

PROFESSIONAL TRAINING REPORT - II

entitled

PET LOST MANAGEMENT SYSTEM

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering with
specialization in Cyber Security

by

GOLUGURI SAMPATH REDDY

[Reg. No. 41614027]



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY) CATEGORY -1 UNIVERSITY BY UGC Accredited with
Grade "A++" by NAAC | 12B Status by UGC | Approved by AICTE JEPPIAAR NAGAR,
RAJIV GANDHI SALAI, CHENNAI - 600 119

APRIL 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

CATEGORY-I UNIVERSITY BY UGC

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **Mr. Goluguri Sampath Reddy (Reg.No.41614027)** who carried out the project entitled **"PET LOST MANAGEMENT SYSTEM"** under my supervision from January 2024 to April 2024.

Internal Guide

Dr. B. U. Anu Barathi, Ph.D

Head of the Department

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **Goluguri Sampath Reddy (41614027)**, hereby declare that the Professional Training Report-I entitled “**PET LOST MANAGEMENT SYSTEM**” done by me under the guidance of **Dr. B. U. Anu Barathi, Ph.D** is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering with specialization in Cyber Security.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneshwari M.E., Ph.D., Head of the Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Internal Guide **Dr. B. U. Anu Barathi, Ph.D** for his/her valuable guidance, suggestions and constant encouragement which paved way for the successful completion of my phase-1 professional Training.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

SAMPLE COURSE CERTIFICATE

< [Icons] ...
HCL - Internship (PT-2) |
Shortlisted Students Inbox ☆

 Sathyabama Place... 19 Jan
to sathyabama-pc-2025 ✓ [Icons] ...

Dear Students,

Greetings! This is to inform you that, attached is the list of students who are final shortlisted for the **HCL Internship (PT-2)** program.

All these students are requested to meet **Dr. Albert, Professor, Department of CSE, and Dr. Ashok, Professor, Department of CSE** for further queries if any.

With Warm Regards
Team Placement
Sathyabama Institute of Science & Technology

189	B.E - CSE with Cyber Security	CSE with Cyber Security	41614027	Sampath Reddy Goluguri
190	B.E - CSE with Cyber Security	CSE with Cyber Security	41614103	Vishal M
191	B.E - CSE with Cyber Security	CSE with Cyber Security	41614104	VISHAL .T
192	B.E - CSE with Cyber Security	CSE with Cyber Security	41614111	manoj kumar

< [Icons] ...
REMINDER - APPRECIATING 10
INTERNS - Re: HCL-
Sathyabama - Mini-Project - ☆
Project Titles and Weekly
Review Call.. Inbox

 Sankaran V 11 Feb
to Sankaran ✓ [Icons] ...

Good Morning HCL-Sathyabama Mini-Project Interns,

I have BCCed all Interns and Director Placement, Professors, Deans and HODs.

APPRECIATING the following 10 interns who have chosen their project titles.

Pls fill up the form before Tuesday, 13th Feb night.

HCL - Sathyabama Mini-Project List			
S.No.	Reg. No.	Name	Project Title
1	41614027	GOLUGURI SAMPATH REDDY	Pet lost management system
2	41733020	DANIEL LAWRENCE	Call taxis management system

< [Icons] ...
WELCOME - HCL-Sathyabama -
Mini-Project Internship Inbox ☆

 Sankaran V 9 Feb
to Sankaran ✓ [Icons] ...

Good Evening HCL-Sathyabama Mini-Project Interns,

I have BCCed all Interns and Director Placement, Professors, Deans and HODs.

This is Dr.V.Sankaran, HCL Technologies.

Happy to WELCOME you all to the mini-project internship.

Today, 7.00pm we have a kick-off call. Pls. attend.

Topic: Orientation Session-HCL Internship
Time: Feb 9, 2024 07:00 PM Mumbai,
Kolkata, New Delhi

Join Zoom Meeting
<https://zoom.us/j/97608953856?pwd=ZGc4RTJYcExNT0hHeDIKRXFZWJOUT09>

Meeting ID: 976 0895 3856
Passcode: 180254

I will send more details from tomorrow.

< [Icons] ...
FINAL - HCL-Sathyabama Mini-
Project GitHub Details- Pls. fill
up the details before end of ☆
tomorrow Tuesday, 30-
Apr-2024 Inbox

 Sankaran V 12:23 PM
to Sankaran, princemary.cs... ✓ [Icons] ...

Good Afternoon HCL-Sathyabama Mini-Project Interns,

I have BCCed all interns.

I have CCed the Dean of CSE, Placement Director, Dr. Vijay Ramalingam, Dr.Albert and HODs and Professors of the Departments.

Pls submit your project details in the following form before end of Tuesday, 30-Apr-2024.

You can modify the code and update the GitHub till end of Sunday, 05-May-2024. We will start our review by 6th May.

FINAL - HCL-Sathyabama Mini-Project GitHub Details

<https://forms.office.com/r/U6s80hbY4h>

Pls. fill up the above form which has fields:

ABSTRACT

CHAPT ER NO.	TITLE	PAGE NO.
-----------------	-------	-------------

This
Pet
Lost
Mana
geme

nt System serves as a user-centric platform dedicated to assisting pet owners in reporting lost or found pets. The journey commences with a user-friendly Home Page, which intuitively directs users to the "Get Started" section. If not already logged in, users are seamlessly guided to the Login Page, where they are presented with the option to either sign in with existing credentials or embark on the sign-up process. Following a successful authentication, users are seamlessly ushered into the Pet Details Form, a vital portal for submitting essential information about their pets. Within the confines of the Pet Details Form, users are provided with a critical choice: they can either report their pet as "Lost" or "Found." Opting for "Lost" results in the website's automatic generation of an Instagram post replete with the appropriate hashtag, subsequently published on the user's Instagram account. Conversely, selecting "Found" triggers a similar process, creating and posting an Instagram update adorned with the "Found" hashtag. With the successful posting of their pet's status on Instagram, users are gracefully redirected to an Alert Page, offering a reassuring confirmation of task completion. The Alert Page goes above and beyond mere confirmation, presenting users with a range of options to further enhance their experience. These options encompass the ability to swiftly return to the Home Page, initiate a fresh pet details form submission, or access the "Pet Community" – a dynamic space where users can peruse and engage with others' lost and found pet reports. In its entirety, "Furry Finder" embodies a user-centric ethos, employing a robust technological stack consisting of HTML, CSS, and JavaScript on the front-end, while harnessing the power of Node.js, Express.js, and MongoDB on the back-end. Seamless integration with Instagram is achieved through the Instagram Post API, ensuring a seamless and efficient platform for pet owners seeking to reunite with their beloved animals.

TABLE OF CONTENTS

	ABSTRACT		vi
	LIST OF FIGURES		viii
1	INTRODUCTION		1
2	LITERATURE SURVEY		3
3	REQUIREMENTS ANALYSIS		
	3.1	Objective	9
	3.2	Requirements	11
	3.2.1	Hardware Requirements	13
	3.2.2	Software Requirements	14
4	DESIGN DESCRIPTION OF PROPOSED PRODUCT		
	4.1	Existing System	16
	4.2	Proposed Methodology	17
	4.2.1	Ideation Map/System Architecture	21
	4.2.2	Various stages	23
	4.2.3	Internal or Component design structure	25
	4.3	Working Principles	27
	4.3.1	Output Screenshots	31
	4.4	Features	35
	4.4.1	Novelty of the Project	36
5	CONCLUSION		38
	REFERENCES		39
	APPENDIX		
		A. Research Paper	40
		B. Source Code	48

LIST OF FIGURES

Figure No	Figure Name	Page No
4.2.1	System Architecture of Furry Finder	21
4.2.2	Stages of Implementation	23
4.2.3	Internal Components of Furry Finder	25
4.3.1.1	Output	31
4.3.1.2	Website Screenshots	32
4.3.1.3	Instagram Screenshots	34

CHAPTER 1

INTRODUCTION

Overview:

In a world where pets are cherished members of countless households, the heart-wrenching experience of a lost pet can be a source of profound distress. "Furry Finder" emerges as a beacon of hope and innovation in this context, offering a dedicated platform that not only simplifies the process of reporting lost or found pets but also leverages the power of social media to facilitate swift reunions. This introduction embarks on a journey into the heart of "Furry Finder", where user-centric design and cutting-edge technology converge to create a transformative experience for pet owners and animal lovers alike.

The concept of "Furry Finder" is rooted in empathy and a genuine understanding of the bond between humans and their animal companions. At its core, it seeks to alleviate the anguish and uncertainty that accompanies the loss of a beloved pet by providing a structured and efficient mechanism for reporting and disseminating crucial information. In this age of digital connectivity, my platform recognizes the pivotal role of social media, particularly Instagram, as a tool for mobilizing communities and amplifying the reach of vital information. Thus, "Furry Finder" seamlessly integrates with the Instagram Post API, enabling users to effortlessly share their pet's status with the world, thereby enhancing the likelihood of a reunion.

As users embark on their " Furry Finder " journey, they are greeted by a user-friendly Home Page that beckons them to "Get Started." Here, a thoughtful and intuitive design guides them towards the next steps. For those not yet logged in, the path leads to the Login Page, where they have the option to access their existing accounts or embark on the user registration process, ensuring a seamless entry into the " Furry Finder " ecosystem.

Once authenticated, users are ushered into the Pet Details Form, a pivotal space where they provide essential information about their pets. Within this form, they face a critical decision - to report their pet as "Lost" or "Found." This choice triggers an automated process where " Furry Finder " generates Instagram posts complete with appropriate hashtags, subsequently posting them to the user's Instagram account. The satisfaction of a successfully completed task is conveyed through a redirection to an Alert Page, providing not only confirmation but also additional options to enhance the user experience.

These options include returning to the Home Page, initiating a fresh pet details form submission, or gaining access to the "Pet Community". In this vibrant community space, users can browse through the reports of other lost and found pets, fostering a sense of unity and solidarity among pet owners and animal lovers. In its entirety, " Furry Finder " embodies a philosophy that transcends mere functionality. It represents a commitment to empathy, innovation, and the power of technology to reunite lost pets with their owners, bringing solace and joy to countless homes.

CHAPTER 2

LITERATURE REVIEW

Survey

The literature survey conducted for this project has illuminated key facets of pet loss management and user experience. The articles examined provide a comprehensive view of the challenges faced by pet owners when their animals go missing, addressing both the practical aspects of tracking lost pets and the emotional toll it takes on individuals. Some of the articles are mentioned below that focuses on the pet lost management.

- 1. Pet Loss and Grieving Strategies: A Systematic Review of Literature**
Author- Ruth E. Cervantes; San Jose State University

This literature review highlights the significant impact of companion animals on human lives, particularly in therapy and emotional support. Pets are often considered necessary due to medical conditions and are integral members of the family. The loss of a pet can result in intense grief, which is often not recognized in the workplace. The review suggests that bereavement leave, typically offered to employees for the loss of immediate human family members, should also include the loss of a pet. Pet owners often experience a more profound sense of grief when their pets pass away compared to the loss of a human family member. Companies offering bereavement leave for pet loss have observed positive outcomes, including improved employee morale, higher retention rates, increased company loyalty, and reduced stress levels among employees.

- 2. "Where is my pet? A global solution for locating lost pets,"**
Author- Montserrat Bassas Cordoba.

The thesis proposal, "Where is my pet?", aims to develop a software system to aid in the recovery process of missing pets. The system will use image recognition and geolocation technologies to facilitate the search for lost animals. The thesis will analyze available technologies, including

native apps, web applications, REST and GraphQL APIs, and image recognition services, to make informed technology choices. The final solution will consist of a web application built with React, a RESTful API developed with NodeJS, and a relational database using PostgreSQL. These components will run on cloud technology, specifically Heroku, and will leverage Google Cloud Vision for image processing capabilities. The proposal emphasizes the importance of technology selection, design, and implementation decisions throughout the development process.

3. "Frequency of Lost Dogs and Cats in the United States and the Methods Used to Locate Them."

Authors- Emily Weiss, Margaret Slater, Linda Lord.

A study conducted between September and November 2010 found that 14% of dogs and 15% of cats were lost within the past five years. The study did not identify any owner demographic variables associated with the likelihood of losing a pet. However, 93% of lost dogs and 75% of lost cats were eventually recovered. Dogs were more likely to get lost multiple times and had less identification than cats. The most common methods for recovery for dogs were searching the neighborhood and returning on their own, with only 14% found through an identification tag. For cats, returning on their own was the most common method. The study concluded that knowledge of successful methods for finding lost dogs and cats is valuable for pet owners. However, since 25% of lost cats were not found, alternative methods for reuniting lost cats and their owners are needed. The study proposes using collars and ID tags or humane trapping as potential approaches to improve the recovery of lost cats.

