PROJECT REPORT

ON

HappyTails

BY

Sakshi Nikalje

SAVITRIBAI PHULE PUNE UNIVERSITY
MASTER IN COMPUTER APPLICATION
MAHARASHTRA EDUCATION SOCIETY's
INSTITUTE OF MANAGEMENT AND CAREER COURSES
(IMCC), PUNE-411038
2024-25

CERTIFICATE

This is to certify that the Sakshi Nikalje has completed the project work entitled "*HappyTails*" under my guidance. The report is submitted in partial fulfillment of M.C.A. Course for the Academic Year 2024-2025 as per the rules & prescribed guidelines of Savitribai Phule Pune University.

His/her work is found to be satisfactory and complete in all respects.

Dr. Jayashree Patil
(Internal Project Guide)

MAHARASHTRA EDUCATION SOCIETY'S
(SINCE 1860)

INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC) (AUTONOMOUS)

Approved by AICTE and Recognized by Savitribai Phule Pune University, Pune
IMCC Campus, 131, Mayur Colony, Kothrud, Pune 411038, Maharashtra, India | Ph.: 020-2546 6271 / 73 | e-mail: info.imcc@mespune.in | https://imcc.mespune.in

NAAC Accredited with Grade A+

Ref. No. MES IMCC /345/2024-25 **Date:03/04/2025**

CERTIFICATE

This is to certify that the Project Report entitled

" HappyTails "

is prepared by

Sakshi Nikalje

M.C.A. Semester IV Course for the Academic Year 2024-25 at M.E. Society's Institute of Management & Career Courses (IMCC), Pune - 411038.

M.C.A Course is affiliated to Savitribai Phule Pune University.

To the best of our knowledge, this is original study done by the said student and important sources used by him/her have been duly acknowledged in this report.

The report is submitted in partial fulfillment of M.C.A Course for the Academic Year 2024-25 as per the rules and prescribed guidelines of Savitribai Phule Pune University.

Dr. Ravikant Zirmite
Head, Dept of MCA MES IMCC

Dr. Santosh Deshpande
Director, MES IMCC

Internal Examiner

External Examiner

Acknowledgement

I would like to express my sincere gratitude to our internal project guide, Dr. Jayashree Patil, her exceptional organizational skills have been instrumental in keeping the project on track and ensuring timely delivery. Her guidance, expertise, and unwavering support have been invaluable throughout the project journey.

I am also deeply thankful to our Head of Department, Dr. Ravikant Zirmite, for his insightful inputs and continuous support. His wisdom and encouragement have greatly contributed to the success of our project.

Special thanks are due to our esteemed Director, Dr. Santosh Deshpande, and Deputy Director, Dr. Manasi Bhate, for their invaluable guidance and encouragement. Their vision and leadership have been a source of inspiration for us.

I extend my heartfelt appreciation to all individuals mentioned above for their strong belief in me and their dedication to our project's success.

INDEX

Chapter 1

INTRODUCTION

1.1. Institute profile

The Institute popularly known as IMCC was established in 1983 by M. E. Society for providing quality education and technical expertise at the Post Graduation Level in the Fields of Computers and Management. The Institute is recognized by SPPU under Section 46 of Pune University Act, 1974 and Section 85 of Maharashtra University Act, 1994. The Institute is located at 131, Mayur Colony, Kothrud, Pune-411 038 .

1.2. Abstract

HappyTails is a MERN-based web platform designed to streamline pet adoption and management by connecting pet owners, adopters, and administrators. The system provides a centralized and user-friendly solution where users can browse, list, and adopt pets, while admins can manage pet listings, adoption requests, and pet-related services. Additionally, the system integrates adoption request tracking, ensuring a transparent and structured process. The inclusion of pet services enhances the platform's functionality by providing users access to essential pet care resources. HappyTails aims to simplify pet adoption and promote responsible pet care in a digital-first approach.

1.3. Existing System and Need for System

1.3.1. Existing System

Currently, pet adoption and management are often handled through traditional methods such as social media posts, word-of-mouth, or offline pet shelters. But there are few limitations like limited service access, manual adoption process, no proper pet management, etc. Also there is no dedicated system where users can browse available pets and request adoptions efficiently.

1.3.2. Need for the System

HappyTails aims to provide a web platform with multiple features. Admins can manage pet listings, approve/reject adoption requests, and update available services. Users can submit adoption requests directly, ensuring a structured and efficient process. A secure login system (Admin & User) to provide a personalized dashboard for each role. This system will streamline pet adoption, improve accessibility to pet services, and provide a centralized platform for pet lovers.

1.4. Project Scope

The project involves developing a user-friendly website where users can browse available pets for adoption, submit adoption applications, and connect with shelters or pet owners. The platform

will include an intuitive admin panel for managing pet listings, adoption requests, and user accounts. It will enable both pet owners and adopters to create and manage their profiles, track adoption requests, and communicate seamlessly. The platform will integrate pet care services, such as grooming and daycare center listings, where users can explore service providers, view their offerings, and contact them. An e-commerce module will allow users to purchase pet food, toys, and accessories through a secure and user-friendly online store.

1.5. Operating Environment

1. Frontend
   - **Framework:** React.js
   - **State Management:** Context API
   - **Styling:** CSS
   - **Port:** localhost:3001
2. Backend
   - **Runtime Environment:** Node.js
   - **Framework:** Express.js
   - **Authentication:** JSON Web Token (JWT) for user and admin login
   - **Database:** MongoDB (petManagement database)
   - **ORM:** Mongoose
   - **Port:** localhost:5000

3. Database

- **Database Type:** NoSQL
- **DBMS:** MongoDB (Cloud: MongoDB Atlas or Local: MongoDB Compass)
- **Collections:**
  - users (for user and login)
  - pets (for pet listings)
  - adoptionrequests (for adoption form submissions)
  - products (for pet-related eCommerce items)
  - services (for service listings)

1.6. Brief description of technology used

1. Frontend (React.js)

- **React.js:** A JavaScript library for building dynamic UI components.
- **React Router:** Enables navigation between different pages without reloading.
- **CSS:** Used for styling and responsive design.

2. Backend (Node.js & Express.js)

- **Node.js:** A JavaScript runtime environment that allows running JavaScript outside the browser.
- **Express.js:** A lightweight framework for building REST APIs and handling server-side logic.

- **JWT (JSON Web Token):** Used for secure authentication of users and admins.

3. Database (MongoDB & Mongoose)
   - **MongoDB:** A NoSQL database to store user data, pets, products, and adoption requests.
   - **Mongoose:** An Object Data Modeling (ODM) library for interacting with MongoDB using JavaScript.

Chapter 2

PROPOSED SYSTEM

2.1. Study of Similar Systems

1. Petfinder

Overview: Petfinder is a widely used pet adoption website that helps users find pets for adoption from shelters and rescue organizations.

Similarities to HappyTails:

- Users can view pet profiles, including breed, age, and location.
- Adoption request submission process.
- Shelters and individuals can list pets.

Differences:

- Petfinder primarily serves as a search engine for rescue shelters, whereas HappyTails includes pet listings from both users and the admin.
- HappyTails has an integrated product shopping and payment system.
- HappyTails features admin-controlled service listings (Grooming & Daycare).

2. Adopt-a-Pet

Overview: Adopt-a-Pet is a pet adoption website where animal shelters, rescues, and pet owners can list pets available for adoption.

Similarities to HappyTails:

- Adoption listing and request system.
- Filtering options for finding pets by category.

- User registration and profile management.

Differences:

- HappyTails includes eCommerce and service booking functionalities.
- Admin has direct control over service management.
- Chewy is solely an eCommerce platform, while HappyTails integrates adoption and pet services.
- HappyTails has an admin panel for managing both pets and services.
- Provides categorized pet services.
- Users can browse available services before booking.

2.2. Feasibility Study

2.2.1. Technical Feasibility-

The project will be developed using the MERN (MongoDB, Express.js, React.js, Node.js) stack, ensuring scalability, efficiency, and maintainability. MongoDB will handle pet listings, adoption requests, user profiles, and product inventories efficiently.

2.2.2. Scalability-

The system can be expanded with more services (vet consultations, pet insurance, etc.). The platform can be easily operated and maintained, making it operationally feasible.

2.2.3. Operational Feasibility-

The platform is user-friendly, with a simple UI/UX for pet parents, admins, and service providers. Admins manage pet listings, adoption approvals, and product inventory. Users (Pet Parents) browse pets, request adoptions, and shop for pet products. Service Providers manage grooming and daycare services.

2.3. Objective of the System

The primary objective of HappyTails is to create a centralized, user-friendly, and efficient pet adoption and management platform using the MERN stack. The system aims to bridge the gap between pet owners, adopters, and administrators, making pet adoption smoother and more accessible. Few key objectives are for enabling pet listings, dashboard for admin and users, provide access for pet services, etc. It aims to create a reliable and efficient platform that simplifies the pet adoption process while also promoting responsible pet care.

2.4. Users of the system

- Admin :

  Role: Manages the entire platform, including pet listings, adoption requests, and product inventory. Admin can Approve, edit, and delete pet listings, approve or reject adoption requests, manage product listings in the eCommerce section, oversee pet services, Handle user complaints and platform maintenance, etc.

- Users (Pet parent) :

  Role: Browse, adopt, and add pets for adoption, as well as purchase pet products. They can Browse pet listings and view pet details, submit adoption requests for pets, add their own pets for adoption, purchase pet product, etc.

- Visitors (unregistered users) :

  Role: Explore services and product catalogs before signing up. They can view pet product listings in the products section, browse available pet services, register and login to access full features, etc.

Chapter 3

ANALYSIS AND DESIGN

3.1. System Requirements –

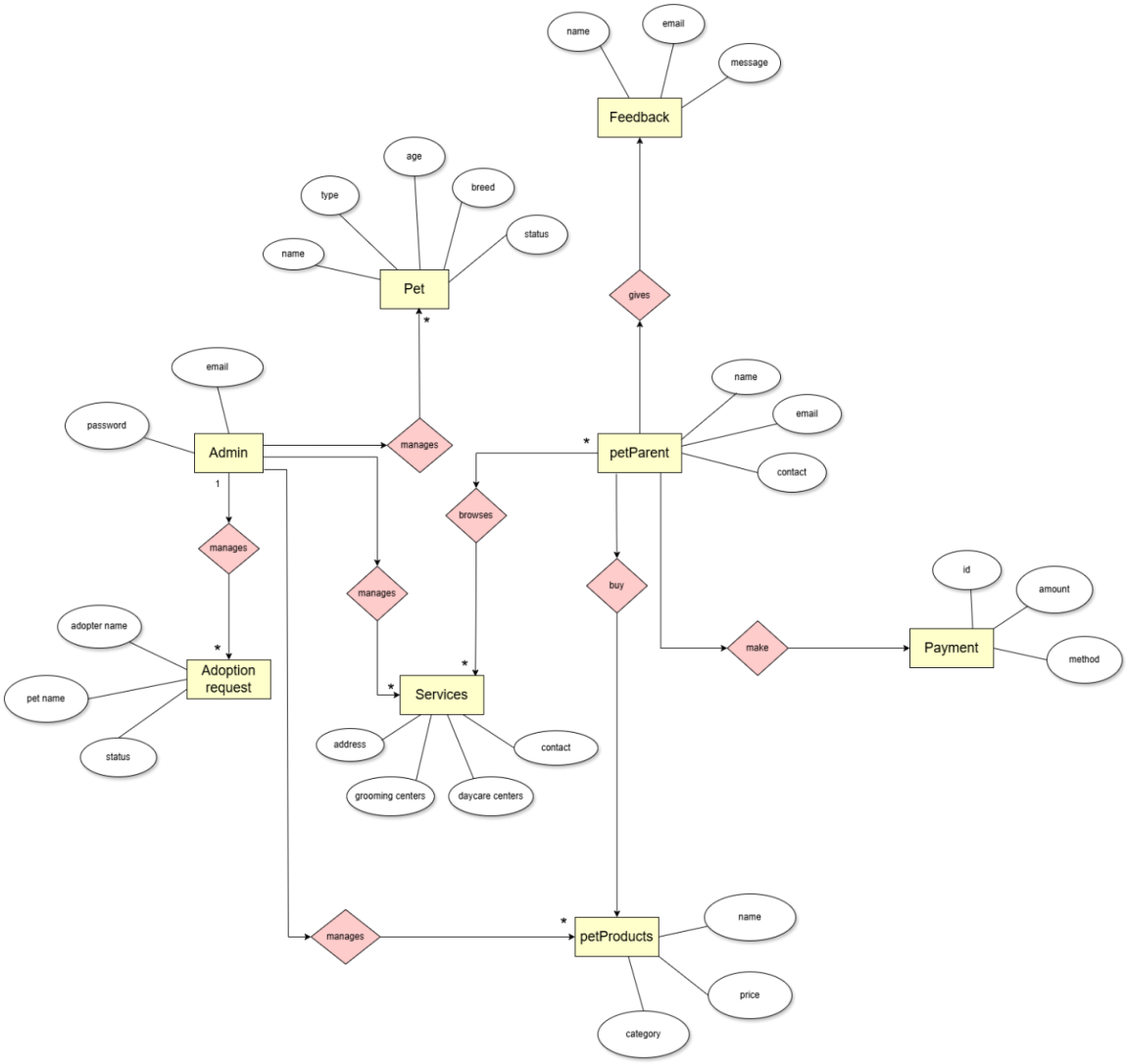
3.1.1 Functional Requirements

1. User management : Users and admins must be able to register and log in securely. Users can view and request adoption of listed pets. Admins can manage pet listings and adoption requests.
2. Pet listing and management : Users can add pets for adoption, including details like name, breed, age, gender, type, and description. Admins can add, edit, or remove pet listings. The system should display all pets (both user-added and admin-added) in a structured format.
3. Adoption request handling : Users can submit adoption requests for a pet. Admins can approve or reject adoption requests. Users can view the status of their adoption requests.
4. Service lisitng : Only admins can add, edit, and delete pet services (e.g., veterinary, grooming, training). Users can view available pet services.
5. Dashboard functionalities : Users should have a dashboard displaying their added pets and adoption requests. Admins should have a dashboard to manage pet listings, adoption requests, and services.
6. Feedback : Users can give a feedback after adoption.

