

PET CARE MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

RITHANYA R R
(Reg. No: 24MCR081)

SANTHOSH P
(Reg. No: 24MCR091)

VAISHNAVI D
(Reg. No: 24MCR121)

*in partial fulfillment of the requirements
for the award of the degree
of*

MASTER OF COMPUTER APPLICATIONS

DEPARTMENT OF COMPUTER APPLICATIONS



KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638 060

DECEMBER 2024

DEPARTMENT OF COMPUTER APPLICATIONS**KONGU ENGINEERING COLLEGE****(Autonomous)****PERUNDURAI, ERODE – 638 060****DECEMBER 2024****BONAFIDE CERTIFICATE**

This is to certify that the project report entitled “**PET CARE MANAGEMENT SYSTEM**” is the bonafide record of project work done by **RITHANYA R R (Reg. No: 24MCR081)**, **SANTHOSH P (Reg. No: 24MCR091)**, **VAISHNAVI D (Reg. No: 24MCR121)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications of Anna University, Chennai during the year 2024-2025.

SUPERVISOR**HEAD OF THE DEPARTMENT****(Signature with seal)****Date:**

Submitted for the end semester viva voce examination held on _____

INTERNAL EXAMINER**EXTERNAL EXAMINER**

DECLARATION

We affirm that the project report entitled “**PET CARE MANAGEMENT SYSTEM**” being submitted in partial fulfilment of the requirements for the award of Master of Computer Applications is the original work carried out by us. It has not formed the part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidates.

Date

RITHANYA R R

(Reg. No: 24MCR081)

SANTHOSH P

(Reg. No: 24MCR091)

VAISHNAVI D

(Reg. No: 24MCR121)

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date:

Name and Signature of the supervisor

(Dr.K.CHITRA)

ABSTRACT

The Pet Care Management System Web Application is designed to address the growing need for a streamlined, secure, and efficient platform that facilitates pet adoption in a user-friendly manner. With the increasing reliance on digital platforms for everyday tasks, the application focuses on creating an intuitive and responsive interface that meets customer expectations for ease of use and accessibility. Users can browse a diverse range of pets available for adoption, whether users are looking to adopt a pet or post one for adoption. By making the adoption process simple, transparent, and secure, the platform aims to revolutionize the pet adoption experience, ensuring a safe and responsible process for both pet seekers and pet owners.

The platform is organized into two core sections: "Home Pets" and "Homeless Pets," providing users with a clear distinction between pets owned by individuals who wish to re-home them and those found by users that need adoption. "Home Pets" are pets that are owned by individuals who want to give them up for adoption, while "Homeless Pets" refers to pets that have been found and are being posted for adoption. The organizational structure allows users to filter and view pets based on their specific preferences, making it easier for them to find the right pet to adopt. Whether user are looking for a pet with a known background or one that has been rescued, users can make more informed decisions based on the pet's status and history.

To enhance the security and reliability of the platform, the system incorporates an One-Time password for user verification during sign-up, ensuring that only legitimate users can access the platform. Additionally, email notifications are sent to users to update them on the status of their adoption requests, approvals, or rejections. The transparent communication helps to build trust among users, as users are kept informed throughout the adoption process. The platform fosters a sense of community and responsibility, ensuring that pet adoption is a seamless and secure process for everyone involved. By offering a secure, transparent, and easy-to-use platform, the project encourages responsible pet ownership and helps reduce the number of abandoned pets.

ACKNOWLEDGEMENT

We express our sincere thanks to our correspondent **Thiru.A.K.ILANGO BCom., MBA., LLB.**, and other philanthropic trust members of Kongu Vellalar Institute of Technology Trust for having provided with necessary resources to complete this project. We are always grateful to our beloved visionary Principal **Dr.V.BALUSAMY BE(Hons)., MTech., PhD** Kongu Engineering College, Perundurai for providing us with the facilities offered.

We convey our gratitude and heartfelt thanks to our Head of the Department **Dr.A.TAMILARASI Msc., MPhil., PhD., MTech.**, Department of Computer Applications, Kongu Engineering College for her perfect guidance and support that made this work to be completed successfully.

We also like to express our gratitude and sincere thanks to our project coordinator **Mrs.S.HEMALATHA MCA.**, Assistant Professor (Sr.G), Department of Computer Applications, Kongu Engineering college who have motivated us in all aspects for completing the project in scheduled time.

We would like to express our gratitude and sincere thanks to our project guide **Dr.K.CHITRA MCA., M.Phil., PhD** Assistant Professor, Department of Computer Applications, Kongu Engineering College for giving her valuable guidance and suggestions which helped us in the successful completion of the project.

We owe a great deal of gratitude to our parents for helping us to overwhelm in all proceedings. We bow our heart and head with heartfelt thanks to all those who thought us their warm services to succeed and achieve our work.

TABLE OF CONTENTS

| CHAPTER NO | TITLE | PAGE NO |
|---------------|--|---------|
| | ABSTRACT | iv |
| | ACKNOWLEDGEMENT | v |
| | LIST OF FIGURES | vii |
| | LIST OF ABBREVIATIONS | xi |
| 1 | INTRODUCTION | 1 |
| | 1.1 ABOUT THE PROJECT | 1 |
| | 1.2 EXISTING SYSTEM | 2 |
| | 1.3 DRAWBACKS OF EXISTING SYSTEM | 2 |
| | 1.3 PROPOSED SYSTEM | 2 |
| | 1.4 ADVANTAGES OF THE PROPOSED SYSTEM | 3 |
| 2 | SYSTEM ANALYSIS | 4 |
| | 2.1 IDENTIFICATION OF NEED | 4 |
| | 2.2 FEASIBILITY STUDY | 4 |
| | 2.2.1 Technical Feasibility | 5 |
| | 2.2.2 Operational Feasibility | 5 |
| | 2.2.3 Economical Feasibility | 5 |
| | 2.3 SOFTWARE REQUIREMENT SPECIFICATION | 6 |
| | 2.3.1 Hardware Requirements | 6 |
| | 2.3.2 Software Requirements | 7 |
| 3 | SYSTEM DESIGN | 12 |
| | 3.1 MODULE DESCRIPTION | 12 |
| | 3.2 USE CASE DIAGRAM | 16 |
| | 3.3 DATA FLOW DIAGRAM | 18 |

| | | |
|----------|--|----|
| | 3.4 DATABASE DESIGN | 20 |
| | 3.5 INPUT DESIGN | 25 |
| | 3.6 OUTPUT DESIGN | 25 |
| 4 | IMPLEMENTATION | 27 |
| | 4.1 CODE DESCRIPTION | 27 |
| | 4.2 STANDARDIZATION OF THE CODING | 27 |
| | 4.3 ERROR HANDLING | 27 |
| 5 | TESTING AND RESULTS | 29 |
| | 5.1 TESTING | 29 |
| | 5.1.1 Unit Testing | 29 |
| | 5.1.2 Integration Testing | 31 |
| | 5.1.3 Validation Testing | 33 |
| 6 | CONCLUSION AND FUTURE ENHANCEMENT | 37 |
| | 6.1 CONCLUSION | 37 |
| | 6.2 FUTURE ENHANCEMENT | 37 |
| | APPENDICES | 38 |
| | A. SAMPLE CODING | 38 |
| | B. SCREENSHOTS | 48 |
| | REFERENCES | 58 |

LIST OF FIGURES

| FIGURE NO | FIGURE NAME | PAGE NO |
|----------------------|------------------------------|----------------|
| 3.1 | Use Case Diagram | 16 |
| 3.2 | Data Flow Diagram Level 0 | 18 |
| 3.3 | Data Flow Diagram Level 1 | 19 |
| 3.4 | Pet Post Database | 23 |
| 3.5 | User Database | 23 |
| 3.6 | Adoption Submission Database | 24 |
| B.1 | SignUp Page | 48 |
| B.2 | Signin page | 48 |
| B.3 | OTP Notification | 49 |
| B.4 | Homeless pet post page | 50 |
| B.5 | Home pet post page | 50 |
| B.6 | Admin login page | 51 |
| B.7 | Admin dashboard | 52 |
| B.8 | Pet post request page | 52 |
| B.9 | View pets page | 53 |
| B.10 | Adoption submission page | 54 |
| B.11 | Adoption request page | 55 |
| B.12 | Adopted History page | 55 |
| B.13 | User profile page | 56 |
| B.14 | Contact page | 57 |

LIST OF ABBREVIATIONS

| | |
|-------------|--|
| MERN | MongoDB, Express.js, React.js, Node.js |
| HTML | Hypertext Markup Language |
| CSS | Cascading Style Sheets |
| JS | JavaScript |
| JSON | JavaScript Object Notation |
| API | Application Programming Interface |
| NPM | Node Package Manager |
| OTP | One-Time Password |
| JWT | JSON Web Token |
| DB | Database |

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE PROJECT

The purpose of the project is to develop a pet care management system that simplifies and enhances the pet adoption process for both admin and users. Traditionally, pet adoption systems are fragmented, with separate processes for pet listings, adoption requests, and management, often leading to inefficiencies and confusion. The platform aims to unify these functions in one seamless application, offering an efficient solution for users to adopt pets and for admin to manage pet posts, adoption requests, and user data. By integrating features like secure user authentication, adoption application forms, and real-time pet listings, the platform aims to streamline pet adoption, ensuring a smooth and transparent experience for all users.

The system enables admin to oversee the adoption process by reviewing and approving pet posts and adoption applications, ensuring that all pets and adopters meet the necessary criteria. For users, the platform provides an easy way to browse available pets, express interest, and submit adoption applications.

Users can also post pets for adoption through detailed forms that capture all relevant information, which is then reviewed by admin. By combining these features into one unified platform, the project eliminates the need for multiple tools and ensures transparency, security, and trust in the pet adoption process. Ultimately, the platform promotes responsible pet adoption, making it easier for pets to find their forever homes while fostering a sense of community among pet owners and adopters.

1.2 EXISTING SYSTEM

The current pet adoption system lacks a comprehensive digital platform to manage the adoption process effectively. While some standalone systems may exist, do not provide an integrated solution for both administrators and users. The admin module, if present, is often limited to basic functionalities like manually tracking pet data and user requests. On the other hand, users must rely on fragmented communication methods to inquire about available pets and their adoption status.

1.3 Drawbacks of Existing System

- Users lack awareness of a centralized platform for pet adoption.
- No secure OTP-based user verification during registration, leading to potential data risks.
- Does not include SMS or email notifications for updates on adoption requests.
- Lacks advanced filters for categorizing pets by type or status, reducing user convenience.

1.4 PROPOSED SYSTEM

The Pet Platform prioritizes secure user registration by implementing OTP verification, where a one-time password is sent to the registered email address for identity validation. The step ensures that only legitimate users gain access to the platform, fostering trust and security from the start.

Once registration is complete, users can explore a wide range of pets available for adoption, submit detailed adoption applications, and express interest in specific pets with ease. The platform's intuitive design ensures a seamless experience, allowing users to navigate listings effortlessly and focus on finding the right pet.

The platform also features robust administrative tools, enabling admins to manage pet listings efficiently, approve or reject adoption requests based on specific criteria, and track all user submissions for accountability. Every action on the platform triggers automated email notifications, ensuring users remain informed in real time about the status of their applications or listings. By combining a secure and user-friendly interface with transparent communication, the platform enhances the overall pet adoption process, making it more accessible, trustworthy, and efficient for both adopters and pet owners.

1.5 Advantages of Proposed System

- **Secure OTP Verification:** Ensures user identity verification, reducing the risk of unauthorized access.
- **Real-Time Email Notifications:** Keeps users informed about adoption requests, approvals, and rejections.
- **Efficient Adoption Process:** Simplifies pet adoption by allowing users to view pets, submit applications, and track their status.

Administrator Control: Admin can efficiently manage pet listings, adoption requests, and approval processes.

SUMMARY

The Pet Adoption Web Application aims to simplify and streamline the pet adoption process by providing a centralized platform where users can browse, post, and adopt pets based on type and status (Home Pets or Homeless Pets). Key features include secure OTP verification for authentic sign-ups and email notifications for updates on adoption requests, approvals, and rejections. By enhancing transparency, preventing fraud, and promoting responsible pet ownership, the platform ensures a safe, efficient, and trustworthy adoption experience while fostering stronger connections between adopters and pets.