4. A Literature Review: Pet Bereavement and Coping Mechanisms

Authors- Rachel Caddiell, Kenneth D Royal, Margaret E Gruen

This review explores the grief response and coping mechanisms of pet owners who have experienced the loss of their pets. It synthesizes information from 48 studies and identifies several major themes. The

review discusses the factors influencing grief response, the disenfranchised nature of pet loss, and the ambiguous nature of pet loss. It also identifies several coping mechanisms used by bereaved pet owners, such as isolation, seeking social support, maintaining bonds with pets' memories, memorialization practices, turning to religion for comfort, and forming new relationships with other animals. The findings suggest that bereaved pet owners often experience disenfranchisement regarding their grief, and the need for greater understanding and acknowledgment of the profound impact of pet loss. Future research should focus on evaluating the effectiveness of coping mechanisms employed by individuals grieving the loss of their pets.

5. Search and identification methods that owners use to find a lost dog.

Authors- L. Lord, T. Wittum, A. Ferketich, J. Funk, P. Rajala-Schultz

The study analyzed the process of dog owners searching for lost dogs in Montgomery County, Ohio, between June 1 and September 30, 2005. The research found that 71% of the 187 lost dogs were recovered, with a median recovery time of 2 days. The primary recovery methods were contact with an animal agency, dog license tags, and posting neighborhood signs. Owners had a higher likelihood of recovering their dogs when they took specific actions, such as contacting an animal agency or posting neighborhood signs. Dogs with a dog license tag also had a higher likelihood of recovery. However, owners were less likely to recover their dogs if they believed their pets were stolen. The study concluded that various factors, including actions taken by owners and the presence of identification, play a crucial role in determining the likelihood of recovering lost dogs.

6. Coping With Losing a pet.

Authors- Lawrence Robinson and Jeanne Segal, Ph.D.

Many of us share an intense love and bond with our animal companions. For us, a pet is not “just a dog” or “[just a cat](#),” but rather a beloved member of our family, bringing companionship, fun, and joy to our lives. A pet can add structure to your day, keep you active and social, help you to overcome setbacks and challenges in life, and even provide a sense of meaning or purpose. So, when a cherished pet dies, it's normal to feel racked by grief and loss. The pain of loss can often feel overwhelming and trigger all sorts of painful and difficult emotions. While some people may not understand the depth of feeling you had for your pet, you should never feel guilty or ashamed about grieving for an animal friend.

While we all respond to loss differently, the level of grief you experience will often depend on factors such as your age and personality, the age of your pet, and the circumstances of their death. Generally, the more significant your pet was to you, the more intense the emotional pain you'll feel. The role the animal played in your life can also have an impact. For example, if your pet was a working dog, service animal, or therapy animal, you'll not only be grieving the loss of a companion but also the loss of a coworker, the loss of your independence, or the loss of emotional support.

7. Pet Humanisation And Related Grief - Authors: Stefania Uccheddu, Lorian De Cataldo, Mariangela Albertini, Stanley Coren.

The aim of this study was the scientific validation of a novel instrument (the Mourning Dog Questionnaire) designed to allow a comprehensive quantitative analysis of grief responses in dog owners after the death of a pet dog, which is a still underestimated issue. This instrument was based on a grid of quantitative measurable outcomes that combines five different questionnaires concerning levels of attachment to pets, responses to the loss of a pet, outlook on life events and grief, perceptions of animals in

terms of emotions, needs and legal right. We found that pet owners have the tendency to humanise their pet and perceive animals no differently from humans in terms of emotions, needs and legal rights. In addition, after the death of the pet, a negative view of life was described according to the findings. The Mourning Dog Questionnaire could be a useful tool considering the high numbers of people who are at risk of experiencing the loss of a companion dog, which makes this type of grief as potential major concern for public health and human welfare.

**8. The Impact of continuing Bonds Between Pet Owners And Their Pets
Authors - Ben Hughes, Beth Lewis Harkin.**

When a pet dies, owners can experience similar levels of grief as when a human dies. Previous research indicates the role of continuing bonds (CB) when a pet is alive. To understand the impact of these bonds after the pet has died, we conducted a systematic narrative synthesis according to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses guidelines (PRISMA). Findings were heterogenous, yet there were still parallels in the literature. CB can sometimes aggravate and intensify grief experiences, particularly when pet grief is perceived as disenfranchised grief. However, identifying appropriate bonds can be useful to moderate the intensity of grief and be a valuable mechanism of support. CB can also help post-traumatic growth of owners.

9. Pet Grief: Tools To Assess Owners Bereavement And Veterinary Communication – Author: Ines Testoni, Lucia Ronconi, Lorian De Cataldo

In Italy, there are approximately 60.5 million companion animals, and 92% of Italian owners consider them as family members. Despite the growing interest in pet bereavement and in end-of-life (EOL) issues in veterinary medicine across the world, there are still very few Italian studies on the psychological impact of losing a pet, and there are no instruments in the Italian language to assess grief, following the death of a companion animal

and the impact of effective veterinary communication skills on pet bereavement. The aim of this study is the Italian adaptation of instruments (Pet Bereavement Questionnaire (PBQ); Regret of Bereaved Family Members (RBFM); Shared Decision-Making Questionnaire (SDM-Q-9); Consultation and Relational Empathy Measure (CARE)), which can be useful in assessing bereavement in companion animal owners and communication skills in veterinary medicine. All the instruments obtained good internal reliability. The results showed that the use of the Italian versions of all the instruments is useful, and that the CARE, the SDM-Q-9, and the Regret of Bereaved Family Members (RBFM) developed for the human healthcare context, may also be used in veterinary medicine, and in further Italian pet bereavement and EOL veterinary studies.

10. Overcoming The Social Stigma Of Losing A Pet
Authors – Michelle Kay Crossley

During the COVID-19 pandemic, individuals found themselves spending more time with their pets and relied on them to maintain normalcy and provide security during isolation. Pets play a significant role in the lives of their caregivers, taking on different attachment roles depending on the needs of the individual. Grieving the death of a pet continues to be disenfranchised in society. Perceptions of judgment can lead individuals to grieve the loss without social support. The present review builds on research in the field of pet loss and human bereavement and factors in the impact of the COVID-19 pandemic on human-animal attachment. A goal of the present review is to provide counselors with perspectives to consider in their practice when working with clients who have attachments to their companion animals and to acknowledge the therapeutic benefits of working through the grief process to resolution as a way to continue the bond with a deceased pet.

By conducting a literature survey on these articles, I gained a deeper understanding of the broader landscape of pet loss management, emotional aspects related to pet

loss, emerging technologies, and global solutions. This knowledge helped me to refine my project's features, user engagement strategies, and long-term development goals to make "Furry Finder" even more effective in reuniting lost pets with their owners.

CHAPTER 3

REQUIREMENTS ANALYSIS

3.1 OBJECTIVE OF THE PROJECT

The primary objectives of the project are:

❖ **User Registration and Authentication:**

- Users can register for an account with a unique username and a valid email address.
- Registered users can log in securely using their credentials.
- Password reset functionality should be available for users who forget their passwords.

❖ **User Profile:**

- Registered users have a profile page where they can edit their personal information (e.g., name, contact details) and change their profile picture.

❖ **Pet Details Submission:**

- Users can submit information about a lost or found pet.
- The submission form includes fields for pet name, description, photos (upload or URL), location, and contact information.
- Users can upload multiple photos of the pet.

❖ **Pet Details Management:**

- Users can view a list of their submitted pet details.
- They can edit the details of pets they've submitted.

- Users can delete a pet submission if needed.

❖ **Viewing Others' Submissions:**

- Users can search and view pet details submitted by other users.
- A search or filter mechanism should be available for users to find specific pets.

❖ **Automatic social media Posting:**

- When a user submits a pet as "LOST" or "FOUND," the system automatically generates and posts a message on Instagram with the appropriate hashtag (#lost or #found).
- The Instagram post includes a link back to the pet's details on the website.

❖ **Pet Status Updates:**

- Users can update the status of a pet from "LOST" to "FOUND" or vice versa if the pet's situation changes.

❖ **Commenting and Communication:**

- Users can comment on pet details to offer assistance or share information.
- A messaging system or contact information exchange feature can be implemented to facilitate communication between users regarding specific pets.

❖ **Notification System:**

- Implement a notification system to alert users when there are updates or comments on their pet submissions.

❖ **Reporting and Flagging:**

- Users can report inappropriate or false pet submissions for moderation.

❖ **Blog or Complaints Section (Alternative Name Suggestion Required):**

- Create a section for users to post pet-related blogs, share experiences, or report complaints about pet-related issues.

❖ **Privacy Settings:**

- Users can control the privacy settings for their pet submissions, deciding whether they want to share contact information publicly or keep it private.

By achieving these objectives, the project aims to create a valuable resource for pet owners and animal lovers while leveraging technology to improve the process of reuniting lost pets with their owners.

3.2 REQUIREMENTS

Functional Requirements:

➤ **User Registration and Authentication:**

Allow users to create accounts, log in securely, and manage their personal information.

➤ **Pet Details Submission:**

Provide users with a form to submit detailed information about lost or found pets, including photos and descriptions.

➤ **Programmed social media Posting:**

Integrate with the Instagram API to enable automatic posting of pet details on Instagram with relevant hashtags (#lost or #found).

➤ **User-Friendly Interface:**

Design a user-friendly and intuitive interface that ensures a seamless and engaging experience for users.

➤ **Data Storage:**

Store user data and pet details in a secure and organized manner using MongoDB.

➤ **Information Sharing:**

Facilitate the sharing of pet details and related information with a wider community of pet lovers and pet owners.

➤ **Additional Features:**

Provide options for users to view other users' submissions, access a blog or complaints section for pet-related issues, and navigate the website easily.

➤ **Efficient Pet Management:**

Help users effectively manage and track the status of lost or found pets, making it easier for them to reunite with their beloved animals or assist others in doing so.

➤ **Social Impact:**

Contribute to the local and global community by aiding in pet reunifications and promoting responsible pet ownership.

Non-Functional Requirements:

▪ **Security:**

Ensure the security of user data and prevent unauthorized access. Implement secure data transmission (HTTPS) and authentication mechanisms.

▪ **Performance:**

The website should load quickly and handle a reasonable number of concurrent users without significant performance degradation.

▪ **Scalability:**

Design the system to be scalable so that it can handle increased traffic and data growth.

- **Usability:**

The user interface should be intuitive and user-friendly.

- **Compatibility:**

Ensure compatibility with popular web browsers (e.g., Chrome, Firefox, Safari).

3.2.1 HARDWARE REQUIREMENTS

1. Web Server:

- A dedicated or cloud-based web server to host the website and handle HTTP requests.
- Sufficient CPU and RAM resources to handle incoming requests and serve web pages efficiently.
- Adequate bandwidth to support web traffic.

2. Database Server:

- A server to host the MongoDB database.
- The server should have sufficient storage capacity to store user data, pet details, and images.
- Adequate CPU and RAM resources to handle database operations efficiently.

3. Storage:

- Storage for hosting user-uploaded pet photos.
- A backup solution to ensure data integrity and recovery in case of data loss.

4. Network Infrastructure:

- Reliable network infrastructure to ensure the website's availability and responsiveness.
- Network security measures to protect against unauthorized access and cyber threats.

3.2.2 SOFTWARE REQUIREMENTS

1. Frontend Technologies:

- HTML, CSS, and JavaScript for building the user interface.
- A modern web browser for testing and viewing the application during development.

2. Backend Technologies:

- Node.js for server-side scripting and handling HTTP requests.
- Express.js for building the web application and defining routes.
- MongoDB as the database management system for storing user data, pet details, and comments.

3. Development Tools:

- A code editor or integrated development environment (IDE) for writing code.
- A version control system (e.g., Git) for managing code changes and collaboration.
- Node.js and npm for package management.

4. Operating System:

- The choice of operating system for both the web server and database server can vary based on your preference and hosting environment. Common options include Windows, MacOS, Linux distributions (e.g., Ubuntu, CentOS) for servers.

5. Web Server Software:

- You'll need web server software to serve your web application to users. Common choices include Nginx or Apache.

6. Database Server Software:

- MongoDB as the NoSQL database management system. You'll need to install and configure MongoDB on the database server.