3.1.2. Non-Functional Requirements

1. Performance : The system should handle multiple users simultaneously without lag. The database should efficiently store and retrieve pet and user data.
2. Security : User passwords must be securely hashed and stored. Only authorized users (admins) should be able to modify critical data. Adoption request details should be protected from unauthorized access.
3. Usability : The platform should have an intuitive and easy-to-navigate interface.
4. Scalability : The system should allow future enhancements, such as adding different types of pet, search filters, or chat features.
5. Reliability : The system should ensure minimal downtime and recover from failures quickly. All user actions (adoption requests, pet additions, etc.) should be accurately recorded.

3.2. Entity Relationship Diagram (ERD)



3.3. Table structure

1. users

```
petManagement> db.users.find()
[
  {
    _id: ObjectId('67e6a3a06e96f858b4064761'),
    name: 'sonal',
    email: 'sonal@gmail.com',
    phone: '1616161616',
    password: 'Sonal@123',
    role: 'user',
    createdAt: ISODate('2025-03-28T13:26:56.367Z'),
    __v: 0
  },
  {
    _id: ObjectId('67e6a3d26e96f858b406476f'),
    name: 'sakshi',
    email: 'sakshi@gmail.com',
    phone: '5656565656',
    password: 'Sakshi@123',
    role: 'user',
    createdAt: ISODate('2025-03-28T13:27:46.519Z'),
    __v: 0
  },
  {
    _id: ObjectId('67e6a3ec6e96f858b4064772'),
    name: 'yash',
    email: 'yash@gmail.com',
    phone: '9766468320',
    password: 'Yash@123',
    role: 'user',
    createdAt: ISODate('2025-03-28T13:28:12.643Z'),
    __v: 0
  },
  {
```

2. pets


```
{
  _id: ObjectId('67e939b12f59ff90088de941'),
  name: 'rebel',
  age: 3,
  breed: 'russian blue',
  type: 'Cat',
  image: '
/2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2P/wAARCAJYAVIDASIAAhEBAxEB/8QAGwAAAgMBAQEAAAAAAAAAAAAAAgMAAQQFBgf/x
AAAAAAAAAAAEDAgQF/8QAIhEBAQACAgMAAgMBAAAAAAAAAAECEQMhEjFBBDITIlFh/9oADAMBAAIRAxEAPwDpVsJhg0fV6Oo+4ER
Ew0H6vS/sC0VAcxUb4Sm6QgYTDf8vS/sCIYTDb4el/YE1ELz5JtdE/Q8NP8PS/sCW7CUA6G4ejHuwtJMaom3gq7Rko4PDBrgcPSJ
fVqJt/wDGEt+Ewpqfw9H+wJ9EktYTyVZcxJGyAaOBwoqQcNR/6YTKeDwhDpwtH/phHT1ad0dMwXDqmwpuDwmUfVaGv/xhJxeDwoY
P8AF6q+Ros4bCNscFS/sCzjCYYVXgYall1uwLoND2GTDh8lmeZxFtwmzRQweG/5aj/YEX0PDR/DUdPwBOAv6q44Vzs0S3CYb/l6P
VtwWGFUA4ajE/gCc8wGu1unAAw5NjA/A4aliB9WpQbeAJ30PCmPq1H/phHizwtduCjB0K62mnzjtNjW9qYtrWNAFZ4AgW4ioi7UJ
fBKFQ0yQRI2TX+A9EMBwBVcl0JL6hO5lHRENPmVGQKrx5I2aHzVoCnZ7ke6BtqpRFQHRtUPko4zTHkqpn2noqOhHmqhlD7JkdUVH
Kknko98Sd0RAGZKyZyTNlBbK82cLIKuUVmZU2GnhNkiqxjMrwZMqoMGekFF0QjQoouFFUDLRCIhCBaAj2KgHaFGWc5WLzzhCJFQd
IcR4qfPOEYAO/JYNRRcKwAhJhE0gygBoEkqDSQrAjTZXE6KBdQkVG8jPyVAaqVBFZnr8kQFp9UqwIE6qDQxqrjRRpvpuo6RhkCdw
KnmCmvJa9x6LquVUxJWipZzCs9EE6J73Q5gndRRNPEVAcwIPxVaVAikFqgzUobSQmoWm1wrtxMSiIMSuiDzB7HHdACRTg6ymCnFG
1VxJUjiUUJgODlbhIIVOHCQpMCRdBbG5qZG4UoB2aqwEHeCoxxD50DreSoRTrB3MQuo5r552pP+kYv3z/1FRX2r/quM9+/9RUWzJ
VnisdCFbPDHJLk6btKMT3nQhBT2+3aeiptmlFUvWp+RQ/dJKtSLYeCQqm91cey9Ett2+SijeJaVThAEKMJNO6a9vA09EgGh9pU8g
+QQucHVcs7oxHtHbRCgCq8ZCBqhltNgbuUDWlzgSJCKGucXHQaLoKogZTYCU4G1klrpLo0Dk0GBHopUiO0Vg6FDMg9VGGwUUTrGe
4oxrVf8yot2L6DXaHPpj+YFMME2CGsQKlH+oKy6HHmLLFqIkReyjDxAocwcIITARKigFqhOysqz4iq2UUurZ7P6ldpMaqqvjZ/UF
DoiAw4+s33aUdURl8kFIfWGHzCZWFgqn1VExUEbpjbV3pLGltRsyBK0Ae2d5IJVJLCUTLj0CCqfZlWyAzrCghA+kt8iFlIcyobEC
+qGSCQrCB3NQMFwqYboQ7L5KF0PsimUzxOHJGZLDl1IS2H2p6iU0pUAJyjNrF1cxA2VTCuAbeqigEtcfiFbyGva8aGxQvtldyN0Z
VH7TRXrsopbxNRnR4THeLyS3iHs/rCafFolIE6lXAUlQaKKCmyGu6GQjJyjN6qwLgcxCmrY12RAV7lkfiCpw0BUqDhYTzCt4mqBs
LS2SQg24g5KZRU4LmTyhYn40VQWlplT6U5jmw2YTQ11fA07sdCuqd1krYkusWwHkTdPrXpteNwqQDTLb81TjxEnZXbuwqfYHzRQ+
CFlBjL5rUAlRXzUbdsq1GbhRVRIIS6boLgRoU3QJLrVuUrqI+f9qD/NcZ79/wCoqKdpT/ieL98/9RUWzF7/ABkE0v61IsdyFMWIf
A02aeiAKwLaZ3yuRTxOKrEGGE7GFbhwlAqDUqcwhMscWkynMaGydyqqUwXB0IogJYmObwM/pSmHgICdJLGW+6iEVGzSKylt4OxWt
2o8kt7iXmDugg08ylN8NTzTWmw5CSlgjuieZVRna0EOIumUrh03VtYS7SAqpXDj6K/A4eKItCmxHRSL+iEOjIDdchbBYTyi6OBoh
```

3. adoptionrequests

```
petManagement> db.adoptionrequests.find()
[
  {
    _id: ObjectId('67dc0e6482c25c7e127f84f4'),
    petId: '67daa8a2886c0a6d97131dd0',
    petName: 'Lola',
    ownername: 'AWES',
    email: 'sakshi@gmail.com',
    phone: '87645678',
    address: 'aaa',
    reason: 'rrrrrrrrrr',
    __v: 0,
    status: 'approved',
    petImage: '/uploads/default.jpg'
  },
  {
    _id: ObjectId('67dc37680a76fe4276555ff0'),
    petId: '67d95d2e7694650682c4e04e',
    petName: 'diddo',
    ownername: 'Sakshi',
    email: 'sakshi@gmail.com',
    phone: '07038920649',
    address: 'F-2/12, State Bank Nagar',
    reason: 'aaaaaaaaaaaaa',
    __v: 0,
    petImage: '/uploads/default.jpg'
  },
  {
    _id: ObjectId('67dc3ec10a76fe427655602f'),
    petId: '67d91c9f29032220f8165d85',
    petName: 'lichi',
    ownername: 'sdf',
    email: 'sakshi@gmail.com',
    phone: '345678765',
    address: 'aaaaaaaaaa',
    reason: 'sssssssss',
```

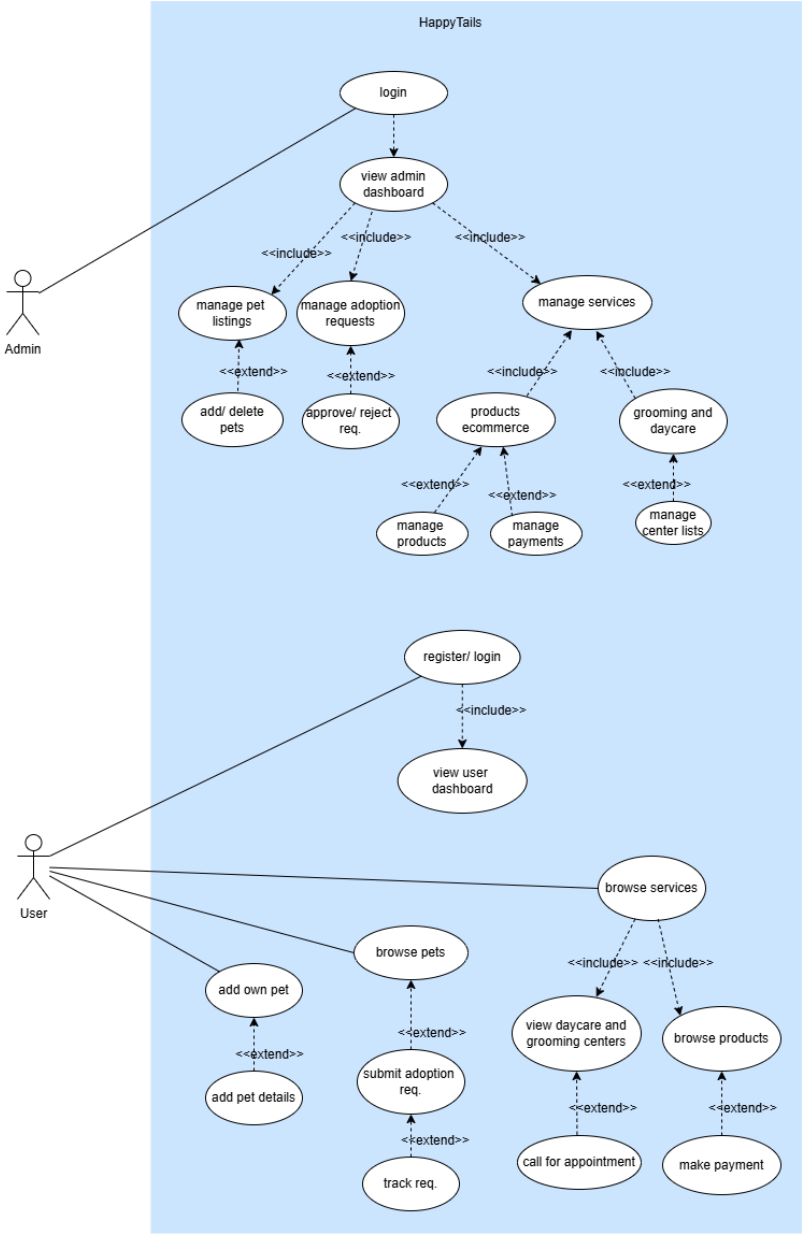
4. product

```
petManagement> db.products.find()
[
  {
    _id: ObjectId('67da77f47317dce24f4d7947'),
    name: 'Cat Food (1kg)',
    type: 'Food',
    price: 799,
    image: '/images/products/catfood.jpg'
  },
  {
    _id: ObjectId('67da77f47317dce24f4d7948'),
    name: 'Pet Bed',
    type: 'Accessory',
    price: 1499,
    image: '/images/products/petbed.jpeg'
  },
  {
    _id: ObjectId('67da77f47317dce24f4d7949'),
    name: 'Dog Leash',
    type: 'Accessory',
    price: 599,
    image: '/images/products/dogleash.jpeg'
  },
  {
    _id: ObjectId('67da77f47317dce24f4d794a'),
    name: 'Pet Shampoo',
    type: 'Accessory',
    price: 349,
    image: '/images/products/shampoo.jpeg'
  },
  {
    _id: ObjectId('67e582b6bd1ae0d6a44d7942'),
    name: 'Pet Sweater',
    type: 'Clothes',
    price: 799,
    image: '/images/products/sweater.jpg'
```

