

PAW CONNECT

SUBMITTED IN PARTIAL FULFILMENT REQUIREMENT FOR THE AWARD OF
DEGREE OF

Bachelor of Technology

(Computer Science & Engineering)



Submitted By:

Harleen Kaur (2104112)

Submitted To:

Dr. Kapil Sharma
Dr. Priyanka Arora
Training Co-ordinator's
CSE Department

Department of Computer Science & Engineering

Guru Nanak Dev Engineering College

Ludhiana 141006

PROVISIONAL CERTIFICATE



SCIENCE & TECHNOLOGY ENTREPRENEURS' PARK

Approved TBI Under MSME DI
(Promoted by DST, Govt. of India, PSCS & T, Govt. of Punjab & NSET)

GURU NANAK DEV ENGINEERING COLLEGE

AN AUTONOMOUS COLLEGE UNDER UGC ACT - 1956

website : www.stepgndec.com



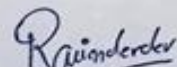
Ref. No. STEP/STD/25/124

Dated 23/05/2025

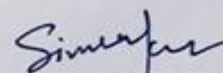
Provisional Certificate

This is to certify that **Ms. Harleen Kaur D/o S. Satinder Singh**, Roll No. **2104112** (B.Tech - CSE), a student of **Guru Nanak Dev Engineering College**, has successfully completed a **six-month Industrial Training in Full Stack** at our institute, for the session **January to June 2025**.

During the course of this training, she has worked on various software technologies in line with our curriculum. This **Provisional Certificate** for six month training **Full Stack Development with MERN** is being issued pending the issuance of the final certificate by **STEP GNDEC**.


Mr. Ravinder Kumar Dev
Training & Placement Officer




Simranjeet Kaur
Co-ordinator STEP-Affairs

Ph. : 0161-2814748, Mob. +91 78371-00954, 78371-00922, 78371-00924
E-mail : step.gne@gmail.com, step_gne@yahoo.com, Website : www.stepgndec.com

ABSTRACT

As a part of my six-month industrial training at STEP, I worked on a MERN-focused project titled "**Paw Connect**". The goal of the project was to design and develop a centralized web-based platform to simplify and digitize the management of pet adoption, selling, veterinary consultation, and pet care services. Paw Connect addresses the inefficiencies and fragmentation in traditional systems by offering a unified, user-friendly, and secure application that brings together pet owners, adopters, veterinary professionals, and animal welfare contributors. The platform aims to provide essential services through a responsive interface, reducing the need for physical visits and manual coordination.

Tailored for modern pet shop environments, the platform includes features such as listing pets for adoption or sale, scheduling vet appointments, purchasing pet-related products via Stripe, and receiving AI-powered pet care suggestions through Hugging Face integration. Users can register under different roles each with access to specific functionalities. The system also incorporates Firebase and Nodemailer for real-time updates and notifications, and Cloudinary for efficient media management. Hosting is handled through scalable services like Netlify and Render. Built using the MERN stack (MongoDB, Express.js, React.js, and Node.js), Paw Connect ensures a dynamic, scalable, and maintainable architecture. It follows modern design practices with secure authentication and routing mechanisms, enhancing reliability. AI integration further personalizes the experience with features like diet plans, remedy suggestions, and general pet care tips based on user input.

Throughout the development process, I was actively involved in both frontend and backend tasks—designing responsive React components, implementing RESTful APIs in Node.js and Express, setting up MongoDB schemas, and integrating third-party services. This hands-on experience significantly improved my skills in full-stack development, deployment, version control using Git, and working with real-time data and external APIs. Collaborating in an agile environment also strengthened my problem-solving and debugging abilities, preparing me for real-world software development challenges.

In conclusion, Paw Connect successfully digitizes and streamlines the pet service ecosystem while promoting responsible pet ownership, transparency, and informed decision-making. The project serves as a scalable and meaningful contribution to the field of pet care and animal welfare.

ACKNOWLEDGEMENT

I take this opportunity to express my heartfelt gratitude to everyone who has supported and guided me throughout the successful completion of this project, *Paw Connect*. This project would not have been possible without the valuable contributions and encouragement from many individuals.

First and foremost, I would like to extend my sincere thanks to the institution where I had the privilege of undergoing my training. The organisation provided me with an excellent learning environment, state-of-the-art facilities, and the opportunity to work on cutting-edge technologies. Their support and resources were instrumental in shaping this project.

I would like to express my sincere gratitude to Dr. Sehijpal Singh (Principal, G.N.D.E.C) and Dr. Kiran Jyoti (Head of Department, CSE, G.N.D.E.C) for granting me the opportunity to undertake this valuable six-month industrial training as part of my academic curriculum. Their support and belief in my capabilities were invaluable.

I am deeply grateful to my instructor Er. Satinderpal Singh, whose expertise, mentorship, and guidance played a pivotal role in the development of this project. Their consistent feedback and encouragement helped me overcome challenges and motivated me to strive for excellence. I am indebted to them for their patience and for sharing their immense knowledge with me.

A special note of gratitude goes to my college teachers, whose constant guidance and encouragement inspired me throughout the project. Their dedication to teaching and mentoring has been instrumental in helping me gain the skills and confidence required to successfully complete this work.

Lastly, I would like to thank my family and friends for their unwavering support and motivation during this journey. Their belief in me has been a source of strength and inspiration.

This project has been a significant milestone in my learning journey, and I am deeply appreciative of everyone who contributed to its successful completion.

Harleen Kaur (2104112)

TABLE OF CONTENTS

CONTENT	PAGE NO.
<i>Provisional Certificate</i>	<i>i</i>
<i>Abstract</i>	<i>ii</i>
<i>Acknowledgements</i>	<i>iii</i>
<i>Table of Contents</i>	<i>iv-v</i>
<i>Definitions, Acronyms and Abbreviations</i>	<i>vi</i>
<i>List of Figures</i>	<i>vii</i>
<i>List of Tables</i>	<i>viii</i>
Chapter 1: Introduction to Company	1-2
Chapter 2: Introduction to Problem	3-9
2.1 Overview	3
2.2 Existing System	3-4
2.3 User Requirement Analysis	5-7
2.4 Feasibility Study	8
2.5 Objectives of Project	9
Chapter 3: Product Design	10-26
3.1 Product Perspectives	10
3.2 Product Functions	10-11
3.3 User Characteristics	11-12
3.4 Constraints	12
3.5 DFDS/Flowchart	13-19
3.6 Database Design	19-20
3.7 Table Structure	20
3.8 ER Diagrams	21-23
3.9 Assumptions & Dependencies	23-25
3.10 Specific Requirements	25-26

Chapter 4: Development & Implementation	27-36
4.1 Introduction to Languages	27-28
4.2 Other Supporting Tools	29-30
4.3 Pseudocodes And Algorithms	30-33
4.4 Test Cases	33-36
Chapter 5: Conclusion & Future Scope	37-38
5.1 Conclusion	37
5.2 Future Scope	38
References	39
Appendix	40-46

DEFINITIONS, ACRONYMS AND ABBREVIATIONS

This section provides definitions, acronyms, and abbreviations used throughout the project report to ensure clarity and understanding for the client.

Definitions

- **Frontend:** The part of the application that interacts with users directly. It includes the graphical user interface (GUI) and ensures a smooth user experience. In this project, it is developed using React.js and styled with Tailwind CSS.
- **Backend:** The server-side component of the application responsible for handling business logic, database interactions, and API endpoints. It is developed using Node.js and Express.js.
- **Database:** A structured storage system for the application's data, such as appointment request and product payment through stripe. This project uses MongoDB, a NoSQL database.
- **API (Application Programming Interface):** A set of rules that allows the frontend to communicate with the backend, enabling data exchange and functionality execution.

Acronyms

Acronym	Full Form
MERN	MongoDB, Express.js, React.js, Node.js
API	Application Programming Interface
JWT	JSON Web Token
CSS	Cascading Style Sheets
HTTPS	Hypertext Transfer Protocol
OTP	One Time Password
NoSQL	Not Only SQL (Database type)

Abbreviations

- **CRUD:** Refers to Create, Read, Update, and Delete operations, fundamental to database management.
- **HTTPS:** Hypertext Transfer Protocol Secure, used for secure data communication.
- **JSON:** JavaScript Object Notation, a lightweight format for data exchange.

LIST OF FIGURES

Fig No.	Fig. Description	Page No.
1	0 Level DFD	13
2	1 Level DFD	13
3	2 Level DFD for Shopping Pet Products	15
4	2 Level DFD for Own/Sell Pet	16
5	2 Level DFD for Booking Appointments	17
6	User Flow for Paw Connect	18
7	Admin Flow for Paw Connect	19
8	Database Design	20
9	ER Diagram Admin managing Users	21
10	ER Diagram Users Purchasing Products	22
11	ER Diagram Sell/Adopt and Book	23
12	Landing into Website	40
13	Signup Page	40
14	Doctor Appointment Form	41
15	Successful Payment for Products	42
16	Mobile Interface	42
17	Chatbot Application for User	43
18	Admin Panel Dashboard	44
19	Appointment request accepted/rejected	44
20	Order history of Products	45
21	Managing Pet details by Admin	45
22	Managing Products Details by Admin	46

LIST OF TABLES

Table No.	Table Description	Page
1	Table of SDLC Stages	20
2	Test Cases Performed	36

Chapter 1 – Introduction to Company

The Science and Technology Entrepreneurs' Park (STEP) at Guru Nanak Dev Engineering College (GNDEC), Ludhiana, is a premier organization that fosters innovation, entrepreneurship, and skill development. Established in 1986 under the guidance of the Department of Science and Technology, Government of India, STEP GNDEC is among the pioneering technology parks in India. It has been instrumental in bridging the gap between academia and industry, creating a dynamic ecosystem for nurturing talent and promoting technological advancements. Over the years, the institution has become a catalyst for creating a strong entrepreneurial ecosystem in Punjab and beyond.

STEP GNDEC operates with a mission to empower individuals and businesses by providing world-class training, incubation, and consultancy services. With a focus on practical learning, it collaborates with various industries, government agencies, and academic institutions to deliver skill development programs tailored to the evolving needs of the market. Its commitment to excellence and innovation has made it a hub for aspiring entrepreneurs and professionals seeking to build their careers in technology-driven domains.

One of the notable features of STEP GNDEC is its dedication to entrepreneurship. It hosts incubation programs that support startups and budding entrepreneurs by providing mentorship, funding opportunities, and access to an extensive network of industry experts. This focus on entrepreneurship aligns with its vision of contributing to economic development by fostering a culture of innovation.

STEP GNDEC is known for organizing workshops, training programs, hackathons, boot camps, and startup mentoring sessions. These events are designed to equip students and entrepreneurs with the latest technological knowledge and business acumen required to thrive in competitive markets. It also provides assistance in securing funding, navigating patent processes, and creating business plans.

The park works closely with various government schemes and startup promotion programs such as Startup India, MSME schemes, and TIDE (Technology Incubation and Development of Entrepreneurs) to support early-stage ventures. This ensures that startups not only receive physical space but also access to necessary financial and advisory support. As a part of the GNDEC ecosystem, STEP benefits from the institute's rich academic heritage, experienced faculty, and strong alumni network. GNDEC itself is one of the oldest and most reputed engineering colleges in northern India, known for its academic excellence and commitment to research and innovation.

During my six-month training in Full Stack Development with MERN at STEP GNDEC, I had the opportunity to gain hands-on experience in modern technologies such as HTML, CSS, JavaScript, Bootstrap, React.js, Node.js, Express and MongoDB. The collaborative and supportive environment, along with regular guidance from mentors, allowed me to improve my technical skills and gain practical exposure to full-stack development.

STEP GNDEC also provided opportunities to interact with other startups and fellow trainees, which helped me understand real-world challenges in software development and entrepreneurship. This dedication innovation makes STEP GNDEC an ideal platform for students, professionals, and entrepreneurs seeking to build a bright and successful future.

Chapter 2 – Introduction to Problem

2.1 Project Overview

In the current digital landscape, pet lovers and pet store owners face significant challenges due to the lack of a centralized and interactive system that integrates all essential pet-related services. Most existing platforms are limited in scope, offering only standalone features such as basic e-commerce for pet products, individual appointment booking modules, or isolated adoption services. These solutions operate independently and fail to address the broader, interconnected workflows that are essential for managing a modern pet store or fulfilling a pet owner's needs. As a result, pet owners are often left navigating multiple disjointed applications just to accomplish simple tasks like booking a grooming session, purchasing food and accessories, or initiating a pet adoption. This fragmented user experience leads to confusion, wasted time, and dissatisfaction.

On the other hand, pet store owners are burdened with manually managing services, customer requests, inventory, and communication using inefficient tools or offline methods, which severely hampers productivity. The lack of integration among these critical operations results in miscommunication, lost customer engagement, missed business opportunities, and overall operational inefficiency. These gaps highlight the urgent need for a comprehensive digital platform that not only combines all key services into a single system but also simplifies user interaction, improves administrative control, and enhances the overall experience for both customers and store operators.