7. API Integration:

- Implementation of the Instagram API for automatic posting. You'll need to use the API's documentation and libraries for integration.

8. Security Software:

- Implement security measures such as SSL/TLS certificates for secure data transmission (HTTPS) and firewall configurations to protect against security threats.

9. Deployment Tools:

- Tools and scripts for deploying your web application to the web server.

10. Monitoring and Logging Tools:

- Implement monitoring tools to track the health and performance of your application and server. Also, set up logging to record errors and activities.

CHAPTER 4

DESIGN DESCRIPTION OF PROPOSED PROJECT

4.1 EXISTING SYSTEM

In the existing system, there may not be a dedicated online platform or website for pet owners and enthusiasts to manage and share information about lost and found pets. Pet owners may rely on various offline methods and social media platforms to post information about their lost or found pets, which can be fragmented and less efficient. Here are the characteristics of the existing system:

- **Fragmented Information:** Pet owners may resort to using social media platforms or local bulletin boards to share information about lost or found pets, resulting in fragmented and unorganized data.
- **Limited Automation:** Posting on social media platforms like Instagram is a manual process, requiring users to perform each post individually.
- **Lack of Centralization:** There is no centralized platform or database specifically dedicated to managing lost and found pet information, making it challenging for users to search for and access relevant information.
- **Limited User Engagement:** Users may not have access to features like comments, notifications, or a dedicated blog or complaints section to discuss and share pet-related information and experiences

4.2 PROPOSED METHODOLOGY

The proposed system is my pet management website project, which aims to address the limitations of the existing system by providing a dedicated online platform with enhanced features and functionalities. Here are the key aspects of the proposed system:

- **Centralized Platform:** The website serves as a centralized platform where pet owners can submit, search for, and manage information about lost and found pets, streamlining the process.
- **Automated social media Posting:** Integration with the Instagram API allows automatic posting of pet details on Instagram with the appropriate hashtags (#lost or #found), increasing the reach and efficiency of pet information dissemination.
- **User Engagement:** Users have access to features like comments, notifications, and a dedicated blog or complaints section to engage with the community, share pet-related experiences, and discuss issues.
- **User-Friendly Interface:** The user interface is designed to be intuitive and user-friendly, making it easy for users to submit pet details and navigate the website.
- **User Registration and Authentication:** Users can create accounts, log in securely, and manage their pet submissions, enhancing security and personalization.

- **Data Management:** The system stores user data, pet details, and comments in a structured manner using MongoDB, facilitating efficient data retrieval and management.
- **Notification System:** Users receive notifications when there are updates or comments on their pet submissions, improving communication and engagement.
- **Privacy Control:** Users can control the privacy settings for their pet submissions, ensuring that their contact information remains private if desired.

1. Project Planning:

- **Define Objectives:** Clearly define the project's objectives, including its core functionalities and expected outcomes.
- **Requirements Gathering:** Collect and document detailed functional and non-functional requirements.
- **Scope Definition:** Determine the project's scope, including the features, components, and timeline.
- **Team Formation:** Assemble a project team with roles such as project manager, developers, designers, and testers.
- **Technology Stack Selection:** Choose the appropriate technologies and tools based on project requirements.

2. Design and Architecture:

- **System Architecture:** Create a system architecture diagram that illustrates how different components will interact.
- **Database Design:** Design the database schema for storing user data, pet details, comments, and other relevant information.
- **User Interface Design:** Design the user interface (UI) for the website, including wireframes and mock-ups.

3. Development:

- **Frontend Development:** Develop the frontend of the website using HTML, CSS, and JavaScript. Implement user registration, login, and user profile features.
- **Backend Development:** Build the backend using Node.js and Express.js. Develop API endpoints for user management, pet submissions, comments, and notifications.
- **Database Implementation:** Set up and configure MongoDB, and implement data storage and retrieval mechanisms.
- **Instagram API Integration:** Integrate the Instagram API to enable automatic posting of pet details.
- **User Interface Implementation:** Translate the UI designs into functional web pages.

4. Testing:

- **Unit Testing:** Test individual components, functions, and modules to ensure they work as intended.
- **Integration Testing:** Verify that different parts of the system work together seamlessly.
- **User Acceptance Testing (UAT):** Involve users or testers to perform UAT to validate that the system meets requirements and is user-friendly.
- **Security Testing:** Conduct security assessments to identify and address vulnerabilities.

5. Deployment:

- **Deployment Environment Setup:** Prepare the production server environment for hosting the website.
- **Deployment:** Deploy the website and database to the production server.
- **Domain Configuration:** Configure domain settings and set up SSL/TLS certificates for secure connections.

- **Monitoring and Logging:** Implement monitoring tools to track system health and performance.

6. User Training and Documentation:

- Provide user documentation and training materials to help users navigate and utilize the website effectively.

7. Launch and Post-Launch Activities:

- Officially launch the website to the public.
- Monitor and address any issues that arise after the launch.
- Collect user feedback and make necessary improvements based on user experiences.

8. Maintenance and Updates:

- Regularly maintain the website, including applying security patches and updates.
- Enhance the website based on user feedback and emerging requirements.

9. Marketing and Community Engagement:

- Promote the website to build a community of users and pet enthusiasts.
- Encourage user participation and contributions to the blog or complaints section.

10. Evaluation:

- Periodically assess the website's performance and effectiveness in achieving its objectives.

- Conduct user surveys and gather metrics to measure success.

This proposed methodology provides a structured approach to the development and deployment of my pet management website project, ensuring that it meets its objectives and provides a valuable resource for users.

4.3.1 Ideation Map/System Architecture

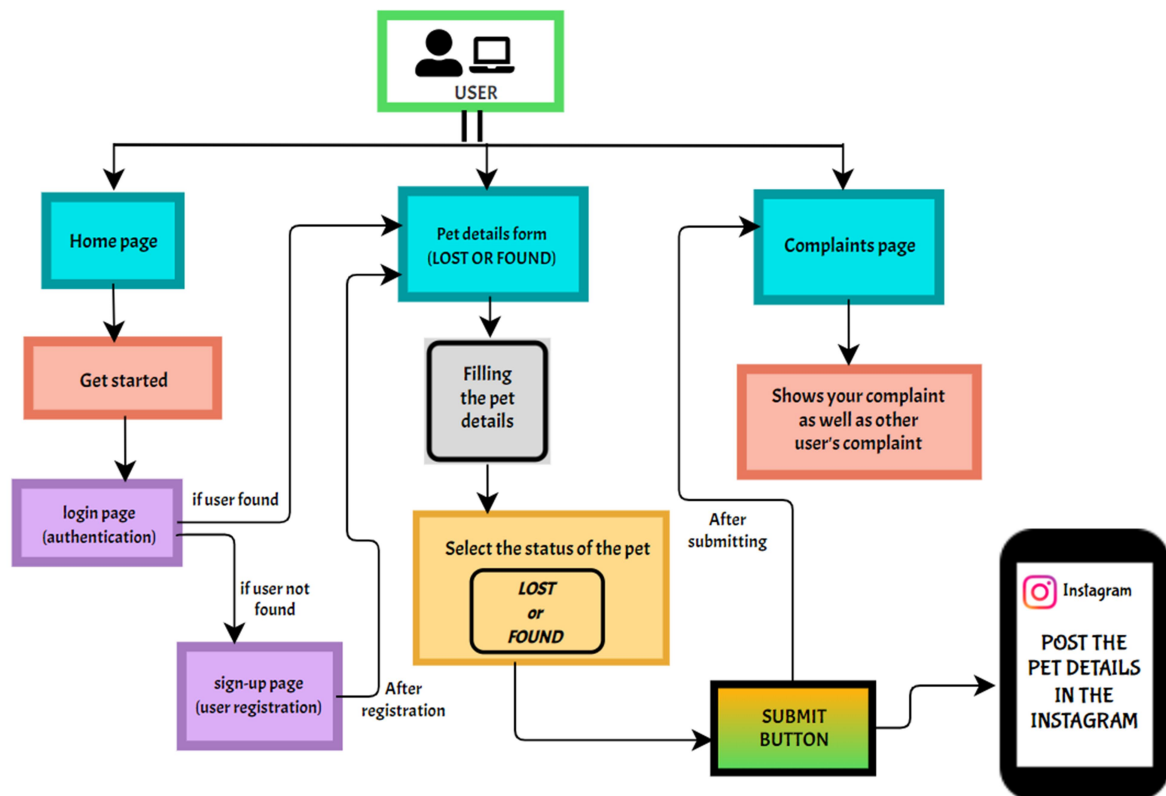


FIG 4.2.1- SYSTEM ARCHITECTURE OF FURRY FINDER

The ideation guide and framework engineering structure the central ideas and outlines for the improvement of my pet administration site project. The ideation map

addresses the inventive conceptualizing process where we recognized the undertaking's center goals and usefulness, for example, client enrollment, pet subtleties accommodation, computerized online entertainment posting, and client commitment through highlights like remarking and a committed blog or objections segment. It additionally exemplifies the soul of centralization and mechanization, which are integral to my venture's incentive.

The framework design, then again, fills in as the specialized system whereupon we'll rejuvenate my ideation map. It portrays the parts and innovations we'll utilize, beginning with a strong frontend constructed utilizing HTML, CSS, and JavaScript for an instinctive UI. my Node.js and Express.js-based backend will deal with the server-side rationale, while MongoDB will act as my proficient NoSQL data set for putting away client information, pet subtleties, and remarks. Coordinating the Instagram Programming interface to mechanize virtual entertainment postings is a vital specialized challenge and a chance for my task. At last, the engineering highlights the significance of safety, adaptability, and ease of use, which are basic contemplations as we move from idea to execution.

In synopsis, the ideation guide and framework design work couple, with the ideation map directing my undertaking's vision and usefulness, and the framework engineering giving the specialized guide to transform that vision into a completely utilitarian pet administration site. These two parts address the cooperative energy among innovativeness and innovation that will drive the outcome of my venture.

4.2.2 Various Stages

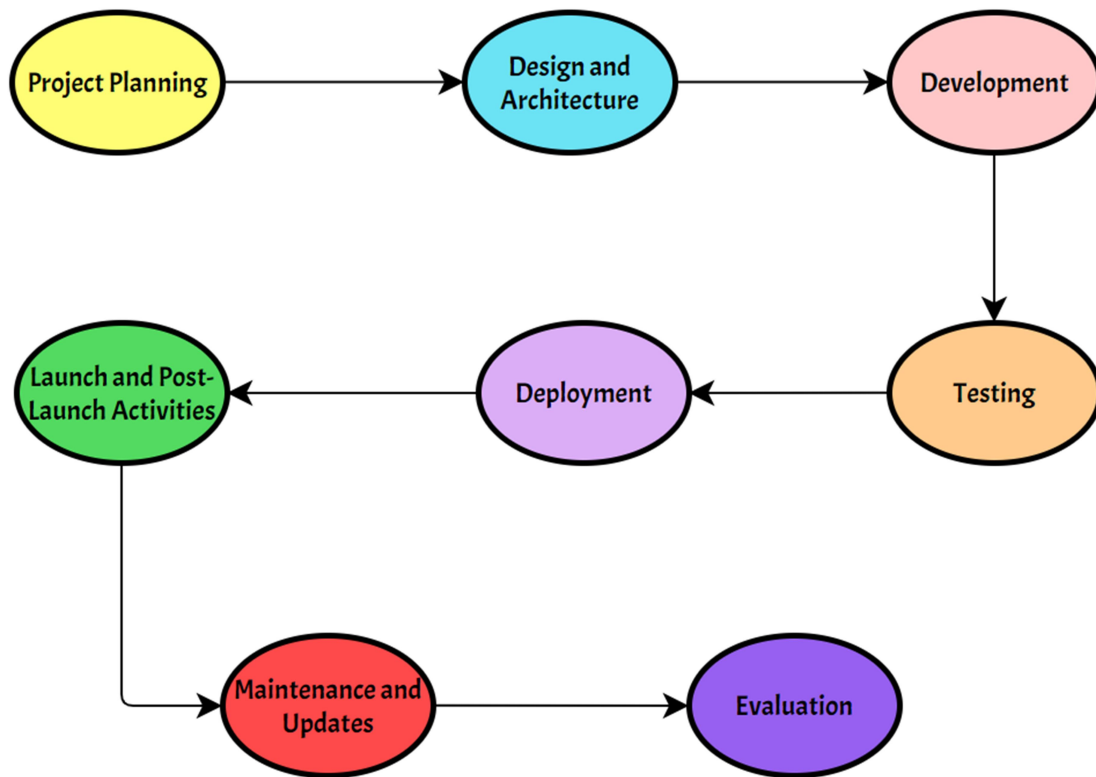


FIG 4.2.2- STAGES OF IMPLEMENTATION

The development of my pet management website project can be broken down into several distinct stages, each of which plays a crucial role in the project's overall success. The initial stage, "Project Planning," involves defining the project's objectives, gathering detailed requirements, and establishing the scope and timeline. It also includes assembling a skilled project team and selecting the appropriate technology stack.