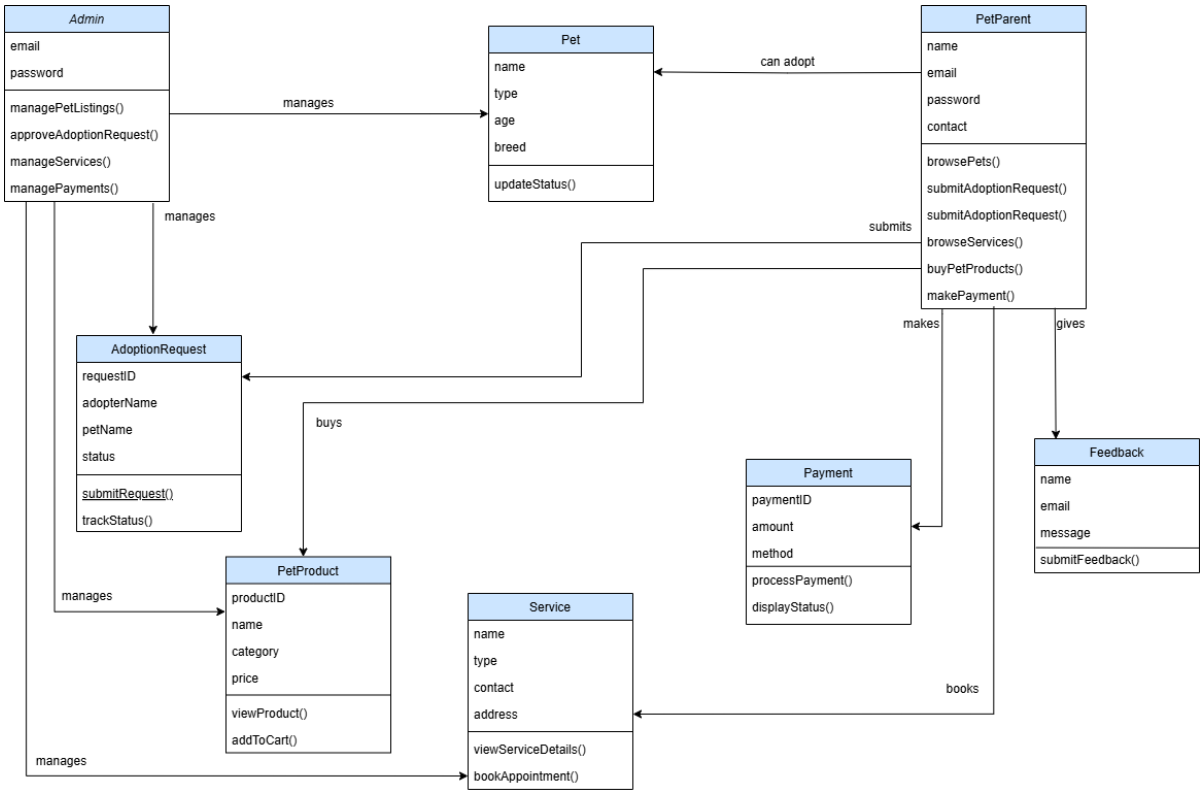
5. services

```
petManagement> db.services.find()
[
  {
    _id: ObjectId('67da9ea9886c0a6d97131d5c'),
    type: 'Grooming',
    name: 'Brush and clip ',
    address: 'Kothrud, Pune',
    contact: '7865477656',
    __v: 0
  },
  {
    _id: ObjectId('67da9ed8886c0a6d97131d5e'),
    type: 'Daycare',
    name: 'Tail stay',
    address: 'Warje, Pune',
    contact: '9965778121',
    __v: 0
  },
  {
    _id: ObjectId('67da9f13886c0a6d97131d67'),
    type: 'Grooming',
    name: 'Happy Pet spa',
    address: 'Deccan, Pune',
    contact: '8688853335',
    __v: 0
  },
  {
    _id: ObjectId('67da9f65886c0a6d97131d74'),
    type: 'Daycare',
    name: 'Paw palace',
    address: 'Karve nagar, Pune',
    contact: '7887446557',
    __v: 0
```