CHAPTER 2

SYSTEM ANALYSIS

2.1 IDENTIFICATION OF NEED

Effective financial management is essential, yet individuals and groups face unique challenges that existing tools fail to address comprehensively. Personal finances often lack proper tracking and budgeting tools. Group expenses, on the other hand, suffer from disputes and inefficiencies. The disconnect between personal and group management tools leads to fragmented data, making decision-making difficult and less effective. The multi-functional finance tracking system provides real-time tracking, transparency, and actionable insights to help you manage your finances effectively. The system simplifies financial management by encouraging better spending habits and promoting collaboration among users.

2.2 FEASIBILITY STUDY

A feasibility study is a complete analysis that is conducted to evaluate the practicality and viability of a proposed project. The prime goal of a feasibility study is to determine whether the project is worth pursuing and if it can be successfully implemented within the defined constraints. Each structure has to be thought of in the developing of the project, as it has to serve the end user in a friendly manner. Three considerations involved in feasibility are

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

2.2.1 Operational Feasibility

The Pet Adopt Platform is designed to meet the operational needs of the pet adoption industry by aligning with user expectations and streamlining the adoption process. The main goal of the platform is to enhance user experience by providing a seamless, secure, and easy-to-navigate interface. The user-friendly design, with intuitive navigation and effective pet search functionalities, allows users to easily find available pets and submit adoption applications. The integration of features like OTP verification and real-time email notifications ensures secure and efficient communication between users and admin

2.2.2 Technical Feasibility

The technical effectiveness of the Pet Adopt Platform relies on the strategic use of advanced web technologies and design principles to create a seamless online adoption experience. The platform utilizes the latest technologies, including robust back-end systems (such as Node.js) and dynamic front-end interfaces (such as React.js), to ensure smooth operation and optimal user experience. Implementing responsive design is essential to make the platform accessible across various devices, including desktops, tablets, and smart phones, ensuring broad accessibility for users. The database management is optimized to handle large amounts of pet data, user profiles, and adoption requests efficiently, improving search functionality and enhancing the visibility of pets available for adoption

2.2.3 Economical Feasibility

Developing the Pet Care Platform is a cost-effective solution, backed by a thorough cost-benefit analysis. The initial investment in technical infrastructure, skilled developers, design elements, and integration of key features like secure adoption processes and real-time notifications is expected to generate significant returns. The revenue generated through increased adoption rates and an expanded user base will far outweigh the operational costs of maintaining the platform, managing content, and providing customer

support. By incorporating features like automated email notifications, an easy-to-use interface, and secure adoption processes.

Pet Adopt will attract more users and build trust, leading to greater brand awareness and customer loyalty. Additionally, advanced analytics tools will provide insights into user behaviour, allowing for more targeted marketing strategies and enhanced user engagement. The expected return on investment (ROI) will come from higher adoption rates, reduced marketing costs due to digital strategies, and long-term cost savings through platform automation. The project is economically viable, providing long-term cost savings and positive impacts on the platform's market presence, successfully addressing the growing demand for efficient and secure pet adoption solutions in today's digital world.

2.3 SOFTWARE REQUIREMENT SPECIFICATION

A declaration that describes the capability that a system needs in order to satisfy the user's requirements is known as a system requirement. The system requirements for specific machines, software or business operations are general. Taking it all the way down to the hardware coding that operates the software. System requirements are the most efficient way to address user needs while lowering implementation cost.

2.3.2 HARDWARE REQUIREMENT

| | |
|-----------|---------------------|
| Processor | : Intel Core i5 |
| RAM | : 4 GB RAM |
| Hard disk | : 500 GB |
| Keyboard | : Standard 102 keys |
| Mouse | : Optical Mouse |

2.3.1 SOFTWARE REQUIREMENTS

| | |
|--------------------|------------------------------|
| Operating System | : Windows 11 and above |
| Environment | : Chrome, Visual Studio Code |
| Frontend | : HTML, CSS, React.JS |
| Scripting language | : JavaScript |
| Backend | : Node.JS |
| Database | : MongoDB |

Front End

HTML

HTML, or Hypertext Mark up Language, serves as the backbone of the World Wide Web, providing a standardized structure for creating documents accessible via browsers. At its core, HTML consists of a series of elements, each with its own purpose and function. These elements range from simple, such as headings (<h1> to <h6>), paragraphs (<p>), and links (<a>), to more complex, like tables (<table>), forms (<form>), and multimedia content (<video>, <audio>,).

Elements are typically composed of tags, which enclose content and provide instructions on how that content should be displayed or behave. Attributes can also be added to tags to further specify their behaviour or appearance. For example, the tag includes attributes like "src" to specify the image source and "alt" to provide alternative text for accessibility purposes.

HTML documents follow a hierarchical structure, with a root <html> element containing <head> and <body> sections. The <head> section typically includes metadata such as the document's title, character encoding (<meta charset="utf-8">), and links to external resources like style sheets and scripts.

CSS

CSS (Cascading Style Sheets) emerges as a pivotal component in shaping the visual allure and structural integrity of the online platform. Operating as the styling language, CSS meticulously dictates the presentation of HTML elements, contributing to a seamless and aesthetically pleasing user experience. The system requirements for CSS encompass the creation of designs that are not only responsive but also adaptive, leveraging features like media queries to ensure optimal display across an array of devices. CSS extends its versatility to the customization of color, fonts, spacing, and other stylistic attributes, fostering a coherent and engaging design language. The incorporation of CSS frameworks, potentially utilizing Bootstrap, streamlines the styling process, offering a systematic and efficient approach to crafting design elements.

Furthermore, CSS introduces the capacity for implementing animations and transitions, elevating the user experience by introducing dynamic and visually captivating effects. Prioritizing adherence to CSS standards ensures not only a unified design language but also promotes a harmonious balance between functionality and aesthetics. The result is a sophisticated, user-friendly interface that seamlessly integrates design and functionality for an immersive and enjoyable online interaction. Moreover, CSS offers selectors that enable precise targeting of HTML elements, allowing developers to apply styles selectively. The feature is particularly advantageous when designing complex layouts or when specific elements require unique styling treatments.

JAVA SCRIPT

JavaScript is a versatile programming language widely used to enhance interactivity and functionality on web pages. Initially developed by Netscape as a client-side scripting language, it has evolved into a powerful tool for both client-side and server-side development. One of its standout features is the ability to manipulate HTML and CSS dynamically, enabling developers to create interactive and responsive web applications that react to user input in real-time. JavaScript can be integrated into web projects either by embedding it directly into HTML documents or by linking external script files, offering

flexibility in implementation. Its syntax, influenced by languages like Java and C, makes it relatively straightforward for developers to learn and apply.

Beyond dynamic content manipulation, JavaScript is integral for handling events such as mouse clicks, keyboard inputs, and form submissions, making it a cornerstone of modern interactive web design. It also supports asynchronous programming through tools like callbacks, promises, and `async/await`, which allow developers to perform concurrent tasks without blocking the main execution thread. The capability is crucial for building efficient, smooth, and user-friendly web experiences. The introduction of modern frameworks like React, Angular, and Vue.js has further expanded JavaScript's capabilities, empowering developers to create complex single-page applications (SPAs).

REACT.JS

React.js, commonly referred to as React, is an open-source JavaScript library developed and maintained by Facebook. It's widely used for building user interfaces (UIs) for web applications. React follows a component-based architecture, where UIs are broken down into reusable components that encapsulate their own logic and presentation. The modular approach simplifies the development process, promotes code reusability, and improves maintainability. React utilizes a virtual DOM (Document Object Model) to efficiently update the UI, minimizing the need for direct DOM manipulation and enhancing performance. state throughout the application, making it easier to debug and reason about.

Additionally, React's ecosystem is rich with tools and libraries like React Router for routing, Redux for state management, and Material-UI for UI components, further enhancing its capabilities and enabling developers to build sophisticated and feature-rich web applications efficiently. Components can be easily composed together to create complex UIs, and user can also be shared across different projects or teams, promoting consistency and reducing development time. React's support for server-side rendering (SSR) and static site generation (SSG) enables developers to optimize performance and improve SEO (Search Engine Optimization) by rendering UI components on the server and sending pre-rendered HTML to the client

2.6.2 Back End

NODE.JS

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to execute JavaScript code outside the web browser. Developed on Chrome's V8 JavaScript engine, Node.js enables server-side scripting, making it possible to build scalable and high-performance network applications. One of the key advantages of Node.js is its event-driven, non-blocking I/O model, which allows it to handle a large number of concurrent connections efficiently. The asynchronous nature makes Node.js particularly well-suited for building real-time applications such as chat applications, online gaming platforms, and streaming services. Node.js facilitates server-side development by providing a rich set of built-in modules and a vast ecosystem of third-party libraries through npm (Node Package Manager). Developers can leverage these modules and libraries to streamline development processes and enhance application functionality.

Additionally, Node.js excels in handling data-intensive applications thanks to its ability to handle I/O operations asynchronously. Its single-threaded event loop architecture ensures that CPU resources are utilized optimally, enabling Node.js applications to scale effectively while maintaining high performance. Furthermore, Node.js can be used in conjunction with frameworks like Express.js to simplify the development of web servers and restful APIs, allowing developers to build robust and scalable back-end systems. Moreover, Node.js is not limited to server-side development; it can also be used for building desktop applications and Internet of Things (IoT) devices, thanks to frameworks like Electron and Node-RED.

MONGODB

MongoDB is a widely used, open-source, NoSQL database that provides high performance, scalability, and flexibility for modern applications. Unlike traditional relational databases, MongoDB stores data in a flexible, JSON-like format called BSON (Binary JSON), which allows developers to store and retrieve complex data structures with

ease. The flexible schema design enables applications to evolve more quickly by allowing the database to handle various data formats without requiring rigid data models.

One of MongoDB's core strengths is its ability to scale horizontally, meaning it can handle large volumes of data and high traffic by distributing data across multiple servers (sharing). The feature makes MongoDB ideal for applications that require large-scale data storage, such as social media platforms, content management systems, and real-time analytics .

With MongoDB's Atlas, a fully-managed cloud service, developers can easily deploy, manage, and scale their databases without worrying about infrastructure. Atlas offers automated backups, monitoring, and performance optimization, enabling developers to focus more on building applications rather than managing databases. Security is a key feature of MongoDB, offering role-based access control, authentication, and encryption to ensure data integrity and confidentiality. The database supports integration with various authentication mechanisms like LDAP and Active Directory to provide secure access to resources.

SUMMARY

The platform is built using advanced technologies like React.js for a dynamic front-end, Node.js for scalable server-side operations, and MongoDB for robust data management. Its responsive design ensures accessibility across devices, while features like email alerts and secure processes enhance user engagement and trust. Economically and technically viable, the system is designed to improve the adoption experience for users and administrators' alike, promoting responsible pet ownership and building a trusted adoption network.

CHAPTER 3

SYSTEM DESIGN

3.1 MODULE DESCRIPTION

The Pet care Web Application's Home Module serves as the central hub, providing an engaging entry point with interactive features, seamless navigation, and dynamic content areas showcasing available pets and adoption services. Key sections offer quick access to adopting pets, posting pets for adoption, and platform features, fostering trust and transparency with users. The Adopt a Pet Module forms the backbone, presenting a comprehensive catalogue of admin-approved pets, with detailed information on each pet. Users can filter pets by type (dog, cat, rabbit, bird, fish, or other) and status (home pets or homeless pets), ensuring an intuitive browsing experience. The Services Module enables users to post pets for adoption through structured forms for home and homeless pets, requesting details like name, justification, location, type, and pictures.

Admin review these submissions and can approve or reject them accordingly. The Authentication Module ensures secure user interactions by enabling OTP-based registration, login, and account management. The module secures user data and provides easy access to both users and admin. The Admin Module empowers administrators with an intuitive dashboard for managing users, monitoring pet adoption requests, and analysing the platform's activity.

Admin can approve or reject pet submissions, manage adoption requests, delete posts, and track adoption history. The Notification Module sends automated email notifications for important actions, such as pet submissions, approvals, rejections, and adoption updates, ensuring clear communication with all users. The Profile Module allows users to manage their personal information, providing the option to view and edit their details for a more personalized experience. The structured, modular approach creates a streamlined Pet Adoption Platform, providing users with a seamless adoption experience while equipping admin with tools for efficient management and communication.

Home Module

The Home Module serves as the central entry point and foundational interface of the Pet Adoption Web Application. The module offers an engaging and visually appealing landing page that introduces users to the platform's purpose, services, and key features. The homepage includes a navigation bar for easy access to other important sections like Pet Listings, Services, About Us, Profile, and Contact Us. It features dynamic content such as promotional banners, featured pets, and announcements regarding adoption drives or events. The Home Module is designed with a responsive layout to ensure optimal viewing and navigation across devices. Additionally, the module presents sections that guide users to popular pets available for adoption, enhancing browsing and fostering quicker connections between adopters and pets. With an informative "About Us" section, users learn about the platform's mission, values, and its impact on pet adoption. The module creates an engaging, user-friendly gateway to the pet adoption platform.

