATABASE CREATION

In the Pet Adoption System, MySQL is seamlessly integrated with Java to establish a robust and scalable database. The database creation process involved defining tables to store user profiles, pet information, and adoption-related data. Utilizing JDBC connectivity, Java interacts with MySQL, ensuring seamless communication between the application and the database. The structured database design facilitates efficient data retrieval, storage, and modification. This integration enhances the overall functionality, enabling secure user authentication, real-time updates, and streamlined pet management. The MySQL database, coupled with Java, forms a foundational element, contributing to the reliability and performance of the Pet Adoption System.

CHAPTER 5
PROJECT DESCRIPTION

5.1 MODULE DESCRIPTION

1. **Login and Registration:**
   - <u>Objective</u>: Allows users to log in or register based on their roles as sellers or seekers.
   - <u>Features:</u> Secure authentication, role-based redirection, and user-friendly registration.
2. **Seller Homepage:**
   - <u>Objective</u>: Serves as the central hub for sellers to manage their profiles and pet-related activities.
   - <u>Features:</u>
     - *Add Pet:* Enables sellers to add new pets with detailed information.
     - *Remove Pet:* Facilitates the secure removal of listed pets.
     - *View Pet:* Presents a comprehensive view of seller-listed pets.
3. **Seeker Homepage:**
   - <u>Objective:</u> Acts as the starting point for seekers to explore available pets.
   - <u>Features:</u>
     - *Dog Page:* Displays a categorized list of available dogs for adoption.
     - *Cat Page:* Presents a categorized list of available cats for adoption.
     - *Other Pets Page:* Offers a platform to explore other types of pets.

Each module is designed to enhance the user experience, providing sellers and seekers with tailored functionalities aligned with their respective roles and preferences within the Pet Adoption System.

5.2 JDBC CONNECTIVITY

1. **Integration with MySQL**:

The JDBC connection forms the bridge between the Java-based Pet Adoption System and the MySQL database. It enables the retrieval, storage, and modification of essential data, such as user profiles, pet information, and adoption-related details.

2. **Secure User Authentication:**

The JDBC connection is crucial for secure user authentication during the login process. It validates user credentials against stored data in the MySQL database, ensuring a secure and authenticated entry into the system.

3. **Efficient Pet Management:**

JDBC facilitates efficient pet management for adopters, enabling activities like adding, removing, and viewing pets. The connection ensures seamless communication, allowing sellers to interact with their pet’s data stored in the MySQL database.

4. **Streamlined Data Retrieval:**

With JDBC, the Pet Adoption System efficiently retrieves data from the MySQL database. Seekers can explore categorized pet listings, such as dogs, cats, and other pets, ensuring a streamlined and responsive user experience.

5. **Real-time Updates and Interaction:**

The JDBC connection enables real-time updates for sellers regarding adoption applications and facilitates communication between sellers and seekers. This ensures that interactions and data exchanges occur seamlessly, enhancing the overall user experience in the Pet Adoption System.