2.2 Existing System

Existing pet care platforms are often fragmented, offering isolated features like pet adoption, product purchasing, or appointment booking without connecting them into a unified experience. For instance, Chewy is a well-known e-commerce platform that provides a wide variety of pet products and auto-shipping options, but it does not offer features like vet appointment scheduling, pet adoption services, or interactive user engagement tools. Similarly, Petfinder focuses solely on pet adoption, allowing users to browse and adopt pets from nearby shelters, yet it lacks functionalities such as online shopping, service booking, or integrated communication channels. As a result, users are forced to juggle between multiple applications or websites to fulfill different needs, leading to a disconnected, inefficient, and frustrating user journey. These platforms commonly lack real-time notifications, personalization, centralized data management, and streamlined communication among pet owners, veterinarians, service providers, and pet store administrators. This not only decreases user engagement but also complicates daily operations for pet businesses and lowers the overall quality of care.

To bridge this gap, we have developed Paw Connect, a fully integrated, scalable, and intelligent pet care ecosystem built using the MERN Stack. Our platform enables users to register securely through two flexible methods: traditional email signup with OTP verification via Nodemailer, or Google OAuth authentication, ensuring both security and convenience. Once logged in, users gain access to a role-specific dashboard where they can explore a variety of services such as adopting pets, listing pets for sale, booking grooming or vet appointments, and shopping for pet-related products all within a single interface. Stripe has been integrated for safe and smooth payments, and Cloudinary is used to store and manage media assets like pet photos and product images, ensuring fast and optimized loading.

Users can easily book appointments with registered veterinarians or service providers, and the assigned professionals receive automated email notifications, enabling quick confirmations and improved coordination. Each form submission whether it's a product purchase, a pet adoption request, or a service booking is routed through the Admin dashboard, where admins can approve or reject records in real time.

To further differentiate Paw Connect from typical platforms, we have introduced a built-in AI-powered chatbot developed using Hugging Face's Transformer model, which provides instant answers to user queries about pet diet, health care, and general tips. This enhances the user experience by offering intelligent, interactive support directly from the dashboard. The system's mobile-friendly design ensures seamless access across all devices phones, tablets, and desktops so users can manage their pet-related needs anytime, anywhere. Admins are empowered with full control over dynamic platform content such as product listings, pet profiles, appointments, and service details, all managed via a robust, role-based dashboard.

By unifying these advanced modules into a single platform, Paw Connect eliminates the need for multiple apps, solving the biggest challenge faced by pet lovers and store owners: the lack of a centralized and cohesive system. With its modern stack, automation tools, responsive design, cloud-based media handling, and role-specific workflows, Paw Connect is set to redefine pet care management in the digital age making it more connected, efficient, and meaningful for everyone involved. From appointment bookings and product purchases to pet adoptions and medical care, everything is seamlessly integrated. This not only enhances user convenience but also empowers store owners and vets to deliver better, faster, and smarter services.

2.3 User Requirement Analysis

This chapter deals with the Software Requirement Specification for the "PawConnect – A Pet Care Platform" project. SRS elaborates the functional, non-functional, and specific requirements required for the successful implementation and operation of the system.

2.3.1 Data Requirement

Data needs include the type, format, and sources of data the system will handle and process.

- **User Data:** Includes details such as name, email, phone number, password, address, and role (Admin or User). Stored in MongoDB and used for login/signup, dashboard access, and personal interactions.
- **Product Data:** Includes product name, description, price, images, and category. Used in the shopping module for browsing, cart, and Razorpay payment.
- **Pet Sell/Adoption Data:** Includes pet type, name, breed, age, description, image, seller/adopter info, and admin status. Used in the sell/adopt module, with data pending until approved by admin.
- **Appointment Data:** Includes service type, date/time, user info, and doctor assignment. Stored for tracking bookings and assigning doctors.
- **Notification Data:** Includes email addresses and content related to actions like pet selling, adoption, and appointment updates. Sent using nodemailer for both user and admin/doctor.
- **Cart and Order Data:** Product cart items with quantity, Razorpay order ID, payment status, and user info are stored for purchase tracking and confirmation.

2.3.2 Functional Requirement

These functional requirements depict the fundamental functionalities and abilities of the system.

- **User Authentication:** OTP and Google-based signup/login using Clerk, with role-based access (Admin or User).
- **Product Shopping:** Users can browse products, add to cart, and pay securely using Razorpay integration.
- **Pet Sell and Adoption:** Users can list pets for selling or request adoption with form submission. Admin approves requests and status is updated.
- **Appointment Booking:** Users can book grooming/doctor appointments. Admin assigns a doctor and sends confirmation emails.
- **Email Notifications:** Admin and users receive email updates on events like booking confirmations, pet approvals, and successful purchases.

- **Role-based Access:** Admin has access to all pet/product/appointment requests for verification and approval. Users have access to their own records.

2.3.3 Non-Functional Requirement

These define the system attributes and operational capabilities.

- **Usability:** Interface is simple and user-friendly, aimed at both regular users and admin.
- **Scalability:** The system is scalable to handle increasing users, pets, products, and appointment bookings.
- **Maintainability:** Modular structure using the MERN stack ensures easy updates and code maintenance.
- **Availability:** System designed for high availability, ensuring access to services at all times.

2.3.4 Performance Requirement

Performance requirements ensure that the system runs efficiently under all conditions.

- **Fast Load Time:** Pages and dashboards should load within 2 seconds on standard broadband connections.
- **Quick Response:** Booking confirmation and email notifications must be processed within 3 seconds of action.
- **Razorpay Checkout:** Payment should be completed in under 10 seconds under normal conditions.
- **Form Submissions:** Pet sell/adopt/appointment forms should submit and reflect status within 1 second.

2.3.5 Dependability Requirement

The dependability requirements ensure that the system remains available and reliable at all times.

- **System Uptime:** Maintain 99.9% uptime for all frontend/backend services and database access.
- **Error Handling:** Graceful error messages and fallbacks for failed submissions or connection issues.
- **Data Persistence:** All records must be stored permanently unless deleted by admin, ensuring traceability.
- **Fallback Handling:** Admin handles fallbacks in case doctors are unavailable or items are unapproved.

2.3.6 Security Requirement

Security ensures protection of user data and system integrity.

- **Authentication:** OTP and Google-based secure authentication using Clerk to verify users.
- **Authorization:** Role-based access control to prevent unauthorized actions.
- **Data Encryption:** Passwords and sensitive information are stored using hashed/encrypted formats.
- **Secure Payments:** Razorpay integration ensures PCI-compliant and encrypted payment transactions.
- **Email Integrity:** Email notifications are verified using proper configurations to avoid spoofing.

2.3.7 Look and Feel Requirement

The look of the system impacts user experience and ease of interaction.

- **Responsive Design:** UI is fully responsive across devices (desktop, mobile, tablet) using React.js and Tailwind CSS.
- **Minimal Layout:** Clean, elegant, and simple interface for quick understanding and usability.
- **Dashboard Views:** Separate dashboards for admin and users, showing only relevant data/actions.
- **Feedback Mechanisms:** Confirmation messages, error alerts, and status updates are shown for all actions performed.

2.4 Feasibility Study

This section discusses the feasibility of developing and deploying the "Paw Connect" web application—a pet care platform built using the MERN stack that allows users to adopt or sell pets, book appointments with doctors, shop for pet products, and perform role-based actions as Admin or User.

2.4.1 Technical Feasibility

The Paw Connect project is technically feasible as it is built using the MERN stack (MongoDB, Express.js, React.js, and Node.js), which is a reliable and widely used technology stack for modern web development. These technologies are open-source and offer a high level of community support and scalability, making them suitable for building responsive and dynamic web applications. The project also integrates Clerk for OTP and Google-based signup/login, Razorpay for secure and smooth payment processing, and Nodemailer for sending email notifications—each of which has detailed documentation and easy-to-use APIs. With a well-structured codebase and a modular approach, all planned functionalities like role-based access, real-time updates, appointment booking, product shopping, and pet adoption workflows can be implemented effectively. The availability of cloud services like MongoDB Atlas, Render, and Vercel ensures that the application can be deployed and scaled with minimal infrastructure issues.

2.4.2 Operational Feasibility

Paw Connect is operationally feasible as it is designed to solve real-life challenges faced by pet owners, sellers, and veterinary professionals. The platform allows users to sign up as either Admin or User, ensuring that features are tailored to their roles. Users can easily adopt or sell pets, book grooming or doctor appointments, and shop for pet-related products, while admins can approve or manage submissions, assign doctors, and maintain the overall system. The automated email notification system keeps both users and doctors updated on bookings and approvals, ensuring smooth coordination. The user interface is simple, mobile-friendly, and accessible, which supports effective user adoption. The functionalities are aligned with the needs of a growing pet care community, making the platform practical and beneficial for its intended users.

2.4.3 Economic Feasibility

The Paw Connect project is economically feasible as it relies primarily on open-source technologies like MongoDB, Express.js, React.js, and Node.js, which significantly reduce the initial development cost. The project also utilizes free or affordable services such as Vercel for frontend hosting, Render for backend deployment, and MongoDB Atlas for cloud-based database management, ensuring that hosting and infrastructure costs remain minimal for initial launch and scaling. Integration with Clerk for authentication and Razorpay for payments is either free or pay-per-use, meaning costs are incurred only as the platform gains users or transactions. Moreover, the project's features, such as pet selling, product shopping, and appointment booking, open multiple opportunities for future revenue generation through premium services like featured pet listings, promoted products, and priority appointments. These potential income streams make the investment in developing and maintaining Paw Connect economically justifiable in both short-term development and long-term sustainability.

2.5 Objectives

The goals outlined for the development of **Paw Connect** aim to address the key challenges in managing pet care services by combining security, usability, and performance into one unified platform. These objectives are thoughtfully designed to ensure a seamless user experience without compromising on safety or efficiency. The primary objectives of the project are:

- a) To enable easy access to pet adoption/selling, vet appointments and professional interaction
- b) To Integrate Stripe for secure and seamless pet product transactions.
- c) To provide AI-based pet care guidance including diet tips and home remedies.

Chapter 3 – Product Design

The design of Paw Connect focuses on delivering an interactive, user-friendly, and secure platform for pet lovers and pet shop owners. The system supports two roles: User and Admin (Pet Shop Owner). The frontend is built using React.js to provide a dynamic, responsive, and mobile-friendly interface. The backend is developed using Node.js and Express.js to handle user data, authentication, and request processing, while MongoDB is used for storing user, pet, and product-related information. Security is ensured through email-based OTP verification and Google OAuth login, while online purchases are handled securely via Stripe integration. An AI-powered chatbot, developed using the Hugging Face Transformer model, is embedded to help users with pet diet advice, quick home remedies, and general queries. Communication between users and admin is managed through integrated email services using Nodemailer, which is used for sending confirmations, appointment details, and responses to user actions.

3.1 Product Perspective

Paw Connect is a standalone web-based application designed to connect pet owners and pet service providers through a single platform. It is not part of any larger system but is built to be modular and scalable, allowing for future expansion or integration. The platform serves as a bridge between pet lovers and pet shop owners, streamlining communication and services. It integrates Google Authentication for social login, Gmail-based OTP verification for secure user registration, and Stripe for safe online product purchases. Email services are used to send notifications, confirmations, and updates regarding user requests. Additionally, a chatbot powered by the Hugging Face API is available 24/7 to assist users with pet-related queries. Each core feature, such as pet adoption, selling, appointments, and services, is developed as a separate module to ensure easy maintenance and scalability.

3.2 Product Functions

The major functionalities offered by Paw Connect are categorized based on the roles of Users and Admin. Each role is provided with specific access and features tailored to their needs. This role-based categorization ensures smooth functionality, enhanced user experience, and secure management of the platform.

3.2.1 For Users:

- **Sign Up / Login:** Users can register using their email through OTP verification or directly via Google login. This ensures a flexible and secure authentication process for all types of users.
- **Dashboard Access:** After successful login, users are redirected to their personalized dashboard. The dashboard serves as a central hub for all user activities and actions.

- **Adopt a Pet:** Users can fill out an adoption request form with their details and submit it to the admin. Once reviewed, the admin sends further instructions and information via email.
- **Sell a Pet:** Users can offer pets for sale by submitting a form containing personal and pet information. The admin reviews the request and responds through email for further discussion.
- **Book a Service / Appointment:** Users can select from available services and book appointments online. The admin sends confirmation emails to both the user and the respective service provider, detailing timing and date.
- **Purchase Products:** Users can browse and purchase pet-related products using secure online payments through Stripe. A confirmation email and digital receipt are sent upon successful transaction.
- **Use Chatbot:** An integrated chatbot helps users with quick pet-related queries, including diet tips and home remedies. It uses Hugging Face's Transformer model to deliver relevant, real-time responses instantly.

3.2.2 For Admin:

- **User Management:** Admin can view all incoming requests related to pet adoption, selling, and appointments. This helps in organizing and tracking user activities efficiently.
- **Approve Requests:** Admin can accept or reject user requests based on their assessment. The action taken is communicated to the user via email notifications.
- **Email Communication:** Admin sends emails to users and service providers with important updates and instructions. This ensures smooth coordination and professional communication.
- **Product & Service Management:** Admin can manage listings by adding, editing, or deleting products and services. This keeps the store offerings updated and relevant to user needs.

These functionalities empower the admin to maintain smooth operations, ensure timely communication, and provide a well-managed experience for all users. Overall, they contribute to the efficient and organized functioning of the Paw Connect platform.

3.3 User Characteristics

The users of Paw Connect are primarily categorized into two main roles: General Users (Pet Owners / Enthusiasts) and Admins (Pet Shop Owners). Each user type has distinct characteristics, needs, and levels of technical understanding.