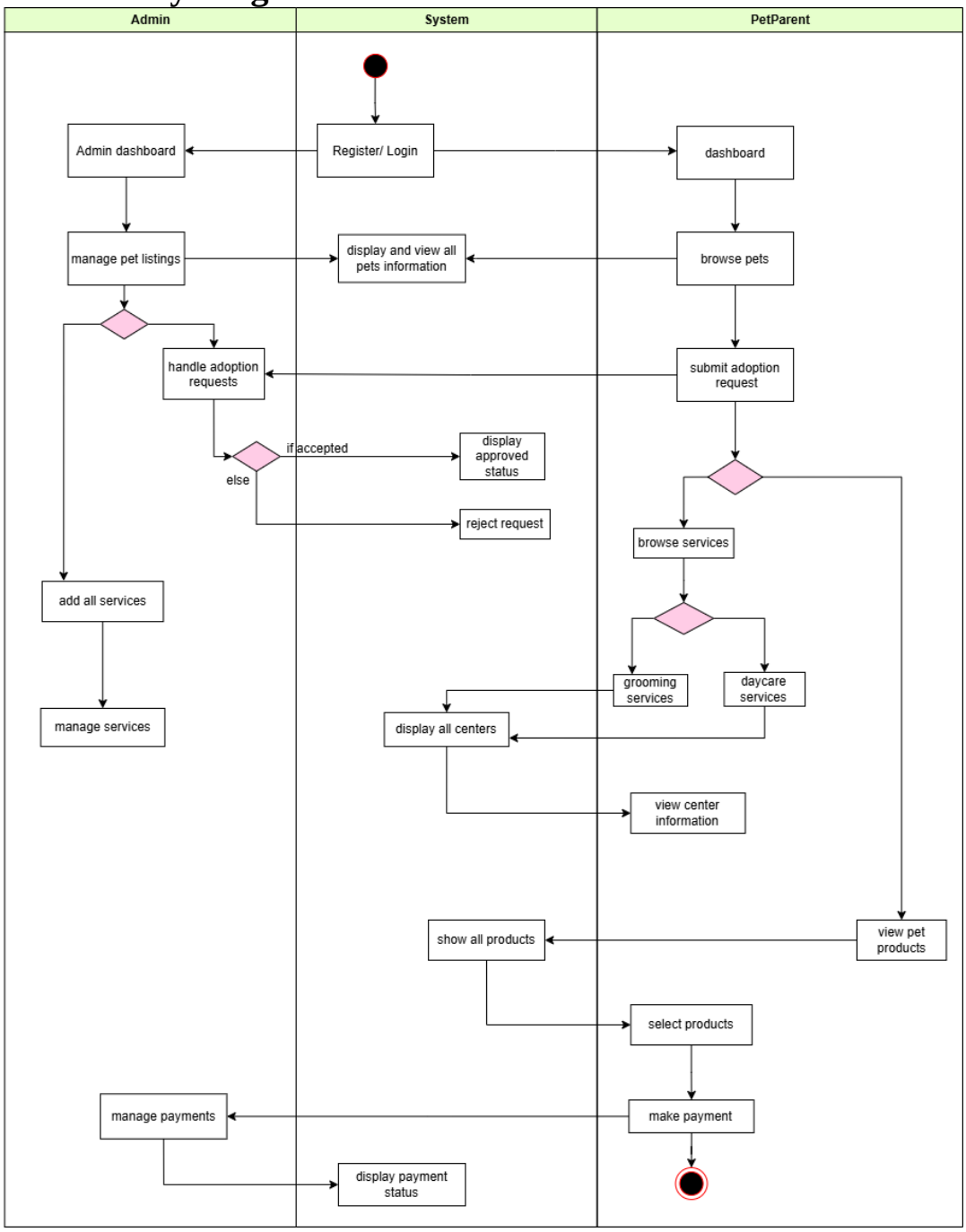
3.4. Use case Diagram



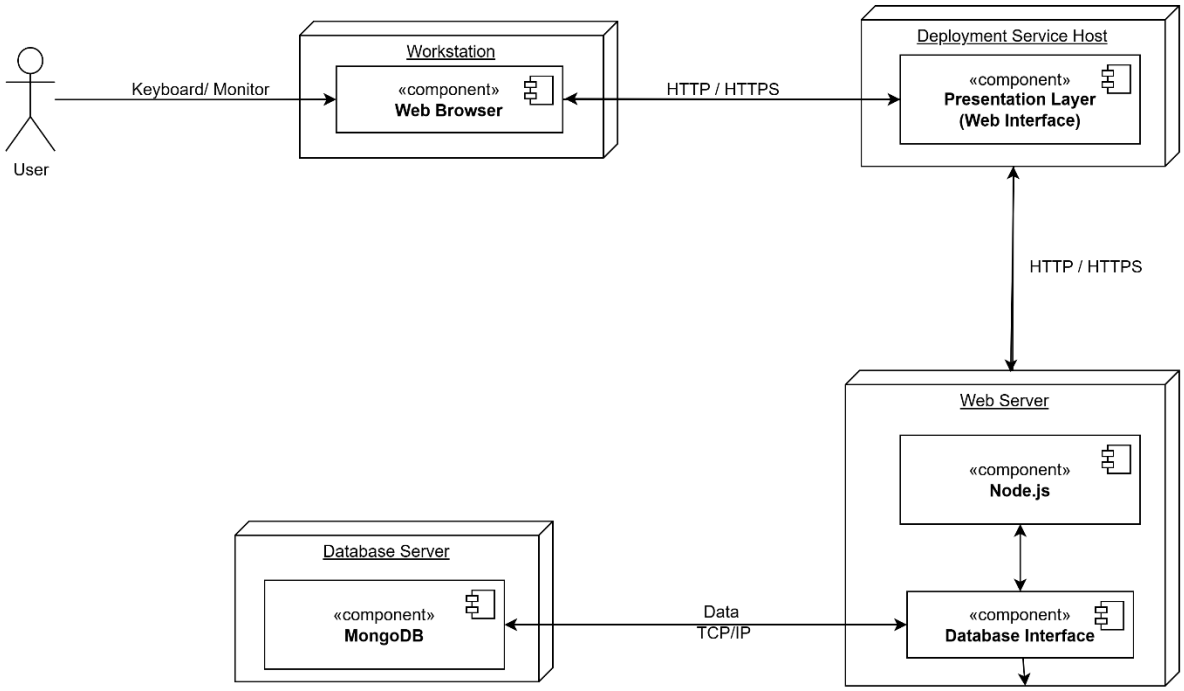
3.5. Class Diagram



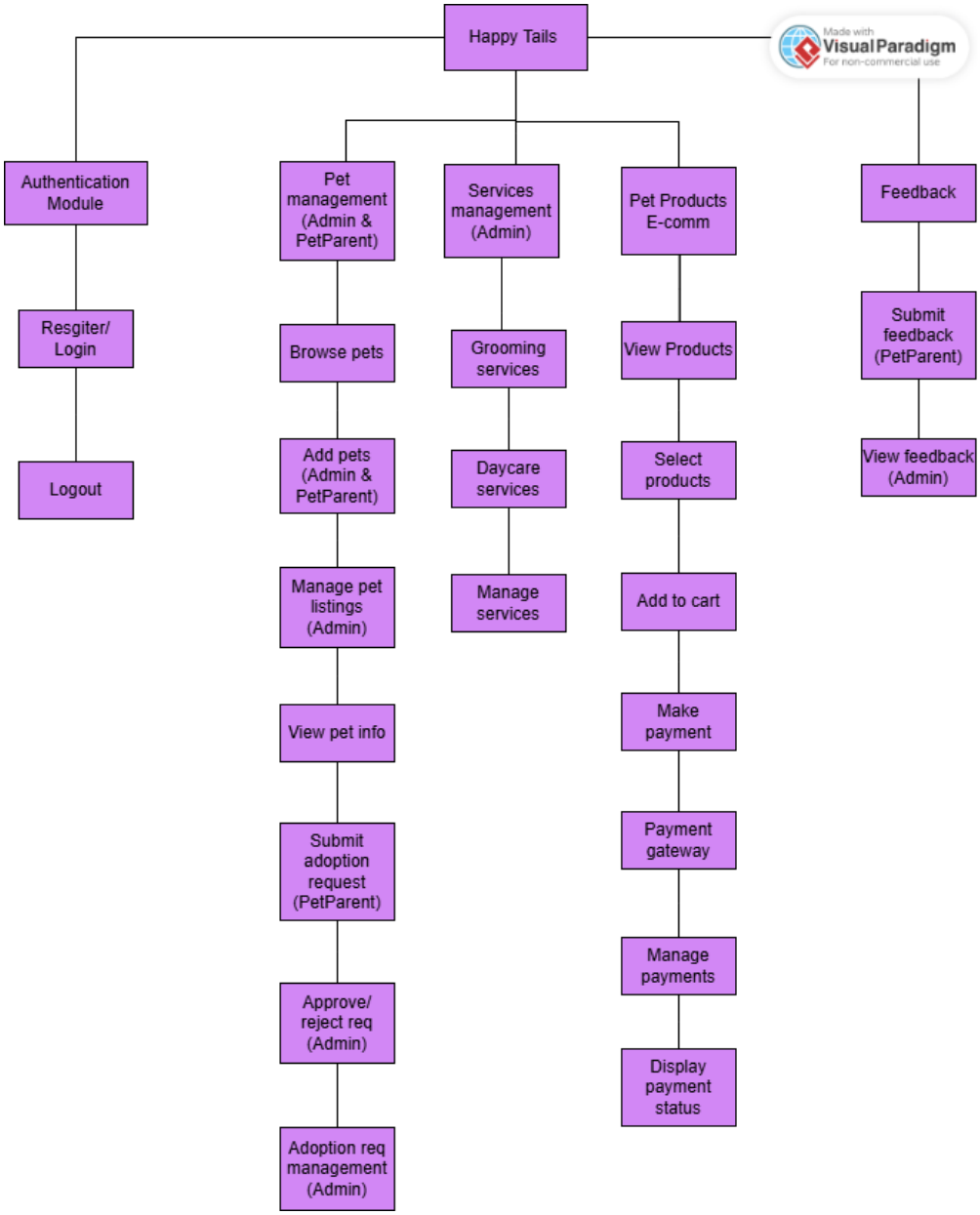
3.6. Activity Diagram



3.7. Deployment Diagram

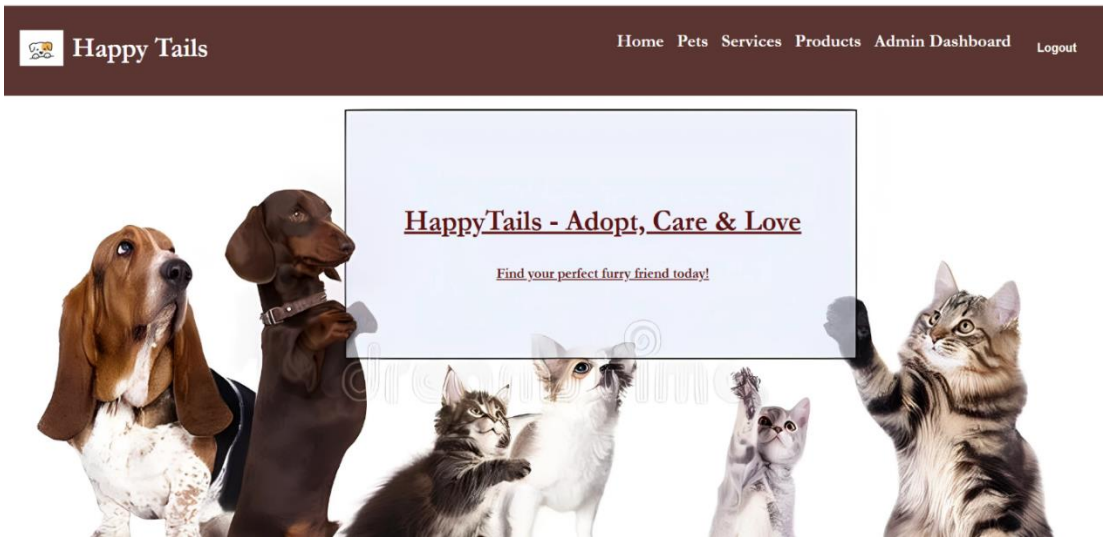


3.8. Module Hierarchy Diagram

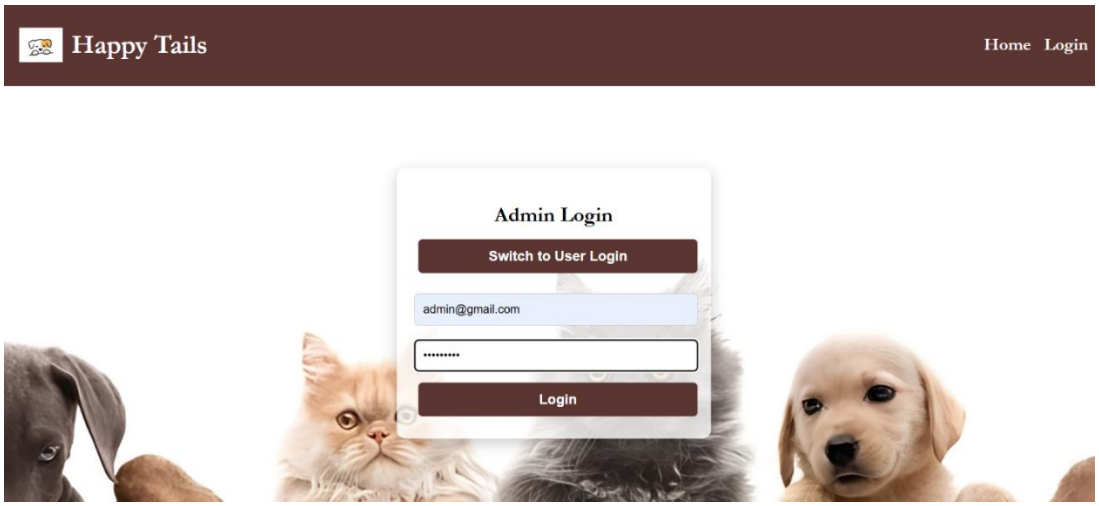


3.9. Sample Input & Output Screens

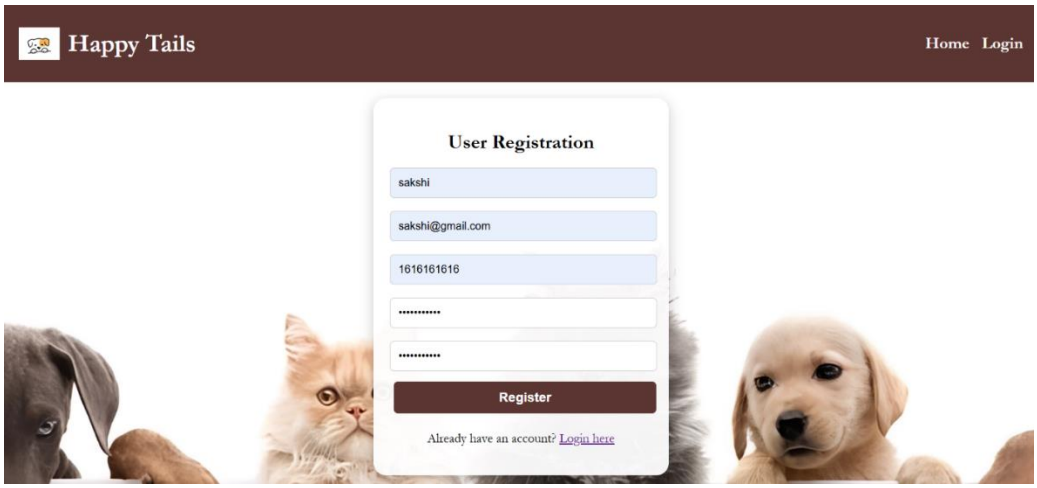
1. Home Page



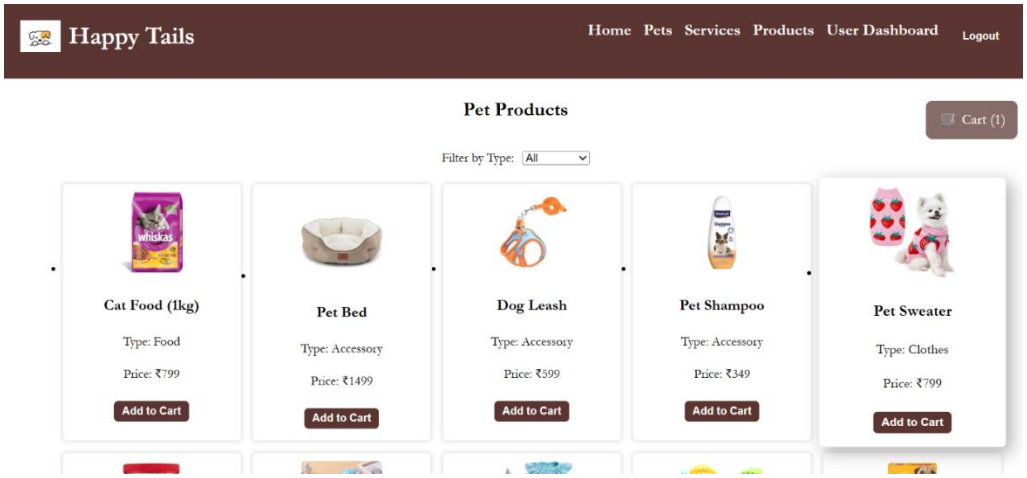
2. Login Page



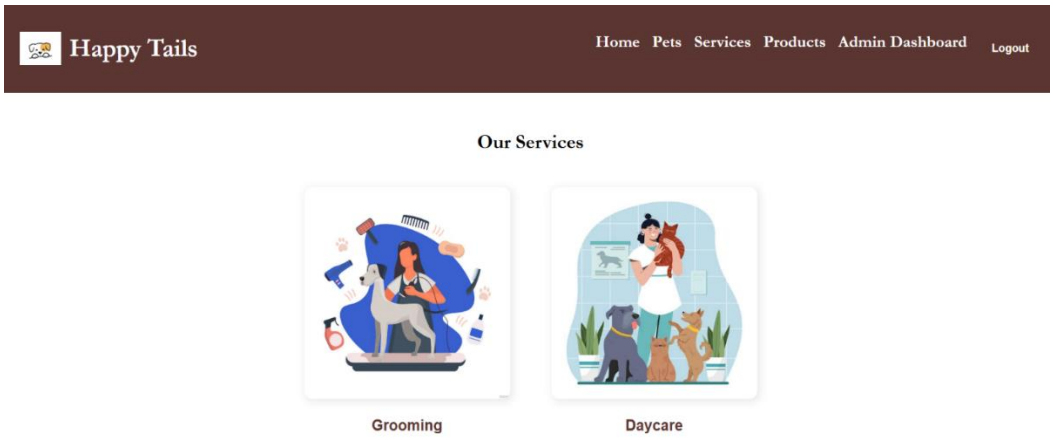
3. Registration Page

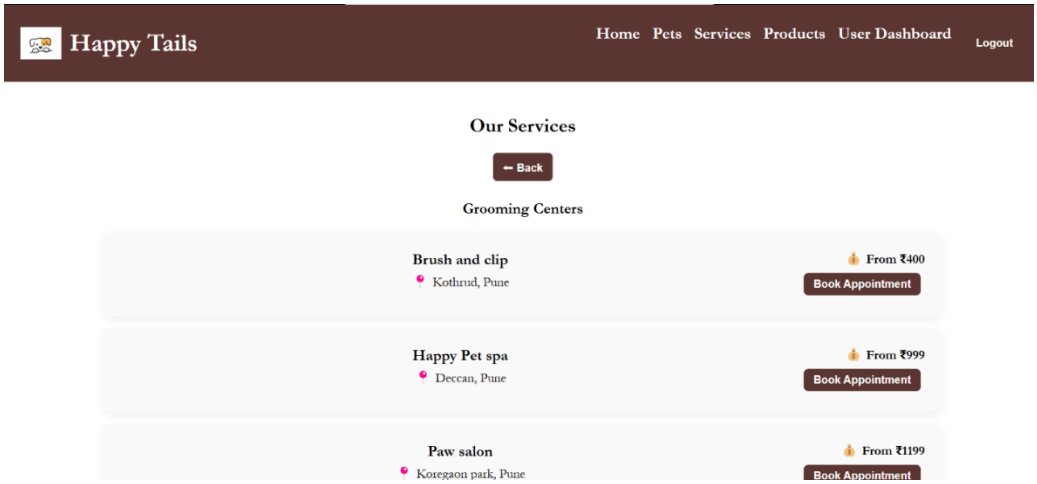


4. Products Page

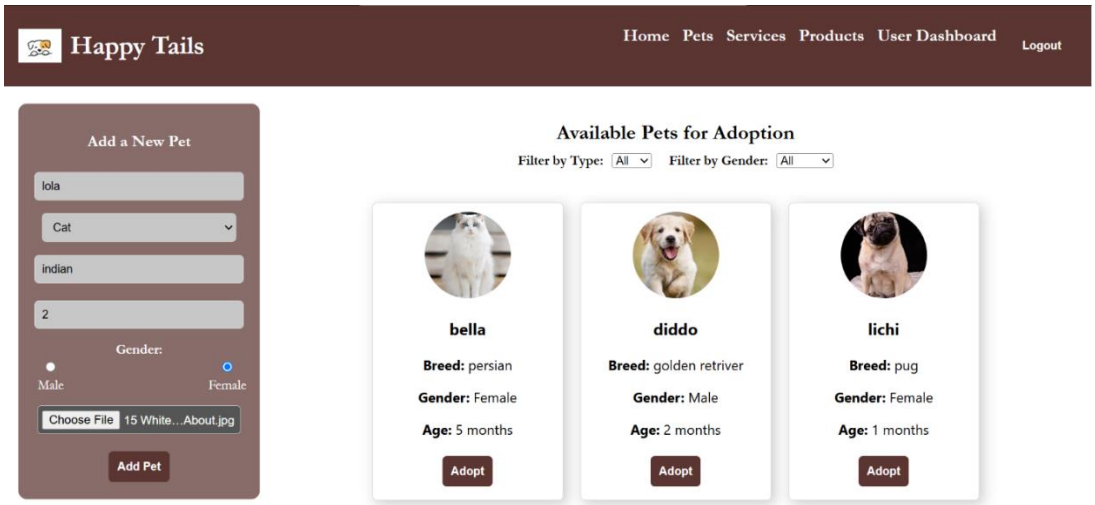


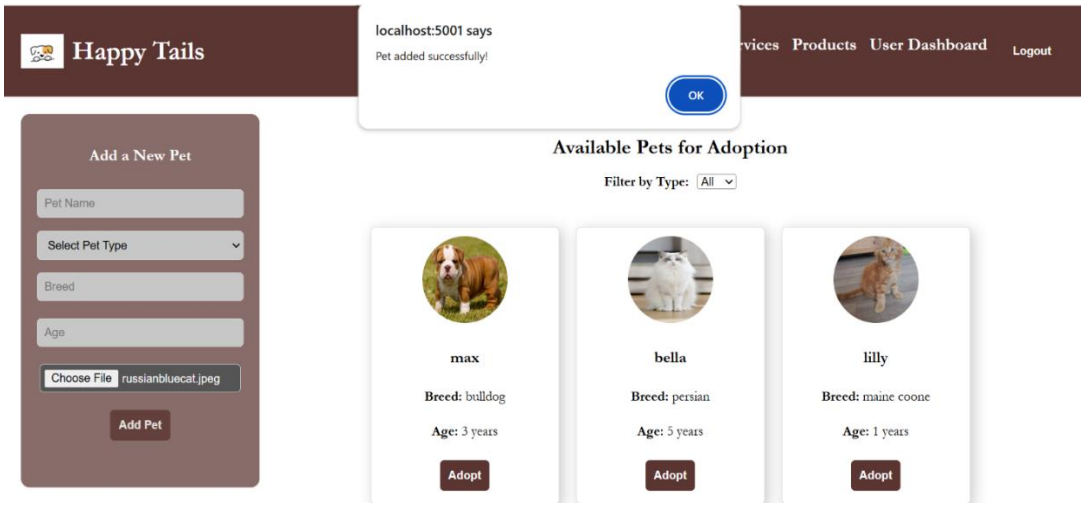
5. Services Page



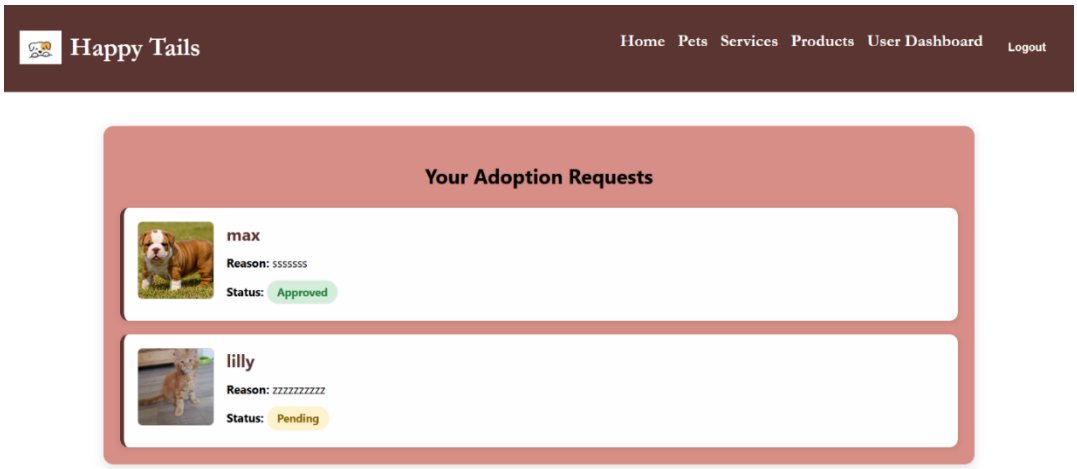


6. Pets Page

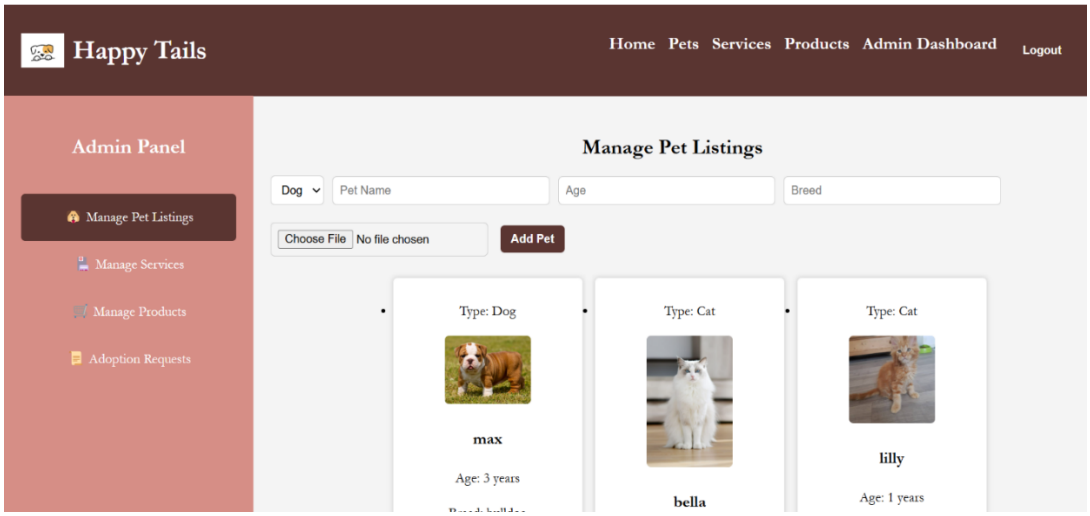

Happy Tails
Home Pets Services Products User Dashboard
Logout
Add a New Pet
lola
Cat
indian
2
Gender:
Male
Female
Choose File 15 White...About.jpg
Add Pet
Available Pets for Adoption
Filter by Type: All
Filter by Gender: All
bella
Breed: persian
Gender: Female
Age: 5 months
Adopt
diddo
Breed: golden retriver
Gender: Male
Age: 2 months
Adopt
lichi
Breed: pug
Gender: Female
Age: 1 months
Adopt


Happy Tails
localhost:5001 says
Pet added successfully!
OK
vices Products User Dashboard
Logout
Add a New Pet
Pet Name
Select Pet Type
Breed
Age
Choose File russianbluecat.jpeg
Add Pet
Available Pets for Adoption
Filter by Type: All
max
Breed: bulldog
Age: 3 years
Adopt
bella
Breed: persian
Age: 5 years
Adopt
lilly
Breed: maine coone
Age: 1 years
Adopt

7. User Dashboard



8. Admin Dashboard



Chapter 4

CODING

4.1. Code snippets

server.mjs –

```
// ✅ Connect to MongoDB
connectDB();

// ✅ Import Models
const { Schema, model } = mongoose;

// 🐶 Pet Schema
const PetSchema = new Schema({
  name: { type: String, required: true },
  age: Number,
  breed: String,
  type: String,
  image: String,
});
const Pet = mongoose.models.Pet || model("Pet", PetSchema);

// 🏥 Service Schema
const ServiceSchema = new Schema({
  type: String,
  name: String,
  address: String,
  contact: String,
});
const Service = mongoose.models.Service || model("Service",
  ServiceSchema);

// 🛍️ Product Schema
const ProductSchema = new Schema({
  type: String,
```

home.js -

```
const Home = () => {
  const [startIndex, setStartIndex] = useState(0);

  // Auto-slide every 10 seconds
  useEffect(() => {
    const interval = setInterval(() => {
      setStartIndex((prev) => (prev + 3) % testimonials.length);
    }, 3000);
    return () => clearInterval(interval);
  }, []);

  // Show 2 at a time
  const visibleTestimonials = [
    testimonials[startIndex],