The "Design and Architecture" stage comes next, where we create a comprehensive system architecture that outlines how different components will interact, design the database schema for efficient data storage, and create user-friendly interface designs. This stage sets the technical and visual foundations for the project.

Following design, we enter the "Development" phase, where the frontend and backend components of the website come to life. Frontend developers use HTML, CSS, and JavaScript to build the user interface, while backend developers implement server-side logic using Node.js and Express.js. Simultaneously, the MongoDB database is set up, and data storage and retrieval mechanisms are implemented. The integration of the Instagram API for automatic posting is a significant milestone during this stage.

Once development is complete, we enter the "Testing" phase. Unit testing ensures that individual components work as intended, while integration testing verifies that different parts of the system work seamlessly together. User acceptance testing (UAT) is conducted to validate that the system meets its requirements, is user-friendly, and performs as expected. Additionally, security testing is vital to identify and address vulnerabilities.

With a thoroughly tested system in place, we proceed to the "Deployment" stage. This involves setting up the production server environment, deploying the website and database, configuring domain settings and SSL/TLS certificates for secure connections, and implementing monitoring and logging tools to ensure the system's health and performance.

As we approach the "Launch and Post-Launch Activities" stage, we officially introduce the website to the public, closely monitoring its performance and addressing any issues that may arise. User feedback is actively sought, and improvements are made based on user experiences.

After launch, the project enters a phase of "Maintenance and Updates," where regular maintenance tasks, including applying security patches and updates, are performed. Enhancements and additional features may be implemented based on user feedback and evolving requirements.

Finally, the "Evaluation" stage involves assessing the website's performance and effectiveness, conducting user surveys, and collecting metrics to measure its success in achieving its objectives. This iterative approach ensures that my pet management website remains a valuable and responsive resource for pet owners and enthusiasts.

4.2.3 Internal or Component design structure

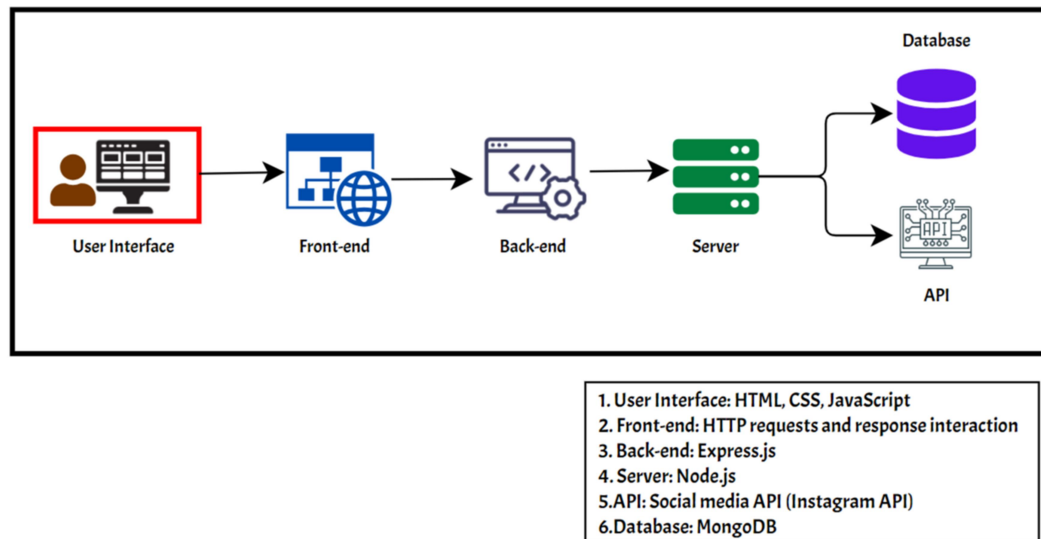


FIG 4.2.3- INTERNAL COMPONENTS OF FURRY FINDER

The internal or component design structure of my pet management website project is crucial aspect of the development process, as it defines the organization and relationships of the various software components that make up my system. This design phase serves as the bridge between the high-level system architecture and the actual implementation, providing a detailed blueprint for how each part of the system will function and interact.

At its core, my internal design structure is modular and follows best practices of software engineering. Each major functional component of the system, such as user management, pet details submission, database interaction, and Instagram API integration, is encapsulated within its own module or component. This

modular approach promotes code reusability, maintainability, and scalability, allowing us to make changes or updates to specific components without affecting the entire system.

Additionally, my component design emphasizes clean and organized code, adhering to coding standards and design patterns to enhance readability and reduce potential bugs. It also incorporates proper error handling mechanisms to ensure that the system gracefully handles unexpected situations.

Furthermore, as my system relies on asynchronous communication with external services like the Instagram API, the internal design includes mechanisms for managing asynchronous operations effectively, such as promises or `async/await` patterns in JavaScript.

In terms of data flow, the internal design structure outlines how data is passed between components and how it is processed and transformed as it moves through the system. This includes defining data models, validation processes, and data access layers to interact with the MongoDB database.

Security is a paramount concern, and my internal design incorporates security measures such as input validation, authentication, and authorization checks at various points in the system to protect user data and ensure secure interactions.

In summary, my internal or component design structure serves as the detailed roadmap for the development phase of my pet management website project. It promotes modularity, clean coding practices, data flow management, and security, all of which are vital for creating a robust and maintainable system that meets the project's objectives and user expectations.

4.3 Working principles

In the following section, we provide a detailed look at the inner workings of our pet management website's key features. This includes the underlying code that powers crucial functionalities such as user registration, pet details submission, and automatic social media posting. As we delve into the code snippets and their corresponding outputs, you'll gain a deeper understanding of how our platform seamlessly integrates user interactions and automation, creating a user-friendly and efficient environment for managing lost and found pets.

Step 1: Install Node.js

- Download Node.js from the official website: <https://nodejs.org/en/download/>
- Choose the LTS (Long Term Support) version for stability.
- Run the installer and follow the installation wizard.
- Check the version of the Node.js and npm to confirm the installation

```
shell

node -v
npm -v
```

Step 2: Install Necessary Dependencies and Initialize

- In the project directory, open a terminal or command prompt and run the following command to install the required dependencies and initialize the listed in your `package.json` file:

```
bash

npm install
```

```
bash Copy code

npm install mongoose express express-session multer csv-parser csv-writer e
```

```
npm install mongoose express express-session multer csv-parser csv-writer
ejs Instagram private-api
```

```
bash
```

```
npm init -y
```

Verify installation:

- After the installation process is complete, you can verify that the packages were installed correctly by checking your package.json file. It should now include the packages you installed under the dependencies section.

```
json
{
  "name": "Paws_Reconnected",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "csv-parser": "^3.0.0",
    "csv-writer": "^1.6.0",
    "dir": "^0.1.2",
    "dotenv": "^16.3.1",
    "ejs": "^3.1.9",
    "express": "^4.18.2",
    "express-session": "^1.17.3",
    "instagram-private-api": "^1.45.3",
    "multer": "^1.4.5-lts.1",
    "request-promise": "^4.2.6",
    "tailwindcss": "^3.3.3",
    "uuid": "^9.0.1"
  }
}
```

Step 3: Code in the Files

Code the program in the following mentioned files

Directories:

1. **public** (Directory):

- Contains a subdirectory named **img**.
- Inside the **img** directory, there is a file named **hero-bg.jpg**.

2. **views** (Directory):

Contains several **.ejs** files:

- **blog.ejs**
- **form.ejs**
- **index.ejs**
- **login.ejs**
- **signup.ejs**

Files:

3. **index.js**:

This is your main JavaScript file where you appear to be implementing some functionality.

4. **package-lock.json**:

This file is automatically generated by npm and keeps track of the exact version of each package installed. You don't need to edit this file manually.

5. **package.json**:

This file contains metadata about your project, including its dependencies. It's also used to define scripts and other project-related information.

6. **server.js**:

This file might be your main server file if you're building a web application. It's not clear from the provided information what this file contains.

Step 4: Integrating Instagram POST API to the website

Integrate the Instagram POST API to the website by typing the following code in the **server.js**

Instagram POST API Integration Code:

```
const ig = new IgApiClient();
    ig.state.generateDevice(process.env.IG_USERNAME);
    await ig.account.login(process.env.IG_USERNAME,
process.env.IG_PASSWORD);
    const imageBuffer = await get({
        url: 'https://images.pexels.com/photos/1643457/pexels-photo-1643457.jpeg?auto=compress&cs=tinysrgb&w=600',
        encoding: null,
    });

    await ig.publish.photo({
        file: await readFileSync(image1),
        caption: caption1,
    });

    console.log('Posted to Instagram successfully.');
```

```
} catch (error) {
    console.error('Error posting to Instagram:', error);
}
};

module.exports = postToInsta;
```

After integrating the Instagram POST API to the website create a new file name “.env”

.env: (no space)

.env Create this file with your Instagram credentials:

```
IG_USERNAME=<your_instagram_username>
IG_PASSWORD=<your_instagram_password>
```

Step 5: Execution

- Open a terminal in your project directory.

- Run the following command to execute your code:

```
bash
node index.js
```

If the code runs successfully, you should see the message "Posted to Instagram successfully."