Pet Listings Module

The Pet Listings Module is the core of the Pet Adoption Web Application, where users can explore available pets for adoption. The module presents a comprehensive and visually appealing display of pets, categorized by type (dog, cat, rabbit, bird, fish, other). Each pet profile includes essential details such as name, age, breed, health status, adoption status (approved or pending), and images. The module offers intuitive filtering options, allowing users to search by pet type, adoption status, and location. Users can view detailed information on each pet and express interest in adopting. The Pet Listings Module empowers users to browse and adopt pets easily, contributing to the smooth flow of the adoption process.

Authentication Module

The Authentication Module ensures secure user interactions with the platform. It includes a streamlined user registration and login process, allowing individuals to create accounts using their email and set passwords. An OTP verification system adds an extra layer of security during the sign-up process. The module also provides secure login, password recovery, and session management to enhance user security. With features like

multi-factor authentication and robust encryption, users can trust that their accounts and personal information are well-protected, ensuring a safe and secure environment for adoption applications and interactions.

Adoption Application Module

The Adoption Application Module facilitates the submission of adoption requests. After users find a pet, are interested in, can fill out a detailed adoption application form that includes personal information, living situation, experience with pets, and any additional details. The module ensures that adoption requests are thoroughly reviewed by the admin, who can approve or reject the application. It also features email notifications that inform users of the status of their application. The Adoption Application Module streamlines the process of applying for pets, ensuring the right homes for animals while maintaining communication between adopters and administrators.

Services Module

The Services Module allows users to submit pets for adoption. It includes two forms: one for pets currently in a home that are being re-homed, and one for homeless pets found by users. For home pets, users fill in details like pet name, age, breed, and reasons for adoption, along with images and contact information. For homeless pets, users provide details about the pet's location, health status, and the number of days the pet has been in the area. Once submitted, these pet requests are reviewed by admin, who can approve or reject them. The module ensures that pets available for adoption are properly vetted before being listed.

Admin Module

The Admin Module is the control centre for administrators, offering tools for managing users, pet submissions, adoption requests, and the overall operation of the platform. Admin can view detailed dashboards that showcase pet post requests, adoption applications, and users' profiles. The module includes features to approve or reject pet submissions, manage user accounts, and track adoption statuses. Admin also have access

to analytics tools that provide insights into adoption trends, pet availability, and user engagement, allowing for data-driven decisions. The Admin Module ensures smooth platform operations and effective management of pet adoption processes.

Notification Module

The Notification Module ensures seamless communication between users and admin by sending automated email notifications for key actions, such as pet submissions, approval or rejection of pets, adoption status updates, and adoption requests. These notifications provide users with real-time updates, helping to maintain transparency and engagement throughout the adoption process. The module enhances user experience and keeps all parties informed of their actions on the platform.

Profile Module

The Profile Module allows users to manage their accounts by viewing and editing their profile information, including name, email, phone number, and password. It also provides users with an option to update their personal details, keeping their contact information current. The module ensures that users can maintain accurate profiles and enhances the user experience by providing a personalized dashboard for each individual.

Contact Module

The Contact Module provides users with an easy way to get in touch with the platform's support team for inquiries, issues, or general feedback. It features a form that users can fill out to send direct messages to the platform administrators. Additionally, the module provides contact information such as email, phone number, and links to the platform's social media profiles for further assistance. The ensures users have a reliable channel to communicate with the platform when necessary.

3.2 USE CASE DIAGRAM

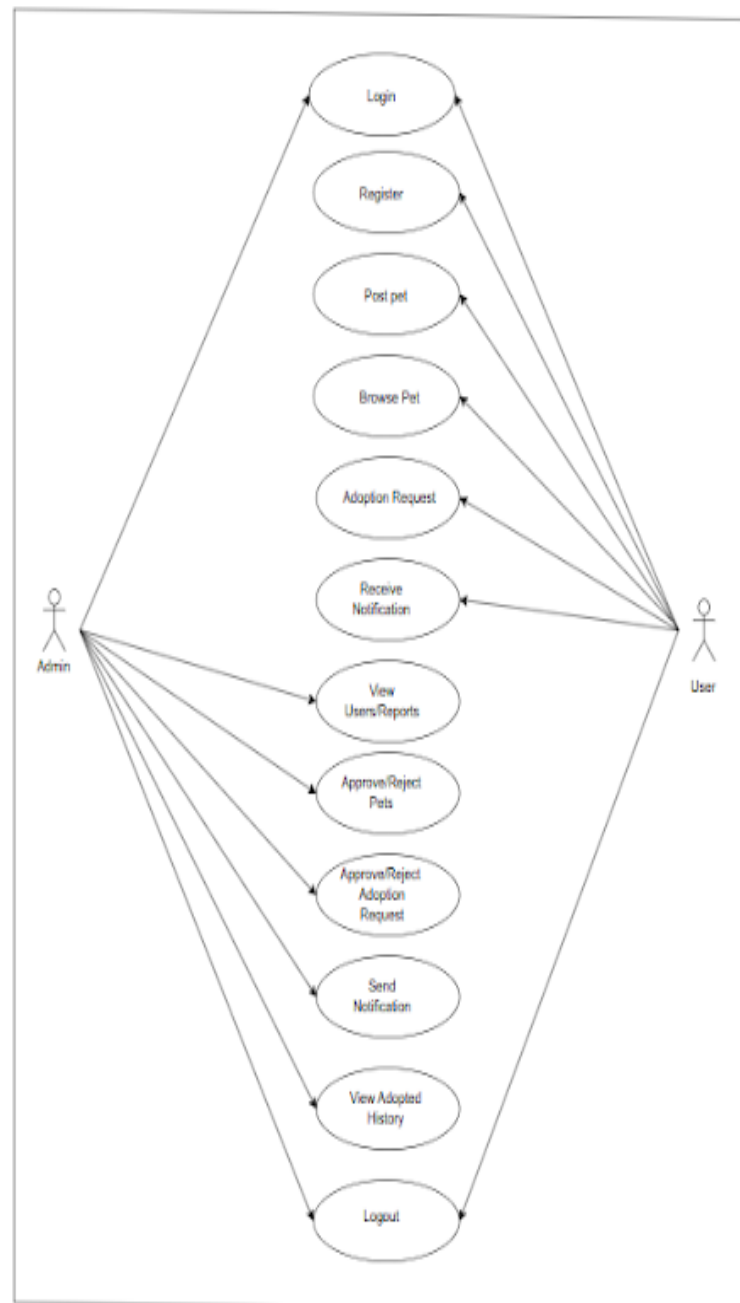


Figure 3.2 use case Diagram

DESCRIPTION:

- **User Use Cases:**
- **Register:** Create an account with OTP.
- **Login:** Log in using username/password.
- **Browse Pets:** View approved pets for adoption.
- **Submit Adoption Request:** Fill and submit adoption form.
- **View Profile:** View and edit user profile.
- **Logout:** Log out of the system.
- **Receive Notifications:** Get notifications about requests.
- **Login:** Admin login with credentials.
- **Approve Pet Post:** Approve/reject pet posts.
- **Manage Adoption Requests:** Review adoption requests.
- **View Users:** View user profiles and history.
- **Send Notifications:** Send email notifications to users.
- **Logout:** Admin logs out of the system.

Relationships:

- Admin manages pet posts, adoption requests, and user profiles.
- Users browse available pets, submit adoption requests, and manage their profiles.
- Admin sends notifications to Users about pet approval/rejection and adoption status.

3.3 DATA FLOW DIAGRAM

LEVEL 0

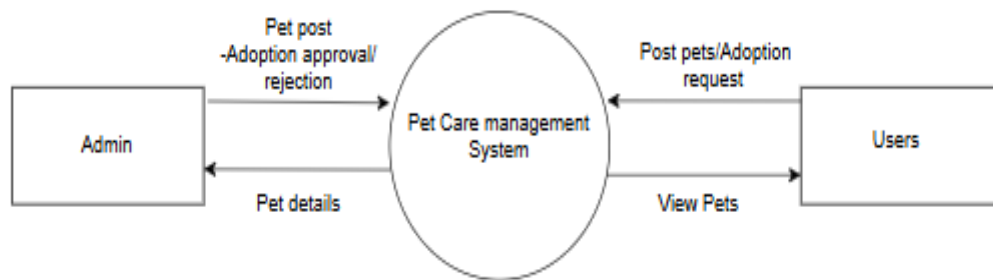


Figure 3.1 Data Flow Diagram level 0

DESCRIPTION:

- **User Registration:** Users register by entering their name, email, and OTP.
- **User Login:** Users log in with their username and password.
- **Browse Pets:** Users view pets approved by the admin.
- **Adoption Request:** Users fill out an adoption request for pets user are interested in.
- **Post Pet for Adoption:** Users post a pet for adoption, choosing between home or homeless pet.
- **Admin Panel:** Admin approves/rejects pet posts and adoption requests.
- **Pet Post Approval:** Admin approves or rejects pet posts submitted by users.
- **Adoption Request Approval:** Admin approves or rejects adoption requests from users.
- **Notifications:** Users receive notifications (via email) on approval/rejection of posts and requests.
- **View Profile:** Users can view and edit their profile.

LEVEL 1

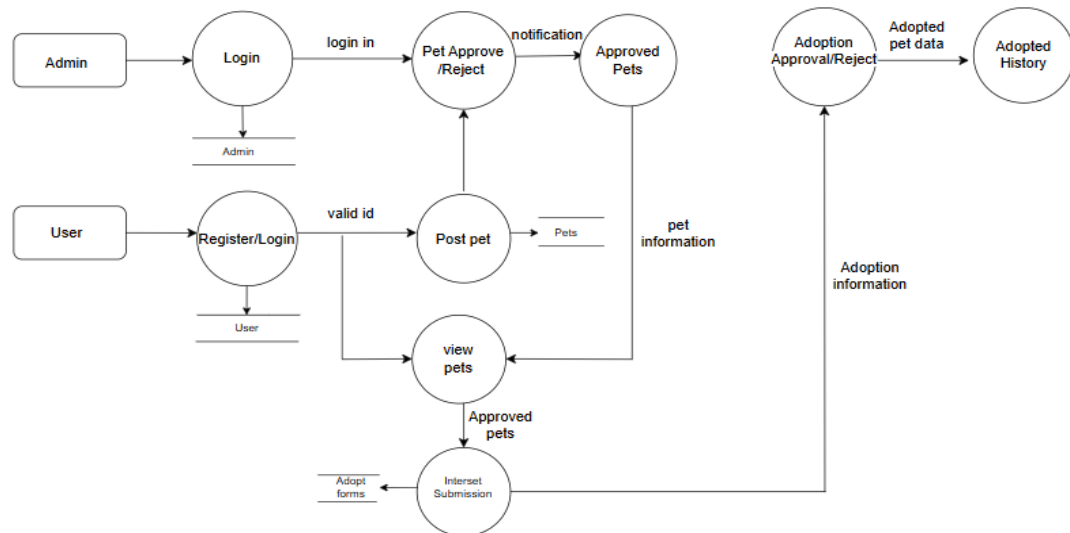


Figure 3.2 Data Flow Diagram Level 1

DESCRIPTION:

User Flow:

- **Login:** Users log in with their existing credentials. If user don't have an account, user register first.
- **Register:** New users can create an account by providing their name, email, and password.
- **View Pets:** Users can browse the pets available for adoption, which have been approved by the admin.
- **Adopt Pet:** Users can select a pet user are interested in adopting.
- **Submit Adoption Request:** After choosing a pet, users submit an adoption request form with details such as email, phone number, and pet living situation.
- **Receive Notification:** Once the request is processed, users receive email notifications regarding the approval or rejection of their adoption request.
- **View Profile:** Users can view and update their profile information as needed.

Admin Flow:

- **Admin Login:** Admin logs in using their credentials.
- **Approve Pet Post:** Admin reviews and approves/rejects pet posts submitted by users.
- **Approve Adoption Request:** Admin reviews and approves/rejects adoption requests.
- **View Pet Post Requests:** Admin can view all pet post requests submitted by users.
- **View Adoption Requests:** Admin can view adoption requests submitted by users.
- **Send Notification:** After approval or rejection of pet posts or adoption requests, notifications are sent to users (via email).