CHAPTER 6

IMPLEMENTATION

```
package application;

import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.RadioButton;
import javafx.scene.control.TextField;
import javafx.scene.layout.BorderPane;
import javafx.stage.Stage;

public class SampleController {
	@FXML
	private Button 
Login,back,addpet,register,addbtn,homeback,view,dog,cat,others;
	@FXML
	private Button SignUp;
	@FXML
	private Label 
label,myLabel,signuplabel,addlabel,breedlabel,breedlabel1,label1,label2,label3,lab
el4;
	@FXML
	TextField txtuser = new TextField();
```

```
@FXML
TextField txtphone = new TextField();
@FXML
TextField txtloc = new TextField();
@FXML
TextField txtnewuser = new TextField();
@FXML
TextField txtemail = new TextField();
@FXML
TextField txtnewpass = new TextField();
@FXML
private RadioButton optadopt;
@FXML
private RadioButton optsell;


@FXML
TextField txtPetname = new TextField();
@FXML
TextField txtbreed = new TextField();
@FXML
TextField txtAge = new TextField();
@FXML
TextField txtColour = new TextField();
@FXML
TextField txtlocation = new TextField();

@FXML
TextField txtSpecies = new TextField();
@FXML
private PasswordField txtpass;

private Stage stage;
private Scene scene;
Connection con;
PreparedStatement pst;
ResultSet rs;

@FXML
```

```
    void login(ActionEvent event) throws IOException {
            String uname = txtuser.getText();
        String pass = txtpass.getText();

        if (uname.isEmpty() && pass.isEmpty()) {
            label.setText("Username and Password should not be empty");
        } else if (uname.isEmpty()) {
            label.setText("Username should not be empty");
        } else if (pass.isEmpty()) {
            label.setText("Password should not be empty");
        } else {
            try {
                Class.forName("com.mysql.jdbc.Driver");
                con = DriverManager.getConnection("jdbc:mysql://localhost/pas",
"root", "root");
                pst = con.prepareStatement("select * from user where username =
? and password = ?");
                pst.setString(1, uname);
                pst.setString(2, pass);
                rs = pst.executeQuery();
                if (rs.next()) {
                    pst = con.prepareStatement("select role from user where
username = ?");
                    pst.setString(1, uname);
                    ResultSet rsRole = pst.executeQuery();

                    if (rsRole.next()) {
                        String role = rsRole.getString("role");

                        if ("Adopt".equals(role)) {
                            BorderPane root = (BorderPane)
FXMLLoader.load(getClass().getResource("Homepage1.fxml"));
                            stage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
                            scene = new Scene(root);
                            stage.setScene(scene);
                            stage.show();
                        } else {
                            BorderPane root = (BorderPane)
FXMLLoader.load(getClass().getResource("Homepage.fxml"));
```

```
                              stage = (Stage) ((Node)
event.getSource()).getScene().getWindow();
                              scene = new Scene(root);
                              stage.setScene(scene);
                              stage.show();
                          }
                      }
                  } else {
                      label.setText("Invalid username or password");
                      txtuser.clear();
                      txtpass.clear();
                  }
              } catch (ClassNotFoundException | SQLException e) {
                  e.printStackTrace();
              }
          }
      }

      @FXML
      void register(ActionEvent event) throws IOException {
             String newuname = txtnewuser.getText();
             String newpassword = txtnewpass.getText();
             String newloc = txtloc.getText();
             String newph = txtphone.getText();
             String newemail = txtemail.getText();
             if(newuname == ""||newpassword == ""||newloc == ""||newemail ==
""||myLabel.getText()==""||newph =="") {
                    signuplabel.setText("All the Fields are compulsory");
             }
             else if(newph.length()!=10)
                    signuplabel.setText("Enter Valid phone number");
             else {
                    try {
                           Class.forName("com.mysql.jdbc.Driver");
                           con =
DriverManager.getConnection("jdbc:mysql://localhost/pas","root","Jeeva2212.");
                           String q = "insert into user (username, password,
location, mobile_no, email, role) values (?, ?, ?, ?, ?, ?)";
                           pst = con.prepareStatement(q);
                           pst.setString(1, newuname);
```

```
                        pst.setString(2, newpassword);
                        pst.setString(3, newloc);
                        pst.setString(4, newph);
                        pst.setString(5, newemail);
                        pst.setString(6, myLabel.getText());
                        int n = pst.executeUpdate();
                        if(n>0) {
                              if(myLabel.getText()=="Adopt") {
                                    BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("Homepage1.fxml"));
                                    stage =
(Stage)((Node)event.getSource()).getScene().getWindow();
                                    scene = new Scene(root);
                                    stage.setScene(scene);
                                    stage.show();
                              }
                              else {
                                    BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("Homepage.fxml"));
                                    stage =
(Stage)((Node)event.getSource()).getScene().getWindow();
                                    scene = new Scene(root);
                                    stage.setScene(scene);
                                    stage.show();
                              }
                        }
                  } catch (ClassNotFoundException | SQLException e) {
                        signuplabel.setText("Username "+newuname+" already
taken.");
                        txtnewuser.clear();
                        e.printStackTrace();
                  }

            }
      }
      @FXML
      void add(ActionEvent event) throws IOException {
            String petname = txtPetname.getText();
            String breed = txtbreed.getText();
            String age = txtAge.getText();
```

```
             String col = txtColour.getText();
             String location = txtlocation.getText();
             String Species = txtSpecies.getText();
             String user = txtuser.getText();

             if(petname == ""||breed == ""||age == ""||col ==
""||Species==""||location ==""||user=="") {
                    breedlabel.setText("All the Fields are compulsory");
                    breedlabel1.setText("");
             }

             else {
                    try {
                           Class.forName("com.mysql.jdbc.Driver");
                           con =
DriverManager.getConnection("jdbc:mysql://localhost/pas","root","Jeeva2212.");
                           String q = "insert into pet
(petname,breed,age,colour,location,species,username) values (?, ?, ?, ?, ?, ?,?)";
                           pst =
con.prepareStatement(q,Statement.RETURN_GENERATED_KEYS);
                           pst.setString(1, petname);
                           pst.setString(2, breed);
                           pst.setString(3, age);
                           pst.setString(4, col);
                           pst.setString(5, location);
                           pst.setString(6, Species);
                           pst.setString(7, user);
                           pst.executeUpdate();
                           try(ResultSet rs = pst.getGeneratedKeys()){
                           if(rs.first()) {
                                  breedlabel.setText("");
                                         breedlabel1.setText(petname+" with petID
"+rs.getLong(1)+" is added Successfully!!");
                                         txtPetname.clear();
                                         txtbreed.clear();
                                         txtAge.clear();
                                         txtColour.clear();
                                         txtlocation.clear();
                                         txtSpecies.clear();
                                         txtuser.clear();
```

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

        }
}

@FXML
void removepet(ActionEvent event) throws SQLException,
ClassNotFoundException {
        String id = txtuser.getText();
        Class.forName("com.mysql.jdbc.Driver");
        con =
DriverManager.getConnection("jdbc:mysql://localhost/pas","root","Jeeva2212.");

        String q = "Delete from pet where petid=?";
        try {
                pst = con.prepareStatement("select petname from pet where
petid = ?");
                pst.setInt(1, Integer.parseInt(id));
         ResultSet rspet = pst.executeQuery();

         if (rspet.next()) {
             String pname = rspet.getString("petname");
                pst = con.prepareStatement(q);
                pst.setInt(1, Integer.parseInt(id));
                pst.executeUpdate();
                breedlabel.setText( pname+" removed");
         }
         else {
                breedlabel.setText("Invalid pet id");
         }
        } catch (SQLException e) {
                breedlabel.setText("Enter valid id");
                e.printStackTrace();
        }

