INSTITUTE FOR ADVANCED COMPUTING AND SOFTWARE DEVELOPMENT, AKURDI, PUNE

'PAWSITIVE FUTURES"

Pet Adoption Platform

PG-DAC March 2024
Submitted By
Group No: 41

243072   Pratiksha Tambe
243083   Punam Salunkhe

Mrs. Sonali Mogal
Mrs. Shraddha Salunkhe   **Mr. Rohit Puranik**
Project Guide   Centre Coordinator

ABSTRACT

"Pawsitive Futures" is a comprehensive pet adoption platform designed to streamline the process of pet adoption and management. This web-based application facilitates interaction among three distinct user roles: Admin, Shelter, and User, each with specific functionalities to enhance the efficiency and effectiveness of the pet adoption process.

The Admin role is responsible for overseeing the entire platform, including managing shelter and user accounts. Administrators have the ability to add, update, or remove shelters and users, ensuring that the platform remains secure and up-to-date.

Shelters are at the heart of the platform, managing their pet listings and adoption applications. Shelters can post details about available pets, review and process adoption applications, and update the status of pets within their care. This role is crucial for ensuring that pets are accurately represented and that the adoption process is smooth and transparent.

Users interact with the platform to browse available pets, submit adoption applications, and track their application status. They can search for pets based on various criteria, view detailed information about each pet, and apply for adoption through a user-friendly interface. The platform aims to simplify the adoption process, making it easier for users to find and adopt their ideal pets.

"Pawsitive Futures" leverages modern web technologies to provide a seamless and intuitive experience for all users. By integrating features such as role-based access control, dynamic pet listings, and real-time application tracking, the platform seeks to improve the overall efficiency of pet adoption and contribute to the welfare of animals in need of loving homes.

Acknowledgements

I am grateful for the opportunity to work on the "Pawsitive Futures" project and would like to acknowledge those who made this possible. My sincere thanks go to **Mrs.Sonali Mogal** and **Mrs.Shraddha Salunkhe** for their invaluable guidance and support throughout the project. Their expertise and encouragement were crucial to our success. I also extend my appreciation to **Mr.Rohit Puranik** for their assistance and coordination, which significantly contributed to the smooth execution of this project. Your contributions were essential to achieving our goals.

Pratiksha Tambe(240341220144)

Punam Salunkhe(240341220165)

Table of Contents

Introduction

1.1   Purpose

The purpose of the "Pawsitive Futures" project is to create a user-friendly web platform that facilitates the adoption of pets. The platform aims to streamline the adoption process by providing a centralized location where potential adopters can browse available pets, shelters can manage their listings, and administrators can oversee platform operations. By leveraging modern web technologies, the project seeks to enhance the efficiency of pet adoption, making it easier for animals to find loving homes.

1.2   Scope

The scope of this project encompasses the development of a full-stack web application with distinct roles for Admin, Shelter, and User. The platform will include features such as role-based access control, dynamic pet listings, application management, and real-time tracking of adoption statuses. The project will focus on ensuring that each user role has access to tailored functionalities, making the adoption process seamless for all participants. Additionally, the platform will be designed to be scalable and maintainable, allowing for future enhancements and expansions.

1.3   Objective

- **Develop a robust, user-friendly platform** that simplifies the pet adoption process for users, shelters, and administrators.

- **Implement role-based access control** to ensure that each user type (Admin, Shelter, User) has appropriate access to features and functionalities.

- **Create dynamic pet listings** that allow shelters to easily post and update information about adoptable pets.

- **Ensure secure and reliable operations** by incorporating bcrypt for password hashing, data validation, and secure communication throughout the platform.

- **Enhance user experience** with intuitive navigation, responsive design, and clear communication of application statuses.

1.4 Functionalities

1.4.1 User Authentication and Role Management

- **Sign Up and Login:** Users can create accounts and log in to the platform, with role-based access control determining their level of access (Admin, Shelter, User).

- **Role-Based Dashboards:** Different dashboards are provided for Admins, Shelters, and Users, each offering specific functionalities tailored to their roles.

1.4.2 Pet Listing Management:

- **Add/Edit/Delete Pets:** Shelters can add new pets to the platform, including details such as species, breed, age, and photos. They can also edit or remove listings as needed.

1.4.3 Adoption Application Management:

- **Submit Adoption Applications:** Users can submit applications for the pets they are interested in adopting, providing necessary information and preferences.

- **Application Review:** Shelters can view, manage, and respond to adoption applications, including approving or rejecting them based on predefined criteria.

- **Application Tracking:** Users can track the status of their adoption applications in real-time, receiving updates on approval, rejection, or further steps.

1.4.4 Admin Management:

- **User and Shelter Management:** Admins can manage user and shelter accounts, including adding, editing, and deleting accounts as necessary. They can also assign roles and manage access permissions.

1.4.5 Payment Processing:

- **Secure Payments:** Users can make payments for adoption fees through the platform, with secure payment gateways integrated for processing transactions.

- **Payment History:** Users and shelters can view payment histories, ensuring transparency and tracking of financial transactions.

1.4.6   Profile Management :

- **User Profiles:** Users can manage their profiles, including updating personal information, viewing their adoption history, and managing their account settings.

- **Shelter Profiles:** Shelters can update their profiles with information about their organization, contact details, and the pets they have listed for adoption.

Requirements

2.1   Functional Requirements

1. **User Registration and Authentication**

   - Users must be able to create an account by providing their email, password, and other necessary details.

   - The system must support role-based authentication, with roles for Admin, Shelter, and User.

   - Users must be able to log in, log out, and reset their passwords.

2. **Role-Based Dashboards**

   - Each user role (Admin, Shelter, User) must have access to a specific dashboard that provides functionalities relevant to their role.

   - Admins should manage user and shelter accounts, as well as oversee platform operations.

   - Shelters must be able to manage pet listings and review adoption applications.

   - Users must be able to browse pets, submit adoption applications, and track their application status.

3. **Pet Listing Management**

   - Shelters must be able to add, edit, and delete pet listings, including details such as species, breed, age, and photos.

4. **Adoption Application Process**

   - Users must be able to submit adoption applications for pets.

   - Shelters must be able to review, approve, or reject adoption applications.

   - The system must provide status updates to users regarding their adoption applications.

5. **Payment Processing**

   - Users must be able to securely process payments for adoption fees through integrated payment gateways.

- The system must provide users and shelters with payment history and transaction details.

6. **Profile Management**

   - Users must be able to view and edit their profiles, including personal information and adoption history.

   - Shelters must be able to update their profiles with contact details and organizational information.

2.2 Non-Functional Requirements

1. **Performance**

   - The platform should maintain fast load times under normal conditions.

   - The system must handle multiple users simultaneously without experiencing performance degradation.

2. **Scalability**

   - The platform must be designed to scale easily, allowing for the addition of new features and supporting a growing number of users and shelters over time.

3. **Usability**

   - The user interface must be intuitive and easy to navigate for users of all technical skill levels.

   - The platform should be responsive and work well on various devices, including desktops, tablets, and smartphones.

4. **Reliability**

   - The platform must have an uptime of 99.9%, ensuring it is available to users whenever needed.

5. **Security**

   - The platform must protect user data through encryption and secure communication protocols (e.g., HTTPS).

   - The system must comply with data protection regulations, ensuring the privacy and security of user information.

6. **Maintainability**

   - The platform codebase must be well-documented and structured to facilitate easy maintenance and updates.

7. **Data Integrity**

   - The system must ensure the consistency and accuracy of data, especially during transactions and user interactions.

2.3   Other Requirements

2.3.1   Hardware and Network Interfaces

Back-end Server Configuration-

- **Processor:** Multi-core processor (e.g., Intel or AMD) for handling concurrent requests efficiently.

- **Memory :**Minimum of 16 GB RAM to support database operations and server-side processing.

- **Storage:** SSD storage with a minimum of 500 GB for fast data retrieval and storage of application data.

- **Database:** MySQL for relational data management.

Front-end Client Configuration-

- **Processor:** Dual-core or higher processor (e.g., Intel Core i5 or AMD Ryzen) for smooth operation of the web interface.

- **Memory:** Minimum of 8 GB RAM to ensure smooth browsing and application usage.

- **Storage:** SSD with at least 256 GB to store temporary files and browser cache.

- **Operating System:** Compatible with major operating systems such as Windows, macOS, and Linux.