- **General Users:** These users include individuals or families who are interested in adopting, selling, or caring for pets. Most of them have basic computer literacy and are familiar with using websites and mobile applications. They expect a smooth and intuitive user interface for browsing pets, booking services, and shopping for pet products. Security and privacy are essential for them, especially during sign-up and payment processes. They also appreciate features like chatbot assistance, email notifications, and easy access to pet health resources.
- **Admins:** Admins are responsible for managing pet-related services and requests. They typically have moderate technical knowledge and are comfortable using web dashboards to manage listings, review forms, and communicate with users. Their primary concern is maintaining accurate information, handling user requests efficiently, and ensuring timely communication through emails. They also require tools to manage appointments, track inventory, and oversee user interactions seamlessly.

Overall, Paw Connect is designed to serve users with varying levels of technical skills while providing both groups with a user-friendly and responsive experience tailored to their specific needs.

3.4 Constraints

The development and deployment of Paw Connect are subject to the following constraints:

- **Internet Dependency:** As a web-based application, Paw Connect requires a stable internet connection for users to access features such as authentication, chatbot assistance, email communication, and online payments.
- **Third-party API Limitations:** Features like Google OAuth, Stripe payments, and the Hugging Face chatbot rely on third-party APIs. Any changes, rate limits, or downtimes in these services can temporarily affect functionality.
- **Browser Compatibility:** Although designed to be responsive and cross-browser compatible, certain UI components may not perform uniformly across outdated or less common web browsers.
- **Scalability:** The current system is built for small to medium traffic. High user volumes may affect performance unless hosting and backend services are scaled accordingly.
- **Limited Admin Panel Automation:** Some admin actions, such as reviewing forms and sending emails, are manually handled, which can lead to delays and increased workload.

These constraints are being considered during development, and efforts are made to minimize their impact through regular updates, testing, and improvements.

3.5 Use Case Model/Flow Chart/DFDS

3.5.1 Overview

This project, “Paw Connect,” is a comprehensive full-stack web application that connects pet sellers, adopters, veterinary service providers, and pet product sellers under a single digital platform. It streamlines various pet-related services, offering an efficient and user-friendly interface for pet selling, pet adoption (with admin approval), vet appointment booking, product shopping, and real-time communication. Developed using Node.js, Express, MongoDB, and React, the system prioritizes modularity, scalability, and secure access, ensuring smooth operation across different roles (Admin, User etc.).

3.5.2 DFD’S:

The basic layout describing the workflow of the project is illustrated in Figure 1. This system involves two primary roles: User and Admin. The system enables users to interact with the Paw Connect platform either through a manual signup/login or via Google Sign-In. Users can register on the platform by providing their details. Upon registration, an OTP is sent to the user's Gmail for email verification. Once the OTP is verified, the user is authenticated and can log in using their credentials. Alternatively, users can bypass the manual registration process and directly sign up through Google, which is handled via Google Cloud Platform APIs and services, ensuring secure and seamless access. In case of invalid login attempts or incorrect OTPs, the system returns an error, prompting the user to reattempt. Admins interact with the system by filling their login credentials into the Paw Connect platform.

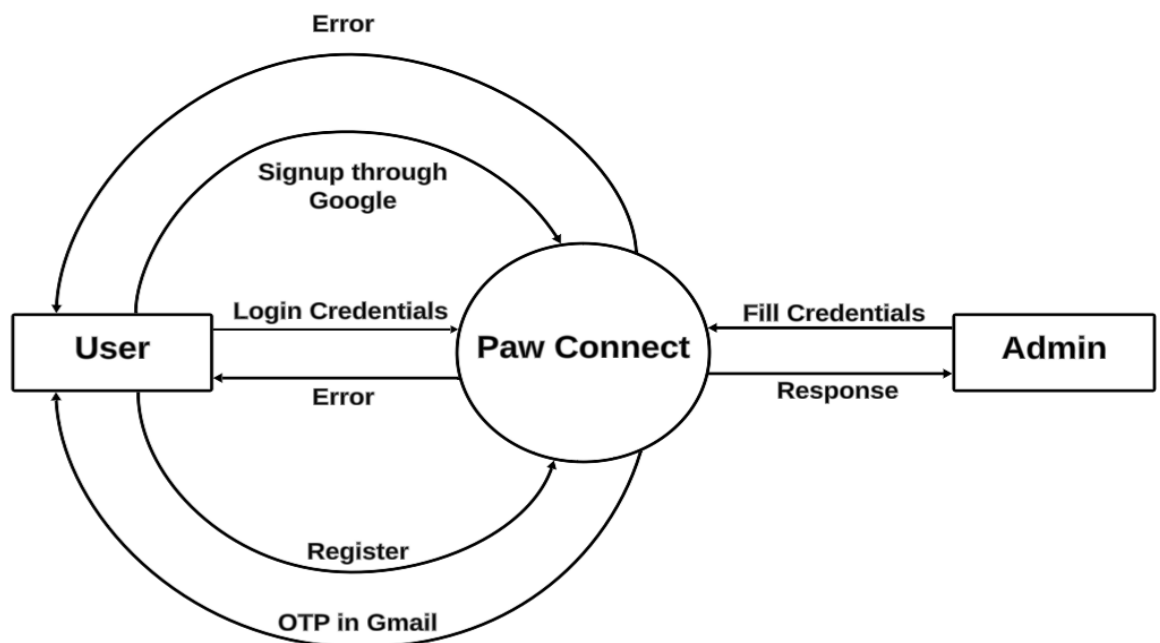


Figure 1: 0 Level DFD

Figure 2 describes how, in this system, the Paw Connect platform manages pet ownership, selling, and appointment features. After logging in or signing up (as shown in Fig 4.1), users can sell or own pets by submitting forms. These requests are then approved or rejected by the Admin. Users can also book appointments, which are likewise subject to Admin approval. The Admin through the Admin Panel, can manage user requests, approve appointments, and add new products to the platform, maintaining smooth system operations.

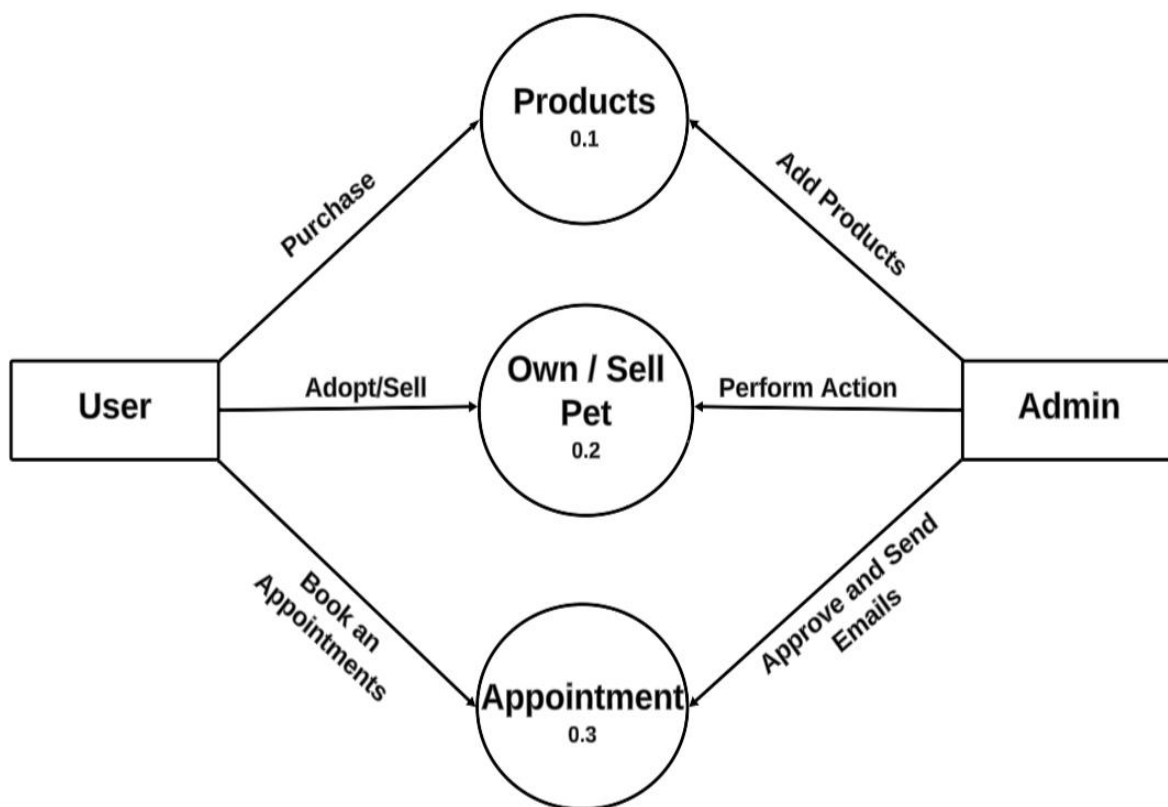


Figure 2: 1 Level DFD

As illustrated in Figure 3, the Data Flow Diagram (DFD) represents the shopping functionality for pet-related products within the Paw Connect system. The process begins with the user entering their details and completing the user registration. After successful registration, the user navigates to the product section, where they can add desired items to the cart. Once the selection is complete, the user proceeds with order processing and initiates payment, which is securely handled through Stripe integration. Upon successful payment, a purchase confirmation message is displayed to the user, and the order details are recorded in the Admin Panel.

The Admin can use this panel to add new products to the system and review the order history of users, ensuring smooth management and tracking of transactions. Additionally, an AI-based chatbot, developed using a Hugging Face transformer model, is integrated into the system. This chatbot is trained to assist users by answering queries related to pet diet plans and quick home remedies, enhancing user support and interaction.

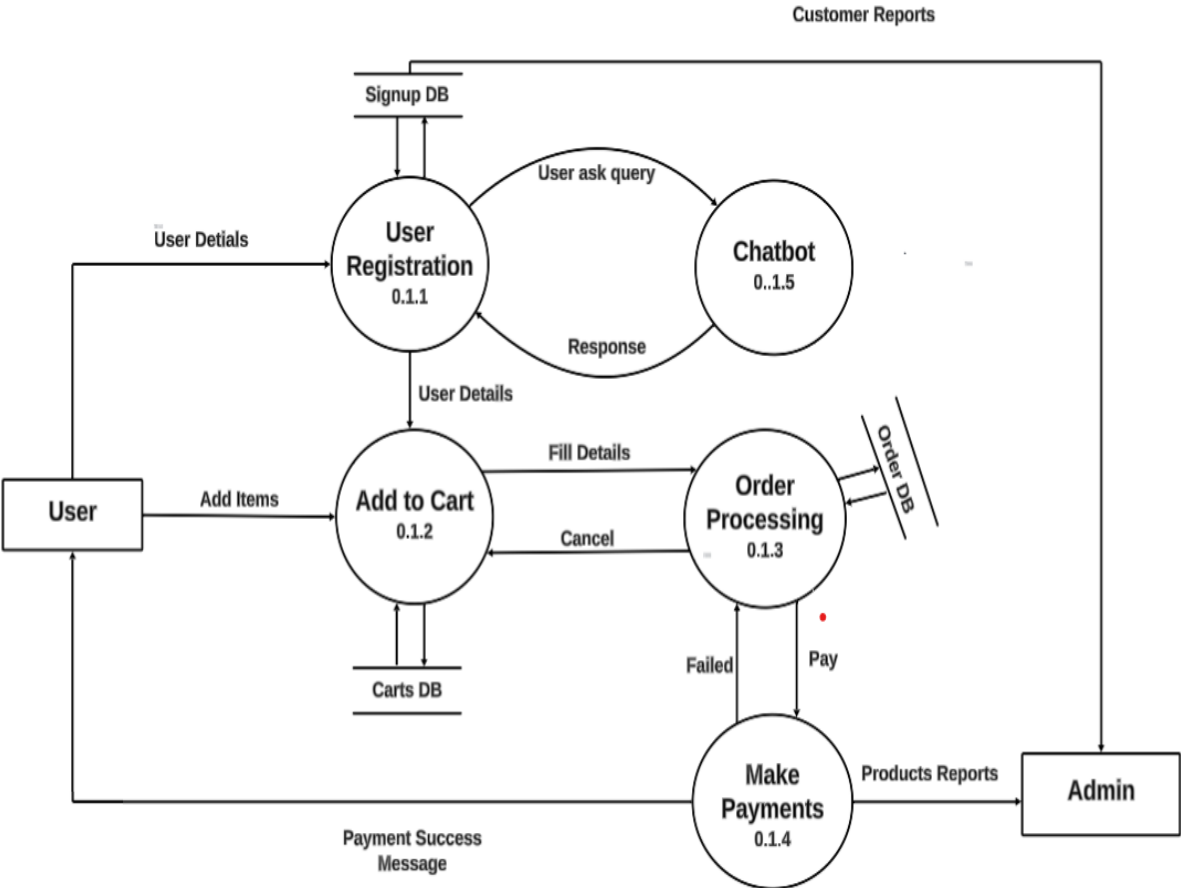


Figure 3: 2 Level DFD for Shopping Pet-Related Products

The Figure 4 is primarily explaining the flow of interactions involving user registration, chatbot support, and pet management within the Paw Connect platform. Initially, the user fills in their details and completes the user registration process. During or after registration, the user has the option to interact with an AI-based chatbot integrated into the system. This chatbot, developed using a Hugging Face transformer model, is specifically trained to respond to user queries related to pet diet plans and quick home remedies, offering intelligent and instant assistance to enhance user engagement and support. Following registration, the user can proceed to fill out the 'Own Pet' form, where they submit relevant information regarding a pet they wish to adopt or claim. The Admin then verifies the submitted details and takes appropriate action. Similarly, the user can fill out the 'Sell Pet' form to offer a pet for sale.