    testimonials[(startIndex + 1) % testimonials.length],
    testimonials[(startIndex + 2) % testimonials.length],
  ];

  return (
    <div>
      {/* Hero Section */}
      <div
        className="home-container"
        style={{
          backgroundImage: "url('/images/wall4.png')",
          backgroundSize: "cover",
          backgroundPosition: "center",
          height: "100vh",
          display: "flex",
```

adoptionForm.js -

```
import React, { useEffect, useState } from "react";

const AdoptionRequestForm = ({ pet, onClose }) => {
  const [formData, setFormData] = useState({
    petName: "",
    ownername: "",
    email: "",
    phone: "",
    address: "",
    reason: ""
  });

  useEffect(() => {
    if (pet) {
      setFormData((prev) => ({
        ...prev,
        petName: pet.name || ""
      }));
    }
  }, [pet]);

  const handleChange = (e) => {
    setFormData((prev) => ({
      ...prev,
      [e.target.name]: e.target.value
    }));
  };

  const handleSubmit = async (e) => {
```

petListings.js –

```
const PetListings = () => {
  const [newPet, setNewPet] = useState({
    name: "",
    age: "",
    breed: "",
    type: "Dog",
    gender: "Male",
    image: "",
  });

  const fetchPets = () => {
    axios
      .get("http://localhost:5000/api/pets")
      .then((response) => {
        setPets(response.data);
        localStorage.setItem("pets", JSON.stringify(response.data));
      })
      .catch((error) => console.error("Error fetching pets:", error));
  };

  useEffect(() => {
    fetchPets();
  }, []);

  const handleAddPet = async () => {
    const { name, age, breed, type, gender, image } = newPet;
    if (!name || !age || !breed || !type || !gender || !image) {
      alert("All fields are required!");
      return;
```

Chapter 5

TESTING

5.1. Test Strategy

5.1.1. Introduction : This document outlines the test strategy for the MERN-based Pet Adoption and eCommerce Platform, ensuring quality, functionality, performance, and security.

5.1.2. Objectives :

- Verify core functionalities, including user authentication, pet adoption, and eCommerce operations.
- Ensure smooth user and admin interactions.
- Validate database operations (MongoDB).
- Test UI/UX for responsiveness and consistency.
- Ensure secure payment transactions.

5.1.3. Scope :

1. Functional Testing –

   - User authentication (login, logout, role-based access).
   - Pet listing, adding, editing, and deletion (admin and users).
   - Adoption request submission, approval, and status updates.
   - Product management (adding, deleting, viewing products).
   - Cart operations (adding, removing items, proceeding to checkout).
   - Payment gateway functionality.