}
@FXML
```

```
        void viewpetdetails(ActionEvent event) throws SQLException,
ClassNotFoundException {
                String id = txtuser.getText();
                Class.forName("com.mysql.jdbc.Driver");
                con =
DriverManager.getConnection("jdbc:mysql://localhost/pas","root","Jeeva2212.");
                try {
                        pst = con.prepareStatement("select petname,breed,age,species
from pet where petid = ?");
                        pst.setInt(1, Integer.parseInt(id));
                  ResultSet rspet = pst.executeQuery();

                  if (rspet.next()) {
                      String pname = rspet.getString("petname");
                      String pbreed = rspet.getString("breed");
                      String page = rspet.getString("age");
                      String pspecies = rspet.getString("species");
                      label1.setText("PET NAME: "+pname);
                      label2.setText("SPECIES: "+pspecies);
                      label3.setText("AGE: "+page);
                      label4.setText("BREED: "+pbreed);
                  }
                  else {
                        breedlabel.setText("Invalid pet id");
                        label1.setText("");
                        label2.setText("");
                        label3.setText("");
                        label4.setText("");

                  }
                } catch (SQLException e) {
                        breedlabel.setText("Enter valid id");
                        e.printStackTrace();
                }

        }



        @FXML
        void switchaddpet(ActionEvent event) throws IOException {
```

```
		BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("addpage.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();

	}
	@FXML
	void switchviewpet(ActionEvent event) throws IOException {
		BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("viewpet.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();

	}
	@FXML
	void switchremovepet(ActionEvent event) throws IOException {
		BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("removepet.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();

	}

	@FXML
	void getOption(ActionEvent event) {
		if(optadopt.isSelected()) {
			myLabel.setText("Adopt");
			optsell.setSelected(false);
		}
		 if(optsell.isSelected())
			myLabel.setText("Sell");
			optadopt.setSelected(false);
	}
	@FXML
	public void switchcreateLogin(ActionEvent event) throws IOException {
		BorderPane root = 
(BorderPane)FXMLLoader.load(getClass().getResource("Signup.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root,600,485);
		stage.setScene(scene);
		stage.show();
	}
@FXML

	public void switchtoHome(ActionEvent event) throws IOException {
		BorderPane root = 
(BorderPane)FXMLLoader.load(getClass().getResource("Homepage.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();
	}
@FXML

public void switchtoHome1(ActionEvent event) throws IOException {
	BorderPane root = 
(BorderPane)FXMLLoader.load(getClass().getResource("Homepage1.fxml"));
	stage = (Stage)((Node)event.getSource()).getScene().getWindow();
	scene = new Scene(root);
	stage.setScene(scene);
	stage.show();
}
	@FXML
	public void switchtoLogin(ActionEvent event) throws IOException {
		BorderPane root = 
(BorderPane)FXMLLoader.load(getClass().getResource("Sample.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();
	}
	@FXML
	void switchdog(ActionEvent event) throws IOException {
		BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("dog.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();

	}
	@FXML
	void switchcat(ActionEvent event) throws IOException {
		BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("cat.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();

	}
	@FXML
	void switchothers(ActionEvent event) throws IOException {
		BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("others.fxml"));
		stage = (Stage)((Node)event.getSource()).getScene().getWindow();
		scene = new Scene(root);
		stage.setScene(scene);
		stage.show();

	}

}
```

Main class:

```
package application;

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


public class Main extends Application {
	@Override
	public void start(Stage primaryStage) {
		try {
			BorderPane root =
(BorderPane)FXMLLoader.load(getClass().getResource("Sample.fxml"));
			Scene scene = new Scene(root,500,350);

	scene.getStylesheets().add(getClass().getResource("application.css").toExter
nalForm());
			primaryStage.setScene(scene);
			primaryStage.show();
		} catch(Exception e) {
			e.printStackTrace();
		}
	}