4.3.1 OUTPUT SCREENSHOTS

FIG 4.3.1.1- OUTPUT

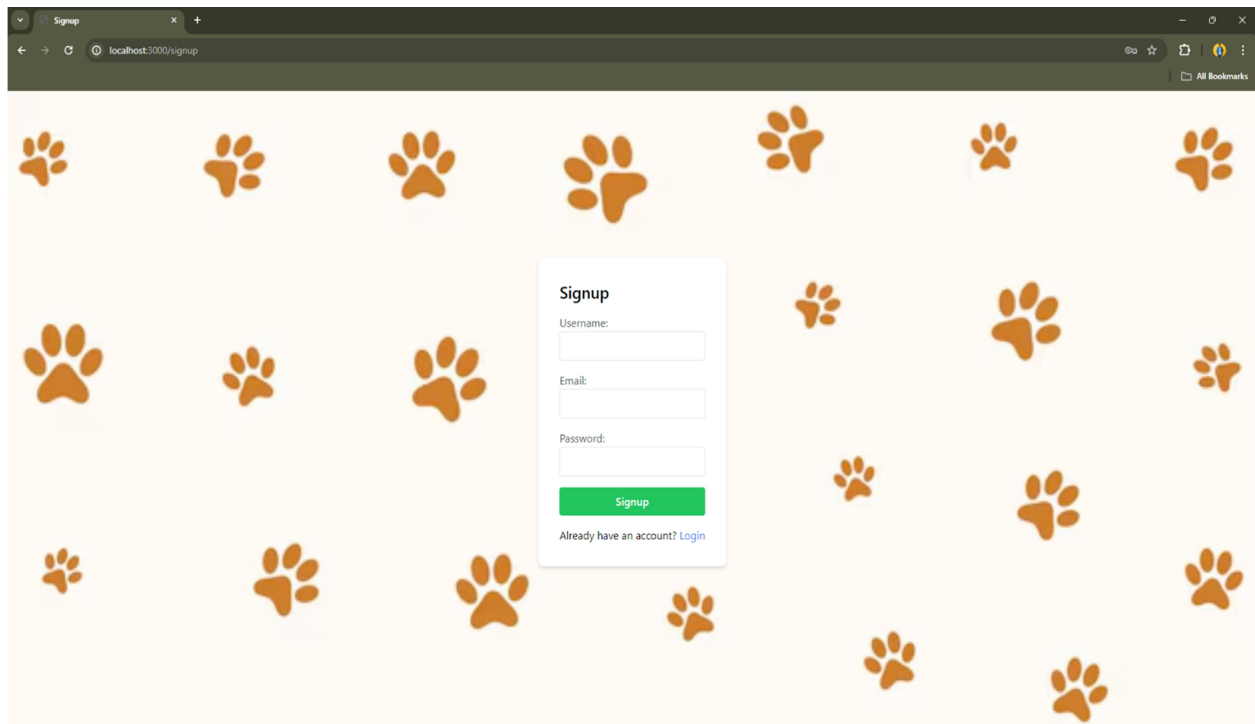
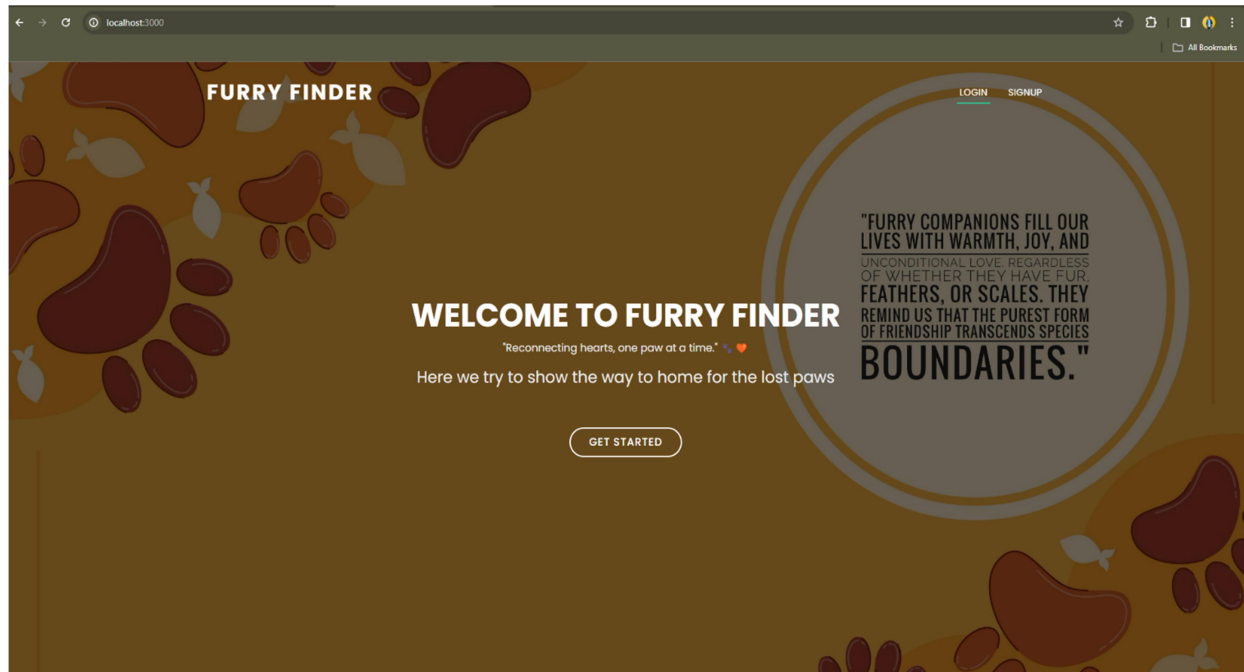
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