2. Non-functional Testing –

   - Performance Testing: Load and stress testing for high traffic handling.
   - Security Testing: Prevent SQL injection, XSS, and ensure secure payment transactions.
   - Usability Testing: Ensure a smooth user experience across different devices and screen sizes.

5.1.4. Testing Approach :

- Manual Testing: UI, functional flows, and edge cases.
- Automation Testing: Use Selenium for UI testing and Jest/Mocha for backend unit testing.
- API Testing: Postman for endpoint validation.

5.1.5. Test Environment :

- Frontend: React.js
- Backend: Node.js, Express.js
- Database: MongoDB
- Test Frameworks: Jest, Mocha, Selenium
- Tools: Postman

5.1.6. Test Execution :

- Unit Testing: Individual component testing.
- Integration Testing: Validate communication between modules.
- System Testing: End-to-end testing.
- User Acceptance Testing (UAT): Validate against business requirements.

5.1.7. Risk and Mitigation Plan :

Risk	**Mitigation**
Payment failures	Implement retry and logging
Data inconsistency	Regular database backups
High traffic crash	Load testing and scaling

5.1.8. Conclusion : This strategy ensures a robust, scalable, and secure platform by covering all essential test areas. Testing will be continuously improved based on feedback and iterations

5.2 Unit Test Plan

5.2.1. Introduction - This document outlines the unit test plan for HappyTails, a MERN-based pet adoption and eCommerce platform. The goal is to ensure individual components function correctly before integration.

5.2.2. Scope - Unit tests will cover backend APIs, frontend components, database interactions, and authentication mechanisms

5.2.3. Testing tools –

- Backend: Jest, Supertest (for API testing)
- Frontend: React Testing Library, Jest
- Database: MongoDB memory server for isolated tests

5.2.4. Test Cases –

Test Case ID	Test Category	Test Case Description	Input Steps	Expected Output	Status
TC-01	Login	Test user login with valid credentials	1. Enter email, password 2. Click "Login"	Redirect to Pets page	Pass
TC-02	Login	Test login with invalid credentials	1. Enter wrong email or password 2. Click "Login"	Show "Invalid credentials" message	Pass

TC-03	Pet Listing	Add pet by admin	1. Login as admin 2. Go to Admin 3. Dashboard Add Pet 4. Submit	Pet is visible in Pets page	Pass
TC-04	Adoption	User sends adoption request	1. Login 2. Pets page 3. Click "Adopt" 4. Fill form Submit	Request is sent and shown in User Dashboard	Pass
TC-05	Adoption	Admin updates status	1. Admin Dashboard 2. View requests 3. Click "Approve/Reject"	Status updates for user	Pass
TC-06	Services	View all services as user	1. Login as user 2. Go to Services page	Both Grooming & Daycare services are visible	Pass
TC-07	Products	User adds product to cart	1. Products page 2. Click "Add to Cart" on a product	Product is shown in cart	Pass

TC-08	Navbar	Navbar visibility before login	1. Load homepage before login	Show: Home, Login	Pass
TC-09	Payment	User buys a product	1. Add to cart 2. Click "Buy Now" 3. Complete payment flow	Show "Payment Successful" and confirmation message	Pass
TC-10	Navbar	Navbar after user login	1. Login as user	Show: Home, User Dashboard, Pets, Products, Services, Logout	Pass
TC-11	Navbar	Navbar after admin login	1. Login as admin	Show: Home, Admin Dashboard, Pets, Products, Services, Logout	Pass

5.2.5. Conclusion - The above unit test plan ensures HappyTails is robust and functional. These test cases will be executed before deployment to guarantee smooth operation of the platform.

5.3. Acceptance Test Plan

1. Project Information

Project Name: HappyTails – Pet Adoption & E-Commerce Platform

Developed By: Sakshi Nikalje

Technology Stack: MERN (MongoDB, Express.js, React.js, Node.js)

Database: MongoDB (Database: petManagement)

2. Objective

The objective of HappyTails is to create a comprehensive platform for pet adoption and pet product sales. The system allows users to browse and adopt pets, purchase pet-related products, and access pet care services. Admins can manage pet listings, adoption requests, and product inventory efficiently.

3. Scope

- Users can register, log in, and manage their profiles.
- Pet adoption functionality, allowing users to browse, request adoption, and track request statuses.
- E-commerce features, including product listings, shopping cart, and payment gateway.

- Admin dashboard for managing pets, products, services, and adoption requests.
- Service listings (Grooming & Daycare) that users can view but only admins can add.

4. Test Environment

- Frontend: React.js (Running on localhost:3001)
- Backend: Node.js with Express.js (Running on localhost:5000)
- Database: MongoDB (petManagement database)
- Testing Tools: Postman (API Testing)
- Browser Compatibility: Chrome, Firefox, Edge

5. Acceptance Criteria

- Users should be able to register and log in successfully.
- Pets should be displayed correctly, and users should be able to view pet details and request adoption.
- Admin should be able to approve/reject adoption requests, and users should see the status updates.
- Products should be listed, added to the cart, and purchased via the payment gateway.
- Services should be visible to all users, but only admins should manage them.

- Admin should be able to add/edit/delete pets, products, and services.
- UI should be responsive and work on desktop and mobile screens.

6. Test Data

- User Login: Test with valid and invalid credentials.
- Pet Adoption: Submit requests with valid details and check status updates.
- Product Purchase: Add items to the cart, remove items, and proceed with payment.
- Admin Management: Add pets, approve adoption requests, and delete products.

7. Test Cases

Test Case ID	**Feature**	**Scenario**	**Preconditions**	**Test Steps**	**Expected Outcome**	**Status**
TC-12	Login System	User logs in successfully	User has a valid account	1. Go to Login 2. Enter email/password 3. Click Login	User is redirected to the User Dashboard	Pass

TC-13	Pet Listing	User adds a pet	User is logged in	1. Go to Pets 2. Fill Add Pet form 3. Click Add Pet	New pet appears in the Pets list	Pass
TC-14	Adoption Process	Admin reviews and approves request	Admin is logged in, requests exist	1. Go to Admin Dashboard 2. View Requests 3. Click Approve	Request status updates to "Approved"	Pass
TC-15	Services	Admin adds a grooming center	Admin is logged in	1. Go to Admin Dashboard 2. Add Service (Grooming) 3. Submit	Service appears under Grooming	Pass
TC-16	Product Management	Admin adds a product	Admin is logged in	1. Go to Admin Dashboard 2. Add product 3. Submit	Product appears on Products page	Pass
TC-17	Shopping Cart	User adds/removes product from cart	User is logged in	1. Go to Products 2. Click Add to Cart	Product appears/gets removed from cart	Pass

				3. View cart 4. Click Remove		
TC-18	Payment Gateway	User completes a purchase	User has items in cart	1. Go to Cart 2. Click Buy Now 3. Complete payment	Show "Payment Successful", "Order Confirmed"	Pass
TC-19	Adoption Status	User sees updated status	Admin has taken action	1. Submit adoption request 2. Admin approves 3. User views dashboard	Status shows "Approved" or "Rejected"	Pass

8. Deliverables

- Test cases and execution report.
- Bug report (if any).
- Deployment-ready system after acceptance testing.
- Final documentation for functionality and database structure.

9. Risk Assessment

- Data Loss**:** Regular database backups needed.
- Security Issues: Implement authentication and authorization properly.
- Payment Gateway Failures: Use a reliable provider and test multiple scenarios.
- Performance Issues: Optimize API calls and database queries.
- User Experience: Ensure a smooth UI with no major usability issues.

5.4. Test Cases

1. Authentication

Test Case ID	Description	Input	Expected output	Status
TC-20	Login with valid credentials	Correct email & password	Successful login, redirect to respective dashboard	Pass
TC-21	Login with invalid credentials	Wrong email or password	Error message: "Invalid credentials"	Pass
TC-22	Register new user	Unique email, valid details	Account created, redirected to login page	Pass

2. Pet Listing & Adoption Requests

Test Case ID	Description	Input	Expected output	Status
TC-23	Add a new pet (User/Admin)	Valid pet details	Pet appears in the pet page	Pass
TC-24	Submit an adoption request	Fill form & submit	Request appears in admin dashboard	Pass

TC-25	Admin approves/rejects request	Click approve/reject	Status updates in user dashboard	Pass

3. Product Management

Test Case ID	Description	Input	Expected output	Status
TC-26	Admin adds a new product	Valid product details	Product appears in the products page	Pass
TC-27	Admin deletes a product	Click delete	Product removed from listing	Pass
TC-28	User views products	Navigate to products page	Product list loads properly	Pass
TC-29	Add product to cart	Click "Add to Cart"	Product appears in cart	Pass
TC-30	Checkout process	Click "Buy Now" & enter payment details	Payment gateway opens	Pass
TC-31	Payment success	Correct details and make payment	Confirmation message & delivery time	Pass

4. Services Management

Test Case ID	Description	Input	Expected output	Status
TC-32	Admin adds a new service	Valid service details (name, category, details)	Service appears on the Services page	Pass
TC-33	Admin tries to add service without address	Leave address blank	Error: "Address is required"	Pass
TC-34	Admin deletes a service	Click "Delete" on a service	Service removed from the Services page	Pass
TC-35	Book appointment	Click “Book appointment”	Contact of service center and whatsapp chat icon appears	Pass

5. Admin Dashboard

Test Case ID	Description	Input	Expected output	Status
TC-36	View all adoption requests	Go to adoption requests section	List of all pending/completed requests	Pass
TC-37	Manage pets	Add/delete pets	Changes reflect in user view	Pass
TC-38	Manage products	Add/delete products	Changes reflect in user view	Pass
TC-39	Manage services	Add/delete services	Changes reflect in user view	Pass

5.5 Defect report

1. Project Information

Project Name: HappyTails – Pet Adoption & E-Commerce Platform
Reported By: Sakshi Nikalje
Reported Date: 02/04/2025
Tested Environment:

- Frontend: React.js (Running on localhost:3001)
- Backend: Node.js with Express.js (Running on localhost:5000)
- Database: MongoDB (petManagement database)
- Browser: Chrome, Firefox
- Testing Tools: Postman

2. Defect summary

- Total Defects Found: [Number of Defects]
- Severity Levels: Critical, Major, Minor

3. Defect list

- **Large Pet Images Not Getting Added**

  Description: When attempting to add a pet with a large image file (e.g., high resolution or large file size), the image upload fails or the form submission is blocked.

Possible Causes:

- Backend/server not handling large file sizes.
- Missing file size validation or compression.
- Frontend timeout during upload.

Severity: High

Suggested Fix:

- Increase file size limit on backend (e.g., using multer for Node.js).
- Add frontend validation and compression before upload.
- Show user-friendly error message on failure.

- **No Notification on Updated Adoption Request Status**

  Description: When the admin updates the status of an adoption request (e.g., Approved/Rejected), the user is not notified, and no real-time or visual update is shown in the user dashboard.

  Possible Causes:

  - Missing socket or polling logic to reflect real-time updates.
  - No toast/alert/notification system implemented.
  - API response not linked to UI update.

  Severity: Medium

Suggested Fix:

- Implement real-time updates using WebSockets or polling.
- Add toast or alert notifications in the UI for status changes.
- Ensure status field in UI reflects latest database state.

4. Conclusion & Next Steps

- Developers will investigate and fix the defects based on priority.
- Testers will re-execute test cases after fixes to confirm resolution.
- A retest report will be generated post-fix implementation.

Chapter 6

LIMITATIONS OF PROPOSED SYSTEM

Limitations :

4.1. No Real-Time Notifications –

- Users do not receive real-time alerts when an adoption request is approved/rejected.
- Admins are not notified immediately when a new pet or adoption request is added.

4.2. No AI-Based Recommendations –

- The system does not suggest pets based on user preferences.
- Product recommendations are not personalized.

4.3. No Multi-Language Support –

- The platform supports only one language (English), limiting accessibility for non-English speakers.

Chapter 7

PROPOSED ENHANCEMENTS

Future Enhancements :

1. Real-Time Notifications

   Enhancement: Use WebSockets (Socket.io) or Firebase to send real-time updates.

   Benefit:

   - Users get instant notifications when their adoption request is approved/rejected.
   - Admins receive alerts when a new adoption request is submitted.

2. AI-Based Pet & Product Recommendations

   Enhancement: Implement Machine Learning (ML) algorithms to suggest pets and products based on user behaviour.

   Benefit:

   - Personalized pet recommendations based on breed, age, and past searches.
   - Users see pet products relevant to their browsing history.