3.4 DATABASE DESIGN

The Pet Care Management project, developed using the MERN stack, offers a seamless and scalable solution for managing pet adoption processes. MongoDB serves as the NoSQL database, providing flexible and efficient storage of pet details, user profiles, and adoption requests. Express.js and Node.js power the back-end, ensuring smooth server-side logic and efficient API handling for tasks such as managing pet submissions, processing adoption applications, and updating statuses. The architecture ensures a reliable and efficient flow of data between the server and database, supporting the core functionality of the platform.

On the front-end, React is used to create a dynamic and interactive user interface. The platform delivers real-time updates for key features like pet listings, adoption requests, and status changes by leveraging restful APIs for communication between the front-end and back-end. Efficient state management ensures instant rendering of updates, eliminating the need for manual page refreshes.

The creates a smooth and engaging user experience, where users can effortlessly browse pets, submit adoption requests, or list pets for re homing. The project further

enhances user experience by implementing offline support through local storage or caching mechanisms.

Users can continue interacting with the platform, such as submitting forms or browsing saved data, even in the absence of internet connectivity, with updates syncing automatically once connectivity is restored. By leveraging the MERN stack's capabilities, the Pet Care Management platform delivers a robust, scalable, and user-friendly solution for fostering responsible pet adoption and creating meaningful connections between adopters and pets.

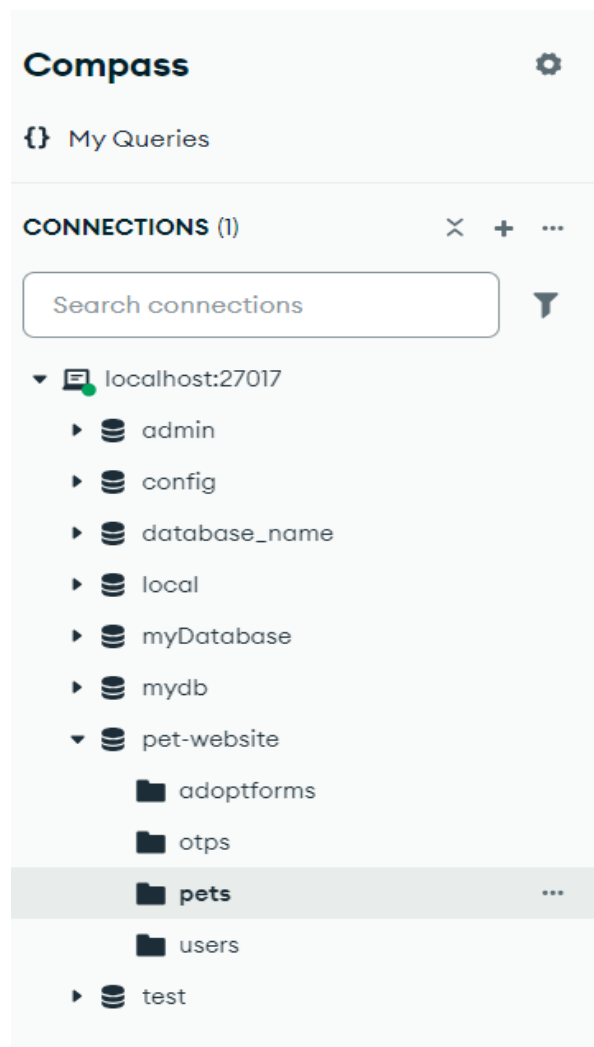


Figure 3.4.1 MongoDB Storage

MongoDB Compass is a graphical user interface (GUI) tool that connects to a MongoDB database instance. It allows users to visualize and explore the database structure. Compass provides an intuitive way to view collections, documents, and indexes. It also offers features like query building, performance optimization, and data validation.

The screenshot shows the MongoDB Compass interface for a database named 'pet-website' on 'localhost:27017'. The interface includes a top bar with 'Open MongoDB shell', 'Create collection', and 'Refresh' buttons. Below the top bar, there's a 'Sort by' dropdown set to 'Collection Name' and a 'View' toggle. The main content area displays four collection cards, each with the following details:

| Collection Name | Storage size | Documents | Avg. document size | Indexes | Total index size |
|-----------------|--------------|-----------|--------------------|---------|------------------|
| adoptforms | 20.48 kB | 3 | 244.00 B | 1 | 36.86 kB |
| otps | 8.19 kB | 0 | 0 B | 2 | 24.58 kB |
| pets | 20.48 kB | 11 | 301.00 B | 1 | 36.86 kB |
| users | 20.48 kB | 1 | 196.00 B | 2 | 40.96 kB |

Figure 3.4.2 Pet-website database

The pet-web application database consists of multiple collections: adopt forms, otp, pets, and users. The adopt forms collection contains 3 documents with an average size of 244.00 B, and a total index size of 36.86 KB. The pets collection has a storage size of 204845 B, with an average document size of 300.00 B and total index size of 36.3648 KB. The user collection has a storage size of 2045-40 B.

localhost:27017 > pet-website > users Open MongoDB shell

Documents 1 Aggregations Schema Indexes 2 Validation

Type a query: { field: 'value' } or [Generate query](#) Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE 75 1-1 of 1 ⌂ ⌕ ⌕

```

_id: ObjectId('6759cbfdbb856e4c6686d4f6')
name: "Rithanya"
email: "rithanya045@gmail.com"
password: "$2b$10$pbAjn94S4aKCND.3Q7eGUu4y5A74XmeE3KdpknhCw10bmhuE2jFD2"
createdAt: 2024-12-11T17:29:33.100+00:00
updatedAt: 2024-12-11T17:29:33.100+00:00
__v: 0

```

Figure 3.4.3 Users Database

The document represents a user record stored in the user collection of the pet-website database hosted on localhost: 27017. It contains the user's name, email (rithanya045@gmail.com), and an encrypted password. The account was created on December 11, 2024, and the last update is recorded for December 22, 2824.

localhost:27017 > pet-website > adoptforms Open MongoDB shell

Documents 3 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#) Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE 75 1-3 of 3 ⌂ ⌕ ⌕

```

_id: ObjectId('67642e97ac12766b2af2eda4')
email: "rithanya045@gmail.com"
phoneNo: "9876547543"
livingSituation: "village"
previousExperience: "no"
familyComposition: "no"
petId: "676422c33b886b6d8a3027be"
createdAt: 2024-12-19T14:32:55.726+00:00
updatedAt: 2024-12-19T14:32:55.726+00:00
__v: 0

```

Figure 3.4.5 Adoption Submission Database

The adoption form belongs to an applicant with email rithanyal45@gmail.com and phone number 9876547543. User live in a village and have no previous experience with pets or family involvement with pets. The form is linked to the pet with ID 676422c33b886b6d8a3107be. The record was created on December 19, 2024, and last updated on December 9, 2004.

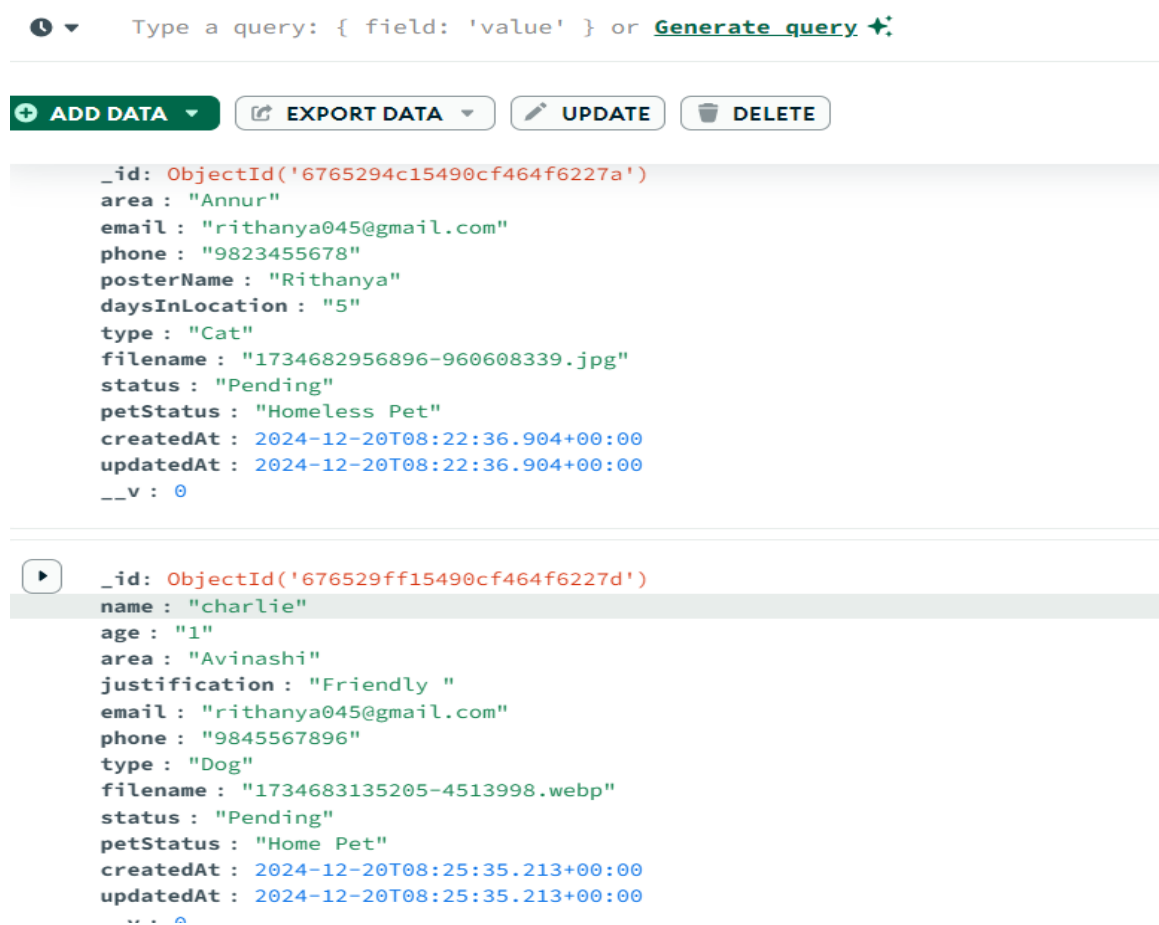


Figure 3.4.5 Pet post Database

MongoDB stores data in the form of documents, which are organized as key-value pairs. Each document is a set of key-value pairs, where keys are field names and values are the associated data. These documents are grouped into collections, similar to tables in relational databases. The flexible, schema-less format allows for dynamic and scalable data storage.

3.5 INPUT DESIGN

Input design is the critical process of transforming user-originated inputs into a computer-readable format. It stands as one of the most resource-intensive phases in the operation of a computerized system and is often a focal point for system challenges. Many issues within a system can be traced back to faulty input design and methodology. Each aspect of input design should undergo meticulous analysis and construction. The design should facilitate the smooth passage of inputs across various networks in a reliable manner, particularly for installations in remote networks. The input database is subject to specific constraints, including the acquisition of all files from the disk by data, suitability for data clearance, and presentation in an understandable and correct format.

The system's functionality relies on taking inputs from users, processing them, and generating outputs. Input design serves as the bridge connecting the information system with its users and should prioritize user-friendliness to provide relevant information effectively. The decisions made during input design aim to minimize time consumption, simplifying application sensitivity.

The input design goes beyond conventional practices, introducing innovative elements to elevate user interaction and system performance. Smart defaults and auto-population features optimize data entry efficiency, remembering frequently used information. Robust data validation algorithms ensure the integrity of inputs, while user-friendly error recovery mechanisms guide users through corrections seamlessly. Emphasis on data privacy includes end-to-end encryption and strict access controls, complying with legal standards. Intuitive feedback and progress indicators enhance user confidence, and cross-platform consistency guarantees a uniform experience.

3.6 OUTPUT DESIGN

Output design encompasses the results and information generated by the system, tailored for comprehension and usability by end-users. It plays a pivotal role in the evaluation of the application's usefulness, forming the basis upon which the system's effectiveness is assessed. The software's output serves as a critical input for remote

installations, triggering immediate alerts to enhance system functionality. The effectiveness of output design is pivotal, directly impacting the user's experience and satisfaction with the system. In the context of computer output, which serves as a primary source of information for users, output design particularly focuses on form design. An efficient output design aims to enhance the user interface, ensuring that the information presented is not only clear and comprehensible but also visually appealing.