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

CHAPTER 7
RESULTS AND DISCUSSIONS

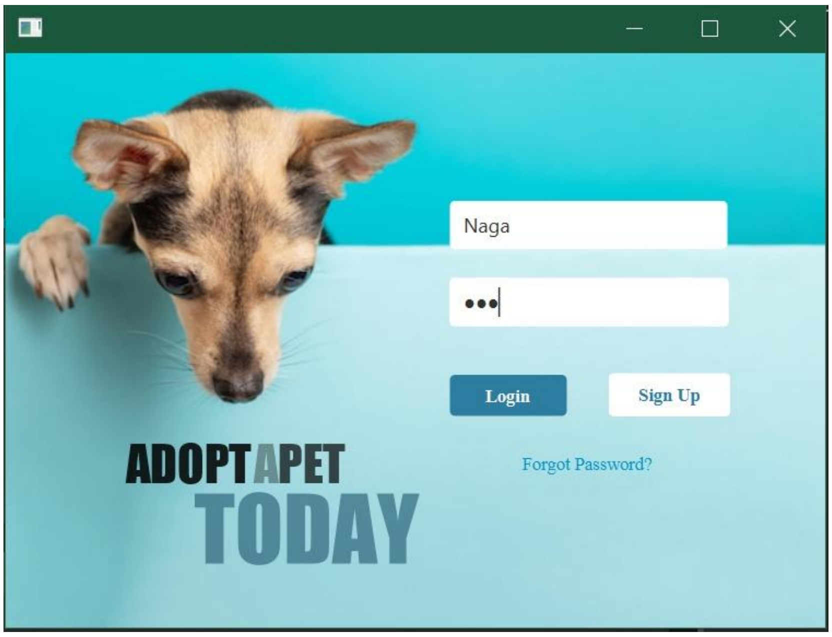
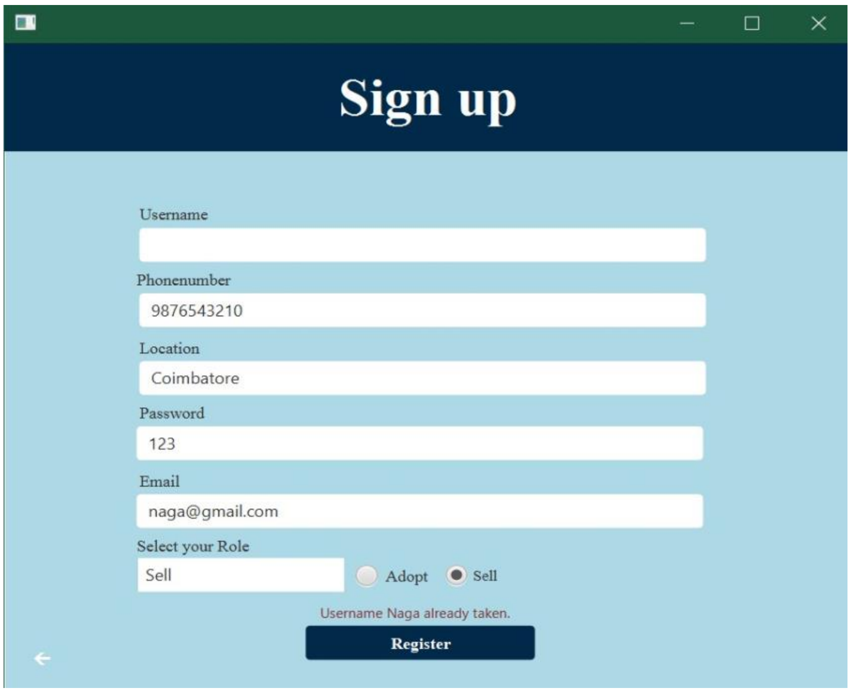


Fig 7.1: Login page which asks for username and password for registered users



Username

Phonenumber 9876543210

Location Coimbatore

Password 123

Email naga@gmail.com

Select your Role Sell   Adopt [✓ Sell]

Username Naga already taken.

Fig 7.2: Register page for new user with phone number and username validation

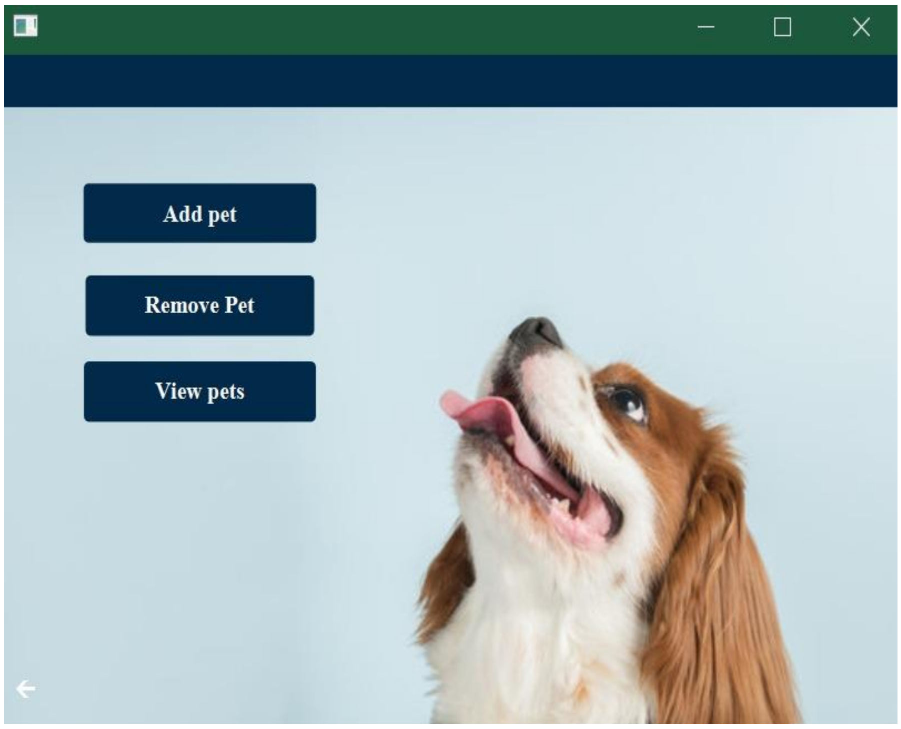
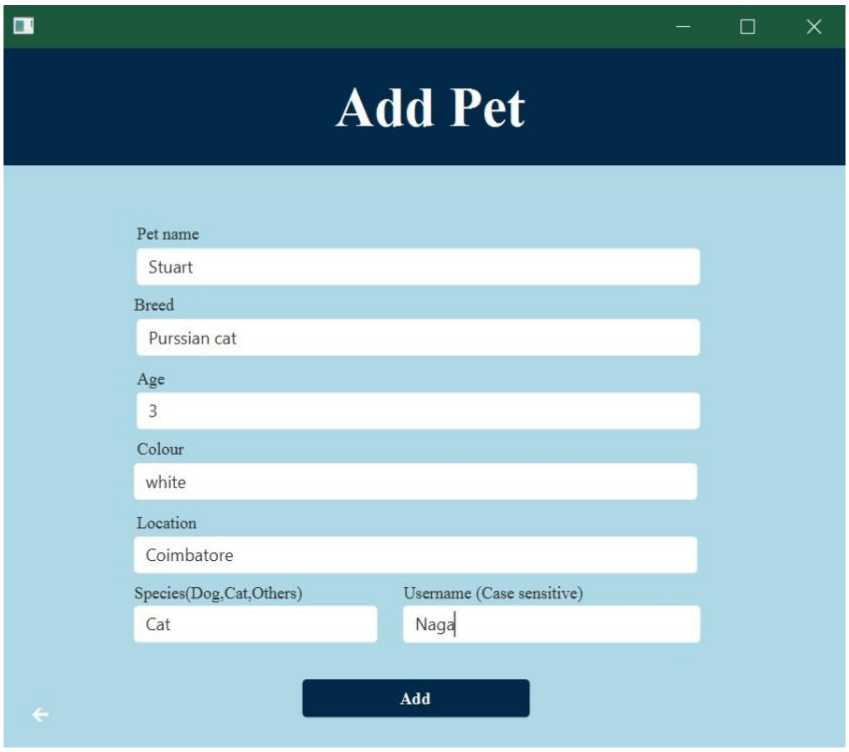