Chapter 8

CONCLUSION

Conclusion :

The HappyTails Pet Adoption & eCommerce Platform is a MERN-based web application designed to streamline pet adoption and pet product purchases. The system enables users to browse and adopt pets, while also offering an admin dashboard for managing pet listings, adoption requests, products, and services. However, with future enhancements like automated adoption processing, role-based access control, multi-language support, and a mobile app, it can become a more user-friendly and scalable platform for pet lovers and pet service providers. This project demonstrates how technology can bridge the gap between pet seekers and pet providers, promoting responsible pet adoption and enhancing pet care services. The integration of admin-controlled pet and service management ensures that only verified listings are displayed, increasing trust and transparency. By leveraging MongoDB, Express.js, React, and Node.js, the platform maintains a robust and scalable architecture that can handle increasing user traffic and data efficiently. Future integrations such as geo-location-based pet searches and AI-powered pet recommendations can further enhance user engagement and personalization.

Chapter 9

Bibliography

Bibliography :

a) **MongoDB Documentation**
MongoDB, Inc. (n.d.). MongoDB Manual. Retrieved from https://www.mongodb.com/docs/manual/

b) **Express.js Guide**
Express.js. (n.d.). Express - Node.js web application framework. Retrieved from https://expressjs.com/

c) **React Official Documentation**
Meta. (n.d.). *React* – A JavaScript library for building user interfaces. Retrieved from https://reactjs.org/docs/getting-started.html

d) **Node.js Documentation**
OpenJS Foundation. (n.d.). Node.js. Retrieved from https://nodejs.org/en/docs/

e) **Mongoose ODM**
Mongoose. (n.d.). Mongoose: Elegant MongoDB object modeling for Node.js. Retrieved from https://mongoosejs.com/

f) **React Router**
Remix Software. (n.d.). React Router Documentation. Retrieved from https://reactrouter.com/

Chapter 10

USER MANUAL

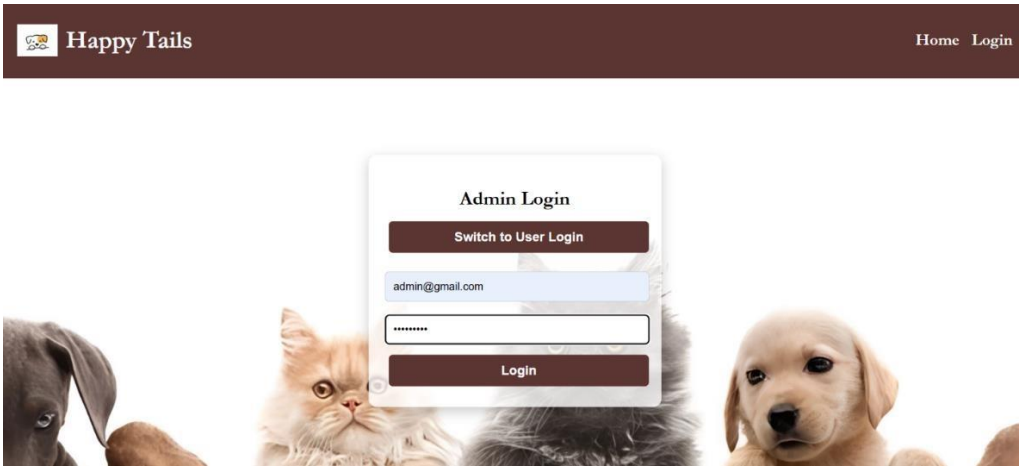
1. Registration & Login

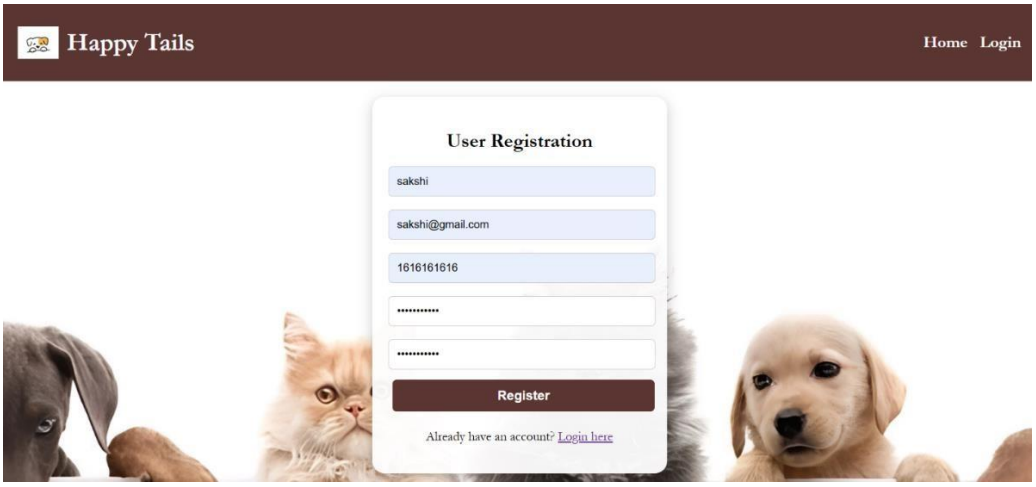
1.1 Registration Screen

- Purpose: To allow new users to create an account.
- Fields:
  - Username: Required, must be unique.
  - Email: Required, must be a valid email format.
  - Password: Required, minimum 6 characters.
- Validations:
  - All fields are mandatory.
  - Proper error messages for duplicate emails or weak passwords.

1.2 Login Screen

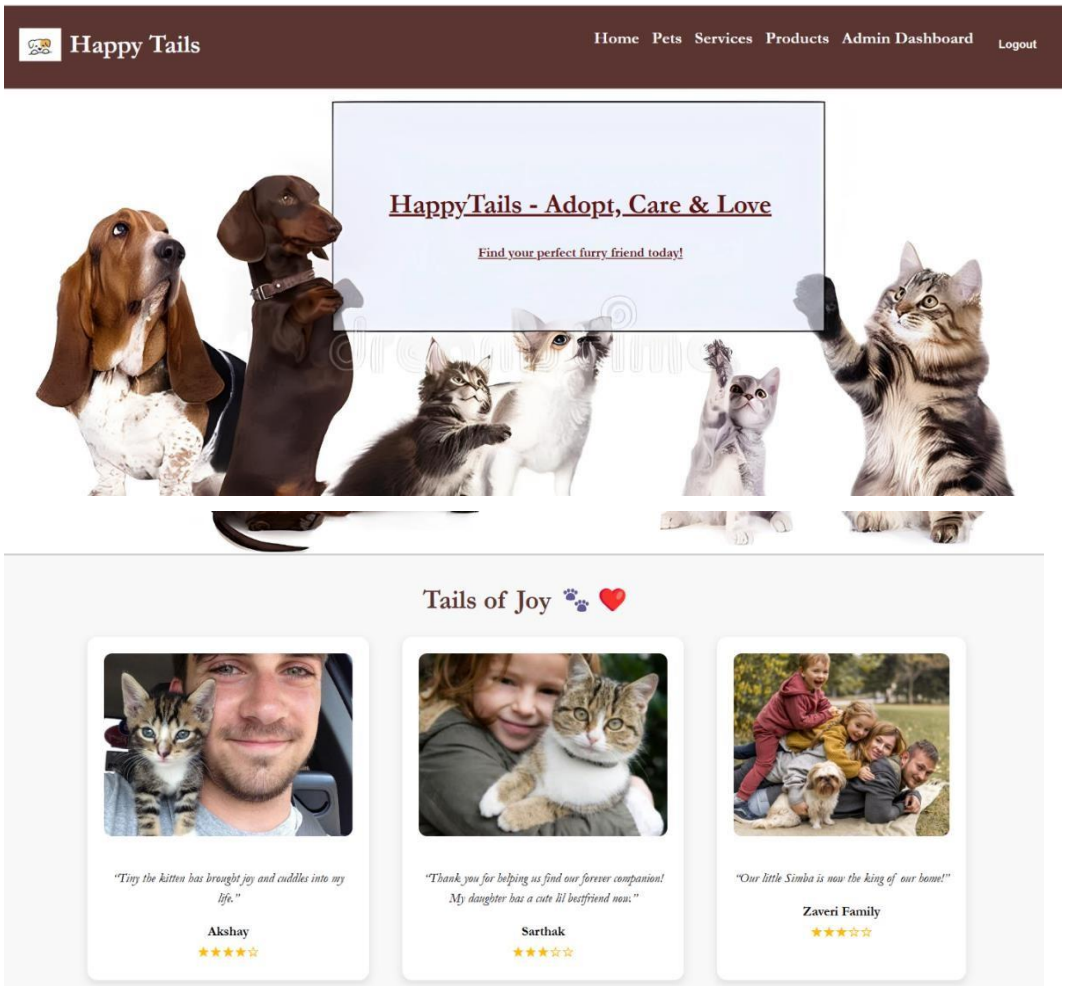
- Purpose: To authenticate existing users/admins.
- Fields:
  - Email
  - Password
- Validations:
  - Must enter correct email-password combination.
  - Error shown for invalid credentials.





2. Home Page

- Purpose: Landing page after login.
- User Role View:
    - welcome message
    - navigation to pets, services, products, etc.
    - happy customers feedback displayed



3. Pets Page

For Users:

- Purpose: View adoptable pets.
- Features:
    - View pet name, type, breed, image, and description.
    - Click "Adopt" to open the adoption request form.
- Validations (Adoption Form):

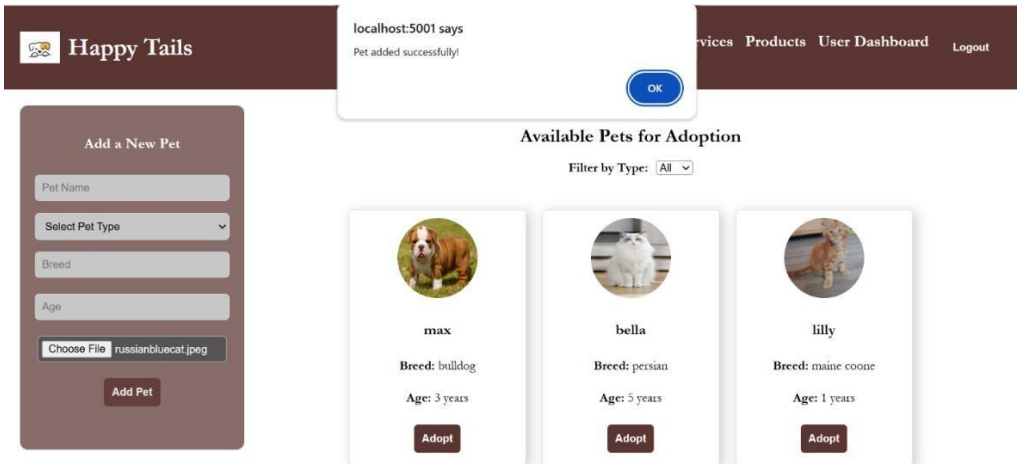
    - Name, contact number, reason for adoption are required.
    - Phone number must be numeric and 10 digits.

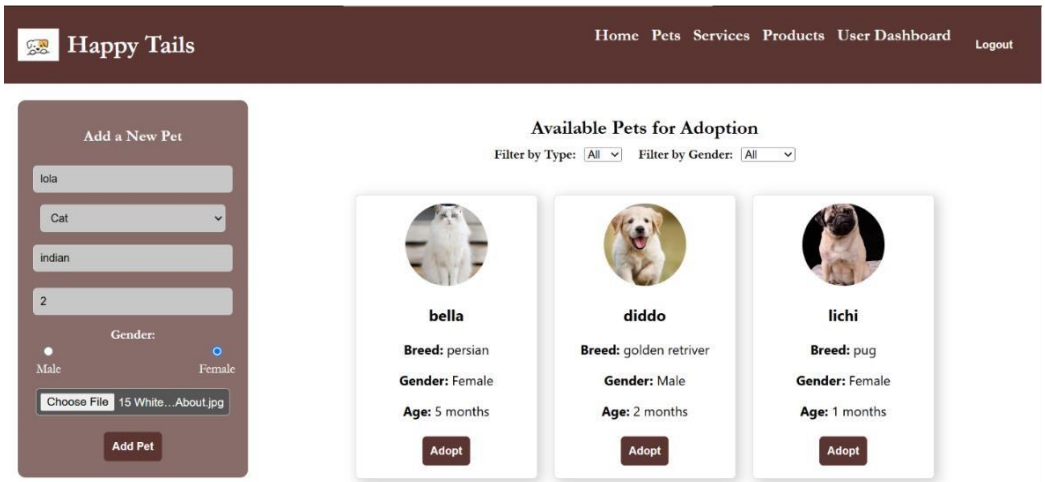
For Admins:

- Purpose: Manage all pet listings.
- Features:
    - Add/Edit/Delete pets.
- Validations:
    - Pet Name, Age, Type, Breed, and Image are mandatory.
    - Age must be a positive number.
    - Image file size limit (e.g., max 2MB).