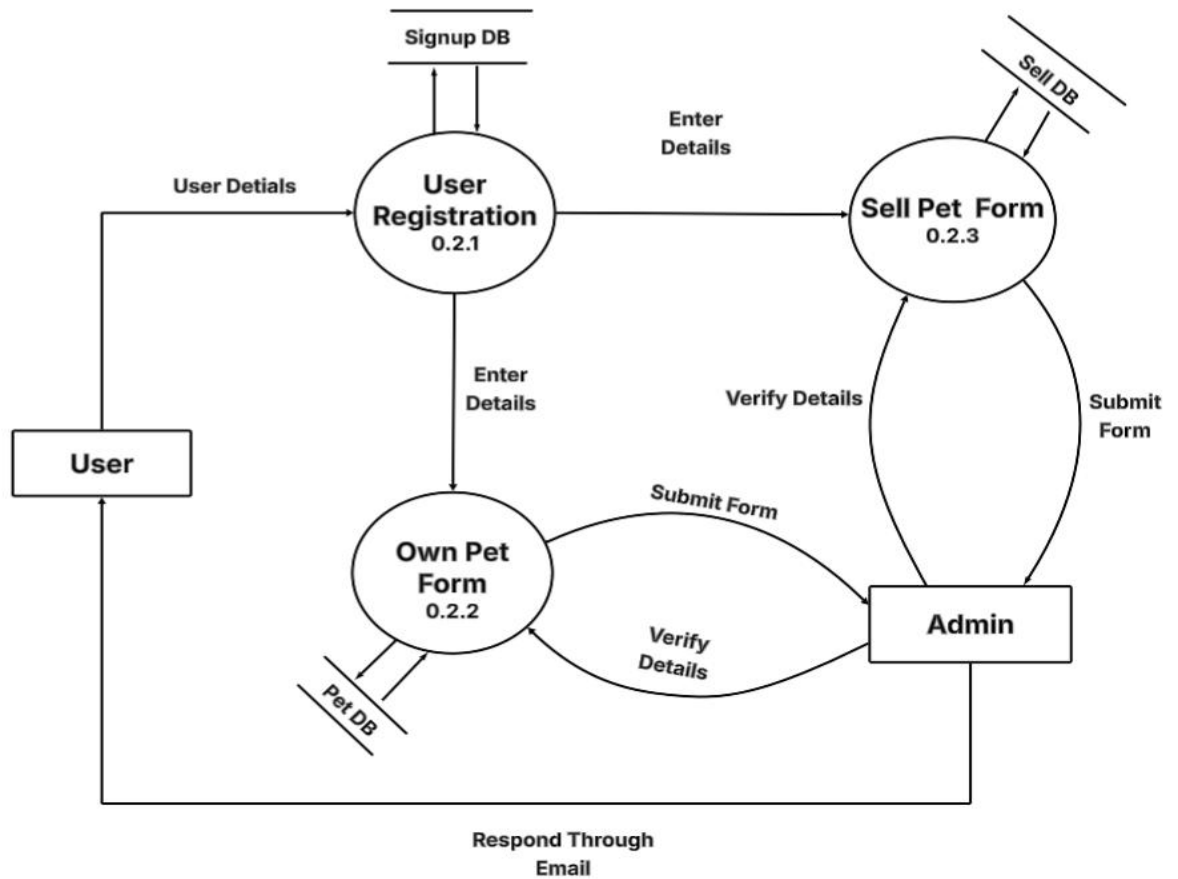


Figure 4: 2 Level DFD for Own/Sell Pet and Chatbot

Figure 5 shows the appointment scheduling process within the Paw Connect platform. The process begins when the user enters their details and completes the user registration. After registering, the user can proceed to fill out the appointment form, providing necessary information for scheduling a pet-related appointment. Once submitted, the appointment form is forwarded to the Admin, who then contacts the appointer to verify the provided details. Upon successful verification, the Admin schedules the appointment and sends confirmation emails to both the appointer and the user, ensuring both parties are informed and updated. Additionally, an AI-based chatbot, developed using a Hugging Face transformer model, is integrated into the system. This chatbot is trained to assist users with queries related to pet diet plans and quick home remedies, thereby enhancing the overall user support and interaction experience within the platform.

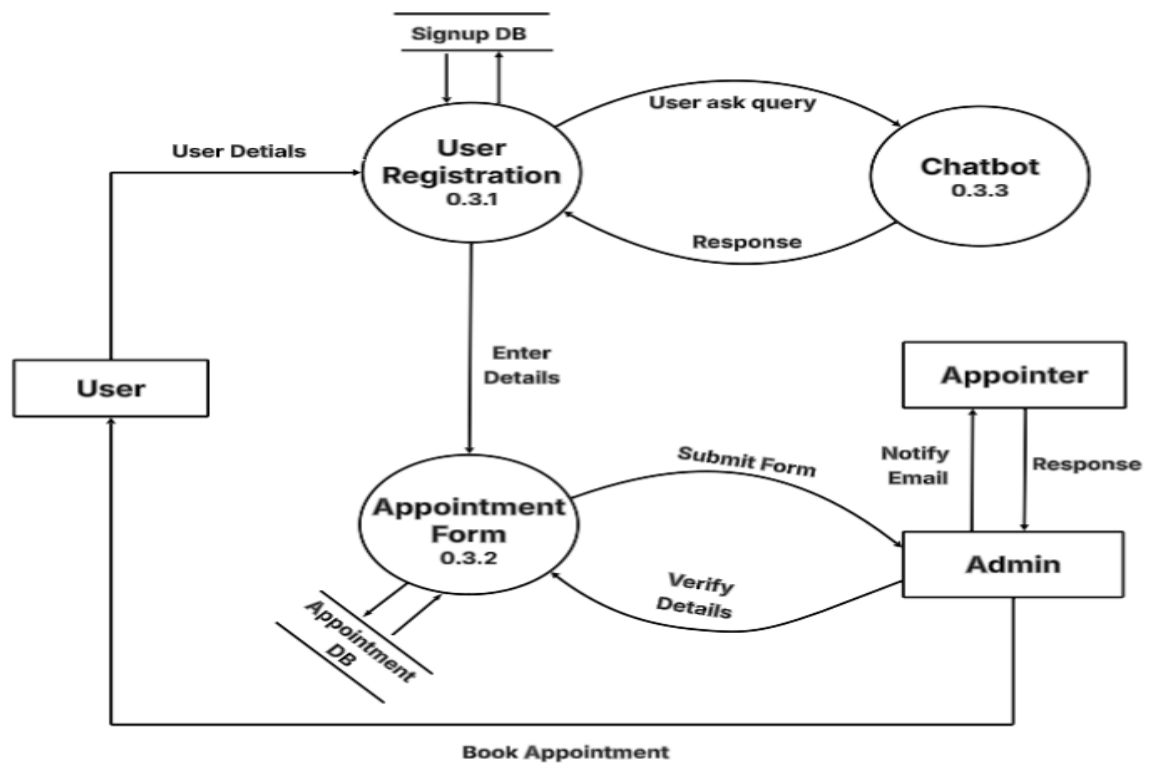


Figure 5: 2 Level DFD for Booking Appointments

3.5.3 Flowchart

The Figure 6 represents the flow of Users interacting with the Paw Connect platform. It begins with the user accessing the platform and checking if they are already signed up. If not, they can choose between email-based OTP verification or Google Signup. If already registered, users proceed with email and password login. After successful login or signup, the user is directed to a dashboard with navbar options: Home, About, Services, Products, Sell Pet, Adopt Pet, Appointment, and Chatbot.

In the Products section, users can add items to the cart and proceed to checkout, followed by a payment success message. For Sell Pet, Adopt Pet, and Appointment options, users fill out respective forms that are sent to the admin. The admin then approves or rejects the requests. Upon approval, emails are sent to users accordingly either confirming a pet sale/adoption or providing appointment details. In the Chatbot section, users can ask diet-related queries, and the chatbot responds with relevant information.

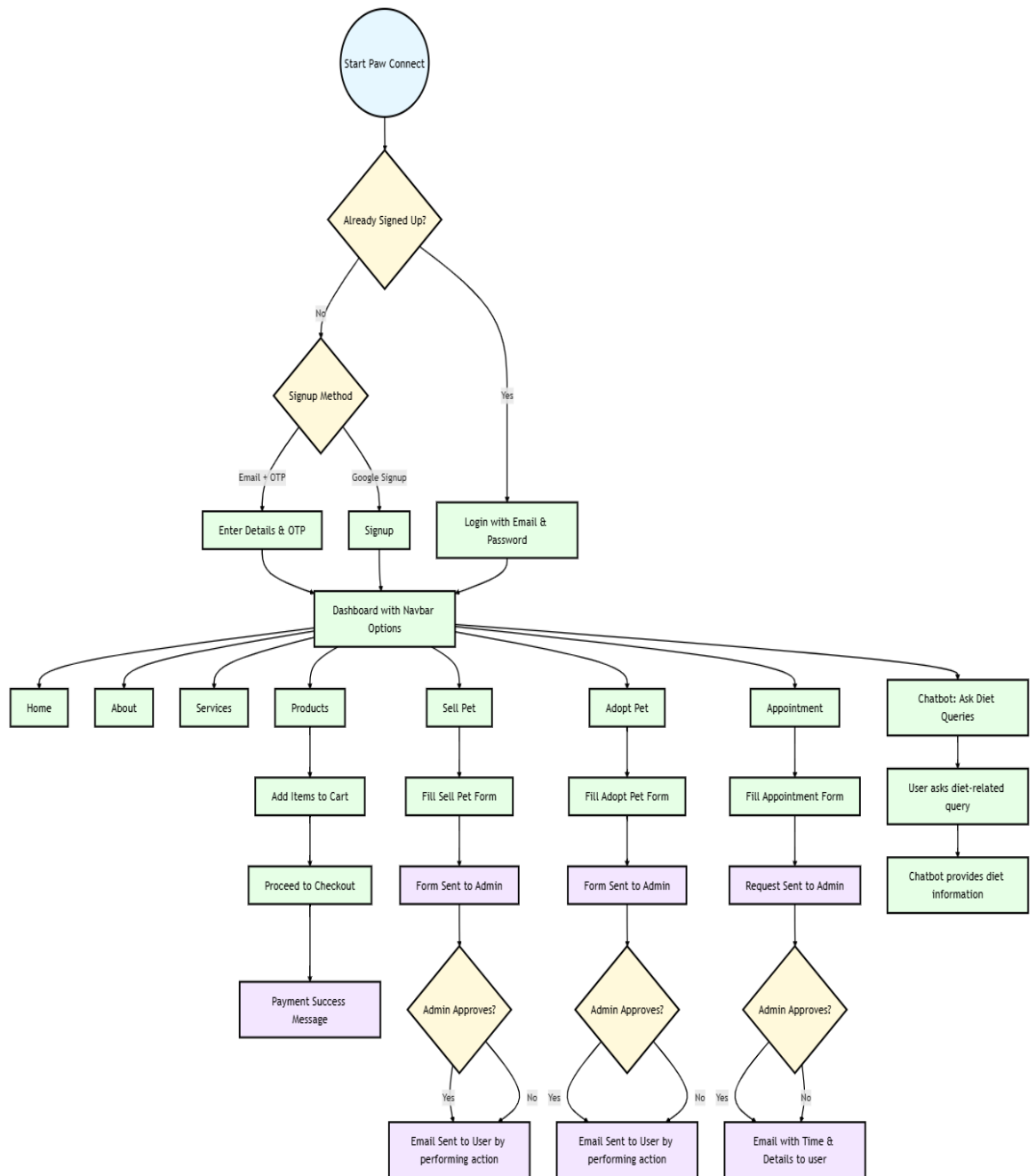


Figure 6: User Flow for Paw Connect

Figure 7 shows the admin-side flow of the Paw Connect system. The process starts with the admin logging in. If the credentials are correct, the admin accesses the dashboard; otherwise, they are prompted to retry. From the dashboard, the admin can add products, check order history, review sell and adopt pet applications, manage appointments, and update their profile. Each review action triggers an email response to the user with the relevant details.