- **Browser:** Latest version of web browsers like Google Chrome, Mozilla Firefox, Safari, or Microsoft Edge for optimal performance.

2.3.2   Software Interfaces

Software Configuration for Back-end Services-

- **Programming Language:** Java with Spring Boot framework for building RESTful APIs and handling server-side logic.

- **Database:** MySQL.

- **Application Server:** Apache Tomcat.

- **APIs:** RESTful API, External API integration's().

- **Build Tools:** Maven

- **Version Control:** Git for source code management, with repositories hosted on platform GitHub.

Software Configuration for Front-end Services-

- **Frameworks and Libraries:** React.js, Redux

- **Languages:** JavaScript, HTML5, CSS3

- **UI/UX:** Bootstrap.

- **Routing:** React Router.

- **APIs:** Axios.

- **Development Environment:** Visual Studio Code.

- **Browser:** Chrome/Microsoft Edge/Firefox/Safari.

UML Diagrams

3.1 ER Diagram
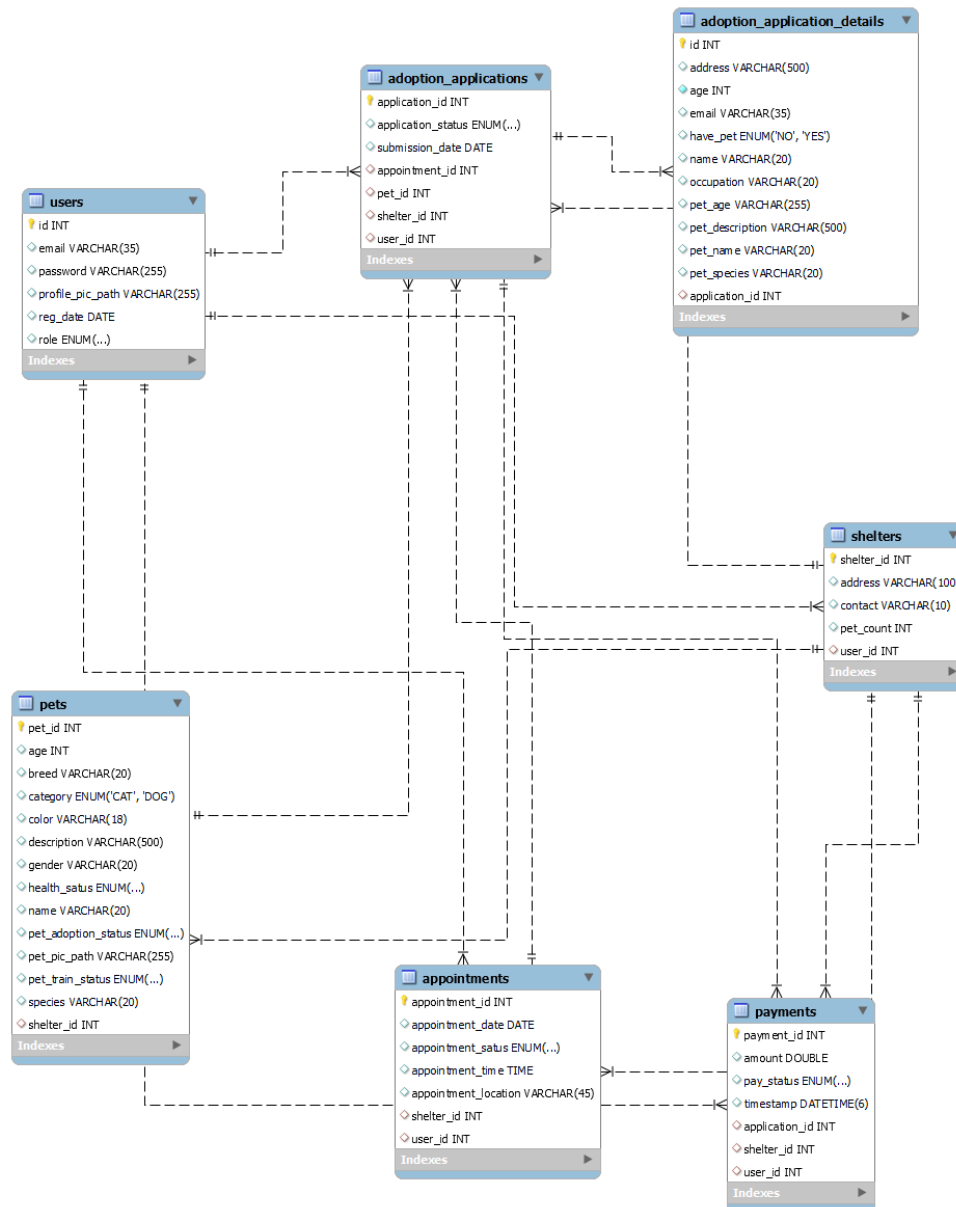

Figure 3.1: ER Diagram
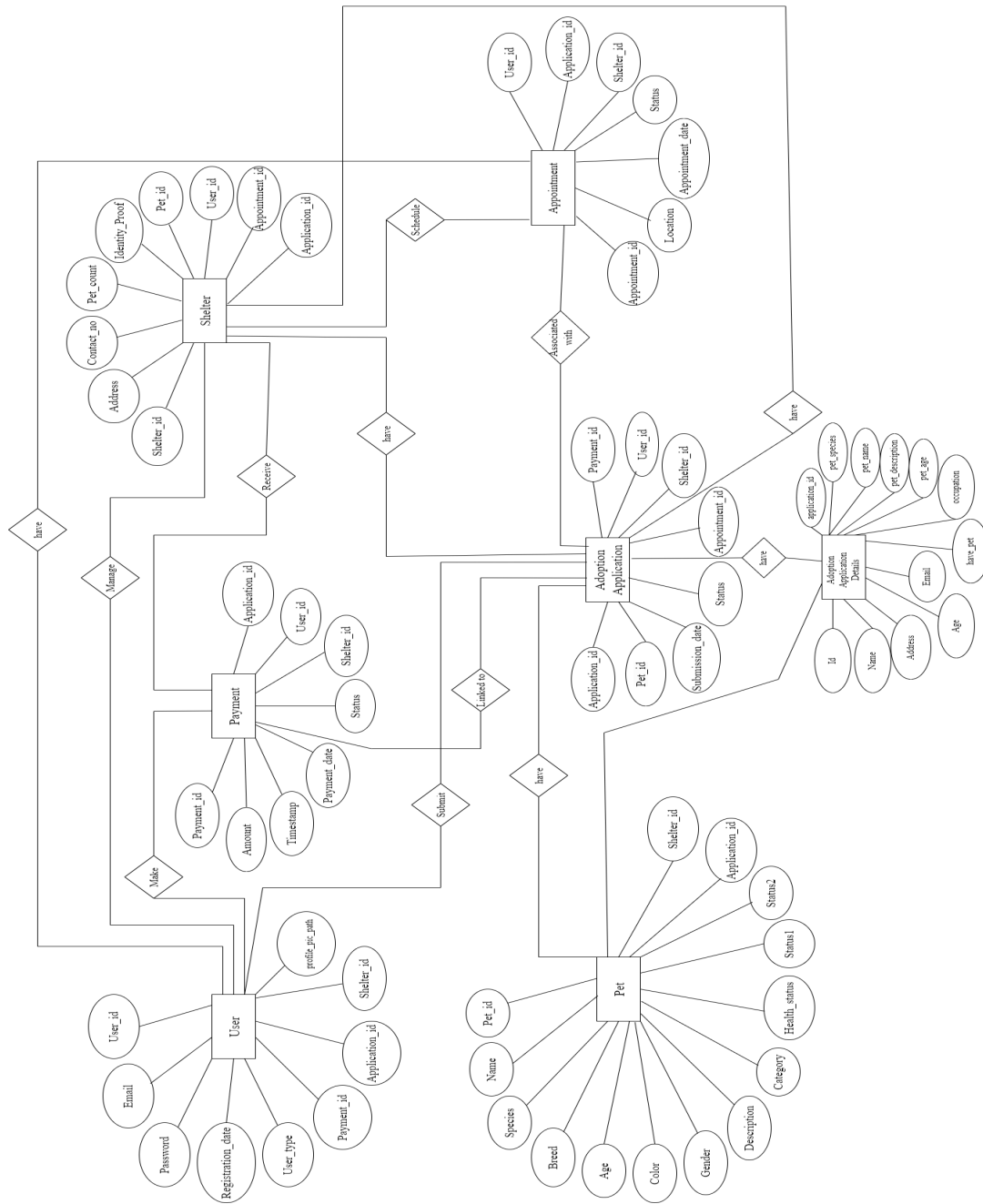
Figure 3.2: ER Diagram