The term "output" extends to any information displayed by the information system. When analysts undertake output design, users identify specific outputs necessary to meet end-user requirements. Previewing output reports by users is of utmost importance, as users are the ultimate judges of output quality and, consequently, the success of the system. The process of designing output involves a comprehensive analysis to determine which applications, websites, or documents are allowed or blocked. Various options for allowing are explored, enhancing the adaptability of the output to user preferences. The design of the output prioritizes attractiveness, convenience, and informativeness. Each output is crafted to be visually appealing, ensuring that it not only conveys information effectively but also provides a positive and engaging user experience. In essence, output design is a strategic process that goes beyond functionality, focusing on creating outputs that are not only technically sound but also user-centric and aesthetically pleasing.

SUMMARY

The Pet Adoption Web Application, built on the MERN stack, simplifies pet adoption with features like pet listings, adoption applications, and user management. It allows users to browse and adopt pets, while admin manage posts and requests. The platform ensures a smooth experience with secure login, real-time updates, and efficient input designs.

CHAPTER 4

IMPLEMENTATION

4.1 CODE DESCRIPTION

Code description can be used to summarize code or to explain the programmer's intent. Good comments don't repeat the code or explain it. They clarify its intent. Comments are sometimes processed in various ways to generate documentation external to the source code itself by document generator or used for integration with systems and other kinds of external programming tools. I have chosen React as front end and mongodb as back end.

4.2 STANDARDIZATION OF THE CODING

Coding standards define a programming style. A coding standard does not usually concern itself with wrong or right in a more abstract sense. It is simply a set of rules and guidelines for the formatting of source code. The other common type of coding standard is the one used in or between development teams. Professional code performs a job in such a way that is easy to maintain and debug. All the coding standards are followed while creating this project. Coding standards become easier, the earlier you start. It is better to do a neat job than cleaning up after all is done. Every coder will have a unique pattern that he adheres to such a style might include the conventions he uses to name variables and functions and how he comments his work. When the said pattern and style is standardized, it pays off the effort well in the long.

4.3 ERROR HANDLING

Exception handling is a process or method used for handling the abnormal statements in the code and executing them. It also enables to handle the flow control of the code/program. For handling the code, various handlers are used that process the exception and execute the code. React provides mechanisms like try-catch blocks to catch exceptions that may occur, such as errors when the application fails to fetch data from mongodb or when users input invalid transaction details.

An error is a serious problem than an application doesn't usually get pass without incident. Errors cause an application to crash, and ideally send an error message offering some suggestion to resolve the problem and return to a normal operating state, there is no way to deal with errors "live" or in production the only solution is to detect them via error monitoring and bug tracking and dispatch a developer or two to sort out the code.

USER INTERFACE DESIGN

Input design is the process of converting user originated inputs to a computer understandable format. Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. A large number of problems with a system can usually be tracked back to fault input design and method. Every moment of input design should be analyzed and designed with utmost care. To provide cost effective method of input. To achieve the highest possible level of accuracy. To ensure that the input is understood by the user. System analysis decide the following input design details like, what data to input, what medium to use, how the data should be arranged or coded, data items and transactions needing validations to detect errors and at last the dialogue to guide user in providing input. Input data of a system may not be necessarily is raw data captured in the system from scratch. These can also be the output of another system or subsystem.

SUMMARY

The above implementation section explained that the purpose of a code description is to summarise the code or to clarify the programmer's intent. Good comments don't repeat or explain the code. A programming style is defined by coding standards. A coding standard isn't usually concerned with what's proper or bad in a broader sense. Exception handling is a method or process for dealing with and executing anomalous statements in code.

CHAPTER 5

TESTING

5.1 TESTING

Software testing serves as the final assessment of specifications, designs, and coding and is a crucial component of software quality assurance. The system is tested throughout the testing phase utilizing diverse test data. The preparation of test data is essential to the system testing process. The system under study is tested after the test data preparation. Once the source code is complete, relevant data structures should be documented. The finished project must go through testing and validation, when errors are explicitly targeted and attempted to be found.

The project developer is always in charge of testing each of the program's separate units, or modules. Developers frequently also perform integration testing, which is the testing phase that results in the creation of the complete program structure.

This project has undergone the following testing procedures to ensure its correctness

- Unit testing
- Integration Testing
- Validation Testing

5.2.1 UNIT TESTING

A testing strategy known as unit and integration testing has been used to check that the system behaves as expected. The testing strategy was based on the functionality and the requirements of the system. In unit checking out, we have to check the applications making up the device.

This enables, to stumble on mistakes in coding and common sense which might be contained with the module alone. The checking out became completed at some stage in programming level itself.

Test Case 1: User Authentication

Module : User Authentication
Test Type : Unit Testing
Input : Username and Password
Expected Output : Authenticate user successfully

Sample Test

Output : User successfully authenticated.

Analysis : The test ensures that the system can authenticate users based on the provided username and password. If authentication fails for valid credentials or successful authentication occurs for invalid credentials, it indicates a problem with the user authentication module.

Test Case 2: Pet Post Submission

Module : Pet Post Submission
Test Type : Unit Testing
Input : Pet Name, Pet Age, Pet Type, Pet Picture, Description
Expected Output : Pet post successfully submitted

Sample Test

Output : Pet successfully submitted for adoption.

Analysis : The test ensures that the system can successfully submit a pet post with all required details. If the pet post fails to be submitted or incorrect details are posted, it indicates an issue in the post submission module.

Test Case 3: Pet Post Approval by Admin

Module : Admin Dashboard (Pet Post Approval)
Test Type : Unit Testing
Input : Pet Post ID (approved), Admin Credentials
Expected Output : Pet post approved by admin

Sample Test

Output : Pet post successfully approved and moved to available pets.

Analysis : The test ensures that the admin can approve a pet post. If the approval fails or the post doesn't appear in the correct category, it suggests an issue with the approval process.

Test Case 4: OTP Validation for User Registration

Module : User Registration (OTP Verification)

Test Type : Unit Testing

Input : OTP (valid or invalid)

Expected Output : Valid OTP grants access, invalid OTP prompts error

Sample Test

Output : OTP successfully validated or error message shown for invalid OTP.

Analysis : The test verifies that the system correctly validates the OTP during user registration. If the OTP verification fails for a valid OTP or accepts an invalid OTP, it indicates a problem with the OTP verification module.

5.1.2 INTEGRATION TESTING

Integration testing is done to test itself if the individual modules work together as one single unit. In integration testing, the individual modules that are to be integrated are available for testing. Thus, the manual test data that used to test the interfaces replaced by that which in generated automatically from the various modules. It can be used for testing how the module would actually interact with the proposed system. The modules are integrated and tested to reveal the problem interfaces

Test Case 1: Authentication & Admin Dashboard Access

Module : Authentication & Admin

Test Type : Integration Testing

Input : Admin login credentials (Username, Password)

Expected Output : Access to admin dashboard

Sample Test

Output : Successfully accessed the admin dashboard.

Analysis : The test validates that when admin login credentials are provided, the user should be granted access to the admin dashboard. If access is denied or the admin is redirected to an incorrect page, it indicates an issue with the integration between the authentication system and the admin module.

Test Case 2: User Registration & OTP Verification

Module : User Registration & OTP Verification

Test Type : Integration Testing

Input : User registration details (Name, Email, Password), OTP verification

Expected Output : Successful user registration and OTP validation

Sample Test

Output : User successfully registered and OTP validated.

Analysis : The test ensures that the user registration process integrates properly with OTP verification. If the user can register without a valid OTP or the registration fails after submitting the OTP, it suggests a problem with the integration between user registration and OTP validation modules.

Test Case 3: Pet Post Submission & Admin Approval

Module : Pet Post Submission & Admin Approval

Test Type : Integration Testing

Input : Pet details (Name, Age, Type, Picture), Admin credentials for approval

Expected Output : Pet post submitted by user and approved by admin

Sample Test

Output : Pet post successfully approved by admin and visible on the adoption page.

Analysis : The test checks the integration between the pet post submission and admin approval. If a pet post is not properly submitted or fails to appear in the approved section after admin approval, it indicates an issue with the integration of the submission and approval processes.

Test Case 4: Adoption Request & Admin Approval

Module : Adoption Request & Admin Approval

Test Type : Integration Testing

Input : Adoption request details (User info, Pet ID, Reason for Adoption), Admin login credentials

Expected Output : Adoption request submitted and approved by admin

Sample Test

Output : Adoption request successfully submitted and approved by admin.

Analysis : The test verifies the integration between adoption requests and the admin approval process. If the adoption request is not forwarded correctly to the admin or fails to be approved and confirmed, it indicates a flaw in the integration between the request and approval systems.

5.1.3 VALIDATION TESTING

Verification and validation checking out are critical tests, which might be achieved earlier than the product has been surpassed over to the customer. This make sure, that the software program checking out lifestyles cycle begins off evolved early. The intention of eachverification and validation is to make certain that the product is made in step with the necessities of the customer and does certainly fulfil the supposed purpose.

Test Case 1: User Registration

Module : User Registration
Test Type : Validation Testing
Input : User tries to register with an invalid email format.
Expected Output : Error message indicating that the email format is incorrect.
Sample Test

Output : Error message displayed, prompting the user to enter a valid email

Analysis : The test ensures that the system properly validates the email format during registration. If an invalid email format is accepted, it indicates a failure in input validation.

Test Case 2: Pet Adoption Request Submission

Module : Pet Adoption Request
Test Type : Validation Testing
Input : filling in required fields
Expected Output : Error message prompting the user to complete the required fields.

Sample Test

Output : Error message displayed, indicating missing information.

Analysis : The test checks that the system validates all required fields in the adoption request form. If the form can be submitted with missing fields, it indicates an issue in form validation.

Test Case 3: Pet Post Submission

Module : Pet Post Submission (User Section)

Test Type : Validation Testing

Input : User tries to post a pet without uploading a photo.

Expected Output : Error message stating that a photo is required to post the pet for adoption.

Sample Test

Output : Error message displayed, prompting the user to upload a photo of the pet.

Analysis : The test ensures that the system enforces the requirement of uploading a photo when submitting a pet post. If the user can submit without a photo, it indicates a flaw in validation.

Test Case 4: Pet Post Approval by Admin

Module : Admin Dashboard

Test Type : Validation Testing

Input : Admin tries to approve a pet post without checking the mandatory details like pet type, age, and location.

Expected Output : Error message or prompt warning the admin to verify the details before approval.

Sample Test

Output : Warning message displayed, indicating missing information that needs to be verified before approval.

Analysis : The test ensures that the admin cannot approve incomplete pet posts. If an incomplete post is approved, it indicates a problem in the post approval validation logic.

Test Case 5: Invalid OTP during Registration

Module : User Registration (OTP Verification)
Test Type : Validation Testing
Input : User enters an invalid OTP during registration or login.
Expected Output : Error message indicating that the OTP is invalid or expired.
Sample Test

Output : Error message displayed, prompting the user to enter a valid OTP.
Analysis : The test ensures that the system validates OTPs correctly. If the system accepts an invalid or expired OTP, it indicates a failure in the OTP validation process.

SUMMARY

The testing process for the Pet Adoption Web Application includes System Testing, Validation Testing, Unit Testing, and Integration Testing. System Testing evaluates the complete system's functionality, ensuring it meets requirements through both functional and non-functional tests. Validation Testing focuses on confirming the system fulfills user requirements and involves real-world tasks, including tests like user registration with invalid inputs. Unit Testing checks individual components for correctness, such as user authentication and pet post submissions. Integration Testing verifies that modules work together properly.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In conclusion, the Pet Adoption Digital Platform represents a transformative approach to pet adoption, catering to the evolving needs of pet lovers and fostering a seamless adoption experience. By emphasizing user-friendly design, secure authentication, and streamlined communication between users and admin, the platform enhances the pet adoption journey for both adopters and pet owners. The inclusion of vital features like real-time adoption requests, profile management, and post-submission pet approvals ensures that the system is both efficient and transparent. With integration of educational resources and personalized pet adoption recommendations, the platform not only simplifies the adoption process but also empowers users to make informed decisions. As the project develops, it will continue to meet the growing demand for digital adoption solutions, positioning itself as a leading and innovative platform for pet adoption in the digital space.

6.2 FUTURE ENHANCEMENT

- **Mobile App Integration:** Develop a mobile application for easier access to pet adoption services, allowing users to track their applications, get notifications, and view pet profiles on the go.
- **Pet Health Tracking:** Allow users to track their adopted pets' health and wellness, integrating with pet care apps or providing advice on health checks, vaccinations, and grooming.
- **Video Profiles:** Enable pet owners to upload video profiles of pets for adoption, offering a more personal touch and helping potential adopters better understand the pets' behaviour.
- **In-App Messaging:** Introduce an in-app messaging system that allows potential adopters and pet owners to communicate directly, asking questions and discussing adoption details in real-time.