Add Pet (User/Admin):

- Form Fields: Name, Age, Breed, Type, Details, Image.
- Validations:
    - All fields are required.
    - Age must be a number.
    - Image should be in .jpg/.png and within size limit.





4. Products Page

- Purpose: Users can browse and add pet-related products to the cart.
- Features:
    - Filter products by category (Food, Clothes, Toys).
    - Add to Cart / Buy Now options.

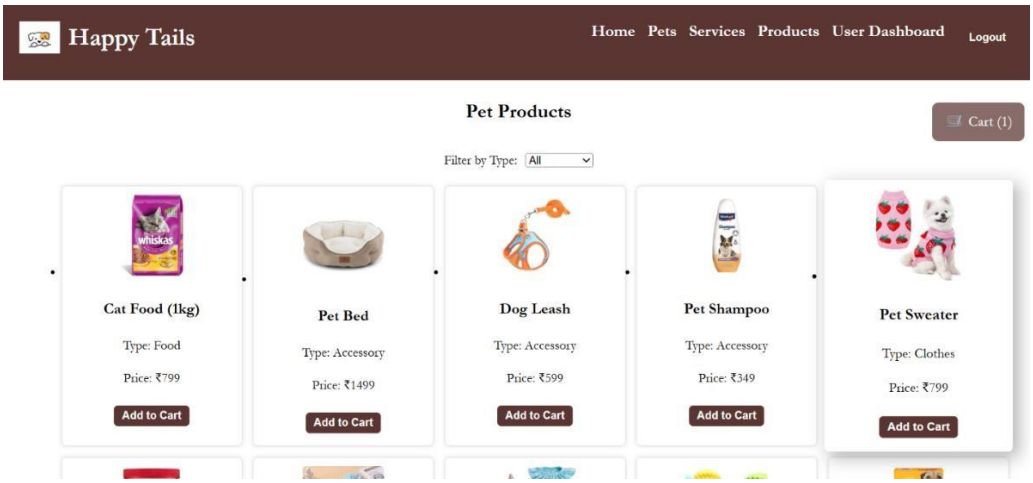
Cart Section:

- Shows all items added by the user.

- Allows removal or proceeding to payment.

Validations:

- Cannot add duplicate items.
- Cart must have at least one item to proceed to payment.



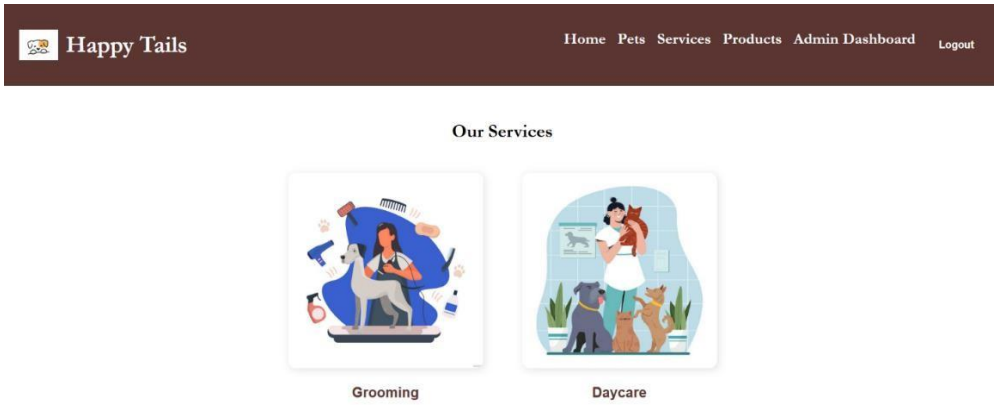
1. **Services Page**

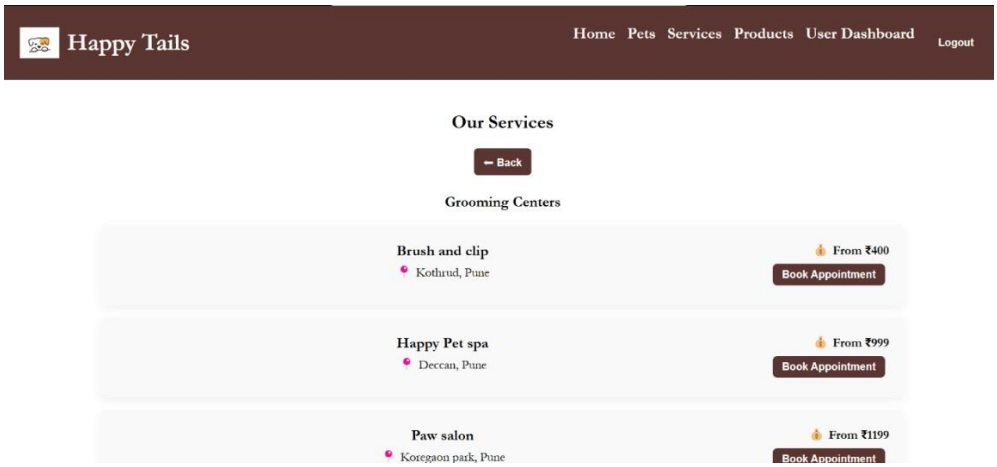
Purpose**:** To display all available pet-related services categorized as Grooming and Daycare, and allow users to book appointments**.**

Features:

- Categorized View:
  - Grooming Centers
  - Daycare Centers
  - Each category has cards/lists showing:
    - Service Name
    - Location
    - Description (optional)

- Book Appointment Button:
    - Available on each service listing.
    - On click, it reveals the Contact Number of the service provider.
    - Optionally, can also show a simple message like "Please contact to schedule your appointment."



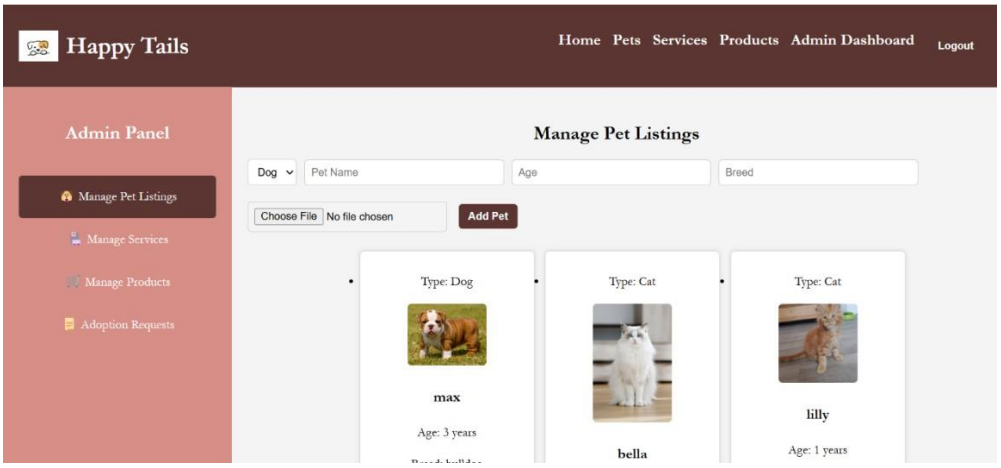


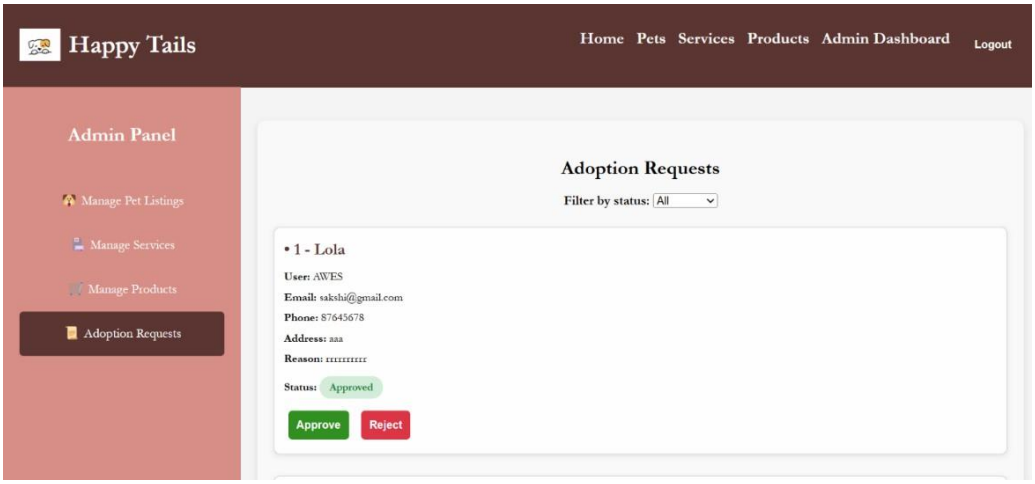
2. **Admin Dashboard**

- Purpose: Central control panel for admin.
- Sections:
    - Manage Users (if needed)
    - View and update Adoption Requests
    - Manage Pets and Products
    - Add Grooming/Daycare Services

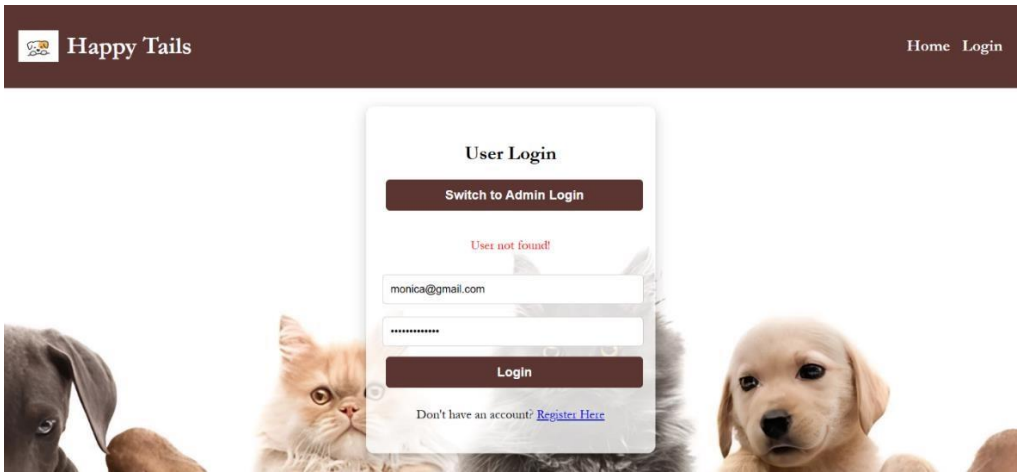
Adoption Requests Management:

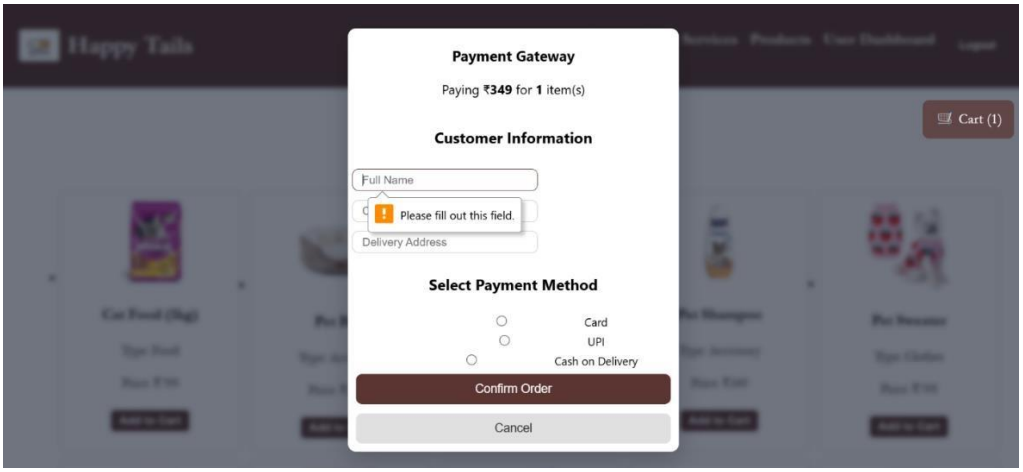
- Admin can view user-submitted requests and click to Approve or Reject.
- Status updates are saved in the database.





3. **Error Handling & Errors**
    - **Invalid Inputs**: Display form-specific validation messages.
    - **No Notification Issue** (Known Bug): Status change messages not popping up yet.




Happy Tails
Cart (1)
Payment Gateway
Paying ₹349 for 1 item(s)
Customer Information
Full Name
Please fill out this field.
Delivery Address
Select Payment Method
Card
UPI
Cash on Delivery
Confirm Order
Cancel


Add a New Pet
richie
Dog
indian
Age (months)
Please fill out this field.
Male
Female
Choose File
22 Pug F...Dog_.jpg
Add Pet