3.2   Use Case Diagram
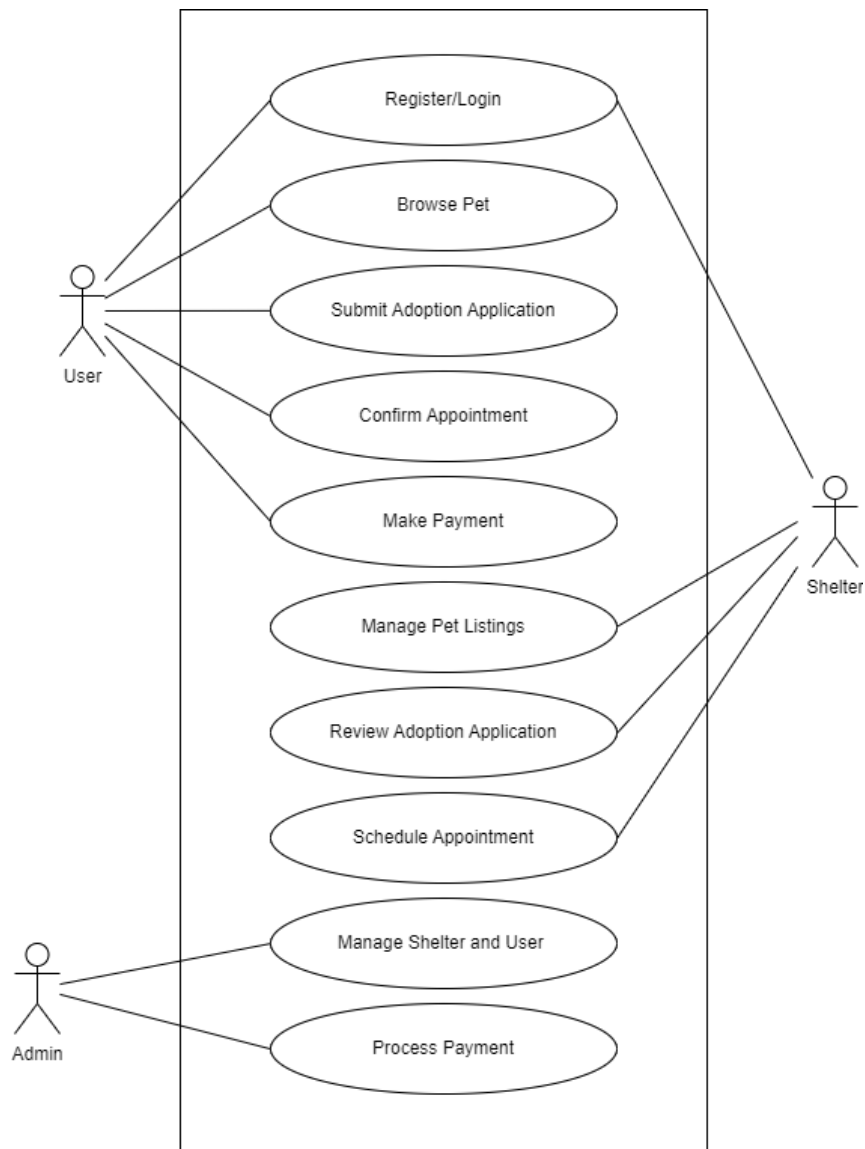

Figure 3.3: Use Case Diagram

3.3 Data Flow Diagram
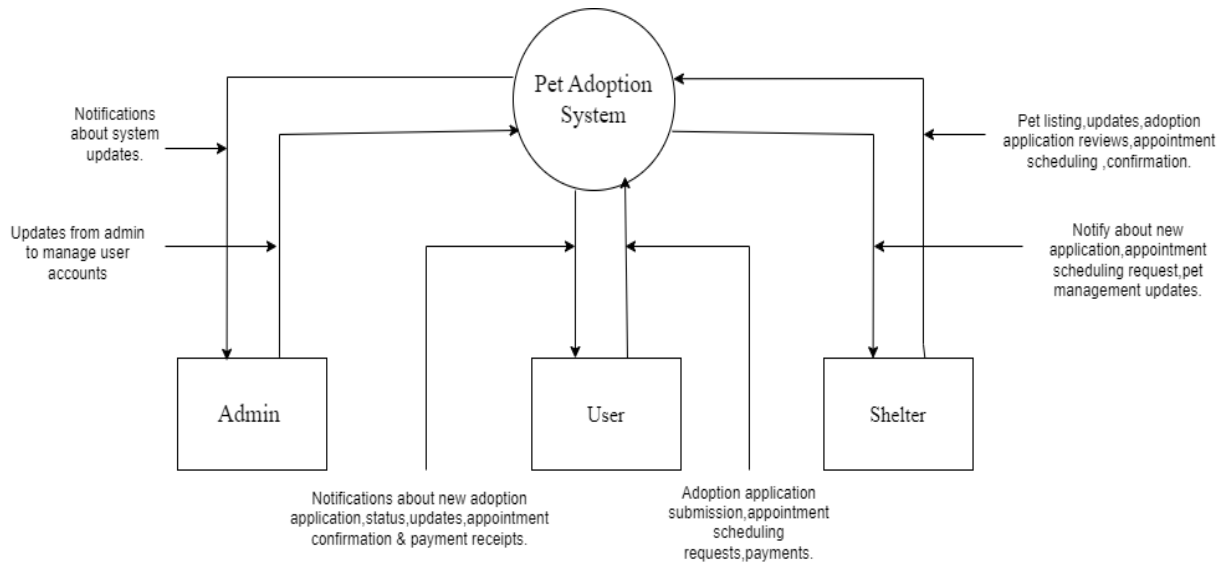
3.3.1 Level 0

Figure 3.4: DFD Level 0

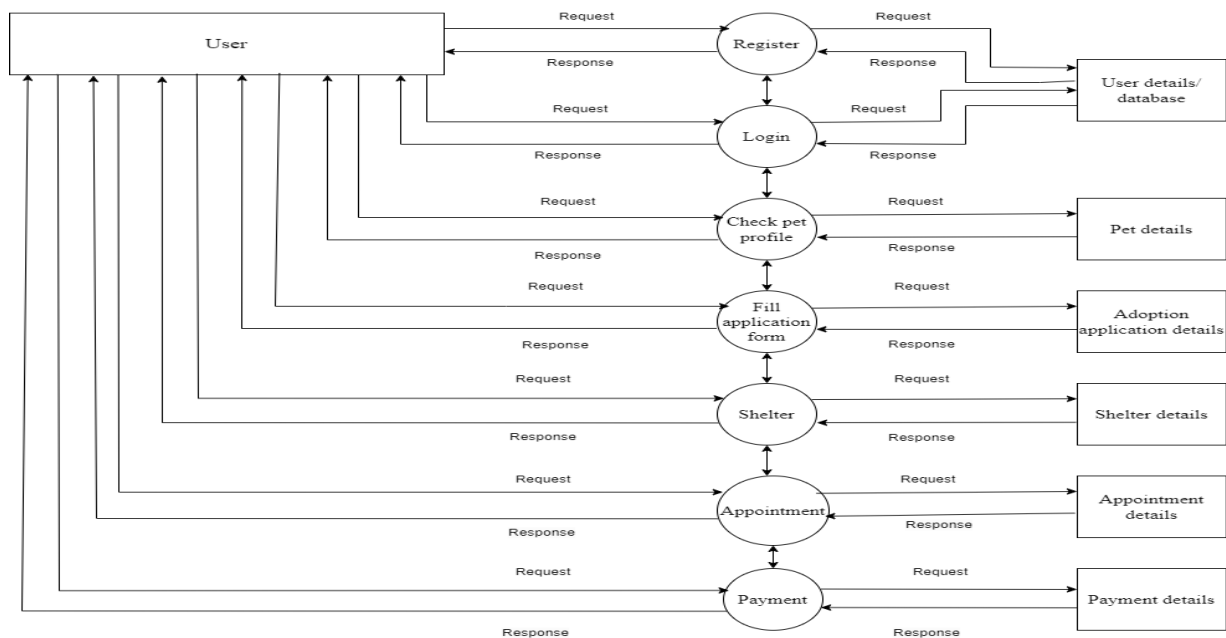## 3.3.2 Level 1

Figure 3.5: DFD Level 1
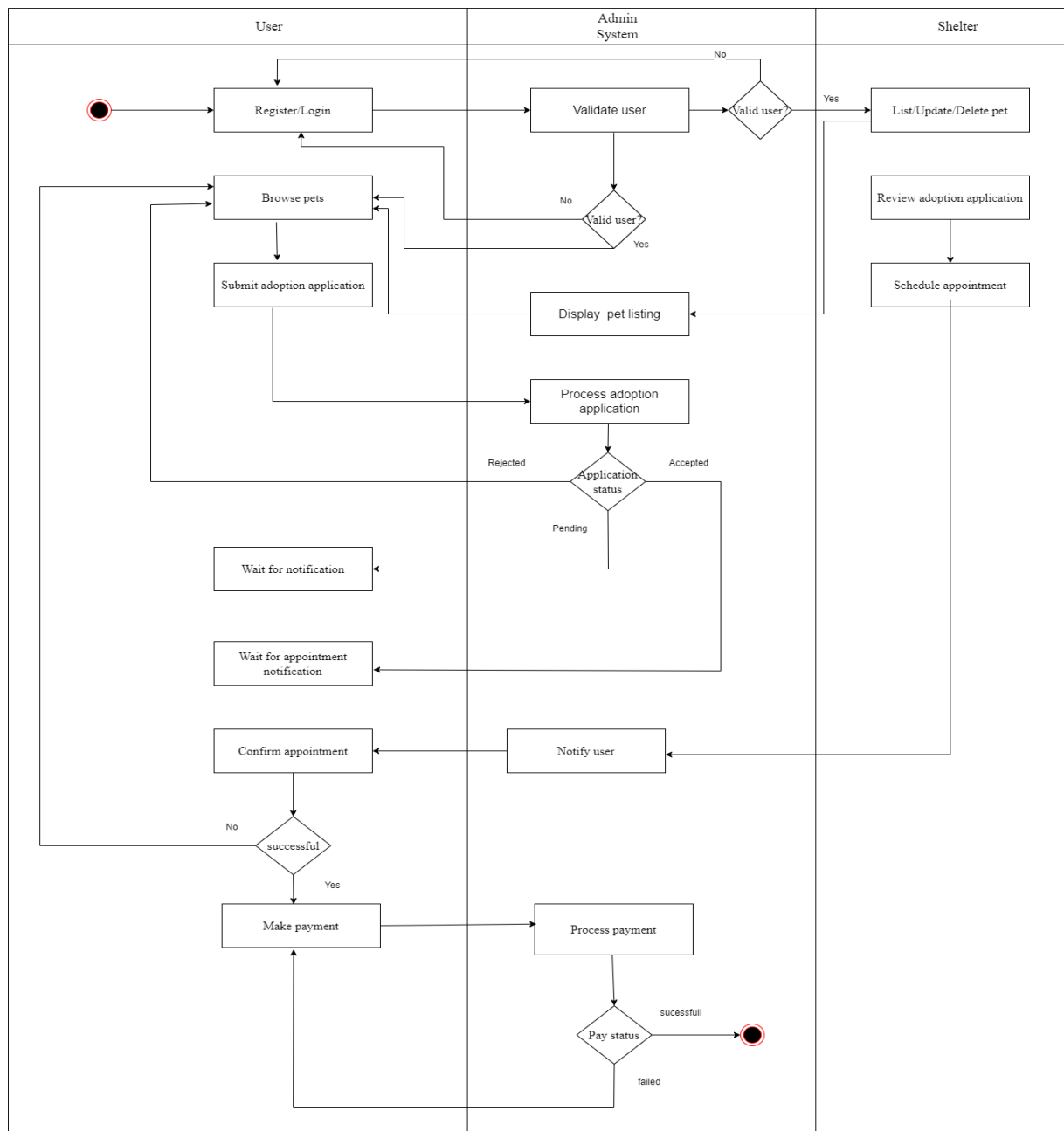
3.4 Activity Diagram