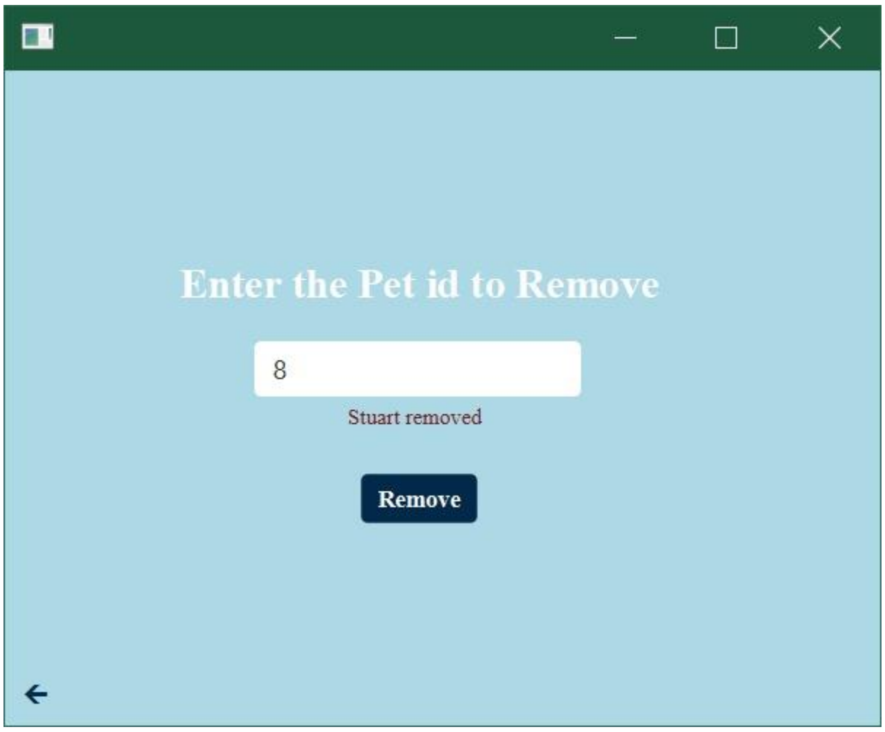
Fig 7.3: Pet seller homepage with multiple options after login or registeration



Add Pet

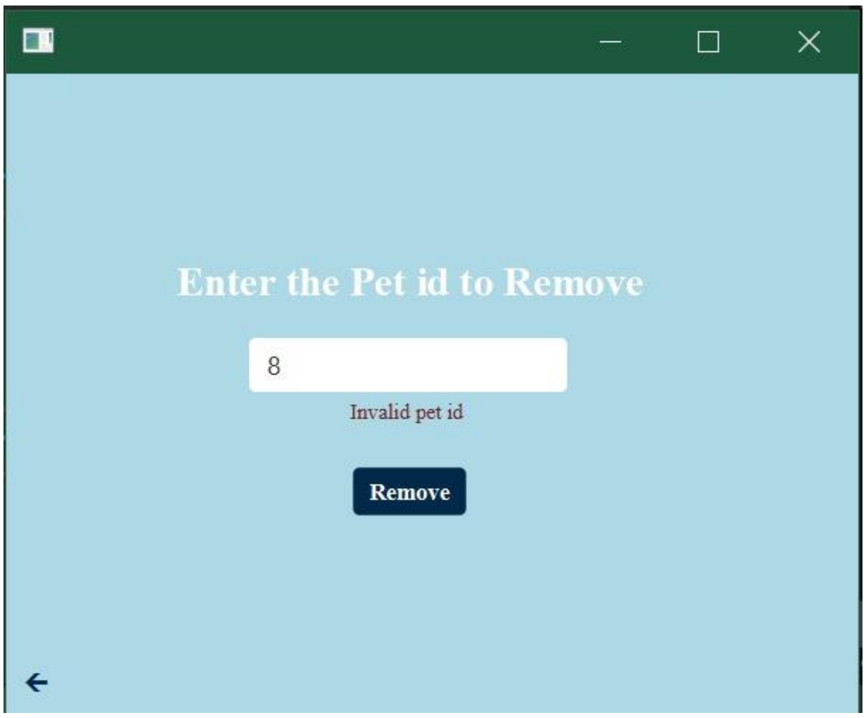
Pet name
Stuart
Breed
Purssian cat
Age
3
Colour
white
Location
Coimbatore
Species(Dog,Cat,Others)
Cat
Username (Case sensitive)
Naga