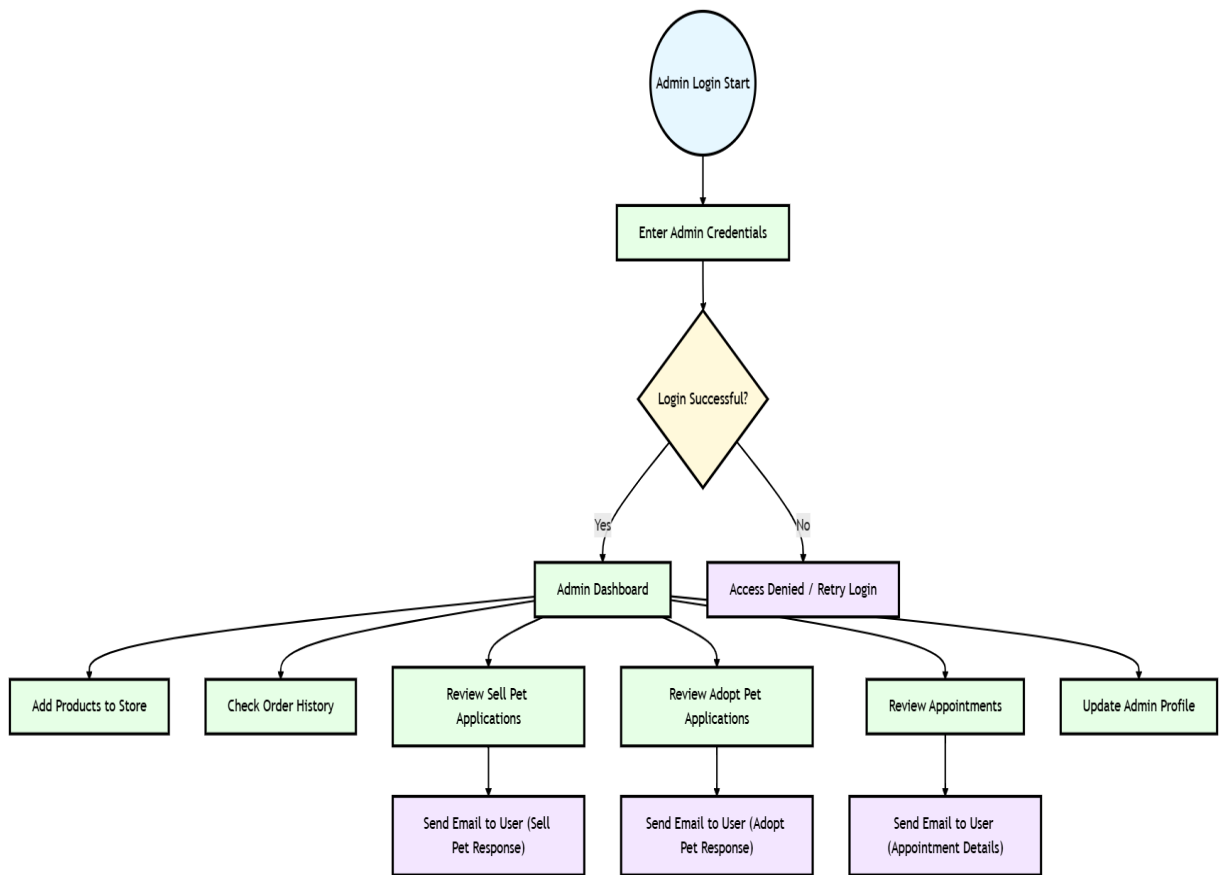


Figure 7: Admin Flow for Paw Connect

3.6 Database Design

The database design of Paw Connect is structured to efficiently manage users, pets, products, services, and transactions, ensuring smooth data flow across the platform. MongoDB is used as the database because of its flexibility in handling complex and dynamic data structures, such as nested documents and arrays. Its scalability and high performance make it ideal for supporting the growing needs of the application while maintaining fast and reliable access to data.

Figure 8 describes the database design of Paw Connect, showcasing how different collections in the MongoDB database are interconnected to manage the platform's core functionalities. The **USERS** collection holds the primary user information such as name, email, Google ID, role (admin or user), address, and login timestamp. It acts as a central reference point for several other collections. The **ORDERS** collection is linked to **USERS** via userEmail, storing data about orders placed, items purchased, payment status, total amount, and timestamps. The **PETS** collection stores pet information including name, description, category, age, and images, also with timestamps. The **SELLING_REQUESTS** collection manages pet selling entries created by users, including pet details, images, price, and the seller's reference ID.

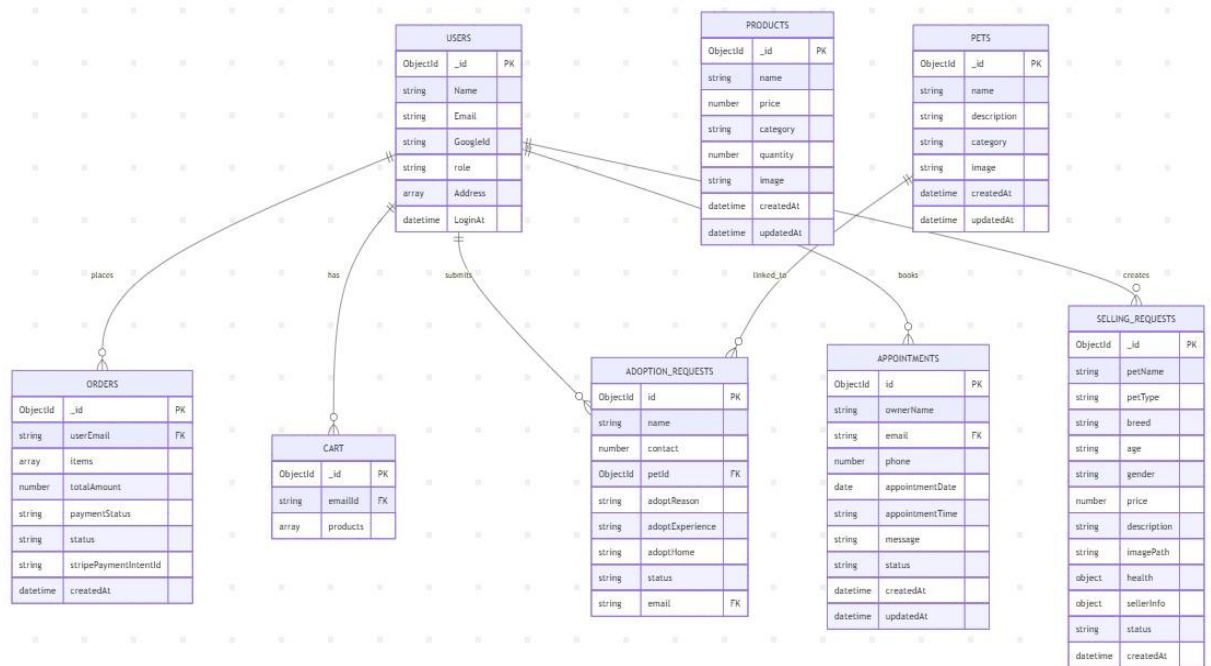


Figure 8: Database Design

3.7 Table Structure

Table 1 describes the Software Development Life Cycle (SDLC) phases followed in the development of the Paw Connect project. It is structured into six stages, each representing critical activities that contribute to building a fully functional, efficient, and user-friendly web application.

Stage1 Planning And Requirement	Stage2 Defining Requirements	Stage3 Design	Stage4 Development	Stage5 Testing	Stage6 Deployment And Maintenance
Define	Defining	Design	Development	System Testing	Release Planning
Project Scope	Functional Requirements	Lower Level Design	Coding Standards	Manual Testing	Development Automated
Set Objective	Technical Requirements	Higher Level Design	Version Control	Automated Testing	Maintenance
Resource Planning	Requirements Review	Design Review	Code Review		Feedback

Table 1: Table of SDLC Stages

3.8 ER Diagrams

As illustrated in Figure 9, the ER Diagram represents the one-to-many relationship between Admin and User entities, where a single Admin can manage multiple Users. Each Admin is identified by an Admin_id and has attributes like Email, Password, and a detailed Address (Line1, Line2, City, State, PIN). Similarly, each User is identified by a User_id and includes Name (Firstname, Lastname), Address, and Email. This structure reflects the Admin's role in overseeing and managing user-related operations within the system.

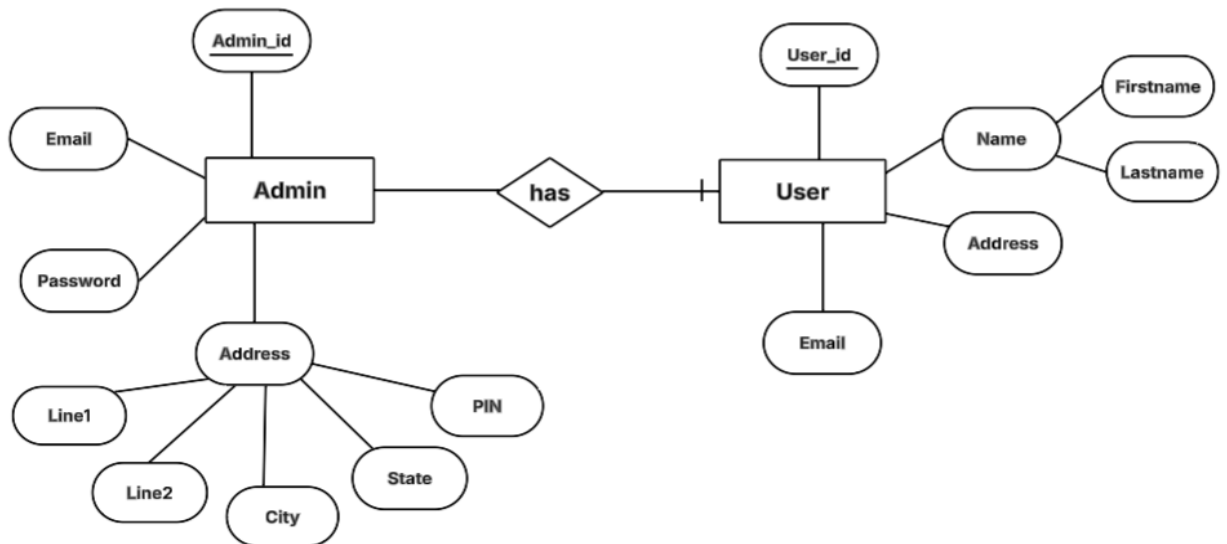


Figure 9: ER diagram showing Admin managing multiple Users

The Figure 10 depicts how Users interact with Products and the Paw Connect shop entity within the system. Each User, identified by a unique User_id, includes attributes like Name (Firstname and Lastname), Email, and Address. Users can purchase Products, where each Product has a unique Products_id along with attributes such as Name, Price, Category, Quantity, and Image. The relationship "purchase" connects Users and Products. The "Paw Connect" entity represents a shop or store, identified by Shopname and containing Address and Description. It is associated with both Users and Products, indicating that a shop manages the products and is accessible to users. This structure supports product management and user interaction within the platform.

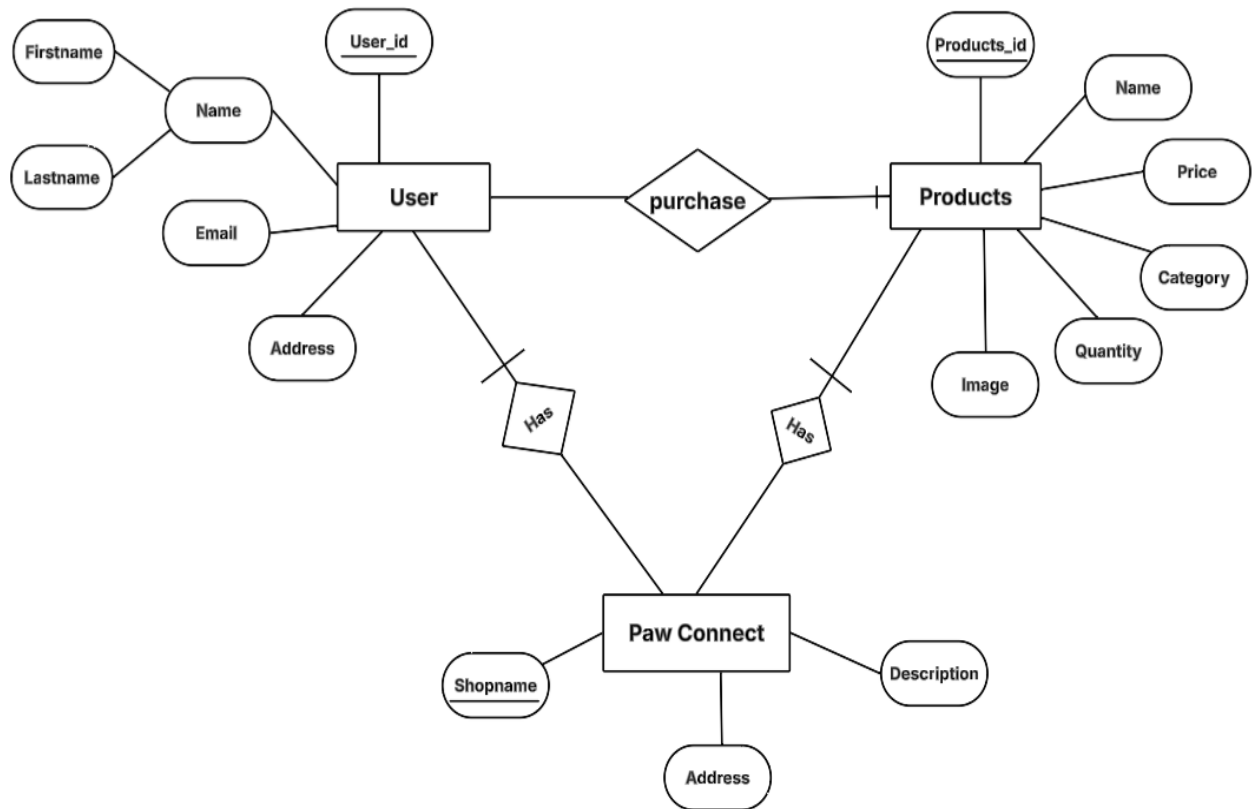


Figure 10: ER diagram showing Users purchasing Products through Paw Connect

Figure 11 illustrates how Users interact with Pets, Appointments, and the Paw Connect shop entity within the system. Each User contains attributes such as Name, Email, and Address. Users can participate in the Sell/Adopt relationship with Pets, where each Pet includes attributes like Name, Breed, Age, Image, and Description. The central Paw Connect entity represents the pet care platform or shop, identified by Shopname and containing details such as Address and Description. It is associated with Users, Pets, and Appointments through the Has relationship, indicating that the shop manages pet data, user records, and appointment services. The Appointment entity allows users to schedule services such as pet care or grooming. Each appointment includes attributes like Username, Email, Phone, and Status. It is linked to Paw Connect, showing that all appointment services are managed by the platform.