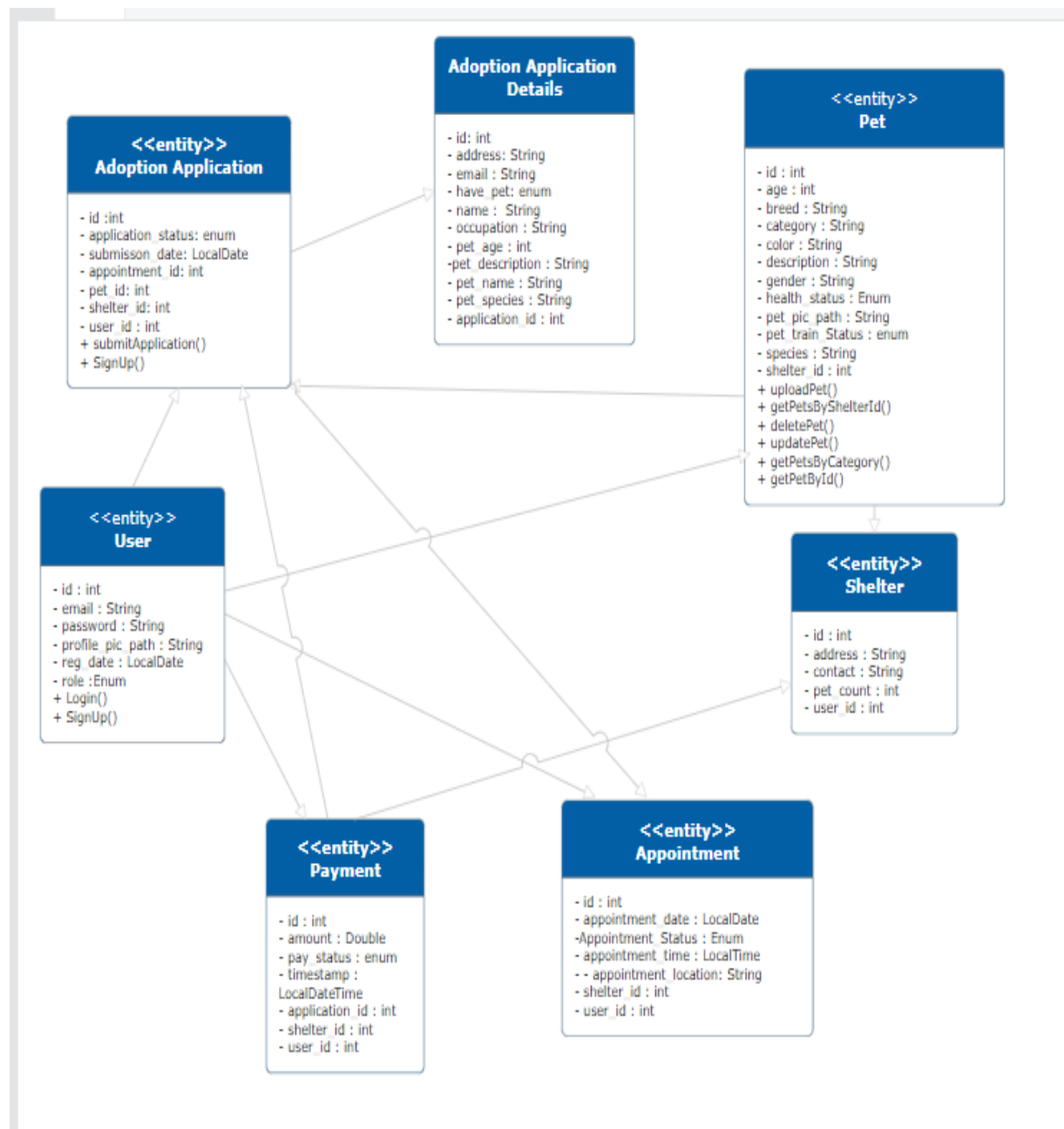**Figure 3.6:** Activity Diagram

3.5 Class Diagram
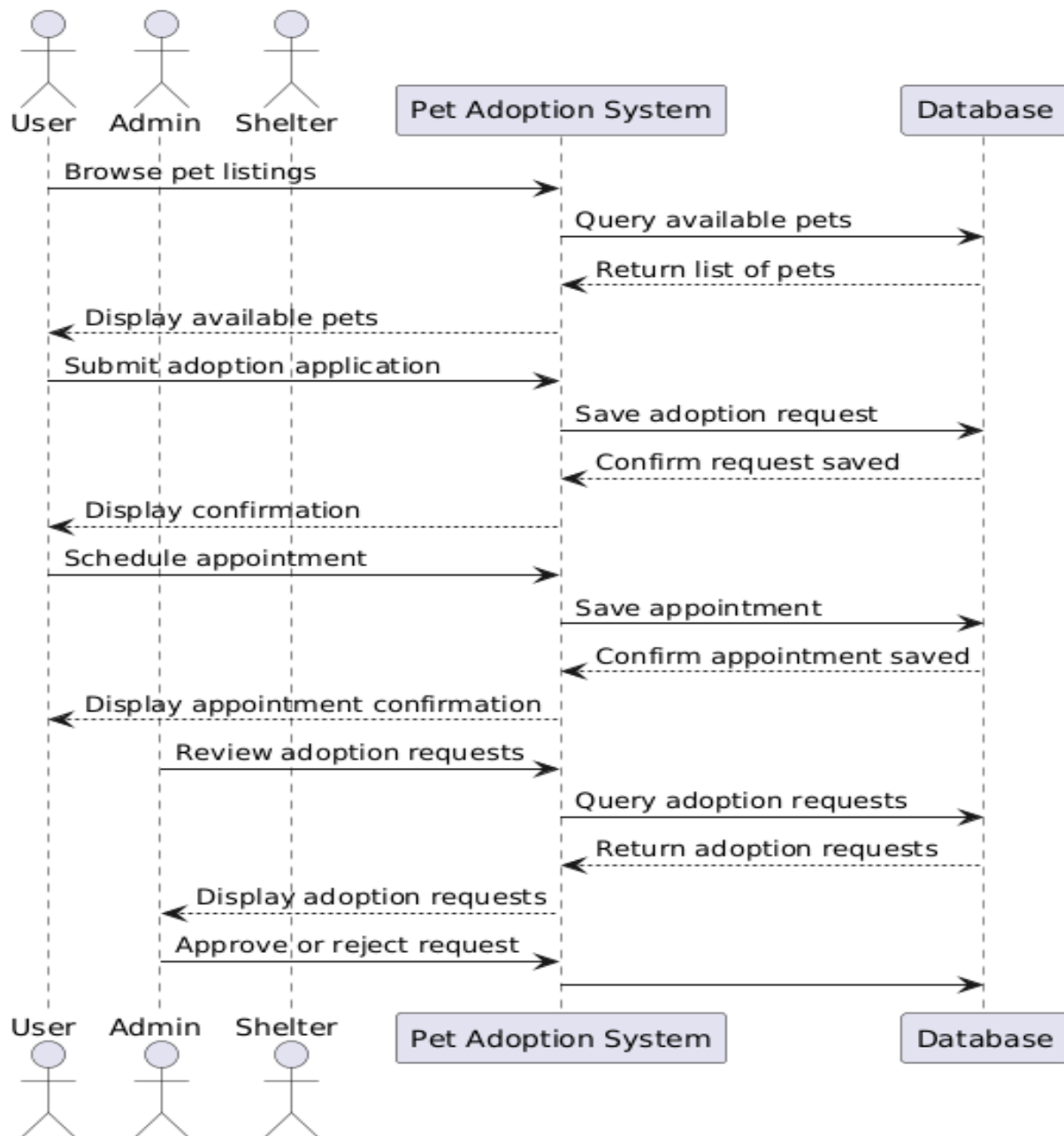

Figure 3.7: Class Diagram

3.6   Sequence Diagram

Figure 3.8: Sequence Diagram

Database Design

4.1 Users Table

```
mysql> desc users;
+-----------------+--------------+------+-----+---------+----------------+
| Field           | Type         | Null | Key | Default | Extra          |
+-----------------+--------------+------+-----+---------+----------------+
| id              | bigint       | NO   | PRI | NULL    | auto_increment |
| email           | varchar(35)  | YES  | UNI | NULL    |                |
| password        | varchar(255) | YES  |     | NULL    |                |
| profile_pic_path| varchar(255) | YES  |     | NULL    |                |
| reg_date        | date         | YES  |     | NULL    |                |
+-----------------+--------------+------+-----+---------+----------------+
5 rows in set (0.01 sec)
```

Figure 4.1: Users Table

4.2 Shelters Table

```
mysql> desc shelters;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| shelter_id | int          | NO   | PRI | NULL    | auto_increment |
| address    | varchar(100) | YES  |     | NULL    |                |
| contact    | varchar(10)  | YES  |     | NULL    |                |
| pet_count  | int          | YES  |     | NULL    |                |
| user_id    | bigint       | YES  | UNI | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

Figure 4.2: Shelters Table

4.3   Pets Table

```
mysql> desc pets;
+--------------------+-------------------------------------------+------+-----+---------+----------------+
| Field              | Type                                      | Null | Key | Default | Extra          |
+--------------------+-------------------------------------------+------+-----+---------+----------------+
| pet_id             | int                                       | NO   | PRI | NULL    | auto_increment |
| age                | int                                       | YES  |     | NULL    |                |
| breed              | varchar(20)                               | YES  |     | NULL    |                |
| category           | enum('CAT','DOG')                         | YES  |     | NULL    |                |
| color              | varchar(18)                               | YES  |     | NULL    |                |
| description        | varchar(500)                              | YES  |     | NULL    |                |
| gender             | varchar(20)                               | YES  |     | NULL    |                |
| health_satus       | enum('HEALTHY','INJURED','RECOVERING','SICK') | YES  |     | NULL    |                |
| name               | varchar(20)                               | YES  |     | NULL    |                |
| pet_adoption_status| enum('ADOPTED','AVAILABLE')               | YES  |     | NULL    |                |
| pet_pic_path       | varchar(255)                              | YES  |     | NULL    |                |
| pet_train_status   | enum('RETIRED','TRAINED','UNTRAINED')     | YES  |     | NULL    |                |