Fig 7.4: Pet add page getting the pet details from the seller (pet id is displayed after successful addition of pet into the database)



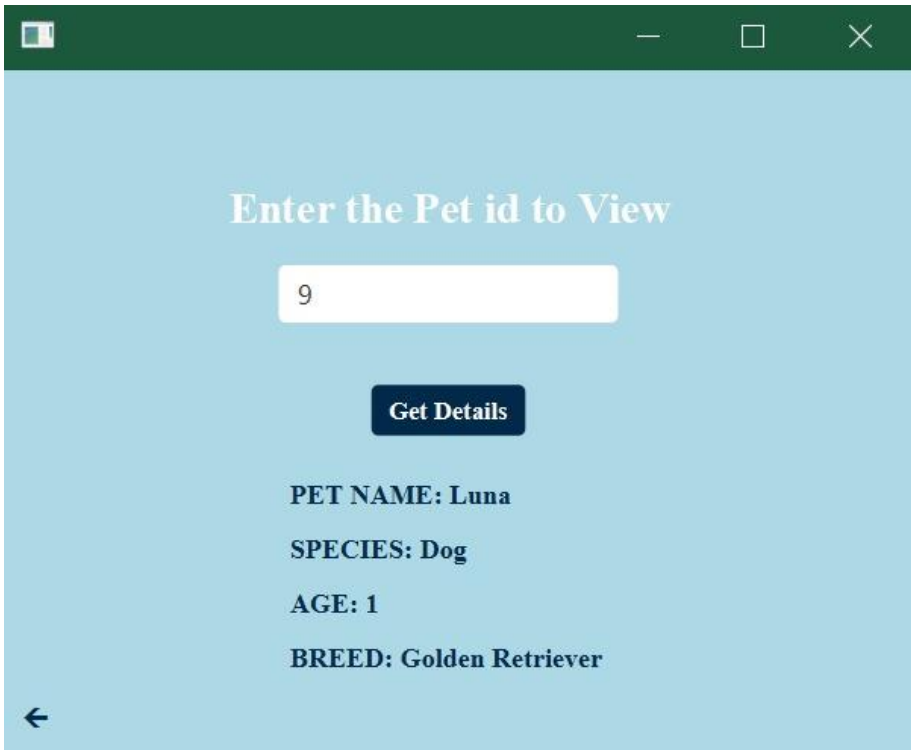
Stuart removed

Fig 7.5



Invalid pet id

Fig 7.6: Removing a pet from the database with pet id 8. Already removed pet or pet with no such entered id informs the user that the id is invalid