This structure supports a comprehensive pet care system, enabling user interaction, pet management, adoption/selling activities, and appointment scheduling through the unified Paw Connect platform.

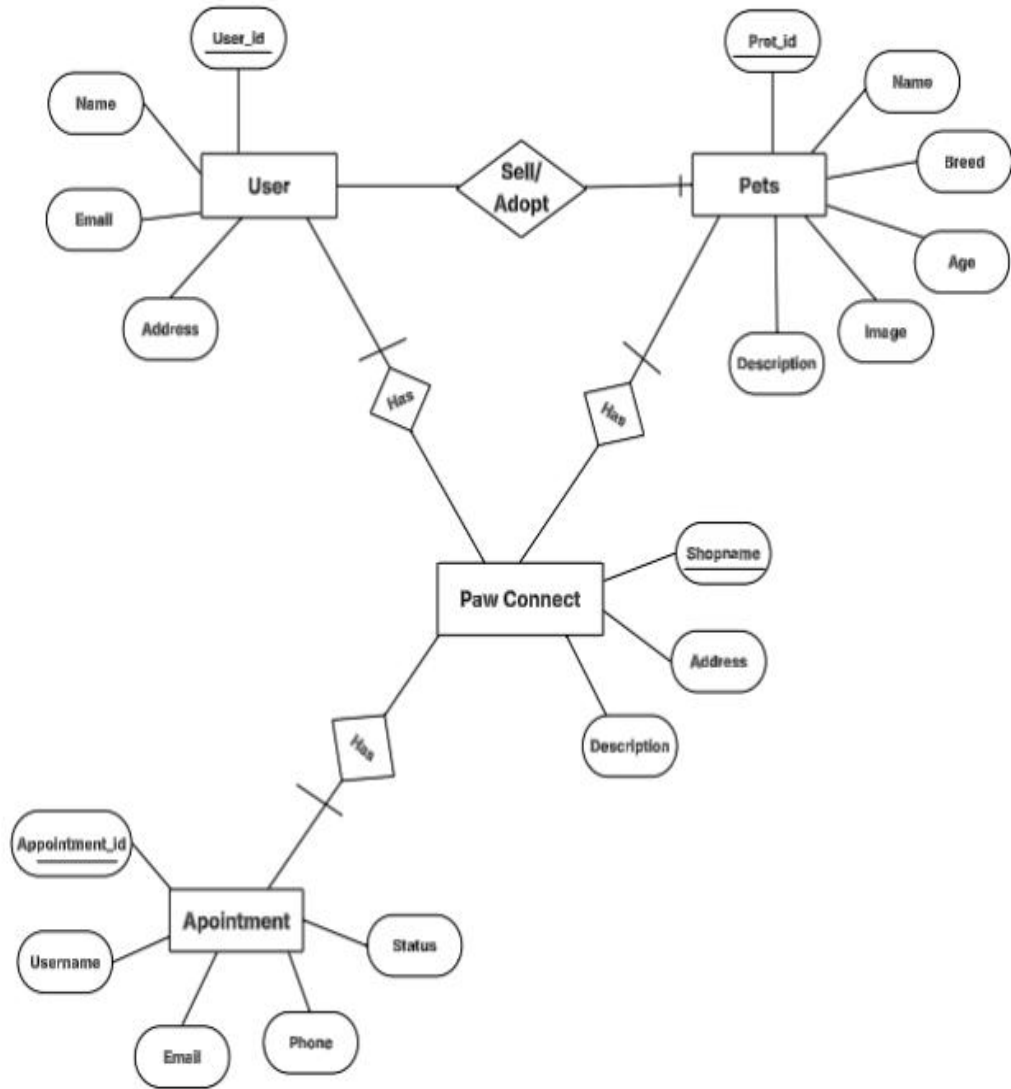


Figure 11: ER diagram showing Users Sell/Adopt and Book an Appointment for Pets

3.9 Assumptions and Dependencies

3.9.1 Assumptions:

- **Users Have Internet Access:** It is assumed that all users (Customers and Admins) have stable internet connectivity to access and interact with the platform.
- **Basic Technical Literacy:** Users are expected to have basic digital literacy, such as navigating a website, filling forms, and using email for communication and verification.
- **Valid Email Addresses:** Users are assumed to have valid and accessible email accounts to receive OTPs, confirmations, and notifications.
- **Google Account Availability:** Users opting for Google Sign-In are assumed to have an active Google account and consent to authentication via Google OAuth.

- **Admin is a Verified Store Owner:** It is assumed that the Admin role is assigned to an authorized and legitimate pet store owner responsible for all approvals and backend operations.
- **Trust in Online Transactions:** Users are comfortable making secure payments through Stripe and sharing basic personal/pet details during the adoption or selling process.
- **Static Service Locations:** The availability of services like grooming or checkups assumes the existence of a physical store or known service location where appointments can be scheduled.
- **Real-Time Communication via Email is Reliable:** It is assumed that email services (via Nodemailer) function reliably for all verification, updates, and notifications.

3.9.2 Dependencies:

- **Technology Stack:** MongoDB: For storing and managing data such as users, pets, products, appointments, and adoption requests.
- **Express.js:** Backend framework for building RESTful APIs and handling server-side logic.
- **React.js:** Frontend framework for building a dynamic, responsive user interface.
- **Node.js:** Runtime environment to execute JavaScript on the server.
- **Authentication & Security:** OAuth 2.0 (Google Cloud): For secure Google login integration.
- **Nodemailer:** To send OTPs and all transactional emails (e.g., adoption approvals, booking confirmations).
- **JWT (JSON Web Tokens):** For maintaining secure user sessions and role-based access control.
- **Payment Gateway Stripe:** For securely processing online payments for pet products and services.
- **Media Management Cloudinary:** To upload, store, and optimize images (e.g., pets, products) with fast loading and reliable delivery.
- **AI Chatbot:** Hugging Face Transformer Model: For powering the intelligent chatbot that assists users with pet care questions.

3.9.3 Deployment Platforms:

- **Render:** Used to deploy and host the backend (Node.js + Express) server and APIs.
- **Netlify:** Used to deploy and host the frontend (React.js) application with continuous integration from GitHub.
- **Version Control & Collaboration:** Git & GitHub: For managing source code, version control, team collaboration, and issue tracking.

- **Email Service (via Nodemailer):** For sending OTPs and notification emails reliably.
- **Mobile Responsiveness:** CSS Media Queries / Frameworks: Used to ensure the platform works seamlessly across mobile, tablet, and desktop devices.

3.10 Specific Requirements

The specific requirements for the "Paw Connect" project are divided into three categories: Functional, Non-Functional, and Security requirements. These define the core features, expected system behaviour, performance benchmarks, and security standards the system must adhere to.

3.10.1 Functional Requirements

- **User Registration & Authentication:** Users must be able to sign up via Standard registration form with Gmail OTP (using Nodemailer). Google OAuth 2.0 login for faster and secure access. After successful login, users should be redirected to their role-based dashboards.
- **Role-Based Access:** The system must support two primary roles User (Pet lover/customer) and Admin (Pet store owner). Each role must have distinct access and feature sets.
- **Pet Adoption Module:** Users must be able to fill out and submit a pet adoption form. Admin receives the application, reviews it, and communicates approval/rejection via email.
- **Pet Selling Module:** Users must be able to submit detailed profiles of pets they want to sell. Admin reviews submissions and continues further communication via email.
- **Appointment Booking System:** Users should be able to book appointments for services like grooming, checkups, or vaccinations. Admin must dynamically manage bookings and notify both users and service providers through email.
- **E-Commerce Module:** Users must be able to browse and purchase pet-related products (food, toys, accessories). Orders must be processed via Stripe for secure online payments. Admin can manage product listings (add/edit/delete).
- **Admin Dashboard:** Admin must be able to Approve/reject adoptions and selling requests. Manage product inventory. View and manage user appointments and requests.
- **AI Chatbot Integration:** Users must be able to interact with a chatbot trained using the Hugging Face Transformer model. The bot should answer pet-related questions (diet, care tips, emergency remedies, etc.).
- **Email Notifications:** All critical interactions (e.g., request approval, payment confirmation, appointment status) must trigger email notifications via Nodemailer.

3.10.2 Non-Functional Requirements

- **Responsiveness:** The web app must be fully responsive across all devices (mobile, tablet, desktop) with no feature degradation.
- **Performance:** The platform must handle a minimum of 100 concurrent users without performance degradation. Standard user operations (e.g., login, booking, checkout) must complete in under 2 seconds.
- **Scalability:** The system should support seamless scaling to accommodate future modules, more users, and higher traffic. The system should support seamless scaling to accommodate future modules, more users, and higher traffic.
- **Availability:** Platform uptime should be maintained at 99.9%, especially during high-traffic periods (e.g., weekends, holidays).
- **Deployment & Version Control:** Frontend deployed via Netlify, and backend via Render. Source code must be version-controlled using Git & GitHub with proper branching strategy and issue tracking.
- **Modular Design:** The architecture should follow a modular structure to allow future updates or third-party integrations with minimal code changes.

3.10.3 Security Requirements

- **Two-Step Verification:** Gmail OTP verification must be implemented during signup to prevent unauthorized account creation.
- **OAuth 2.0 Integration:** Secure user login must be supported through Google OAuth 2.0 with token-based authentication.
- **Role-Based Access Control (RBAC):** Features must be restricted based on user roles. Users can view pets, book services, buy products, and manage their own profiles. Admins have full access to all system modules and data. Unauthorized access attempts should be logged.
- **Secure Payment Handling:** All payment transactions must be processed using Stripe, ensuring PCI-DSS compliance.
- **Media Management Security:** Images must be uploaded and managed securely via Cloudinary, using signed upload presets to prevent abuse.
- **Audit Logs:** System must maintain logs of admin approvals, bookings, and transactions to ensure traceability and accountability.

Chapter 4 – Development and Implement

4.1 Introduction to Frontend and Backend

4.1.1 Languages Used in Frontend

The frontend of Paw Connect is responsible for what users see and interact with. It has been built using a combination of markup languages, styling frameworks, and interactive libraries to create a responsive and dynamic user interface.

- **React.js:** React.js is a powerful JavaScript library developed by Meta and is used for building fast and scalable user interfaces. It allows for the creation of reusable components such as navigation bars, dashboards, pet cards, and booking forms. React enables Single Page Application (SPA) behaviour, allowing users to move between pages like “Adopt”, “Products”, and “Appointments” without reloading the page. It efficiently manages data using state and props, making it possible to dynamically update adoption statuses, display forms conditionally based on user roles, and integrate third-party libraries like Stripe and Cloudinary for advanced functionality.
- **HyperText Markup Language (HTML):** HTML forms the structural backbone of all web pages in Paw Connect. It defines the layout and elements of the application, such as forms, buttons, navigation links, headers, and text areas. Every component rendered on the screen starts with well-structured HTML, ensuring browser compatibility and semantic clarity across the platform.
- **Cascading Style Sheets (CSS):** CSS is used to style and visually enhance the HTML content. It controls the appearance of the interface, including colour schemes, font choices, spacing, layout design, and responsiveness.
- **JavaScript:** JavaScript is the scripting language that powers the dynamic and interactive behaviour of the platform. It handles functionalities such as form validations, click-based events, modal pop-ups, and live chat interactions. JavaScript also plays a crucial role in communicating with the backend by sending and receiving data via APIs. Its real-time behaviour enables the platform to respond immediately to user actions without refreshing the page.
- **Bootstrap:** Bootstrap is a widely-used frontend framework that helps create responsive, mobile-first layouts quickly. It provides a library of pre-designed UI components such as modals, forms, carousels, cards, and buttons, which are used throughout Paw Connect to maintain a consistent and attractive design. class-based and React-Bootstrap component usage.

4.1.2 Languages Used in Backend

The backend of Paw Connect is designed using robust and scalable technologies that ensure secure data handling, smooth user interactions, and efficient communication between the frontend and the database. These technologies work together to manage user authentication, appointment bookings, pet adoptions, order processing, and real-time interactions.

- **Node.js:** Node.js is a powerful JavaScript runtime environment used to build the server-side logic of Paw Connect. It handles tasks such as user registration, login sessions, form submissions, appointment bookings, product orders, and sending email notifications. Since it's built on the same language as the frontend (JavaScript), it creates a seamless development experience and improves code reusability across the project.
- **Express.js:** Express.js is a minimalist and flexible framework built on top of Node.js, used to create the backend APIs and manage routes. It powers all the core backend functionalities, including handling user requests like signing up, logging in, updating pet adoption statuses, booking appointments, and contacting support. Express simplifies the process of routing and middleware handling, making backend development faster and more organized.
- **MongoDB:** MongoDB is a NoSQL database used to store all dynamic data in the Paw Connect application. It stores information related to users, pets, adoption requests, appointments, products, and orders in a JSON-like flexible structure. Its schema-less nature allows smooth handling of various types of data and makes it a perfect match for the dynamic requirements of a pet care platform.
- **Mongoose:** Mongoose is an Object Data Modelling (ODM) library for Node.js that provides a structured way to interact with MongoDB. It allows the developer to define schemas and models for the data, ensuring that the stored information follows a consistent structure. Mongoose also simplifies querying, validation, and data manipulation, making database operations cleaner and more reliable.