APPENDICES

A. SAMPLE CODING

Index.html

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="utf-8" />
```

```
<link rel="icon" href="icon.jpg" />
```

```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

```
<meta name="theme-color" content="#000000" />
```

```
<meta
```

```
  name="description"
```

```
  content="Web site created using create-react-app"
```

```
/>
```

```
<link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
```

```
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
```

```
<link rel="preconnect" href="https://fonts.googleapis.com" />
```

```
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin /
href="https://fonts.googleapis.com/css2?family=Oxygen:wght@300;400&display=sw
ap"
```

rel="stylesheet"

/>

<link rel="preconnect" href="https://fonts.googleapis.com" />

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

<link

href="https://fonts.googleapis.com/css2?family=Oxygen:wght@300;400&family=Preahvihear&display=swap"

rel="stylesheet"

/>

<title>PawFinds: Your Path to Pet Adoption</title>

<link

rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css"

/>

<link rel="preconnect" href="https://fonts.googleapis.com" />

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

<link

href="https://fonts.googleapis.com/css2?family=Varela+Round&display=swap"

rel="stylesheet"/>


```

    <script
src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&libraries=pla
ces"></script>

</head>

<body class="hide-scrollbar">

    <noscript>You need to enable JavaScript to run the app.</noscript>

    <div id="root"></div>

</body>

</html>

}

```

App.js

```

import React from "react";

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

import Navbar from "./Components/NavBar/Navbar";

import Home from "./Components/Home/Home";

import Footer from "./Components/Footer/Footer";

import Services from "./Components/Services/Services";

import Contact from "./Components/Contact/Contact";

import Pets from "./Components/Pets/Pets";

import AdoptForm from "./Components/AdoptForm/AdoptForm";

```

```
import AdminLogin from "../Components/AdminPanel/AdminLogin";
```

```
import "../App.css";
```

```
const Layout = ({ children }) => (
```

```
  <
```

```
    <Navbar title="PawFinds" />
```

```
    {children}
```

```
    <Footer title="PawFinds" />
```

```
  </>
```

```
);
```

```
const App = () => {
```

```
  return (
```

```
    <Router>
```

```
      <Routes>
```

```
        />
```

```
        <Route
```

```
          path="/admin"
```

```
          element={<AdminLogin />}
```

```
        />
```

```
      </Routes>
```

```
</Router>

);

};
```

Index.css

```
body {

  margin: 0;

  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;

  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

code {

  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}

.hide-scrollbar::-webkit-scrollbar {
```

```
display: none;  
  
}
```

Index.js

```
import React from 'react';  
  
import ReactDOM from 'react-dom/client';  
  
import './index.css';  
  
import App from './App';  
  
import reportWebVitals from './reportWebVitals';  
  
const root = ReactDOM.createRoot(document.getElementById('root'));  
  
root.render(  
  
  <React.StrictMode>  
  
    <App />  
  
  </React.StrictMode>  
  
);  
  
// If you want to start measuring performance in your app, pass a function  
  
// to log results (for example: reportWebVitals(console.log))  
  
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals  
  
reportWebVitals();
```

AdminController.js

```
const express = require('express')

const getCredentials = (req, res) => {

  const Credentials = { "username": "admin", "password": "123" }

  res.status(200).json(Credentials)

}

module.exports = { getCredentials }
```

AdoptFormController.js

```
const AdoptForm = require('../Model/AdoptFormModel')

const express = require('express')

const saveForm = async (req, res) => {

  try {

    const { email, livingSituation, phoneNo, previousExperience, familyComposition,
    petId } = req.body

    const form = await AdoptForm.create({ email, livingSituation, phoneNo,
    previousExperience, familyComposition, petId })

    res.status(200).json(form)

  } catch (err) {
```

```

        res.status(400).json({ message: err.message })
    }
}

const getAdoptForms = async (req, res) => {
    try {
        const forms = await AdoptForm.find().sort({ createdAt: -1 });

        res.status(200).json(forms)
    } catch (err) {
        res.status(400).json({ message: err.message })
    }
}

const deleteForm = async (req, res) => {

const deleteAllRequests = async (req, res) => {
    try {
        const result = await AdoptForm.deleteMany({ petId: id });

        console.log("Forms not found");

        return res.status(404).json({ error: 'Forms not found' });
    }

    res.status(200).json({ message: 'Forms deleted successfully' });

    } catch (error) {

```

```

        res.status(400).json({ message: error.message });
    }
};

module.exports = {
    saveForm,
    getAdoptForms,
    deleteForm,
    deleteAllRequests
}

```

PetRoute.js

```

const express = require('express');

const router = express.Router();

const multer = require('multer');

const path = require('path');

const { postPetRequest, approveRequest, deletePost, allPets } =
require('../Controller/PetController');

const storage = multer.diskStorage({

    destination: function (req, file, cb) {

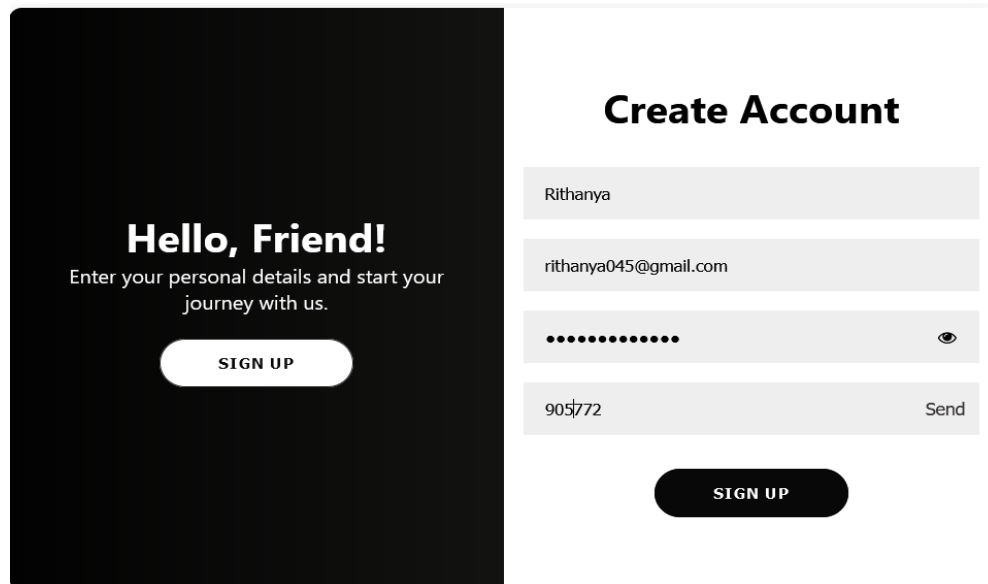
        cb(null, path.join(__dirname, '../images'));

    },

```

```
filename: function (req, file, cb) {  
  
    cb(null, Date.now() + '-' + Math.round(Math.random() * 1E9) +  
    path.extname(file.originalname));  
  
    }  
  
});  
  
const upload = multer({ storage: storage });  
  
router.get('/requests', (req, res) => allPets('Pending', req, res));  
  
router.get('/approvedPets', (req, res) => allPets('Approved', req, res));  
  
router.get('/adoptedPets', (req, res) => allPets('Adopted', req, res));  
  
router.post('/services', upload.single('picture'), postPetRequest);  
  
router.put('/approving/:id', approveRequest);  
  
router.delete('/delete/:id', deletePost);  
  
module.exports = router;
```

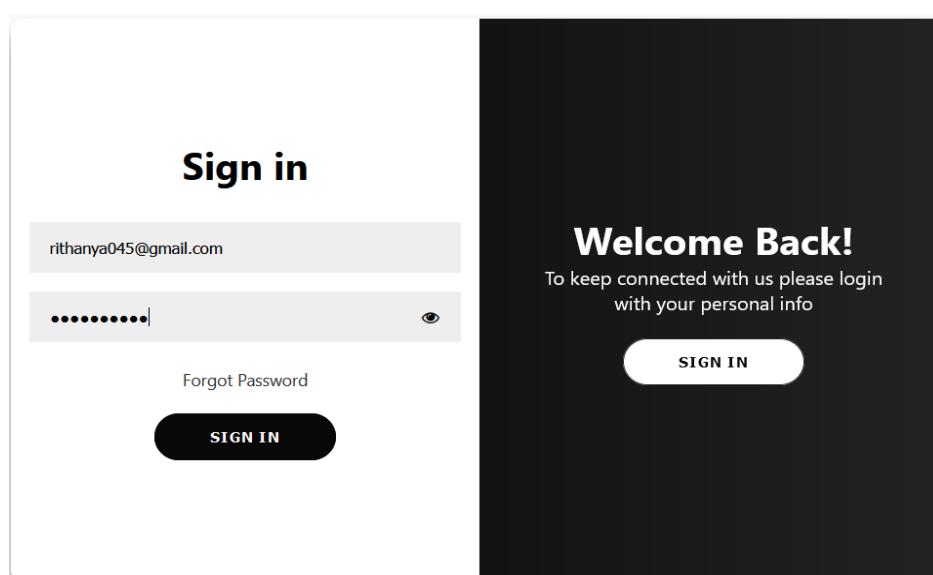

B. SCREENSHOTS



The screenshot displays a sign-up interface with a dark left panel and a light right panel. The left panel features the text "Hello, Friend!" and "Enter your personal details and start your journey with us." with a "SIGN UP" button. The right panel is titled "Create Account" and contains input fields for name ("Rithanya"), email ("rithanya045@gmail.com"), password (masked with dots), and a verification code ("905772" with a "Send" link). A "SIGN UP" button is at the bottom of the right panel.

Figure B.1 Sign up page

Capture the essence of ease and accessibility with our Create Account page, featuring seamless Sign Up and Sign In options. The clean, user-friendly interface ensures a quick start for users eager to adopt or list pets for adoption. Designed to prioritize simplicity, making joining 'Pet care' a breeze for all pet lovers.



The screenshot displays a sign-in interface with a light left panel and a dark right panel. The left panel is titled "Sign in" and contains input fields for email ("rithanya045@gmail.com") and password (masked with dots), a "Forgot Password" link, and a "SIGN IN" button. The right panel features the text "Welcome Back!" and "To keep connected with us please login with your personal info" with a "SIGN IN" button.

Figure

B.2 Sign in page

Screenshot of a 'Sign In' page with a welcoming message saying 'Welcome Back!' encouraging users to log in. Includes fields for email and password, along with a prominent 'Sign In' button. Additional options like 'Forgot Password' and quick links for easy access are visible.

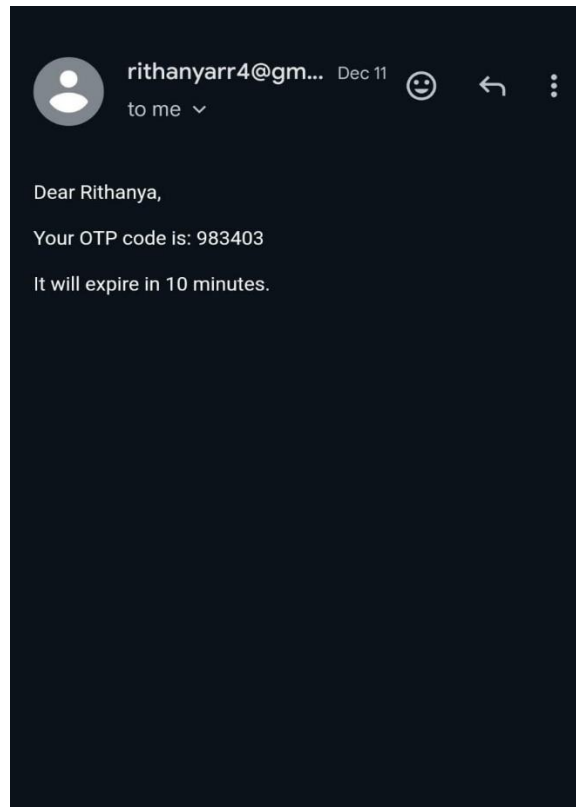




Figure B.3 OTP Notification

OTP Verification: An OTP has been sent to your registered email ID for secure login. Enter the 6-digit code in the provided field to proceed. Please note that the OTP is valid for 10 minutes only. If it expires, request a new code to continue.


[Home](#)
[Services](#)
[Pets](#)
[Profile](#)
[Contact Us](#)

Welcome Rithanya! [Logout](#)

Adopt a Pet



Benefits of Pet Adoption

- Provide a loving home to a pet in need
- Experience the unconditional love of a pet
- Create lasting memories and cherished moments

Adoption Process


- Fill out an adoption application
- Meet potential pets in person
- Complete the necessary paperwork

Responsibilities

Adopting a pet comes with responsibilities, including feeding, grooming, regular exercise, and providing medical care.