Enter the Pet id to View

9

Get Details

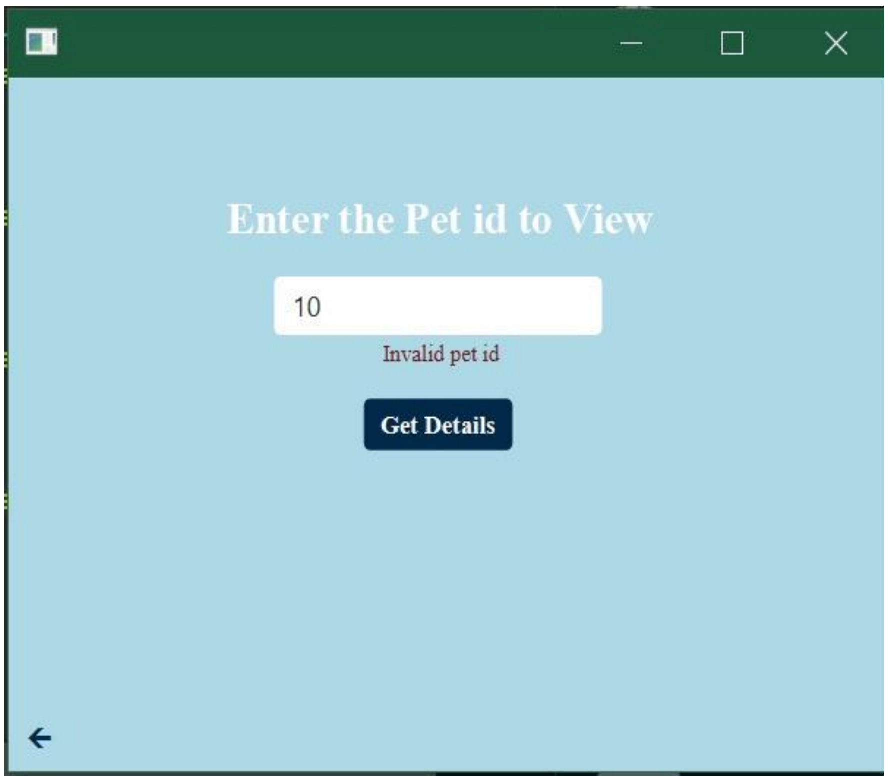
PET NAME: Luna

SPECIES: Dog

AGE: 1

BREED: Golden Retriever

Fig 7.7


Enter the Pet id to View

10

Invalid pet id

Get Details

Fig 7.8: Accessing pet details registered in the database using pet id

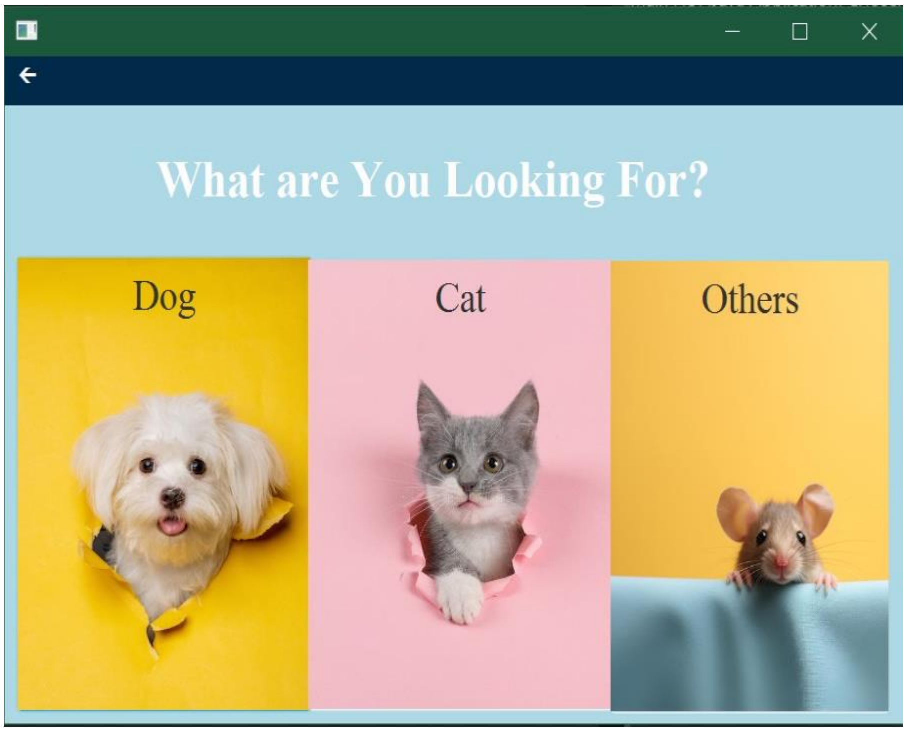
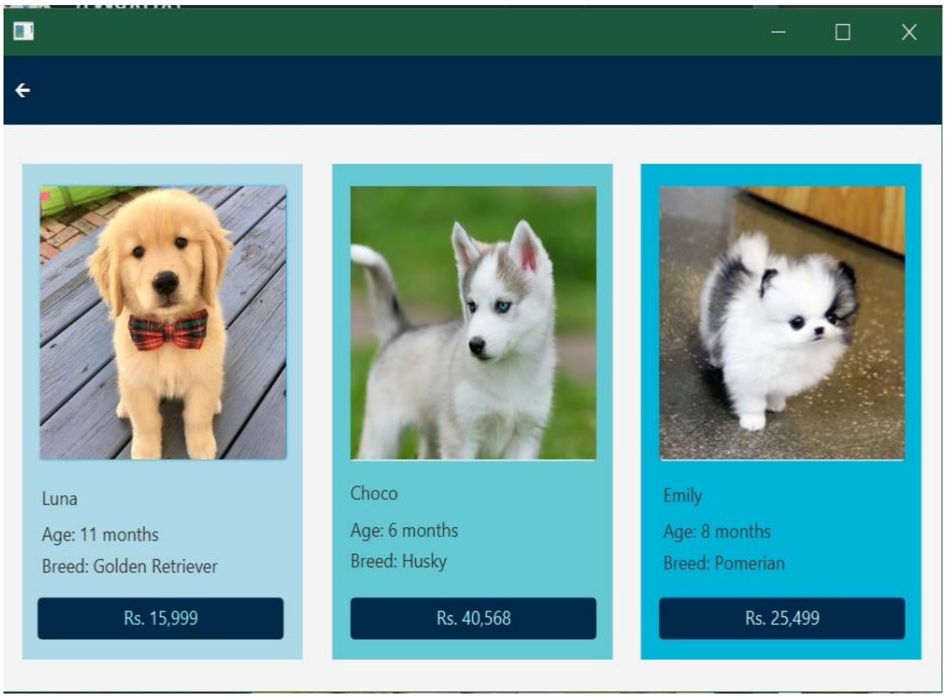