| species            | varchar(20)                               | YES  |     | NULL    |                |
| shelter_id         | int                                       | YES  | MUL | NULL    |                |
+--------------------+-------------------------------------------+------+-----+---------+----------------+
14 rows in set (0.01 sec)
```

Figure 4.3: Pets Table

4.4   Adoption Application Table

```
mysql> desc adoption_applications;
+--------------------+------------------------------------+------+-----+---------+----------------+
| Field              | Type                               | Null | Key | Default | Extra          |
+--------------------+------------------------------------+------+-----+---------+----------------+
| application_id     | int                                | NO   | PRI | NULL    | auto_increment |
| application_status | enum('ACCEPTED','PENDING','REJECTED')| YES |     | NULL    |                |
| submission_date    | date                               | YES  |     | NULL    |                |
| appointment_id     | int                                | YES  | UNI | NULL    |                |
| pet_id             | int                                | YES  | MUL | NULL    |                |
| shelter_id         | int                                | YES  | MUL | NULL    |                |
| user_id            | int                                | YES  | MUL | NULL    |                |
+--------------------+------------------------------------+------+-----+---------+----------------+
7 rows in set (0.03 sec)
```

Figure 4.4: Adoption Applications Table

4.5 Adoption Application Details Table

```
mysql> desc adoption_application_details;
+-----------------+-----------------+------+-----+---------+----------------+
| Field           | Type            | Null | Key | Default | Extra          |
+-----------------+-----------------+------+-----+---------+----------------+
| id              | int             | NO   | PRI | NULL    | auto_increment |
| address         | varchar(500)    | YES  |     | NULL    |                |
| age             | int             | NO   |     | NULL    |                |
| email           | varchar(35)     | YES  | UNI | NULL    |                |
| have_pet        | enum('NO','YES')| YES  |     | NULL    |                |
| name            | varchar(20)     | YES  |     | NULL    |                |
| occupation      | varchar(20)     | YES  |     | NULL    |                |
| pet_age         | varchar(255)    | YES  |     | NULL    |                |
| pet_description | varchar(500)    | YES  |     | NULL    |                |
| pet_name        | varchar(20)     | YES  |     | NULL    |                |
| pet_species     | varchar(20)     | YES  |     | NULL    |                |
| application_id  | int             | YES  | UNI | NULL    |                |
+-----------------+-----------------+------+-----+---------+----------------+
12 rows in set (0.03 sec)
```

Figure 4.5: Adoption Application Details Table

4.6 Payments Table