Together, these technologies enable secure data handling, smooth user experiences, and real-time communication across the system. Whether it's managing users, handling adoption requests, booking appointments, or processing orders, the backend system of Paw Connect is designed to be scalable, modular, and maintainable ensuring the platform remains responsive and reliable as it grows.

In conclusion, the integration of the SDLC process with a powerful backend tech stack has laid a solid foundation for Paw Connect to serve as a trustworthy and user-friendly pet care platform, supporting future enhancements with ease.

4.2 Other Supporting Languages and Tools Used

4.2.1 Supporting Languages and Libraries

These are code-based tools or packages (usually installed using npm or yarn) that enhance core functionality and handle specific tasks in the app:

- **JSON Web Tokens (JWT):** Used to securely transmit user authentication data between client and server. Maintains login sessions without storing sensitive data on the client. Helps protect restricted routes and ensures only authenticated users can access specific features like donation, adoption, and dashboard views.
- **Dotenv:** Manages environment variables such as database URIs, secret keys, API tokens, etc., through a .env file. Keeps sensitive information secure and out of source code. Supports different environments (development, staging, production) without changing code.
- **CORS:** Enables the frontend (React) to safely access backend APIs during cross-origin requests. Allows controlled communication between frontend on different domains/ports and the backend server. Prevents security issues like unauthorized external API calls.
- **React Router:** Handles frontend navigation between pages in your React app without refreshing the browser. Helps build multi-page experiences such as Home, Shop, Adopt, Appointments, etc., while staying on one page. Supports nested routes and route protection based on user roles.

4.2.2 Supporting Tools Used

- **Stripe:** Secure payment processing service used for handling user payments (e.g. product purchases). Handles product purchases and transactions. Supports payment intents and integrates with order confirmation emails.
- **OAuth 2.0:** Secure login protocol used for integrating third-party login methods. Simplifies the login process for users. Reduces friction and increases security with token-based login.
- **Cloudinary:** Secure login protocol used for integrating third-party login methods like Google or Facebook. Supports image uploads, automatic resizing, and fast delivery. Optimizes media files for performance and storage efficiency.
- **Bootstrap:** CSS framework used with React and custom CSS to build mobile-responsive, styled interfaces. Ensures responsiveness across devices. Speeds up UI development with pre-designed components.

- **Git & GitHub:** Git is used for version control, and GitHub is the remote repository for collaboration and history tracking etc. Tracks all changes made to the project. Supports collaborative development, issue tracking, and deployment via GitHub
- **Nodemailer:** A Node.js package used for sending confirmation emails directly from the backend. Sends confirmation emails for registration, order placement, appointment booking, and password recovery. Easily configurable with email providers like Gmail, Outlook, etc.
- **Render:** Used to deploy and host the Node.js + Express backend API. Enables continuous deployment via GitHub. Supports automatic deployments from GitHub pushes. Offers HTTPS, logging, and environment variable management.
- **Netlify:** Used to host the React frontend, enabling auto-deployment from GitHub, SSL, and custom domains. It provides features like continuous deployment from GitHub, free SSL, custom domain setup, and form handling support for a production-ready experience.
- **Postman:** API testing tool used to verify backend endpoints (e.g., login, signup, appointment) during development. It verifies whether routes like /signup, /login, /create-order, /book-appointment, etc., are functioning correctly and returning expected responses.
- **Visual Studio Code (VS Code):** The primary code editor used for writing, debugging, and managing the codebase with support for extensions. VS Code is the main code editor used by developers. It supports syntax highlighting, Git integration, debugging, extension support and terminal access, streamlining the entire development workflow.

4.3 Implementation of Problem

- **Pseudocode:** OTP Sending Function in Registration Flow

Function: handleSendOtp

Check if the Email field is empty:

- If yes, alert the user: "Please enter your email first."
- Return to stop further processing.

Set loading state to true to show a loader or disabled button.

Make a POST request to the backend endpoint "http://localhost:4000/Signup/verifyotp":

- Use headers: Content-Type: application/json
- Body should include Email from the formData

Await the backend response and parse it as JSON.

Set loading state to false after receiving a response.

If the response is successful (status OK):

- Show success alert: "OTP sent to your email."

b. Set otpSent state to true

If the response indicates failure:

a. Show an alert with the error message returned from the server

Catch any network or unexpected errors:

a. Set loading state to false

b. Show alert: "Error sending OTP. Try again."

- **Algorithm:** OTP Sending Function in Registration Flow

Algorithm Send_OTP

Input: formData.Email (string)

Output: Alert indicating success or failure, updated state (otpSent, loading)

Step 1: IF formData.Email is empty THEN

 SHOW alert "Please enter your email first"

 RETURN

Step 2: SET loading \leftarrow true

Step 3: SEND POST request to "http://localhost:4000/Signup/verifyotp"

 WITH headers { "Content-Type": "application/json" }

 AND body { Email: formData.Email }

Step 4: WAIT for response

 PARSE response JSON as data

Step 5: SET loading \leftarrow false

Step 6: IF response is OK THEN

 SHOW alert "OTP sent to your email."

 SET otpSent \leftarrow true

ELSE

 SHOW alert with data.message

Step 7: IF any error occurs DURING the above steps THEN

 SET loading \leftarrow false

 SHOW alert "Error sending OTP. Try again."

END

This part of the code handles the process of sending a One Time Password (OTP) to the user's email during the signup procedure. It ensures that a user enters their email address and then makes a backend API call to send the OTP. Proper loading indicators and error handling are also implemented to enhance user experience. Once the OTP is sent successfully, the frontend state is updated to move to the next step of verifying the OTP and completing the registration.

- **Pseudocode:** Google Signup Function in Registration Flow

Function: handleSendOtp

Check if the Email field is empty:

- If yes, alert the user: "Please enter your email first."
- Return to stop further processing.

Set loading state to true to show a loader or disabled button.

Make a POST request to the backend endpoint "http://localhost:4000/Signup/verifyotp":

- Use headers: Content-Type: application/json
- Body should include Email from the formData

Await the backend response and parse it as JSON.

Set loading state to false after receiving a response.

If the response is successful (status OK):

- Show success alert: "OTP sent to your email."
- Set otpSent state to true

If the response indicates failure:

- Show an alert with the error message returned from the server

Catch any network or unexpected errors:

- Set loading state to false
- Show alert: "Error sending OTP. Try again."

- **Algorithm:** Google Signup Function in Registration Flow

Algorithm Send_OTP

Input: formData.Email (string)

Output: Alert indicating success or failure, updated state (otpSent, loading)

Step 1: IF formData.Email is empty THEN

 SHOW alert "Please enter your email first"

 RETURN

Step 2: SET loading ← true

Step 3: SEND POST request to "http://localhost:4000/Signup/verifyotp"

 WITH headers { "Content-Type": "application/json" }

 AND body { Email: formData.Email }

Step 4: WAIT for response

 PARSE response JSON as data

Step 5: SET loading ← false

Step 6: IF response is OK THEN

```

        SHOW alert "OTP sent to your email."
        SET otpSent ← true
    ELSE
        SHOW alert with data.message
    Step 7: IF any error occurs DURING the above steps THEN
        SET loading ← false
        SHOW alert "Error sending OTP. Try again."
    END

```

This part of the code handles the process of sending a One Time Password (OTP) to the user's email during the signup procedure. It ensures that a user enters their email address and then makes a backend API call to send the OTP. Proper loading indicators and error handling are also implemented to enhance user experience. Once the OTP is sent successfully, the frontend state is updated to move to the next step of verifying the OTP and completing the registration.

4.4 Test Cases

4.3.1 Unit Testing

- **Purpose:** To test individual components or features in isolation to ensure they behave as expected.
- **Tool Used:** Console logs, Postman for API testing, browser developer tools (Inspect → Console & Network tabs).
- **Example:** Verify that the email OTP is sent when a user signs up.

4.4.2 Integration Testing

- **Purpose:** To ensure different modules like adoption forms, email notifications, user login, and dashboard updates work together correctly.
- **Tool Used:** Postman for backend routes testing, form testing in the browser.
- **Example:** Verify that when a user submits an adoption form, an email is sent to the admin and the form data appears in the admin dashboard.

4.4.3 End-to-End (E2E Testing)

- **Purpose:** To simulate a full user journey, ensuring all steps from registration to action completion (e.g., adoption or booking) work properly.
- **Tool Used:** Browser testing step-by-step (manual click-through testing), screenshots for validation.

- **Example:** User signs up, logs in, fills adoption form, and sees confirmation or gets an email.

4.4.4 Regression Testing

- **Purpose:** To check that previous features still work after adding new features like the chatbot or Stripe payment system.
- **Tool Used:** Browser testing, rechecking older forms and routes after updates.
- **Example:** After integrating Stripe, recheck that signup and adoption forms still work.

4.4.5 Performance Testing

- **Purpose:** To test how the system handles basic load (e.g., multiple users submitting forms at the same time).
- **Tool Used:** Browser console for loading time, stopwatch for email delay, submitting forms quickly in different tabs.
- **Example:** Open 3 browser tabs and book appointments at the same time.

4.4.6 Security Testing

- **Purpose:** To ensure that users cannot access admin-only features or dashboards without permission.
- **Tool Used:** Browser testing by entering admin URLs directly without logging in.
- **Example:** Try opening the admin dashboard while logged in as a normal user.

4.5 Relevant Test Cases for Paw Connect

4.4.1 Test Case 1: OTP Email Trigger on Signup.

- **Test:** Enter email in signup form and submit.
- **Expected Outcome:** An OTP email should be received within a few seconds.

4.4.2 Test Case 2: Chatbot Greeting

- **Test:** Open the chatbot and ask about diet plan of your pet.
- **Expected Outcome:** Chatbot should reply with the diet plan.

4.4.3 Test Case 3: Pet Adoption Form + Email Notification

- **Test:** Submit adoption form as a user.

- **Expected Outcome:** Admin should receive a notification email, and form should be visible in admin panel.
- 4.4.4 Test Case 4: Appointment Booking + Confirmation**
- **Test:** Book an appointment through the website.
 - **Expected Outcome:** Appointment details should be stored, and confirmation email should be sent to both user and admin.
- 4.4.5 Test Case 5: Complete Signup to Adoption Flow**
- **Test:** Sign up with OTP or through Google Log in in it. Fill adoption form.
 - **Expected Outcome:** Adoption request should be stored in the admin panel and admin will perform an action.
- 4.4.6 Test Case 6: Product Purchase using Stripe**
- **Test:** Log in, add product to cart, click “Buy Now”, and enter test card details.
 - **Expected Outcome:** Payment success message is shown, and order is recorded.
- 4.4.7 Test Case 7: Multiple Form Submissions**
- **Test:** Submit 3 adoption forms quickly using 3 tabs.
 - **Expected Outcome:** All should be stored in the admin panel and the admin must perform an action either to accept or reject.
- 4.4.8 Test Case 8: Page Load Time on Home Page**
- **Test:** Open the home page and measure how long it takes to load navbar.
 - **Expected Outcome:** Load time should be under 5 seconds on average internet.
- 4.4.9 Test Case 9: Admin Panel Access by Normal User**
- **Test:** Log in as a user and try opening /admin route.
 - **Expected Outcome:** Access should be denied or redirected.

The table below lists key test cases for the website's main features, including signup, chatbot interaction, adoption process, appointment booking, and product purchase. Each test case outlines the steps to execute and the expected outcomes to ensure the system functions correctly.

Test Case ID	Test Description	Steps to Execute	Expected Outcomes
TC-01	OTP Email Trigger on Signup	Enter email in signup form and submit	OTP email receive within a few seconds
TC-02	Chatbot Greeting Diet-Plan	Open Chatbot and ask for a diet plan	Chatbot responds with appropriate diet plan
TC-03	Pet Adoption Form + Email	Fill and Submit the Adoption form	Admin can see form in the admin panel
TC-04	Appointment Booking + Confirmation	Book appointment via website	Details stored confirmation email sent to user
TC-05	Complete Signup to Adoption Flow	Sign up to log in to submit adoption form	Adoption request appears in admin dashboard
TC-06	Product Purchase via Stripe	Log in → Add product → Buy Now → Enter test card	Payment success message; order stored

Table 2: Test Cases Performed

Chapter 5 – Conclusion and Future Scope

5.1 Conclusion

Paw Connect successfully bridges the gap between pet owners, adopters, veterinary professionals, and animal welfare contributors by offering a comprehensive, user-friendly digital platform tailored to a wide range of pet-related needs. It is thoughtfully designed to streamline and simplify various tasks such as pet adoption, selling, appointment scheduling, and expert consultation—making these services easily accessible from a single unified interface. Built using the powerful and scalable MERN stack, the system ensures efficient performance and smooth interaction for all user roles, while maintaining modern design standards for intuitive navigation.