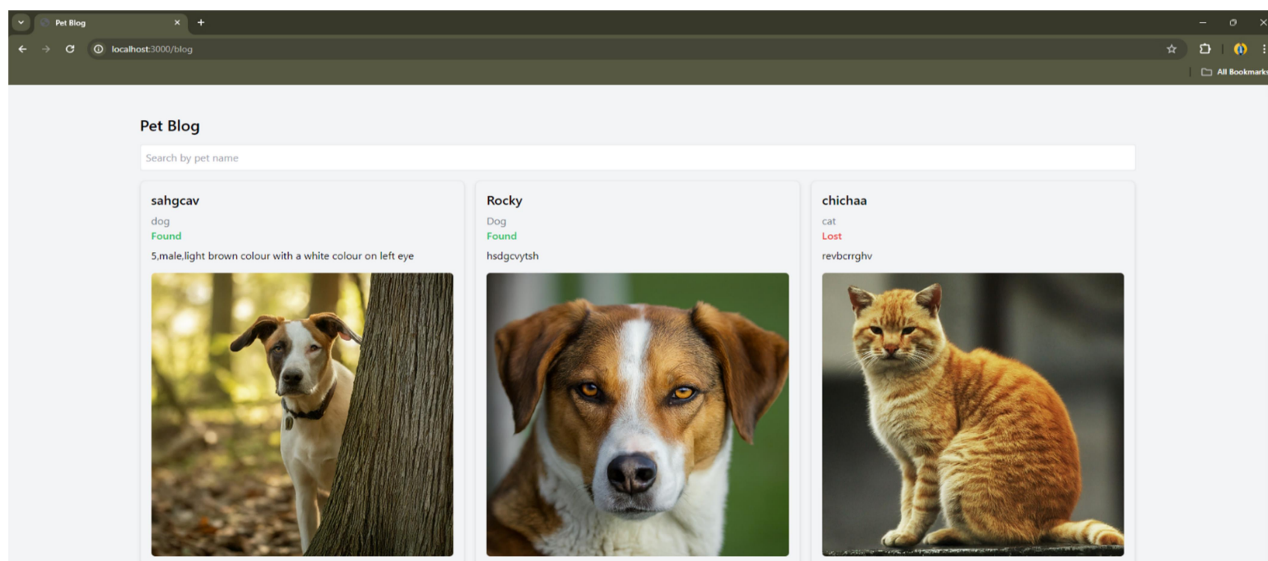
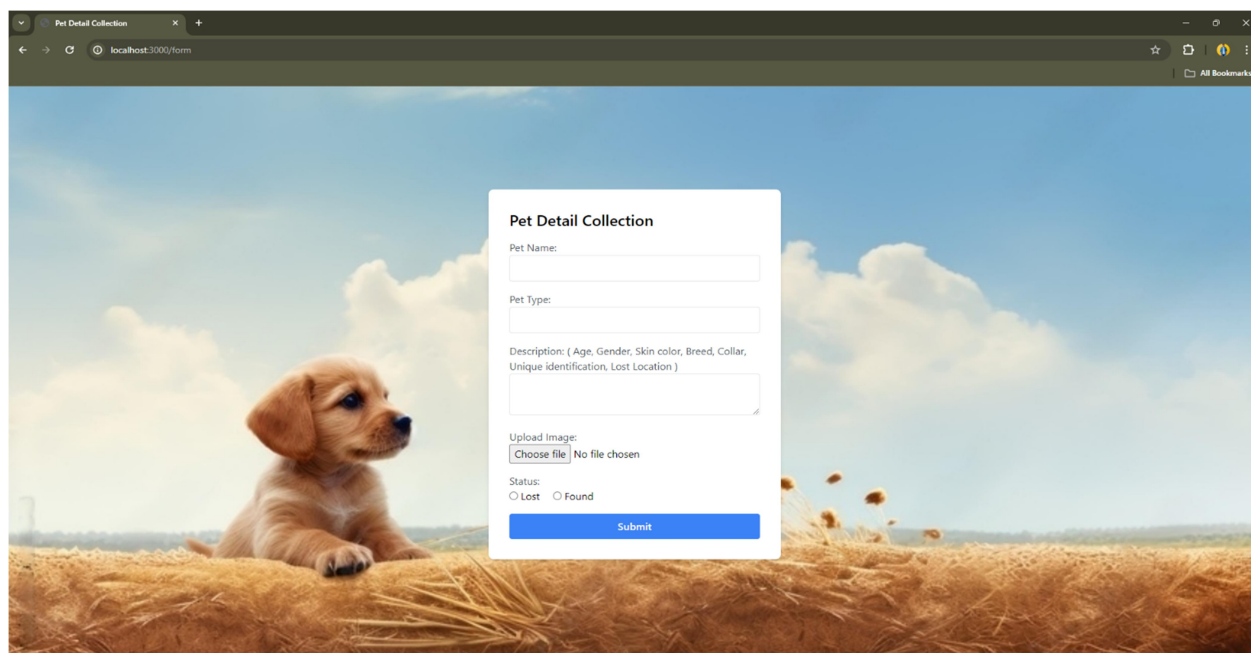
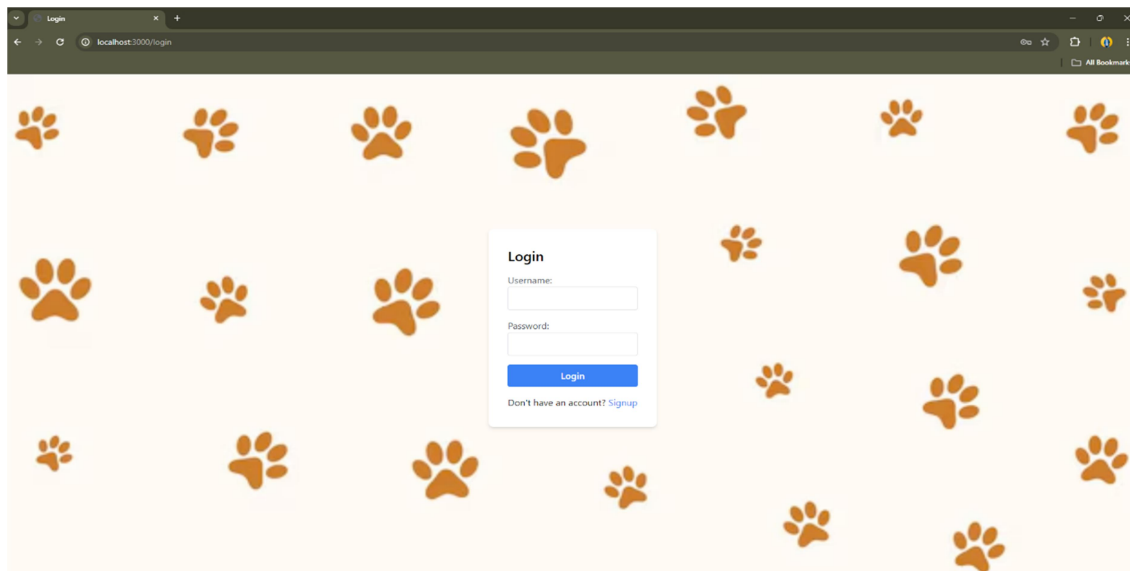
PS C:\Users\sampa\Downloads\Furry Finder-main\Furry Finder-main> node index.js
Server is running on port 3000
User data loaded.
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\sampa\Downloads\Furry Finder-main\Furry Finder-main> node index.js
Server is running on port 3000
User data loaded.
/uploads/image-1711029921193.jpg
Data saved to CSV
Name: Rocky
Category: Dog
Description: hsdgcvytsh
Status: Found
Image URL: /uploads/image-1711029921193.jpg
Error posting to Instagram: IgResponseError: POST /api/v1/accounts/login/ - 400 Bad Request; Please wait a few minutes before you try again.
    at Request.handleError (C:\Users\sampa\Downloads\Furry Finder-main\Furry Finder-main\node_modules\instagram-private-api\dist\core\request.js:54:28)
    at Request.send (C:\Users\sampa\Downloads\Furry Finder-main\Furry Finder-main\node_modules\instagram-private-api\dist\core\request.js:54:28)
Posted to Instagram successfully.
```

FIG 4.3.1.2- WEBSITE SCREENSHOTS





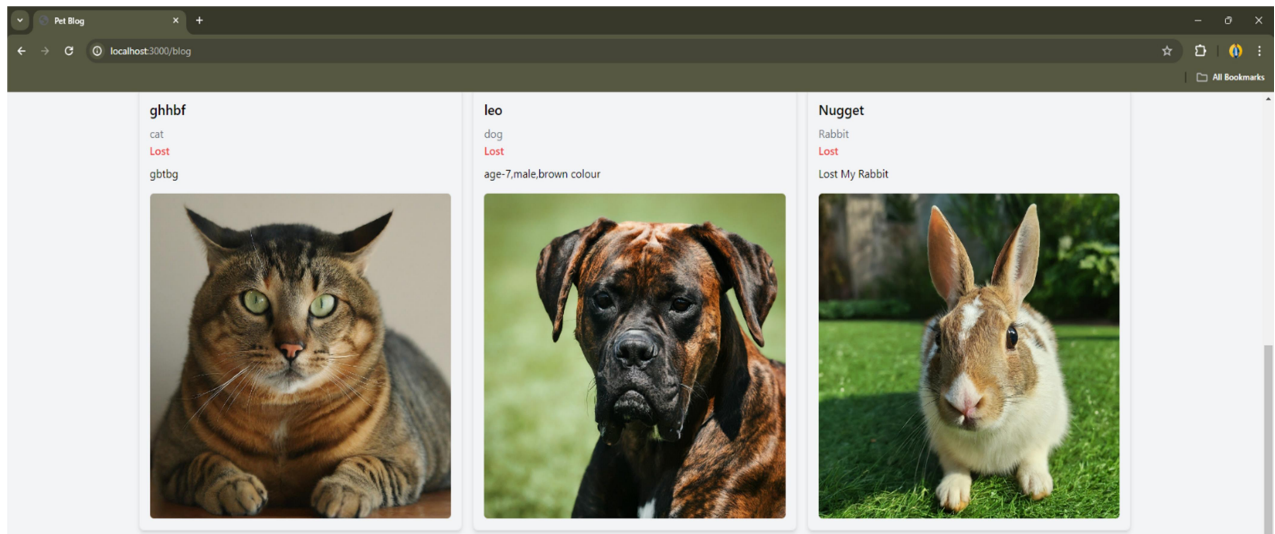
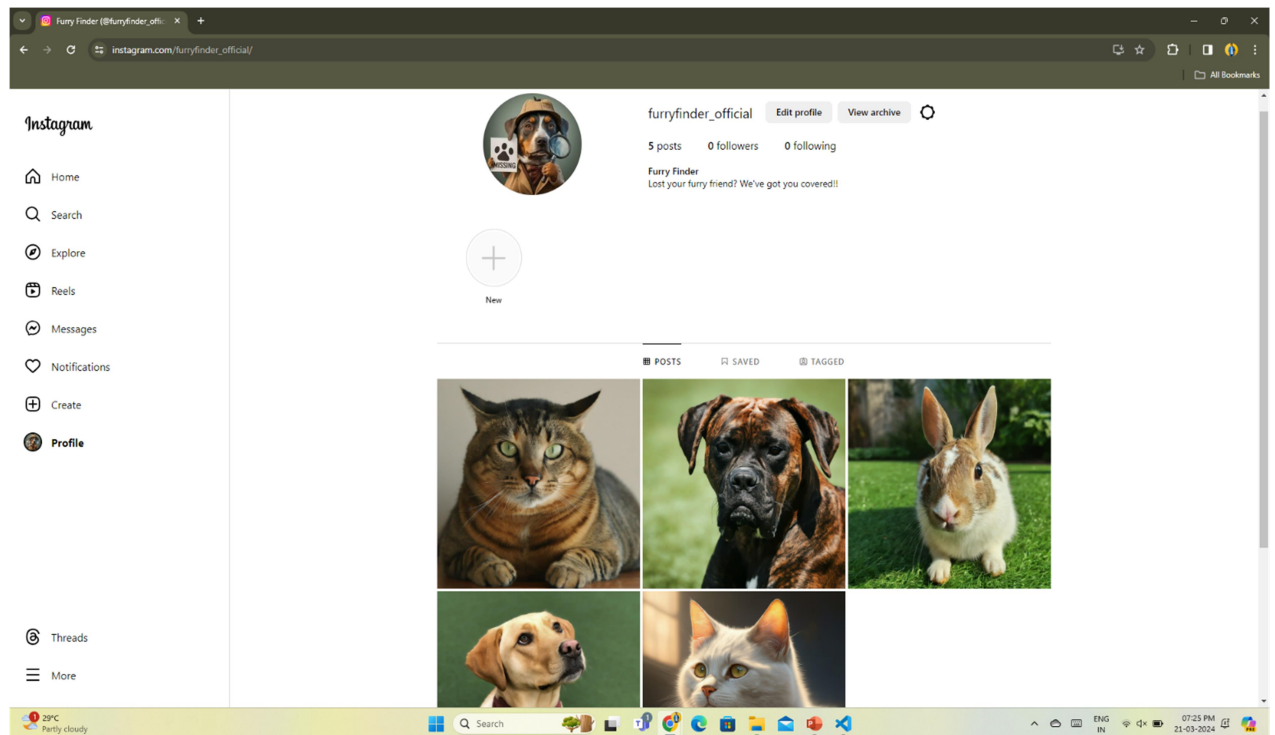
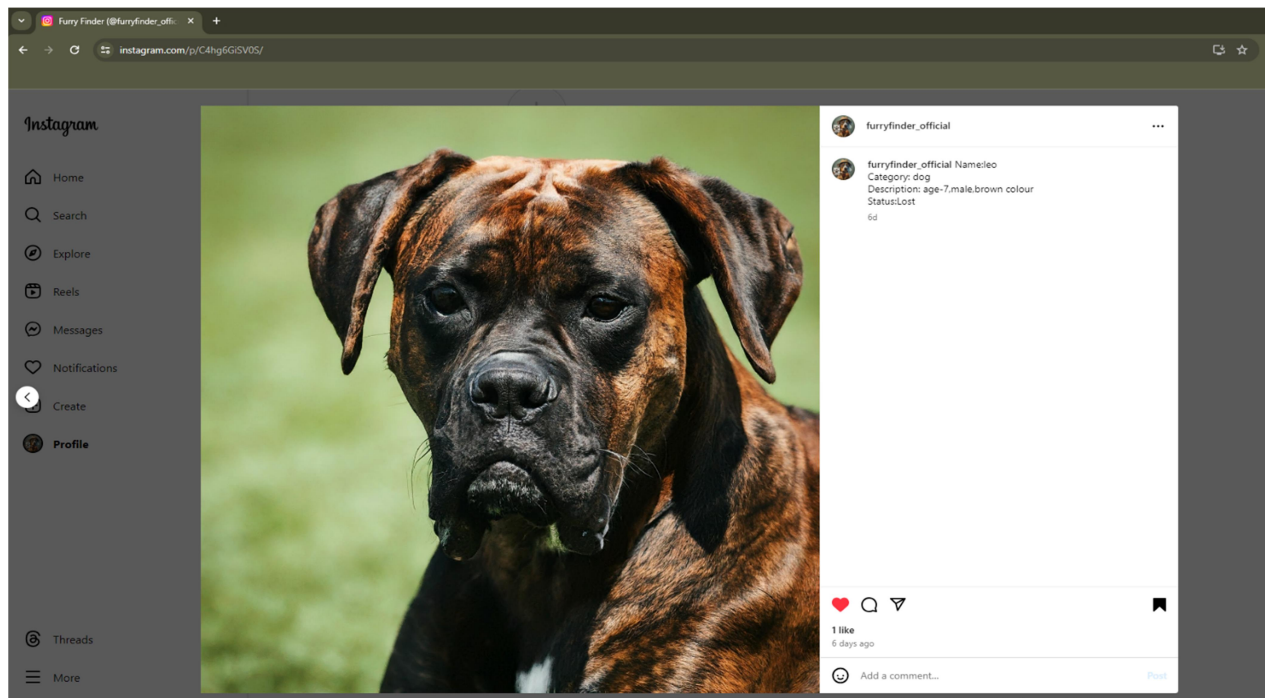


FIG 4.3.1.3- INSTAGRAM SCREENSHOTS





4.4 FEATURES

1. **User Registration:** Users can register with unique usernames and valid email addresses.
2. **User Authentication:** Secure user authentication ensures data privacy and access control.
3. **User Profiles:** Registered users have editable profiles with personal information and profile pictures.
4. **Pet Details Submission:** Users can submit comprehensive information about lost or found pets.
5. **Scheduled Social Media Sharing:** Integration with Instagram API enables automatic posting with relevant hashtags.
6. **User Engagement:** Features include commenting, notifications, and a dedicated blog or complaints section.
7. **Data Privacy:** Users can control the privacy settings for their pet submissions.
8. **Data Validation:** Input validation and error handling ensure data integrity.

- 9. Database Storage:** MongoDB stores user data, pet details, and comments efficiently.
- 10. Pet Status Updates:** Users can update a pet's status from "LOST" to "FOUND" and vice versa.
- 11. Reporting and Flagging:** Users can report inappropriate submissions for moderation.
- 12. Security Measures:** Implemented security features protect user data and system integrity.
- 13. Documentation:** User documentation and training materials facilitate easy website navigation.
- 14. Community Building:** Promotion strategies encourage user engagement and contributions. These features collectively make my pet management website a comprehensive and user-friendly platform for managing and sharing information about lost and found pets while fostering a supportive online community.

4.4.1 Novelty of the proposal

The novelty of my proposal is a critical aspect that can set my pet management website project apart from others and make it appealing to stakeholders, users, and potential investors. To ensure that my proposal stands out, consider the following elements of novelty:

1. Unique Features: Highlight any innovative features or functionalities that are not commonly found in existing pet management or social media platforms. For example, if your website offers a novel way to search for pets or leverages artificial intelligence for pet identification, emphasize these aspects.

2. User Experience: Focus on providing an exceptional user experience. Novelty can stem from a highly intuitive and user-friendly interface that makes it easier for users to accomplish their goals.

3. Automation: Emphasize the automation aspect, especially regarding the integration with the Instagram API for automatic posting. Highlight how this

automation simplifies and streamlines the process of sharing pet information on social media, saving users time and effort.

4. Community Building: Stress the importance of building a supportive and engaged community of pet owners and enthusiasts. Novelty can come from unique community features or engagement mechanisms that foster a sense of belonging.

5. Privacy and Security: If your project introduces novel privacy and security features, such as advanced user data protection or innovative ways to prevent misuse of the platform, make sure to showcase these aspects.

6. Data Management: If your website offers novel ways to manage and present data about lost and found pets, such as advanced search algorithms or data visualization tools, highlight these innovations.

7. Integration with Instagram: Discuss how your integration with Instagram goes beyond simple posting, perhaps by offering additional insights or analytics related to pet posts or using Instagram's features in a unique way.

8. Scalability: If your project has been designed with scalability in mind, meaning it can grow and adapt to meet future needs effectively, mention this as it demonstrates forward-thinking and long-term value.

9. Impact and Purpose: Describe the broader purpose and societal impact of your project. Novelty can also derive from the meaningful and positive contributions your platform aims to make to the pet owner and animal lover community.

10. Future Development: Mention your vision for the future development of the project, including plans for introducing new and innovative features. This demonstrates your commitment to ongoing novelty and improvement.

Ultimately, the novelty of your proposal should reflect the unique value that your pet management website brings to users and the broader community. It's about presenting a vision that distinguishes your project from existing solutions and offers something fresh, exciting, and valuable.

CHAPTER 5

CONCLUSION

All in all, MY pet administration site project addresses a critical step in the right direction in tackling innovation to address the requirements and worries of pet people and creature devotees. By effectively making an online stage with easy-to-understand highlights like secure client enlistment, proficient pet subtleties accommodation, and programmed virtual entertainment posting through the Instagram Programming interface, we have established the groundwork for significant asset in the pet-cherishing local area.

My undertaking not just smoothes out the most common way of sharing data about lost and tracked down pets yet additionally encourages a feeling of local area through highlights like remarks, notices, and a committed blog or protests segment. In addition, the accentuation on information protection and security guarantees that clients can entrust the stage with their touchy data.

As I push ahead, we stay focused on the continuous improvement and development of my pet administration site, with a dream to upgrade its capacities, draw in a more extensive client base, and contribute emphatically to the universe of pet proprietorship. This undertaking epitomizes my devotion to advancement, client driven plan, and social effect, and I anticipate the proceeded with development and outcome of this significant asset for pet sweethearts and proprietors.

REFERENCES

1. https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=2284&context=etd_projects
2. https://upcommons.upc.edu/bitstream/handle/2117/340404/WhereIsMyPet-Final_Thesis.pdf?sequence=3
3. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6912713/>
4. Node application installation: <https://www.geeksforgeeks.org/installation-of-node-js-on-windows/>
5. Installing Dependencies: <https://chat.openai.com/>
6. HTML, CSS, JavaScript: <https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners/>
7. Node.js: <https://www.tutorialspoint.com/nodejs/index.html>
8. EJS: <https://www.geeksforgeeks.org/use-ejs-as-template-engine-in-node-js/>
9. Mongoose npm: <https://www.npmjs.com/package/mongoose>
10. Instagram POST API: <https://www.getphyllo.com/post/how-to-use-instagram-api-to-post-photos-on-instagram>
<https://rapidapi.com/blog/how-to-navigate-and-connect-to-instagrams-api/>

My project in GitHub:

<https://github.com/sampathreddygoluguri/Furry-Finder-HCL-project.git>

B. SOURCE CODE

1. server.js:

```
1. require("dotenv").config();
2. const { lgApiClient } = require('instagram-private-api');
3. const { get } = require('request-promise');
4. const fs2 = require('fs');
5. const { readFile } = require('fs')
6. const csv = require('csv-parser');
7. const { promisify } = require('util')
8. const readFileAsync = promisify(readFile);
9. var caption1;
10. let image1;
11. var imager;
12.
13. const csvFilePath = 'petData.csv';
14.
15. const postToInsta = async () => {
16.   try {
17.     function readLastEntryFromCSV(csvFilePath, callback) {
18.       const data = [];
19.
20.       fs2.createReadStream(csvFilePath)
21.         .pipe(csv())
22.         .on('data', (row) => {
23.
24.           data.push(row);
25.         })
26.         .on('end', () => {
27.           const lastEntry = data[data.length - 1];
28.           callback(null, lastEntry);
29.         })
30.         .on('error', (error) => {
31.           callback(error, null);
32.         });
33.     }
34.
35.     readLastEntryFromCSV(csvFilePath, (error, lastEntry) => {
36.       if (error) {
37.         console.error('Error reading CSV:', error);
38.       } else {
39.         // Access the data fields (e.g., name, category, etc.) from the last entry
40.         const name = lastEntry.Name;
41.         const category = lastEntry.Type;
42.         const description = lastEntry.Description;
43.         const status = lastEntry.Status;
44.         const imageUrl = lastEntry.Image;
45.         imager=imageUrl;
46.         image1=`./public${imageUrl}`;
47.         console.log('Name:', name);
48.         console.log('Category:', category);
```

```

49.         console.log('Description:', description);
50.         console.log('Status:', status);
51.         console.log('Image URL:', imageUrl);
52.         caption1 = `
53.             Name:${name}
54.             Category: ${category}
55.             Description: ${description}
56.             Status:${status} `;
57.     }
58.
59. });
60. const ig = new IgApiClient();
61. ig.state.generateDevice(process.env.IG_USERNAME);
62. await ig.account.login(process.env.IG_USERNAME, process.env.IG_PASSWORD);
63. const imageBuffer = await get({
64.     url: 'https://images.pexels.com/photos/1643457/pexels-photo-
1643457.jpeg?auto=compress&cs=tinysrgb&w=600',
65.     encoding: null,
66. });
67.
68. await ig.publish.photo({
69.     file:await readFileSync(image1),
70.     caption: caption1,
71. });
72.
73.     console.log('Posted to Instagram successfully.');
```

```

74. } catch (error) {
75.     console.error('Error posting to Instagram:', error);
76. }
77. };
78.
79. module.exports = postToInsta;
```

2. index.js:

```

const express = require('express');
const session = require('express-session')
const multer = require('multer');
const csv = require('csv-parser');
const fs = require('fs');
const createCsvWriter = require('csv-writer').createObjectCsvWriter;
const ejs = require('ejs');
const app = express();
const port = process.env.PORT || 3000;
app.use(express.static('public'));
app.use(express.urlencoded({ extended: true }));

const storage = multer.diskStorage({
    destination: 'public/uploads',
```

```

    filename: function (req, file, cb) {
      cb(null, file.fieldname + '-' + Date.now() + '.jpg');
    },
  });

const upload = multer({ storage });

const csvFilePath = 'petData.csv';
if (!fs.existsSync(csvFilePath)) {
  const csvWriter = createCsvWriter({
    path: csvFilePath,
    header: [
      { id: 'Name', title: 'Name' },
      { id: 'Type', title: 'Type' },
      { id: 'Description', title: 'Description' },
      { id: 'Image', title: 'Image' },
      { id: 'Status', title: 'Status' },
    ],
  });

  csvWriter.writeRecords([]);
  console.log('CSV file created.');
```

```

}

const userCSVFilePath = 'users.csv';
if (!fs.existsSync(userCSVFilePath)) {
  fs.writeFileSync(userCSVFilePath, 'Username,Email,Password\n', 'utf8');
}
const userData = [];

// Load user data from the CSV file
fs.createReadStream('users.csv')
  .pipe(csv())
  .on('data', (row) => {
    userData.push(row);
  })
  .on('end', () => {
    console.log('User data loaded.');
```

```

  });

// Set up express-session for session management
app.use(session({
  secret: 'welcome', // Replace with a strong secret key
  resave: false,
  saveUninitialized: false,
}));

// Middleware to check if the user is authenticated
function requireAuth(req, res, next) {
  if (req.session.username) {
    next();
  }
}

```

```

    } else {
      res.redirect('/login');
    }
  }
}

// Routes
app.get('/', (req, res) => {
  res.render('index');
});

app.get('/login', (req, res) => {
  res.render('login', { message: " " });
});
app.post('/login', (req, res) => {
  const { username, password } = req.body;

  const user = userData.find((user) => user.Username == username && user.Password == password);

  if (true) {
    // Set the session variable to remember the logged-in user
    req.session.username = username;
    res.redirect('/form'); // Redirect to a dashboard page
  } else {
    res.redirect('/form');
  }
});

app.get('/signup', (req, res) => {
  res.render('signup', { message: " " });
});

app.post('/signup', (req, res) => {
  const { username, email, password } = req.body;

  if (userData.some((user) => user.Username === username)) {
    res.render('signup', { message: 'Username already exists. Please choose a different username.' });
  } else {
    // Add the new user to the CSV file
    userData.push({ Username: username, Email: email, Password: password });
    fs.appendFile('users.csv', `\n${username},${email},${password}`, (err) => {
      if (err) {
        res.status(500).send('Error creating user');
      } else {
        // Set the session variable to remember the logged-in user
        req.session.username = username;
        res.redirect('/form'); // Redirect to a dashboard page
      }
    });
  }
});
});

```

```

app.get('/form', requireAuth, (req, res) => {
  // Access the logged-in user's username using req.session.username
  const currentUser = req.session.username;
  res.render('form', { currentUser });
});

app.get('/form', (req, res) => {
  res.render('form');
});

// Handle form submission
app.post('/submit', upload.single('image'), async(req, res) => {
  const { name, type, description, imageUrl, lostOrFound } = req.body;
  const imagePath = req.file ? `/uploads/${req.file.filename}` : "";
  console.log(imagePath)

  const csvRow = {
    Name: name,
    Type: type,
    Description: description,
    Image: imagePath,
    Status: lostOrFound,
  };

  // Append the CSV row to the CSV file
  const csvWriter = createCsvWriter({
    path: csvFilePath,
    header: [
      { id: 'Name', title: 'Name' },
      { id: 'Type', title: 'Type' },
      { id: 'Description', title: 'Description' },
      { id: 'Image', title: 'Image' },
      { id: 'Status', title: 'Status' },
    ],
    append: true,
  });

  csvWriter.writeRecords([csvRow]).then(() => {
    console.log('Data saved to CSV');
    res.redirect('/blog');
  });

  try {
    const postToInsta = require('./server');
    postToInsta();
  } catch (error) {
    console.error('Error posting to Instagram:', error);
    res.status(500).send('Error posting to Instagram.');
  }
});

```

```

app.set('view engine', 'ejs');
app.get('/blog', (req, res) => {
  const petData = [];

  // Read data from the CSV file and push it to the petData array
  fs.createReadStream(csvFilePath)
    .pipe(csv())
    .on('data', (data) => {
      petData.push(data);
    })
    .on('end', () => {
      // Render the blog page with pet details
      res.render('blog', { petData });
    });
});

app.use((req, res, next) => {
  res.locals.currentUser = req.session.username;
  next();
});

// Start the server
app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});

```

3. index. ejs:

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">
  <title>Furry Finder</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,700,700i|Poppins:300,400,500,700" rel="stylesheet">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-T3c6Coli6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
crossorigin="anonymous">
  <style>

  body {
    background: #fff;
    color: #666666;
    font-family: "Open Sans", sans-serif;

```

```

}

a {
  color: #2dc997;
  text-decoration: none;
}

a:hover,
a:active,
a:focus {
  color: #2dca98;
  outline: none;
  text-decoration: none;
}

p {
  padding: 0;
  margin: 0 0 30px 0;
}

h1,
h2,
h3,
h4,
h5,
h6 {
  font-family: "Poppins", sans-serif;
  font-weight: 400;
  margin: 0 0 20px 0;
  padding: 0;
}

.back-to-top {
  position: fixed;
  visibility: hidden;
  opacity: 0;
  right: 15px;
  bottom: 15px;
  z-index: 996;
  background: #2dc997;
  width: 40px;
  height: 40px;
  border-radius: 4px;
  transition: all 0.4s;
}

.back-to-top i {
  font-size: 28px;
  color: #fff;
  line-height: 0;
}

```

```

.back-to-top:hover {
  background: #49d6a9;
  color: #fff;
}

.back-to-top.active {
  visibility: visible;
  opacity: 1;
}

/*-----
# Disable AOS delay on mobile
-----*/
@media screen and (max-width: 768px) {
  [data-aos-delay] {
    transition-delay: 0 !important;
  }
}

/*-----
# Header
-----*/
#header {
  height: 90px;
  transition: all 0.5s;
  z-index: 997;
  background: rgba(52, 59, 64, 0.9);
}

#header #logo h1 {
  font-size: 32px;
  margin: 0;
  line-height: 1;
  font-family: "Poppins", sans-serif;
  font-weight: 700;
  letter-spacing: 3px;
  text-transform: uppercase;
}

#header #logo h1 a,
#header #logo h1 a:hover {
  color: #fff;
}

#header #logo img {
  padding: 0;
  margin: 0;
}

#header.header-transparent {
  background: transparent;
}

```

```

}

#header.header-scrolled {
  background: rgba(52, 59, 64, 0.9);
  height: 70px;
  transition: all 0.5s;
}

.scrolled-offset {
  margin-top: 70px;
}

/*-----
# Navigation Menu
-----*/
/**
 * Desktop Navigation
 */
.navbar {
  padding: 0;
}

.navbar ul {
  margin: 0;
  padding: 0;
  display: flex;
  list-style: none;
  align-items: center;
}

.navbar li {
  position: relative;
}

.navbar>ul>li {
  white-space: nowrap;
  padding: 10px 0 10px 24px;
}

.navbar a,
.navbar a:focus {
  display: flex;
  align-items: center;
  justify-content: space-between;
  font-family: "Poppins", sans-serif;
  color: #fff;
  font-size: 14px;
  padding: 0 4px;
  white-space: nowrap;
  transition: 0.3s;
  letter-spacing: 0.4px;
  position: relative;

```

```

    text-transform: uppercase;
}

.navbar a i,
.navbar a:focus i {
    font-size: 12px;
    line-height: 0;
    margin-left: 5px;
}

.navbar>ul>li>a:before {
    content: "";
    position: absolute;
    width: 100%;
    height: 2px;
    bottom: -6px;
    left: 0;
    background-color: #2dc997;
    visibility: hidden;
    transform: scaleX(0);
    transition: all 0.3s ease-in-out 0s;
}

.navbar a:hover:before,
.navbar li:hover>a:before,
.navbar .active:before {
    visibility: visible;
    transform: scaleX(1);
}

.navbar a:hover,
.navbar .active,
.navbar .active:focus,
.navbar li:hover>a {
    color: #fff;
}

.navbar .dropdown ul {
    display: block;
    position: absolute;
    left: 24px;
    top: calc(100% + 30px);
    margin: 0;
    padding: 10px 0;
    z-index: 99;
    opacity: 0;
    visibility: hidden;
    background: #fff;
    box-shadow: 0px 0px 30px rgba(127, 137, 161, 0.25);
    transition: 0.3s;
}

```

```

.navbar .dropdown ul li {
  min-width: 200px;
}

.navbar .dropdown ul a {
  padding: 10px 20px;
  font-size: 14px;
  text-transform: none;
  color: #666666;
}

.navbar .dropdown ul a i {
  font-size: 12px;
}

.navbar .dropdown ul a:hover,
.navbar .dropdown ul .active:hover,
.navbar .dropdown ul li:hover>a {
  color: #2dc997;
}

.navbar .dropdown: hover>ul {
  opacity: 1;
  top: 100%;
  visibility: visible;
}

.navbar .dropdown .dropdown ul {
  top: 0;
  left: calc(100% - 30px);
  visibility: hidden;
}

.navbar .dropdown .dropdown: hover>ul {
  opacity: 1;
  top: 0;
  left: 100%;
  visibility: visible;
}

@media (max-width: 1366px) {
  .navbar .dropdown .dropdown ul {
    left: -90%;
  }

  .navbar .dropdown .dropdown: hover>ul {
    left: -100%;
  }
}

/**
 * Mobile Navigation