Fig 7.9: Pet seeker's homepage after login or registration

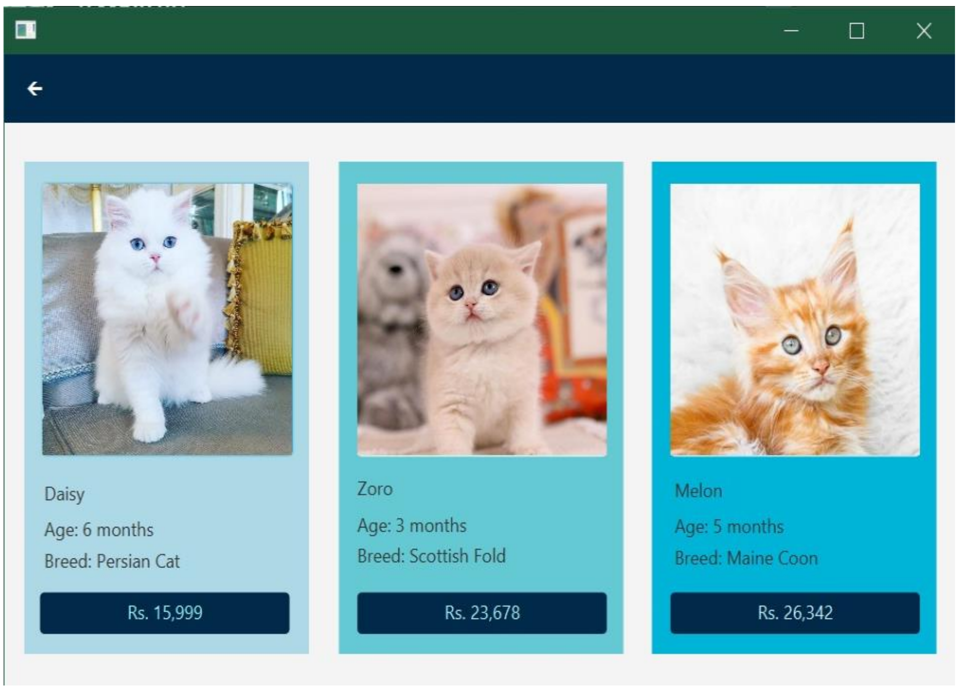


Luna
Age: 11 months
Breed: Golden Retriever
Rs. 15,999

Choco
Age: 6 months
Breed: Husky
Rs. 40,568

Emily
Age: 8 months
Breed: Pomerian
Rs. 25,499

Fig 7.10: Available dog profiles for adoption



Daisy

Age: 6 months

Breed: Persian Cat

Rs. 15,999

Zoro

Age: 3 months

Breed: Scottish Fold

Rs. 23,678

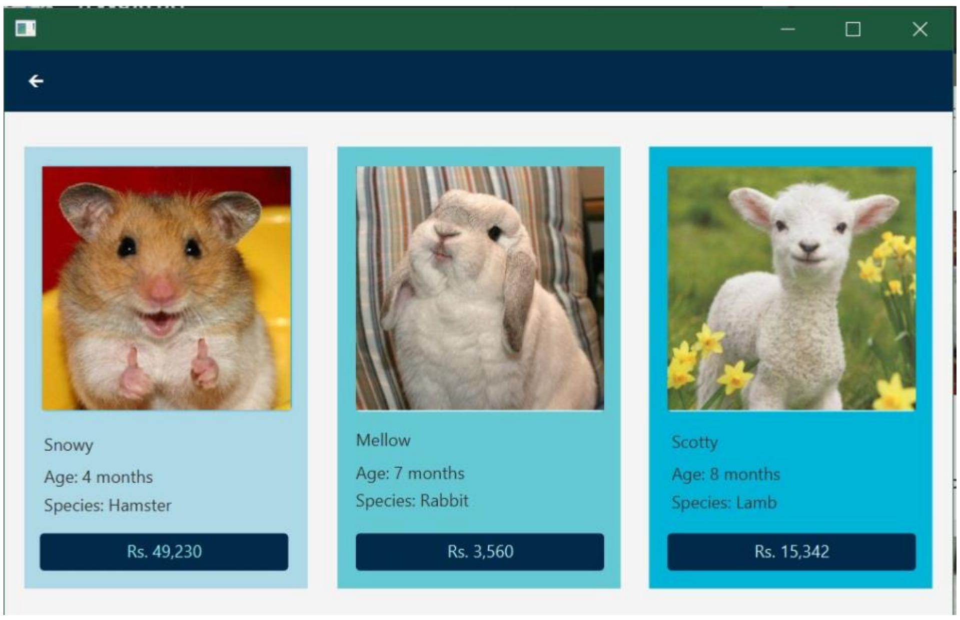
Melon

Age: 5 months

Breed: Maine Coon

Rs. 26,342

Fig 7.11: Avalaible cat profiles for adoption



Snowy

Age: 4 months

Species: Hamster

Rs. 49,230

Mellow

Age: 7 months

Species: Rabbit

Rs. 3,560

Scotty

Age: 8 months

Species: Lamb

Rs. 15,342

Fig 7.12: Available other animals for adoption

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

In summary, the Pet Adoption System represents a significant stride toward transforming pet adoption processes. Through its innovative JavaFX application seamlessly connected to a MySQL database, the system provides an organized and efficient platform. The modular class construction, exemplified by "Seller," "Seeker," and others, establishes a versatile foundation. The JDBC connection ensures smooth interaction with the database, facilitating secure user authentication and streamlined pet management.

Looking forward, the Pet Adoption System has exciting avenues for growth. Future developments may include enriched user profiles with multimedia content, an advanced pet matching algorithm for more accurate suggestions, mobile application expansion for increased accessibility, integration with external APIs for additional pet information, and the introduction of an online payment gateway for streamlined transactions. These enhancements aim to elevate the system, fostering a more engaging and user-friendly experience while continually contributing to the welfare of animals and promoting responsible pet adoption.

REFERENCES

[1] https://www.youtube.com/watch?v=YyaO1cZIRd8

[2] https://www.youtube.com/watch?v=ipz3Ezdeu3M

[3] https://www.youtube.com/watch?v=hcM-R-YOKkQ

[4]
https://www.academia.edu/98308313/Mobile_Application_of_Pet_Adoption_System

[5] https://petrehomer.org/