[Find Your Perfect Pet](#)

Post a Pet for Adoption




☐ Home Pet
☒ Homeless Pet

Poster Name:
 Phone No:
 Days in Location:
 Picture:
 Location:
 Type:
 Email:

[Submit Your Pet](#)

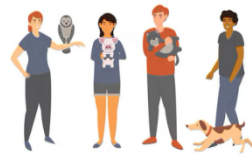
Figure B.4 Homeless pet post page

Pet Post for Adoption Form: Choose between 'Home Pet' and 'Homeless Pet' categories for posting. In the screenshot, 'Homeless Pet' has been selected, and the form is filled out with all necessary details. The post is ready but has not been submitted yet. Review the information and submit once everything is confirmed.


[Home](#)
[Services](#)
[Pets](#)
[Profile](#)
[Contact Us](#)

Welcome Rithanya! [Logout](#)

Adopt a Pet



Benefits of Pet Adoption

- Provide a loving home to a pet in need
- Experience the unconditional love of a pet
- Create lasting memories and cherished moments

Adoption Process


- Fill out an adoption application
- Meet potential pets in person
- Complete the necessary paperwork

Responsibilities

Adopting a pet comes with responsibilities, including feeding, grooming, regular exercise, and providing medical care.

[Find Your Perfect Pet](#)

Post a Pet for Adoption



☒ Home Pet
☐ Homeless Pet

Name:
 Pet Age:
 Justification for giving a pet:
 Phone No:
 Picture:
 Location:
 Type:
 Email:

[Submit Your Pet](#)

Figure B.5 Home pet post page

Pet Post for Adoption Form: Choose between 'Home Pet' and 'Homeless Pet' categories for posting. In the screenshot, 'Home Pet' has been selected, and the form is filled out with all necessary details. The post is ready but has not been submitted yet. Review the information and submit once everything is confirmed.

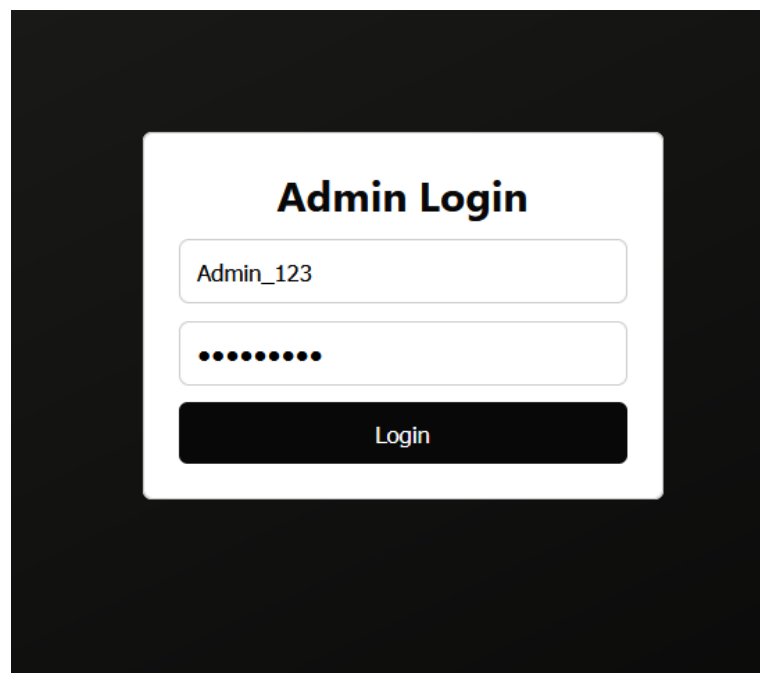
A screenshot of an 'Admin Login' page. The page has a black background. In the center, there is a white rectangular box with rounded corners. Inside this box, the title 'Admin Login' is displayed in bold black text. Below the title, there are two input fields: the first contains the text 'Admin_123' and the second contains ten black dots representing a password. Below these fields is a black button with the word 'Login' in white text.

Figure B.6 Admin Login page

Admin Login Page: The interface includes two input boxes one for entering the admin user ID and the other for the password with login box .Once the details are entered, click the 'Login' button to access the admin panel.

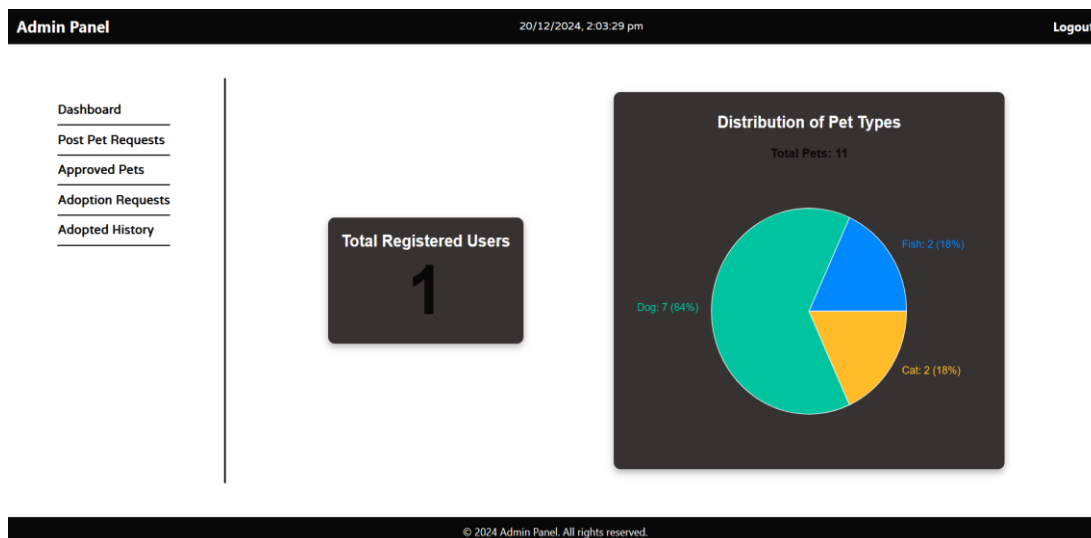


Figure B.7 Admin Dashboard Page

Admin Panel: Manage the platform seamlessly with an intuitive dashboard. Access sections for post request approvals, pet adoption requests, and adoption history. View a detailed pie chart showing the distribution of pet types and a column displaying the total registered users. Streamline operations and monitor platform activity efficiently.

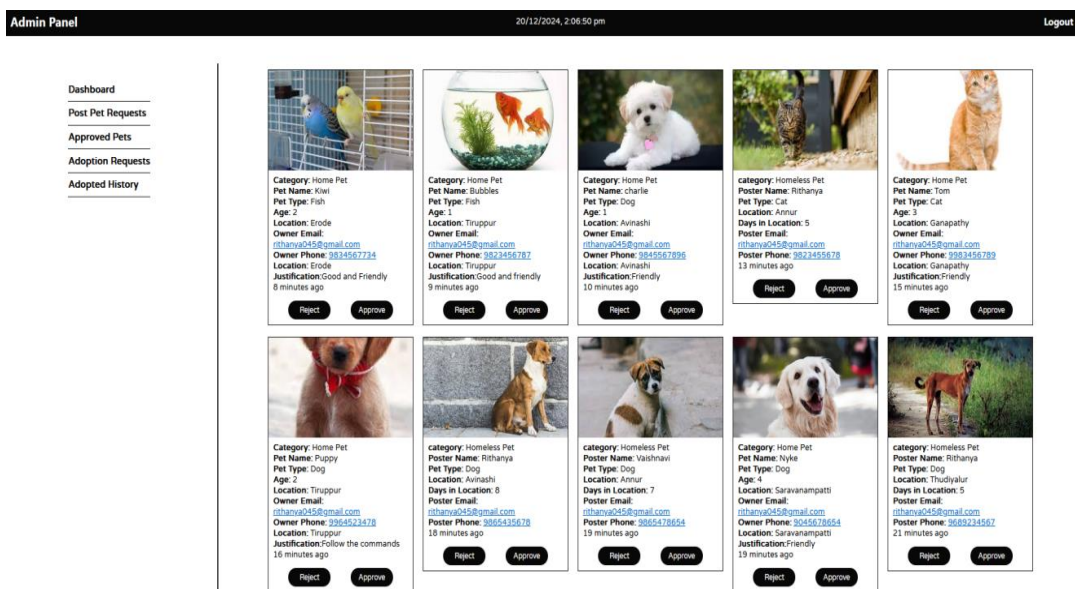


Figure B.8 Pet post request Page

Admin Panel - Pet Approval Requests: View and manage the list of pets awaiting approval for posting. Includes both homeless pets and home pets with their respective details like category, pet name, type, location, and requester information. The interface ensures clear visibility of all pending requests. Approve or reject requests seamlessly from the dashboard.

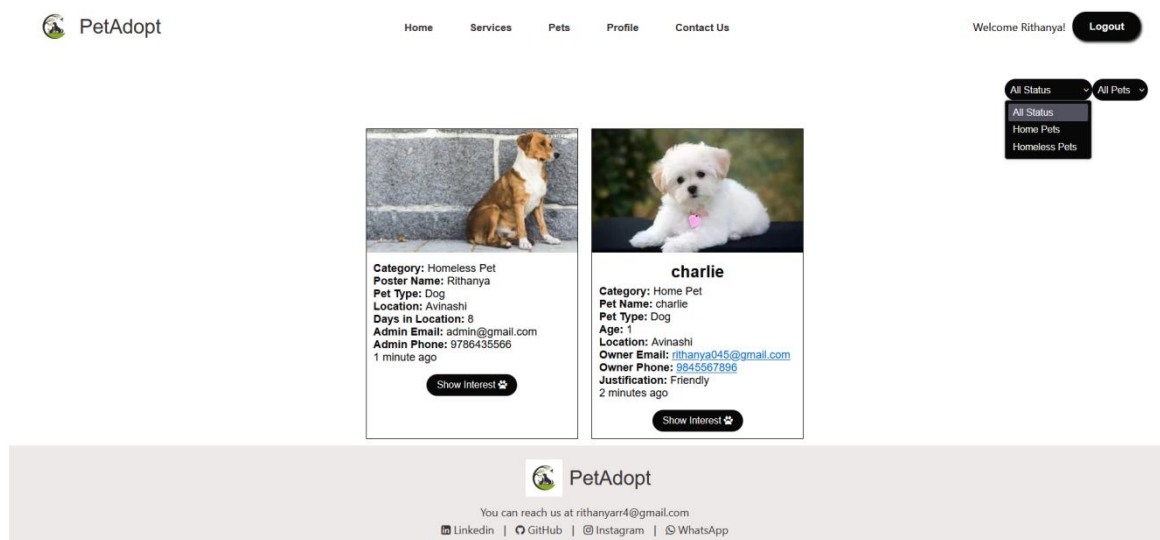
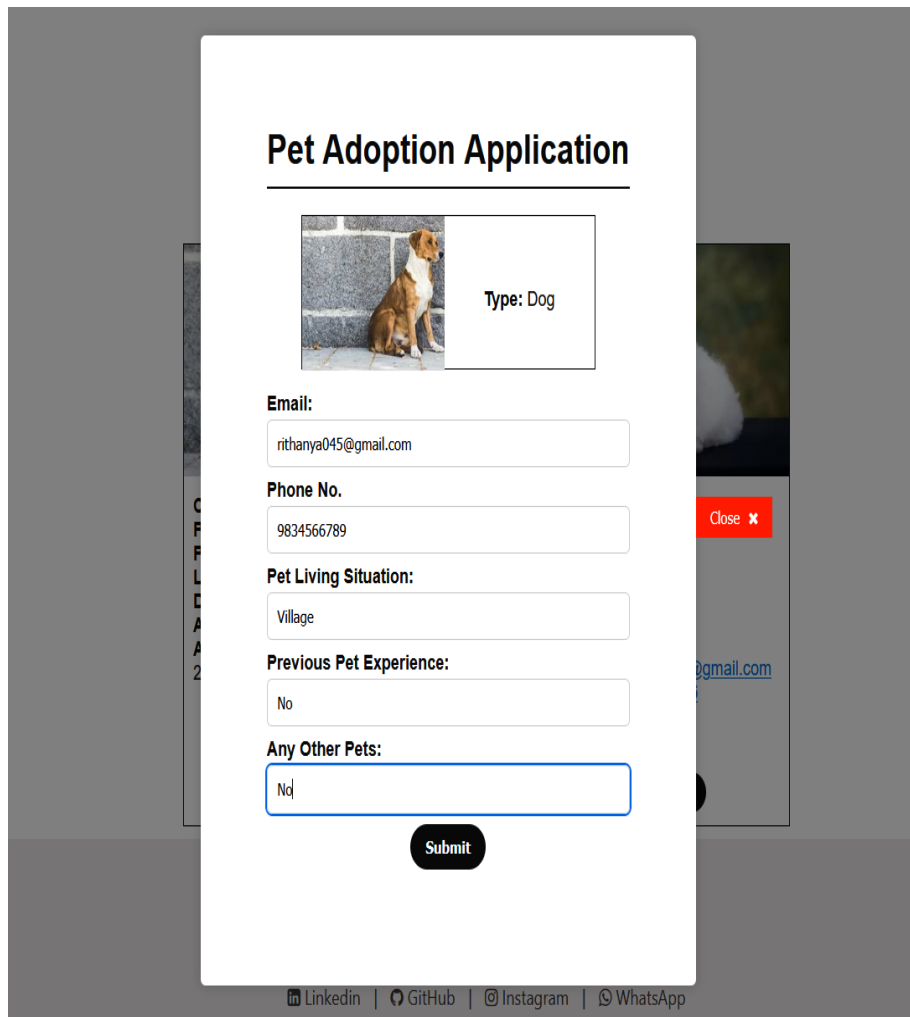



Figure B.9 View pets page

Pets Adoption List: View comprehensive details for both homeless and home pets in a single page. For homeless pets, see category, poster name, pet type, location, days at that same location, admin email, and phone number. For home pets, details include category, pet name, pet type, age, location, owner name, and owner contact information. Organized in clear columns for easy navigation and management.