```

```

*/
.mobile-nav-toggle {
  color: #fff;
  font-size: 28px;
  cursor: pointer;
  display: none;
  line-height: 0;
  transition: 0.5s;
}

@media (max-width: 991px) {
  .mobile-nav-toggle {
    display: block;
  }

  .navbar ul {
    display: none;
  }
}

.navbar-mobile {
  position: fixed;
  overflow: hidden;
  top: 0;
  right: 0;
  left: 0;
  bottom: 0;
  background: rgba(77, 77, 77, 0.9);
  transition: 0.3s;
  z-index: 999;
}

.navbar-mobile .mobile-nav-toggle {
  position: absolute;
  top: 15px;
  right: 15px;
}

.navbar-mobile ul {
  display: block;
  position: absolute;
  top: 55px;
  right: 15px;
  bottom: 15px;
  left: 15px;
  padding: 10px 0;
  background-color: #fff;
  overflow-y: auto;
  transition: 0.3s;
}

.navbar-mobile>ul>li {

```

```

padding: 0;
}

.navbar-mobile a:hover:before,
.navbar-mobile li:hover>a:before,
.navbar-mobile .active:before {
  visibility: hidden;
}

.navbar-mobile a,
.navbar-mobile a:focus {
  padding: 10px 20px;
  font-size: 15px;
  color: #666666;
}

.navbar-mobile a:hover,
.navbar-mobile .active,
.navbar-mobile li:hover>a {
  color: #2dc997;
}

.navbar-mobile .getstarted,
.navbar-mobile .getstarted:focus {
  margin: 15px;
}

.navbar-mobile .dropdown ul {
  position: static;
  display: none;
  margin: 10px 20px;
  padding: 10px 0;
  z-index: 99;
  opacity: 1;
  visibility: visible;
  background: #fff;
  box-shadow: 0px 0px 30px rgba(127, 137, 161, 0.25);
}

.navbar-mobile .dropdown ul li {
  min-width: 200px;
}

.navbar-mobile .dropdown ul a {
  padding: 10px 20px;
}

.navbar-mobile .dropdown ul a i {
  font-size: 12px;
}

.navbar-mobile .dropdown ul a:hover,

```

```

.navbar-mobile .dropdown ul .active:hover,
.navbar-mobile .dropdown ul li:hover>a {
  color: #2dc997;
}

.navbar-mobile .dropdown>.dropdown-active {
  display: block;
}

/*-----
# Hero Section
-----*/

#hero {
  width: 100%;
  height: 100vh;
  background: url("hero-bg.jpg") ;
  background-size: cover;
  position: relative;
}

@media (min-width: 1024px) {
  #hero {
    background-attachment: fixed;
  }
}

#hero:before {
  content: "";
  background: rgba(0, 0, 0, 0.6);
  position: absolute;
  bottom: 0;
  top: 0;
  left: 0;
  right: 0;
}

#hero .hero-container {
  position: absolute;
  bottom: 0;
  top: 0;
  left: 0;
  right: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  flex-direction: column;
  text-align: center;
}

#hero h1 {
  margin: 30px 0 10px 0;
  font-size: 48px;
}

```

```

font-weight: 700;
line-height: 56px;
text-transform: uppercase;
color: #fff;
}

@media (max-width: 768px) {
  #hero h1 {
    font-size: 28px;
    line-height: 36px;
  }
}

#hero h2 {
  color: #eee;
  margin-bottom: 50px;
  font-size: 24px;
}

@media (max-width: 768px) {
  #hero h2 {
    font-size: 18px;
    line-height: 24px;
    margin-bottom: 30px;
  }
}

#hero h5 {
  color: #eee;
  margin-bottom: 20px;
  font-size: 16px;
}

@media (max-width: 768px) {
  #hero h5 {
    font-size: 14px;
    line-height: 24px;
    margin-bottom: 30px;
  }
}

#hero .btn-get-started {
  font-family: "Poppins", sans-serif;
  text-transform: uppercase;
  font-weight: 500;
  font-size: 16px;
  letter-spacing: 1px;
  display: inline-block;
  padding: 8px 28px;
  border-radius: 50px;
  transition: 0.5s;
}

```

```

margin: 10px;
border: 2px solid #fff;
color: #fff;
}

#hero .btn-get-started:hover {
background: #2dc997;
border: 2px solid #2dc997;
}

/*-----
# Sections
-----*/
section {
overflow: hidden;
}

/* Sections Header
-----*/
.section-header .section-title {
font-size: 32px;
color: #111;
text-transform: uppercase;
text-align: center;
font-weight: 700;
margin-bottom: 5px;
}

.section-header .section-description {
text-align: center;
padding-bottom: 40px;
color: #999;
}

/*-----
# Breadcrumbs
-----*/
.breadcrumbs {
padding: 20px 0;
background-color: whitesmoke;
min-height: 40px;
margin-top: 92px;
}

.breadcrumbs h2 {
font-size: 24px;
font-weight: 300;
margin: 0;
}

@media (max-width: 992px) {
.breadcrumbs h2 {

```

```

        margin: 0 0 10px 0;
    }
}

.breadcrumbs ol {
    display: flex;
    flex-wrap: wrap;
    list-style: none;
    padding: 0;
    margin: 0;
    font-size: 14px;
}

.breadcrumbs ol li+li {
    padding-left: 10px;
}

.breadcrumbs ol li+li::before {
    display: inline-block;
    padding-right: 10px;
    color: #6c757d;
    content: "/";
}

@media (max-width: 768px) {
    .breadcrumbs .d-flex {
        display: block !important;
    }

    .breadcrumbs ol {
        display: block;
    }

    .breadcrumbs ol li {
        display: inline-block;
    }
}
</style>
</head>
<body>
<header id="header" class="fixed-top d-flex align-items-center header-transparent">
<div class="container d-flex justify-content-between align-items-center">
    <div id="logo">
        <a href="index.html" style="color: #ffff;"><h1>Furry Finder</h1></a>
    </div>
    <nav id="navbar" class="navbar">
        <ul>
            <li><a class="nav-link scrollTo active" href="/login">Login</a></li>
            <li><a class="nav-link scrollTo" href="/signup">Signup</a></li>
        </ul>
    </nav>
</div>
</header>

```

```

<section id="hero">
  <div class="hero-container" >
    <h1>Welcome to Furry Finder</h1>
    <h5>"Reconnecting hearts, one paw at a time." 🧠❤️</h5>
    <h2>Here we try to show the way to home for the lost paws</h2>
    <a href="/login" class="btn-get-started">Get Started</a>
  </div>
</section>
</body>
</html>

```

4. login. ejs:

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <style>
    body {
      background: linear-gradient(90deg, #ce0aff36, #09f1cb);
    }
  </style>
</head>
<body class="bg-gray-100 flex items-center justify-center h-screen">

  <div class="max-w-md">
    <div class="bg-white p-8 rounded-lg shadow-md">
      <h1 class="text-2xl font-semibold mb-4">Login</h1>
      <form action="/login" method="POST">
        <div class="mb-4">
          <label for="username" class="block text-gray-600">Username:</label>
          <input type="text" id="username" name="username" required class="w-full p-2
border rounded">
        </div>
        <div class="mb-4">
          <label for="password" class="block text-gray-600">Password:</label>
          <input type="password" id="password" name="password" required class="w-full p-2
border rounded">
        </div>
        <button type="submit" class="bg-blue-500 text-white font-semibold py-2 px-4 rounded
hover:bg-blue-600 transition duration-300 w-full">Login</button>
      </form>
      <p class="mt-4 text-red-500"><%= message %></p>
      <p class="mt-4">Don't have an account? <a href="/signup" class="text-blue-500
hover:underline">Signup</a></p>
    </div>
  </div>
</body>

```

```
</html>
```

5. sign. ejs:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Signup</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <style>
    body {
      background: linear-gradient(90deg, #ce0aff36, #09f1cb);
    }
  </style>
</head>
<body class="bg-gray-100 flex items-center justify-center h-screen">
  <div class="max-w-md">
    <div class="bg-white p-8 rounded-lg shadow-md">
      <h1 class="text-2xl font-semibold mb-4">Signup</h1>
      <form action="/signup" method="POST">
        <div class="mb-4">
          <label for="username" class="block text-gray-600">Username:</label>
          <input type="text" id="username" name="username" required class="w-full p-2 border rounded">
        </div>
        <div class="mb-4">
          <label for="email" class="block text-gray-600">Email:</label>
          <input type="email" id="email" name="email" required class="w-full p-2 border rounded">
        </div>
        <div class="mb-4">
          <label for="password" class="block text-gray-600">Password:</label>
          <input type="password" id="password" name="password" required class="w-full p-2 border rounded">
        </div>
        <button type="submit" class="bg-green-500 text-white font-semibold py-2 px-4 rounded hover:bg-green-600 transition duration-300 w-full">Signup</button>
      </form>
      <p class="mt-4 text-red-500"><%= message %></p>
      <p class="mt-4">Already have an account? <a href="/login" class="text-blue-500 hover:underline">Login</a></p>
    </div>
  </div>
</body>
</html>
```

6.form. ejs:

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

```

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Pet Detail Collection</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <style>
    body {
      background: linear-gradient(90deg, #ce0aff36, #09f1cb);
    }
  </style>
</head>
<body class="bg-gray-100 flex items-center justify-center h-screen">
  <div class="bg-white max-w-md p-8 rounded-lg shadow-md">
    <h1 class="text-2xl font-semibold mb-4">Pet Detail Collection</h1>
    <form action="/submit" method="POST" enctype="multipart/form-data">
      <div class="mb-4">
        <label for="name" class="block text-gray-600">Pet Name:</label>
        <input type="text" id="name" name="name" required class="w-full p-2 border
rounded">
      </div>
      <div class="mb-4">
        <label for="type" class="block text-gray-600">Pet Type:</label>
        <input type="text" id="type" name="type" required class="w-full p-2 border rounded">
      </div>
      <div class="mb-4">
        <label for="description" class="block text-gray-600">Description: ( Age, Gender, Skin
color, Breed, Collar, Unique identification, Lost Location )</label>
        <textarea id="description" name="description" required class="w-full p-2 border
rounded"></textarea>
      </div>
      <div class="mb-4">
        <label for="image" class="block text-gray-600">Upload Image ( JPG Format only
):</label>
        <input type="file" id="image" name="image" accept="image/*" required class="w-full">
      </div>
      <div class="mb-4">
        <label class="block text-gray-600">Status:</label>
        <input type="radio" id="lost" name="lostOrFound" value="Lost" required>
        <label for="lost" class="mr-4">Lost</label>
        <input type="radio" id="found" name="lostOrFound" value="Found" required>
        <label for="found">Found</label>
      </div>
      <button type="submit" class="bg-blue-500 text-white font-semibold py-2 px-4 rounded
hover:bg-blue-600 transition duration-300 w-full">Submit</button>
    </form>
  </div>
</body>
</html>

```

7.blog. ejs:

```
<!DOCTYPE html>
```

```

<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Pet Blog</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <style>
  </style>
</head>
<body class="bg-gray-100">
  <div class="container mx-auto mt-8 p-4">
    <h1 class="text-2xl font-semibold mb-4">Pet Blog</h1>
    <!-- Search input field -->
    <input type="text" id="searchInput" placeholder="Search by pet name" class="w-full p-2 border rounded mb-4">
    <div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4" id="petList">
      <!-- Loop through petData and display details -->
      <% petData.reverse().forEach((pet) => { %>
        <div class="border p-4 rounded-lg shadow-md">
          <h2 class="text-xl font-semibold mb-2"><%= pet.Name %></h2>
          <p class="text-gray-500"><%= pet.Type %></p>
          <p class="text-<%= pet.Status === 'Lost' ? 'red' : 'green' %>-500 font-semibold">
            <%= pet.Status %>
          </p>
          <p class="mt-2"><%= pet.Description %></p>
          " class="mt-4 rounded-md">
        </div>
      <% }); %>
    </div>
  </div>

  <script>
    // JavaScript function to filter pet details by name
    document.getElementById('searchInput').addEventListener('input', function () {
      const searchValue = this.value.toLowerCase();
      const petList = document.getElementById('petList');
      const petItems = petList.getElementsByClassName('border');

      for (let i = 0; i < petItems.length; i++) {
        const petName = petItems[i].querySelector('.text-xl').textContent.toLowerCase();
        if (petName.includes(searchValue)) {
          petItems[i].style.display = 'block';
        } else {
          petItems[i].style.display = 'none';
        }
      }
    });
  </script>
</body>
</html>

```