```
mysql> desc payments;
+----------------+-------------------------------------------+------+-----+---------+----------------+
| Field          | Type                                      | Null | Key | Default | Extra          |
+----------------+-------------------------------------------+------+-----+---------+----------------+
| payment_id     | int                                       | NO   | PRI | NULL    | auto_increment |
| amount         | double                                    | YES  |     | NULL    |                |
| pay_status     | enum('CANCELLED','PENDING','SUCCESSFULL') | YES  |     | NULL    |                |
| timestamp      | datetime(6)                               | YES  |     | NULL    |                |
| application_id | int                                       | YES  | UNI | NULL    |                |
| shelter_id     | int                                       | YES  | MUL | NULL    |                |
| user_id        | bigint                                    | YES  | MUL | NULL    |                |
+----------------+-------------------------------------------+------+-----+---------+----------------+
7 rows in set (0.01 sec)
```

Figure 4.6: Payments Table

4.7 Appointments Table

```
mysql> desc appointments;
+---------------------+----------------------+------+-----+---------+----------------+
| Field               | Type                 | Null | Key | Default | Extra          |
+---------------------+----------------------+------+-----+---------+----------------+
| appointment_id      | int                  | NO   | PRI | NULL    | auto_increment |
| appointment_date    | date                 | YES  |     | NULL    |                |
| appointment_satus   | enum('DONE','PENDING') | YES |     | NULL    |                |
| appointment_time    | time(6)              | YES  |     | NULL    |                |
| appointment_location | varchar(45)         | YES  |     | NULL    |                |
| shelter_id          | int                  | YES  | MUL | NULL    |                |
| user_id             | bigint               | YES  | MUL | NULL    |                |
+---------------------+----------------------+------+-----+---------+----------------+
7 rows in set (0.01 sec)
```

Figure 4.7: Appointments Table

Screenshots of Project

5.1   Login
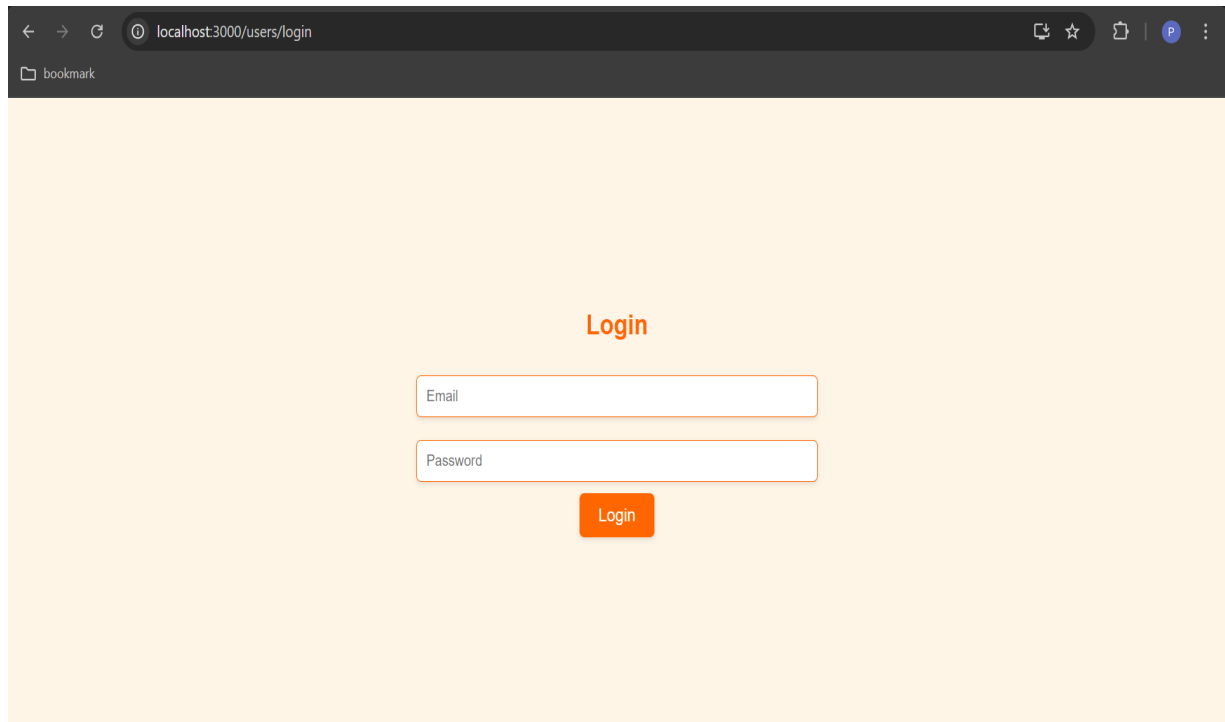
Figure 5.1: Login Page

5.2   SignUp
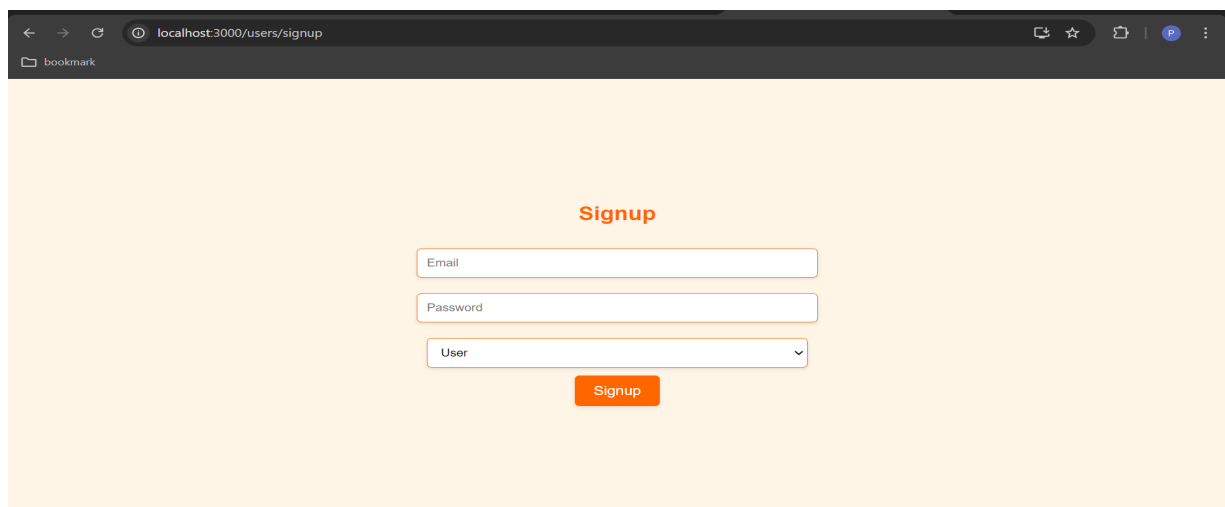
Figure 5.2: SignUp Page

5.3   User Dashboard
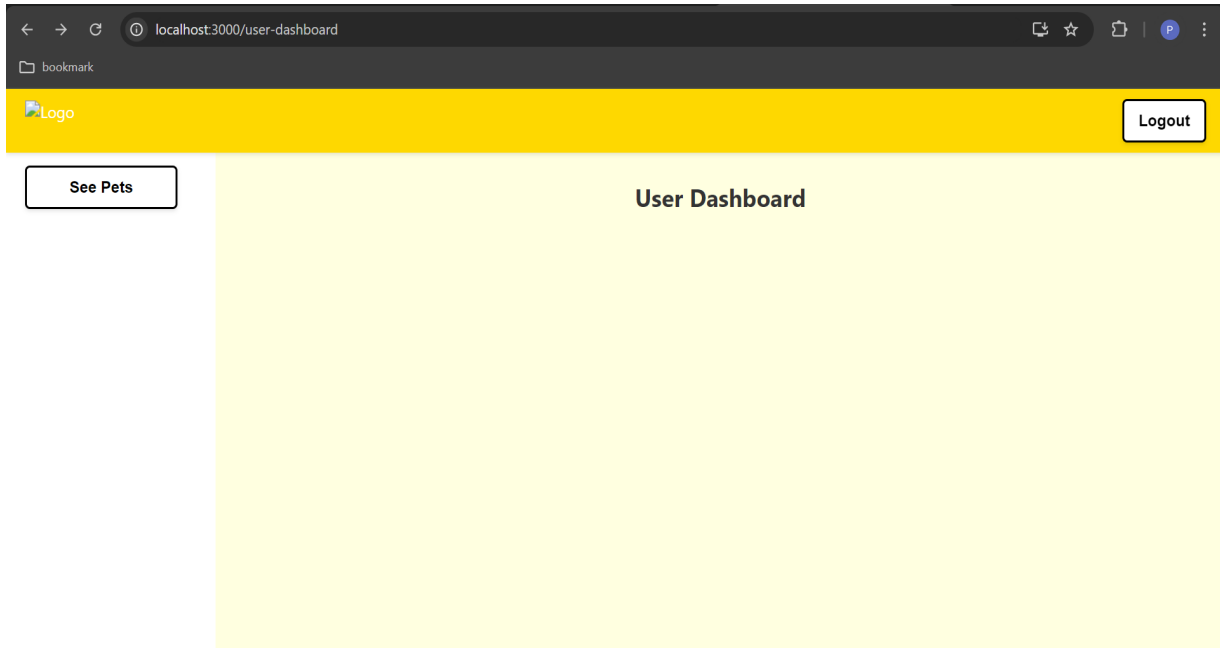
Figure 5.3: User Dashboard

See Pets by DOG Category

Figure 5.4: see Pets by DOG Category

See Pets by CAT Category
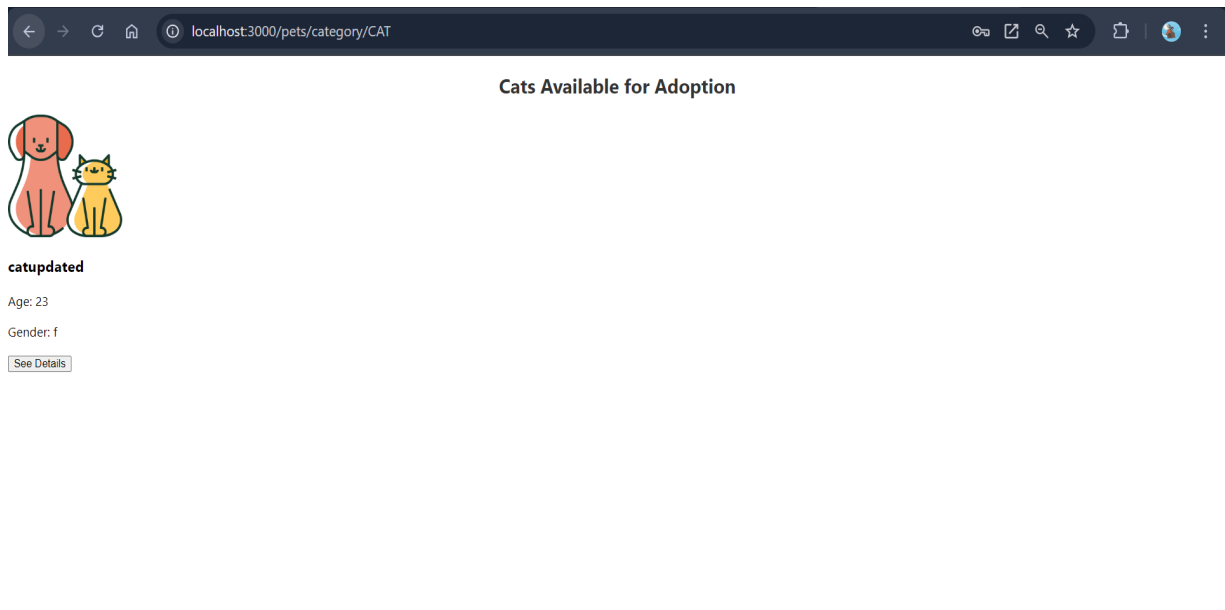
Figure 5.5: see Pets by CAT Category

See Pet Details Page
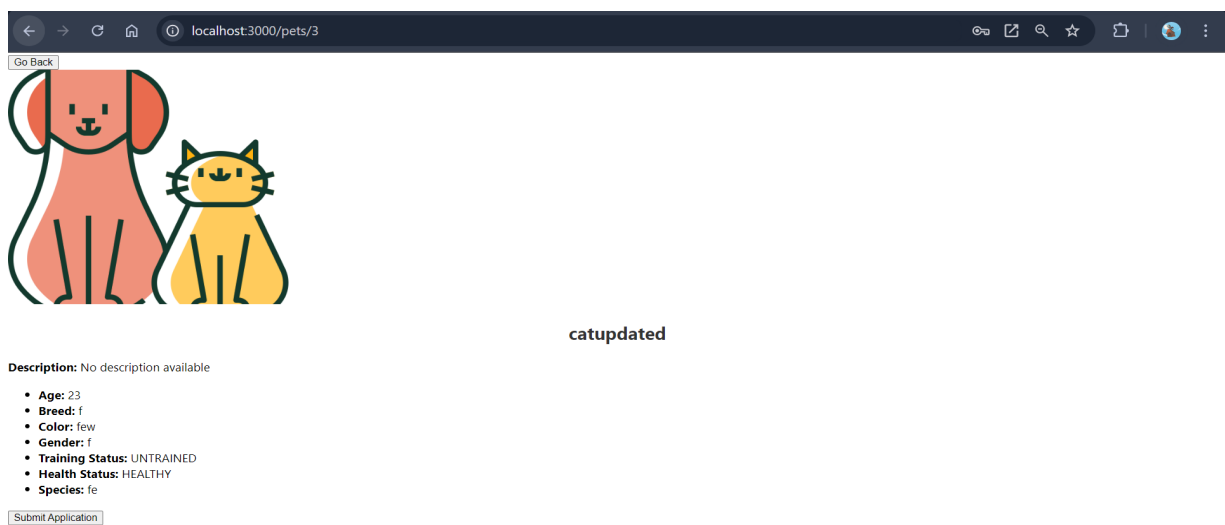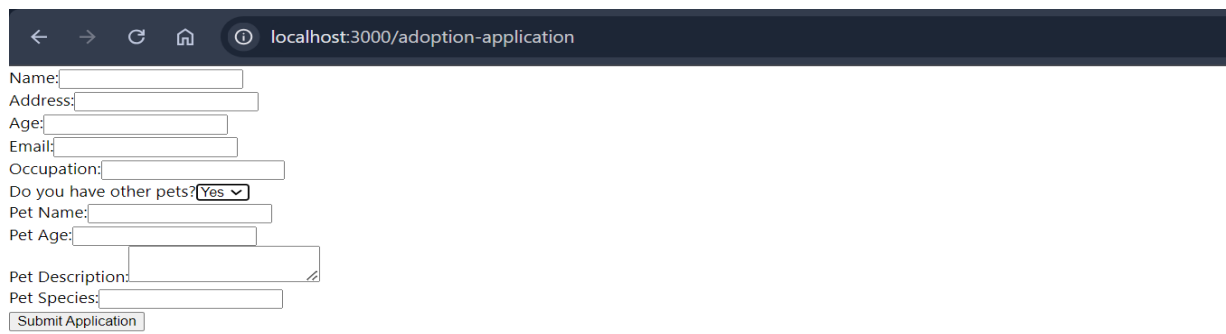
Figure 5.6: See Pet Details Page

Submit Application Form page

Figure 5.7: Submit Application Form Page

5.4   Shelter Dashboard