A screenshot of a web application showing a 'Pet Adoption Application' form. The form is a white modal box centered on a dark grey background. At the top, the title 'Pet Adoption Application' is underlined. Below the title is a pet information section with a photo of a brown and white dog and the text 'Type: Dog'. The form contains several input fields: 'Email' with the value 'rithanya045@gmail.com', 'Phone No.' with '9834566789', 'Pet Living Situation' with 'Village', 'Previous Pet Experience' with 'No', and 'Any Other Pets' with 'No'. A black 'Submit' button is at the bottom. A red 'Close' button is visible on the right side of the modal. At the bottom of the dark grey background, there are social media links for LinkedIn, GitHub, Instagram, and WhatsApp.

Pet Adoption Application



Type: Dog

Email:

Phone No.

Pet Living Situation:

Previous Pet Experience:

Any Other Pets:

Submit

[LinkedIn](#) | [GitHub](#) | [Instagram](#) | [WhatsApp](#)

Figure B.10 Adoption submission form page

Pet Adoption Application Form: Fill in your details to proceed with the adoption process. Provide your email ID and phone number for communication. Share information about your current living situation and any experience with previous pets. Ensure all fields are completed for a smooth application review.

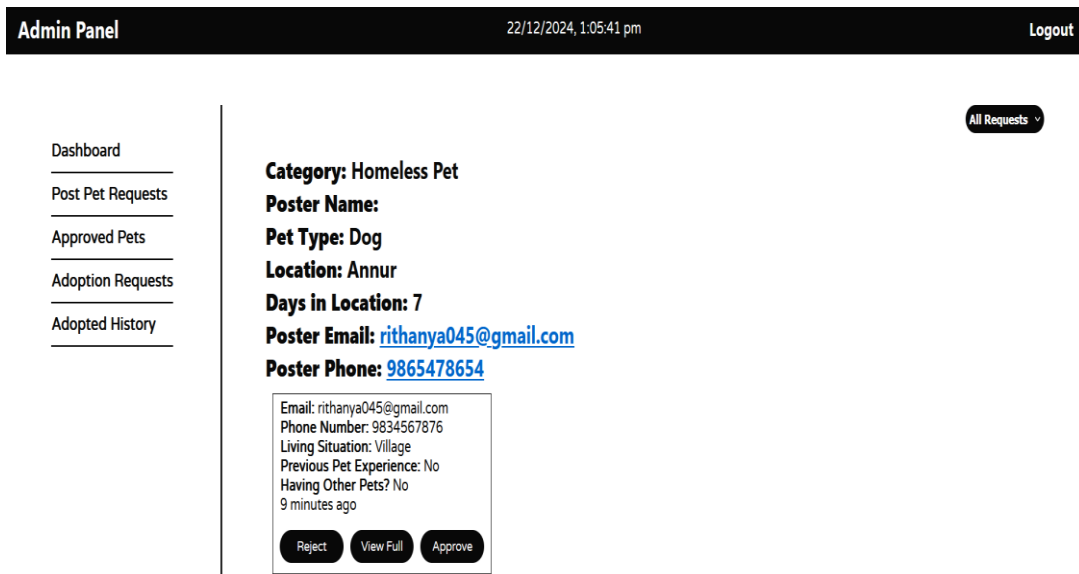


Figure B.11 Adoption request page

Admin Panel - Request Management: The panel displays a list of pending requests for pet adoption posts. For each request, the admin has options to 'Reject,' 'View Full Form,' or 'Approve' the submission. The interface allows the admin to easily manage and take action on each request. Decisions are made efficiently from the centralized view.

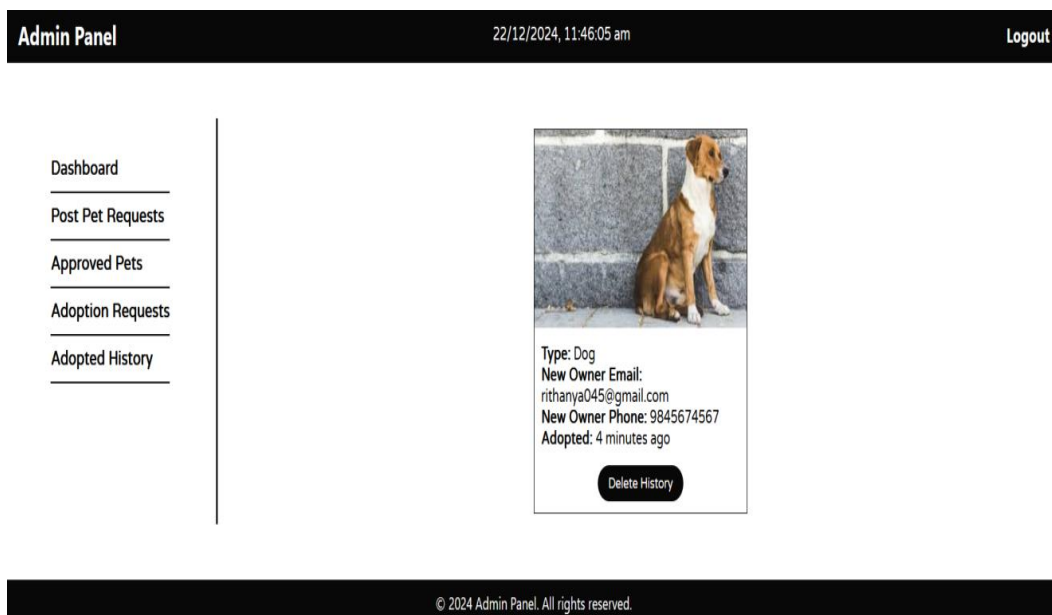


Figure B.12 Adopted History page

Adopted History: The section displays details of adopted pets, including the dog's picture, type, new owner's email, phone number, and adoption time. A 'Delete History' checkbox is provided to remove records. Each entry is clearly listed for easy tracking of adoption activity. Admin can manage adoption history efficiently through the interface.

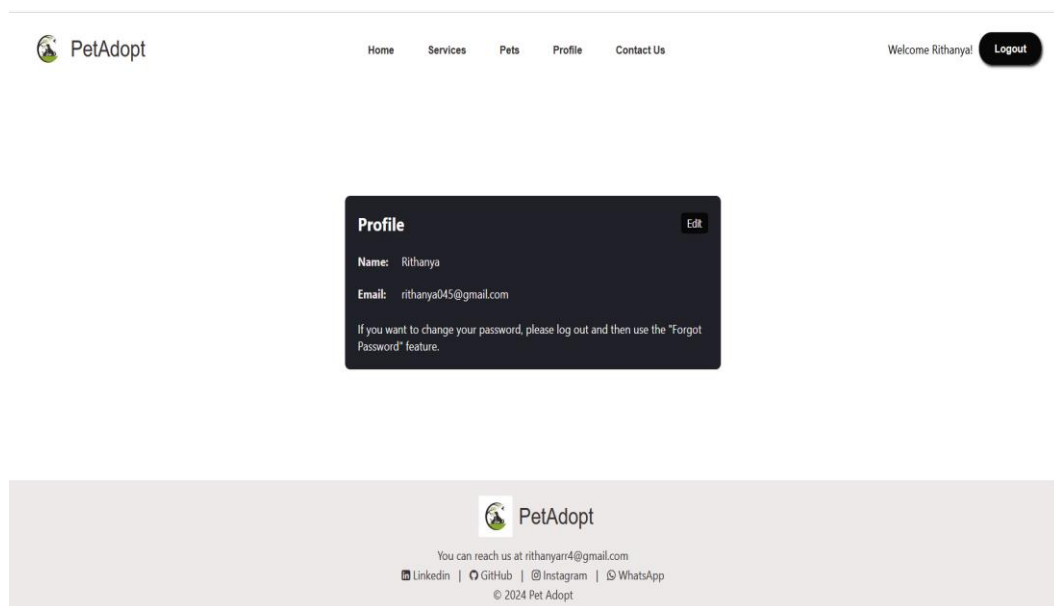


Figure B.13 User profile page

Profile Page: Displays the user's name and email ID for quick reference. To update your password, please log out and use the 'Forgot Password' feature. Manage your account details securely and effortlessly.

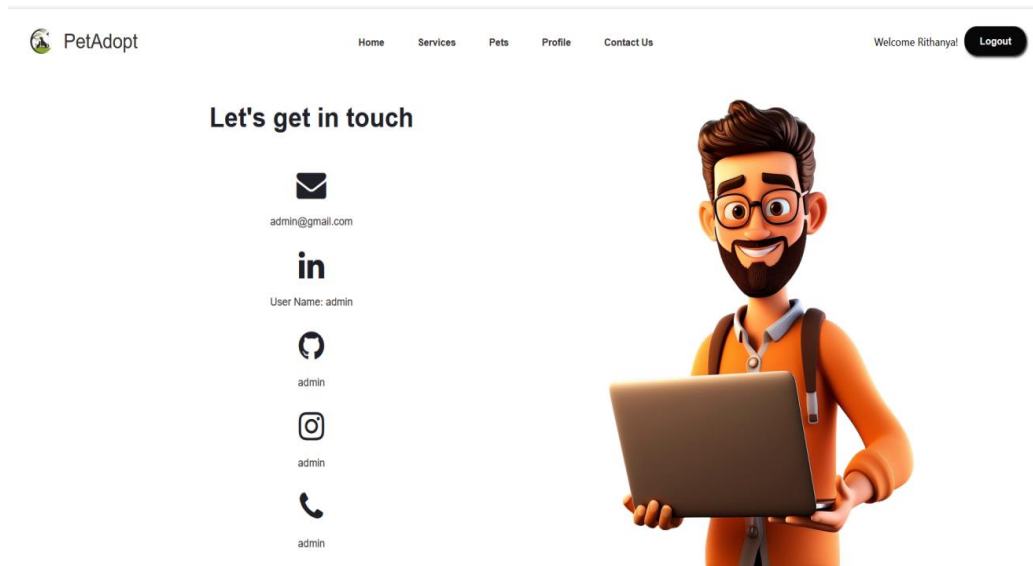


Figure B.14 contact page

Get in Touch with the Admin: Reach out for assistance or inquiries. Contact via email at admin@example.com or visit the admin portal at admin.in. Follow the admin's updates on Instagram at @adminpage.

REFERENCES

- [1] Alex Banks & Eve Porcello (2020), “Learning React” 2nd Edition, O'Reilly Media.
- [2] David Herron (2020), “Node.js Web Development” 5th Edition, O'Reilly Media.
- [3] JonDuckett (2011), “HTML and CSS: Design and Build Websites” 1st Edition, Wiley.
- [4] M. Murphy (2019), “Mongodb: Up and Running” 2nd Edition, O'Reilly Media.
- [5] www.geeksforgeeks.org, “Learn about Node.js and React.js”.
- [6] www.javatpoint.com, “Learn about JavaScript”.
- [7] www.tutorialspoint.com, “Learn about Mongodb and Realtime Database”.
- [8] www.w3schools.com, “Learn about HTML, CSS and React.js”.