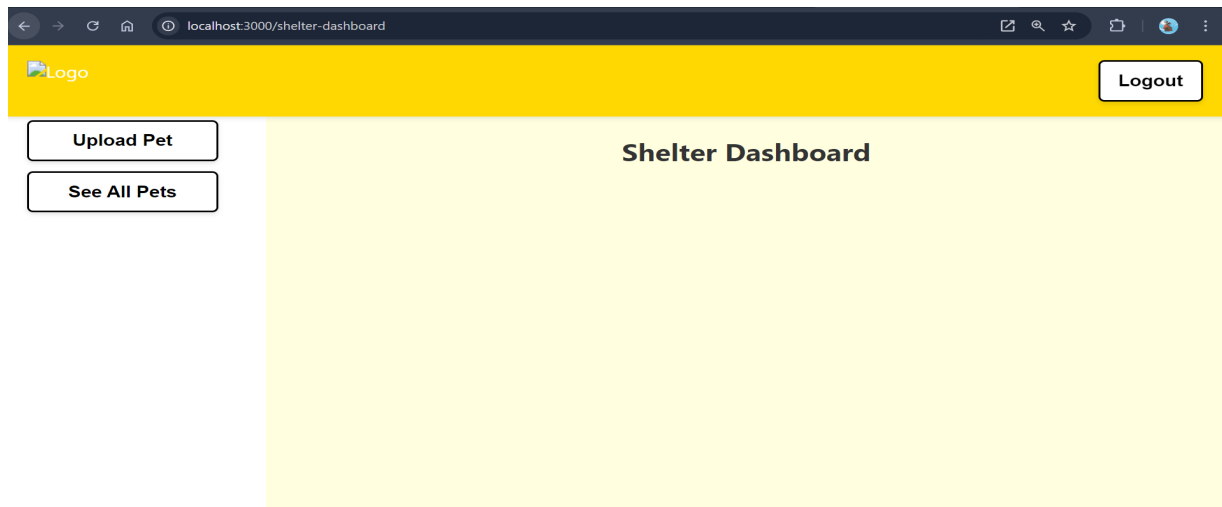
Figure 5.8: Upload Pet Page

Upload Pet
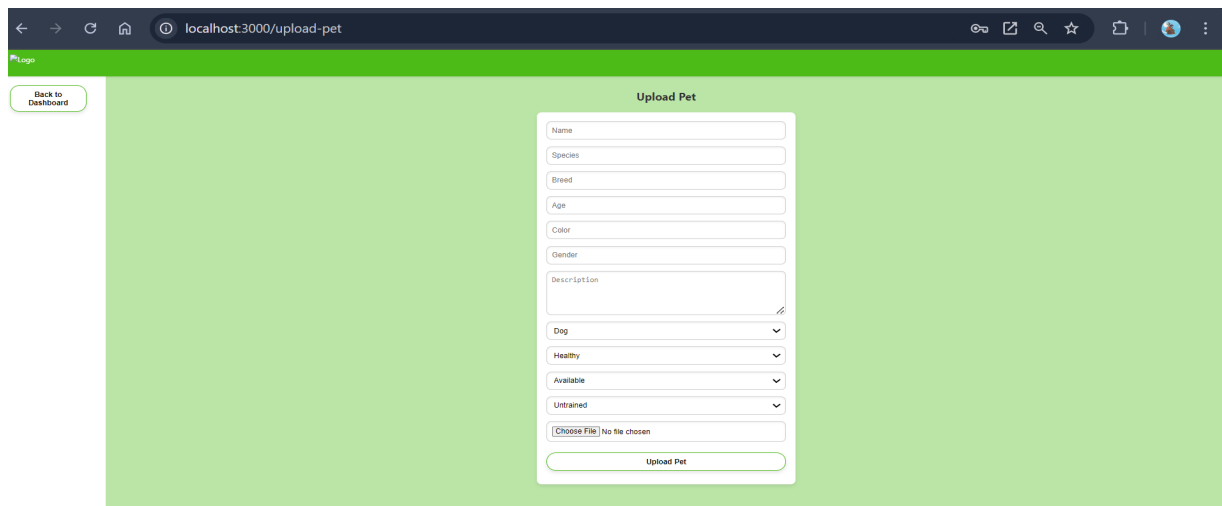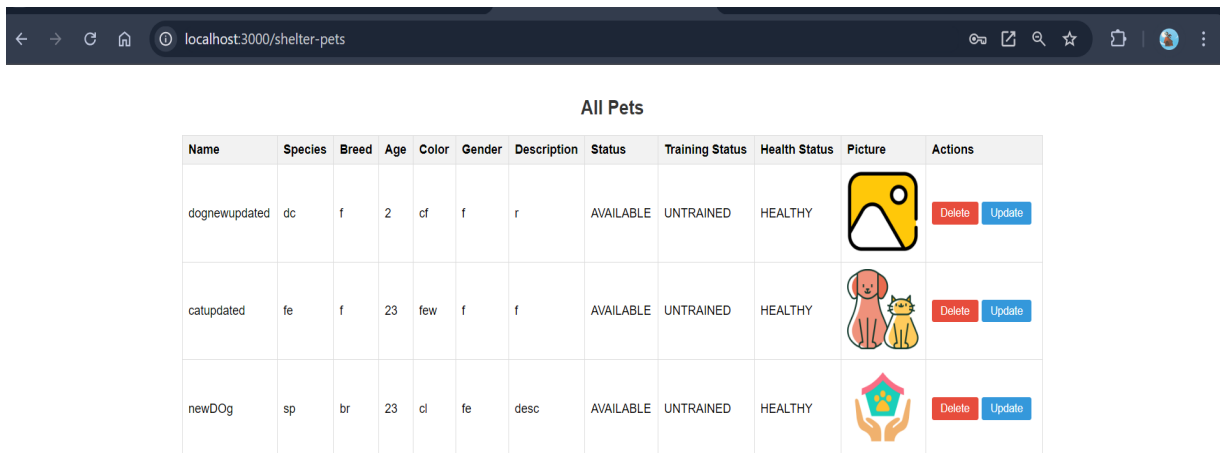
Figure 5.9: Upload Pet Page
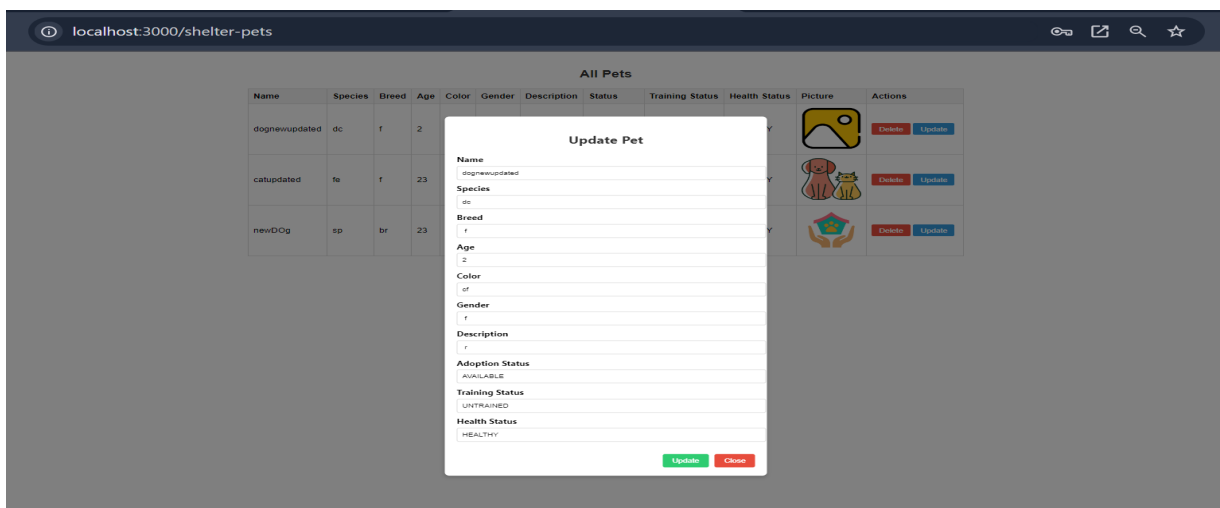
See All Pets Page

Figure 5.10: See All Pets Page

Update Page

Figure 5.11: Update Pet Page

Conclusion

The "Pawsitive Futures" platform effectively streamlines the pet adoption process through advanced web technologies. By offering role-based access, dynamic pet listings, and real-time tracking, the platform enhances efficiency for Admins, Shelters, and Users alike. This project not only simplifies adoption but also supports animal welfare, demonstrating the positive impact of technology in improving adoption experiences and connecting pets with loving homes.

The successful implementation of this project highlights the potential for technology to drive meaningful change in animal care and adoption, providing a model for future developments in similar domains. The platform's user-centric design ensures that it meets the needs of all stakeholders, fostering a more efficient and compassionate adoption process.

References

- Protected Routes in React-

  https://dev.to/collins87mbathi/reactjs-protected-route-m3j

- Animal WelFare Board of India Website-

  https://awbi.gov.in/

- Standard Protocol for Adoption Community Animals-

  https://awbi.gov.in/uploads/regulations/165597075151Adoption%20of%
  20community%20animals_0001.pdf