The platform offers secure and reliable product purchasing experiences through Stripe integration and enhances user engagement through AI-driven features like intelligent pet care suggestions, diet plans, and home remedies powered by Hugging Face. Cloud services like Cloudinary and Firebase contribute to data handling, media storage, and real-time responsiveness, while Netlify and Render ensure stable deployment and hosting.

By bringing together essential services and intelligent automation under one digital umbrella, Paw Connect not only empowers pet owners and caregivers with convenient tools but also encourages responsible pet care, ethical adoption practices, and timely veterinary attention. Ultimately, the project represents a meaningful contribution toward animal welfare through thoughtful design and modern technology.

5.2 Future Scope

Our project is currently performing exceptionally well, successfully delivering essential functionalities such as pet adoption, product purchases, appointment bookings, and AI-based pet care assistance. It ensures a seamless, reliable, and user-friendly experience, making it a comprehensive solution for pet owners and service providers alike. The platform has demonstrated strong usability, functional correctness, and user satisfaction through its core modules and interactions. However, to further scale its impact and continue meeting evolving user needs, there is ample scope for meaningful enhancements that can add significant value to the system.

One key enhancement involves the integration of real-time GPS-based delivery tracking with live status updates. This feature will empower users to track their product deliveries directly on a live map, get updates such as estimated arrival time, current delivery location, and status changes like "Out for Delivery" or "Delivered". Such transparency is especially crucial when dealing with sensitive or perishable pet-related products, as it builds trust and reduces uncertainty. Technologies like Google Maps API, Mapbox, and GPS-based event triggers can be used for smooth integration with the existing backend.

Another forward-looking enhancement includes upgrading the AI pet care assistant by incorporating voice-enabled interaction and multi-language support. While the current chatbot efficiently handles text-based queries, enabling voice communication will greatly enhance accessibility, especially for elderly users, visually impaired individuals, or users who prefer conversational interaction. Additionally, multi-language support will break language barriers, allowing the platform to cater to a broader and more diverse user base across different regions. This can be achieved using speech-to-text and text-to-speech APIs to translate and localize the chatbot's responses.

By integrating these enhancements, the platform can evolve into an even more dynamic, inclusive, and feature-rich ecosystem. It will not only improve the convenience and satisfaction of current users but also attract new users by offering modern, smart, and personalized pet care services in a scalable and future-ready manner.

REFERENCES

- [1] D. Gawande, V. V. Gawande, and S. S. Gawande, "NOOR: Cross–Platform Pet Adoption Application," International Research Journal of Modernization in Engineering Technology and Science (IRJMETS), vol. 6, no. 4, Apr. 2025. [Online].
- [2] S. S. Patil, A. S. Patil, and V. V. Gawande, "The Paw Project: A Mobile Application for Identifying Stray Dogs," International Journal of Engineering Research & Technology (IJERT), vol. 11, no. 9, Sep. 2023. [Online].
- [3] Nodemailer, "Nodemailer Documentation," 2025. [Online].
- [4] W. C. Brooks et al., "The Pet Health Library," Educational Director, vol. 4, p. 2010, 2008. [Online]. Available: <https://www.pets.ca/health> (Date reviewed/revised: 2011).
- [5] A. S. Patil, V. V. Gawande, and S. S. Gawande, "PetHub: A Platform for Pet Adoption," International Journal of Modern Research in Science, Engineering and Technology (IJMRSET), vol. 2, no. 7, Jul. 2024. [Online].
- [6] Razorpay, "Razorpay Payment Gateway Documentation," 2025. [Online].

APPENDIX

OUTPUT SCREENS

- **Landing Pages**

Figure 12 shows the landing page of the website, where users are first directed after accessing the platform. It provides a clear and welcoming interface, guiding users to explore key features or navigate further.

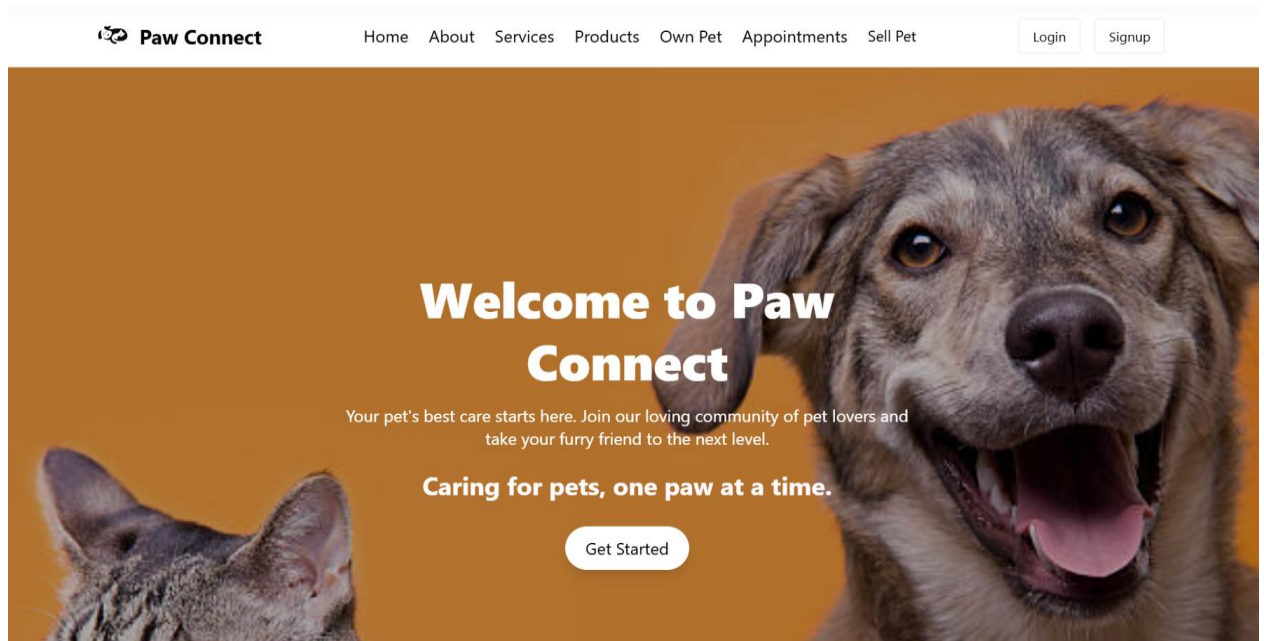


Figure 12: Landing into Website

Figure 13 displays the Signup Page of the website, allowing new users to register by entering their details or using the convenient "Sign in with Google" option for quick access

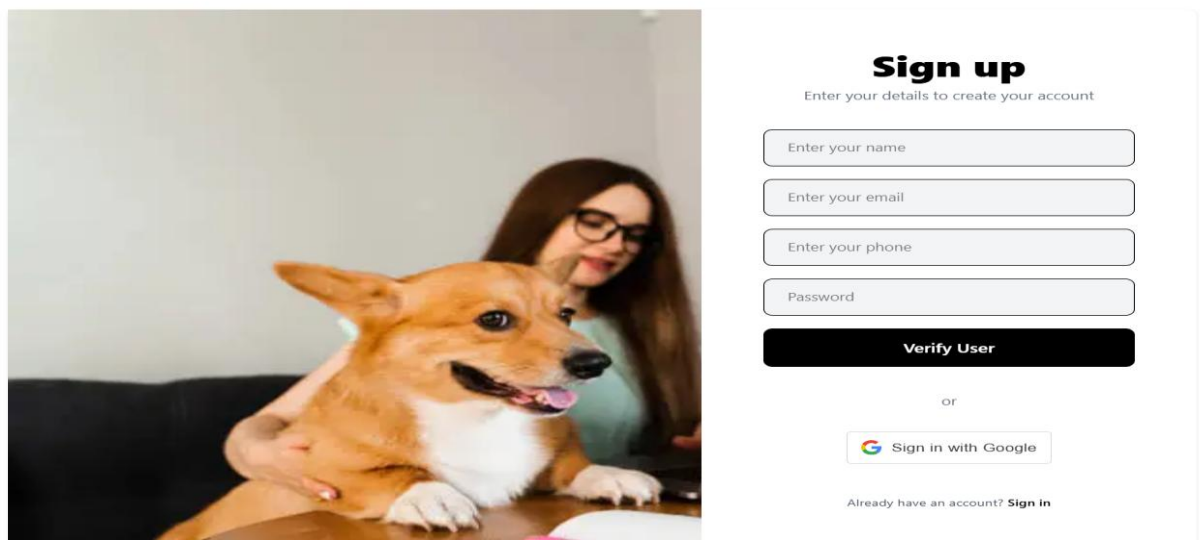
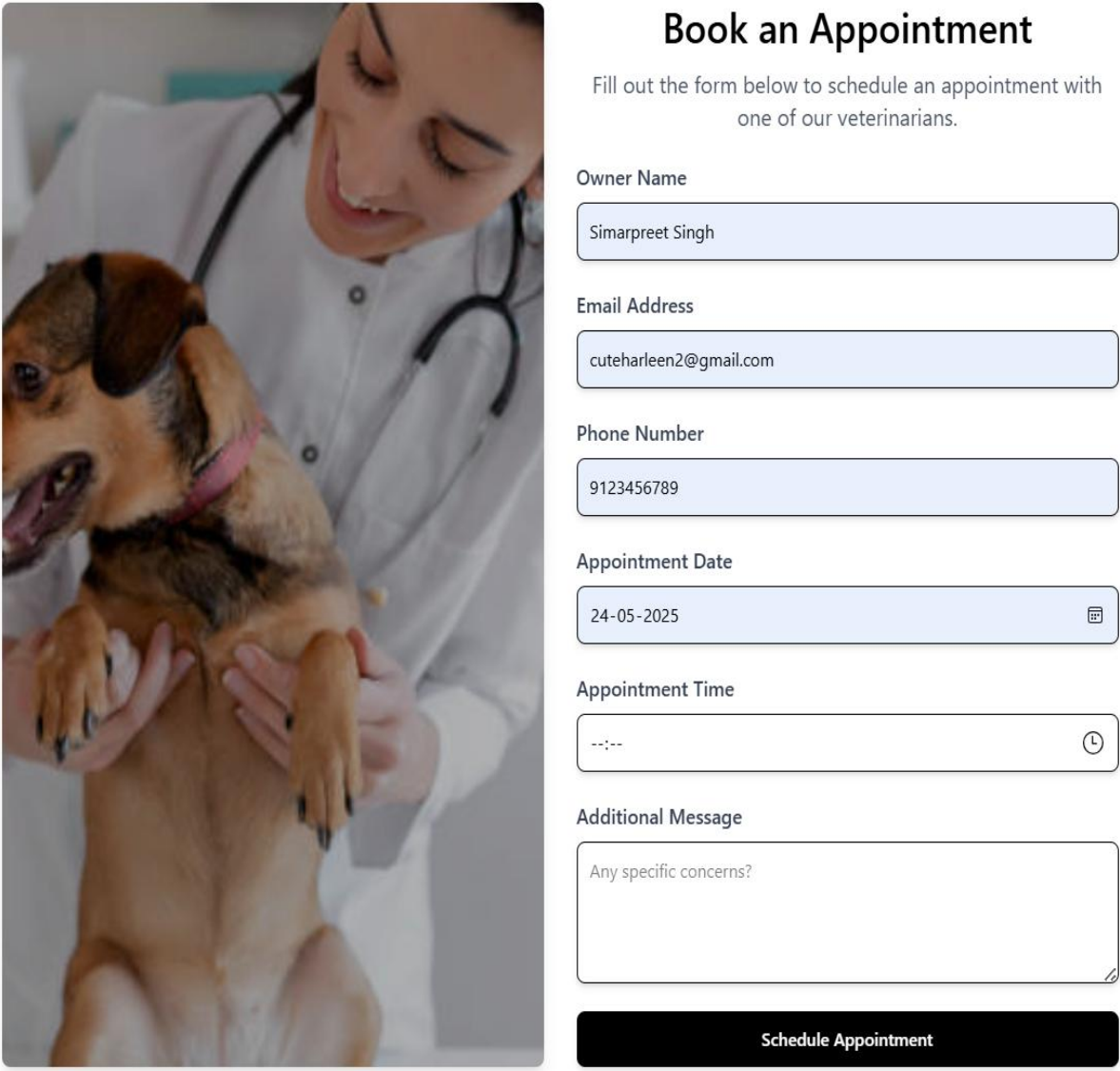


Figure 13: Signup Page

- **User Panel**

Figure 14 shows the Doctor Appointment Form, where users can book appointments by providing details like date, time, and pet information. The form ensures a smooth and efficient appointment booking process.



Book an Appointment

Fill out the form below to schedule an appointment with one of our veterinarians.

Owner Name

Email Address

Phone Number

Appointment Date

Appointment Time

Additional Message

Schedule Appointment

Figure 14: Doctor Appointment Form

Figure 15 displays the Successful Payment confirmation page for products, indicating that the transaction has been completed. It provides users with assurance and details of their purchase.

Success! Payment Completed

Order Summary:

Email:

harleenparmar158@gmail.com

Total Amount:

\$4.50

Items Purchased:

Nail Caps

\$4.50 x 1

\$4.50

Figure 15: Successful Payment for Products

Figure 16 showcases the Mobile Interface of the website, demonstrating its responsive design and user-friendly layout on smaller screens. It ensures seamless access and navigation for users on mobile devices.



Figure 16: Mobile Interface

Figure 17 displays the Chatbot Application integrated for user support, allowing users to ask questions and receive instant assistance. It enhances user experience by providing real-time, interactive help.

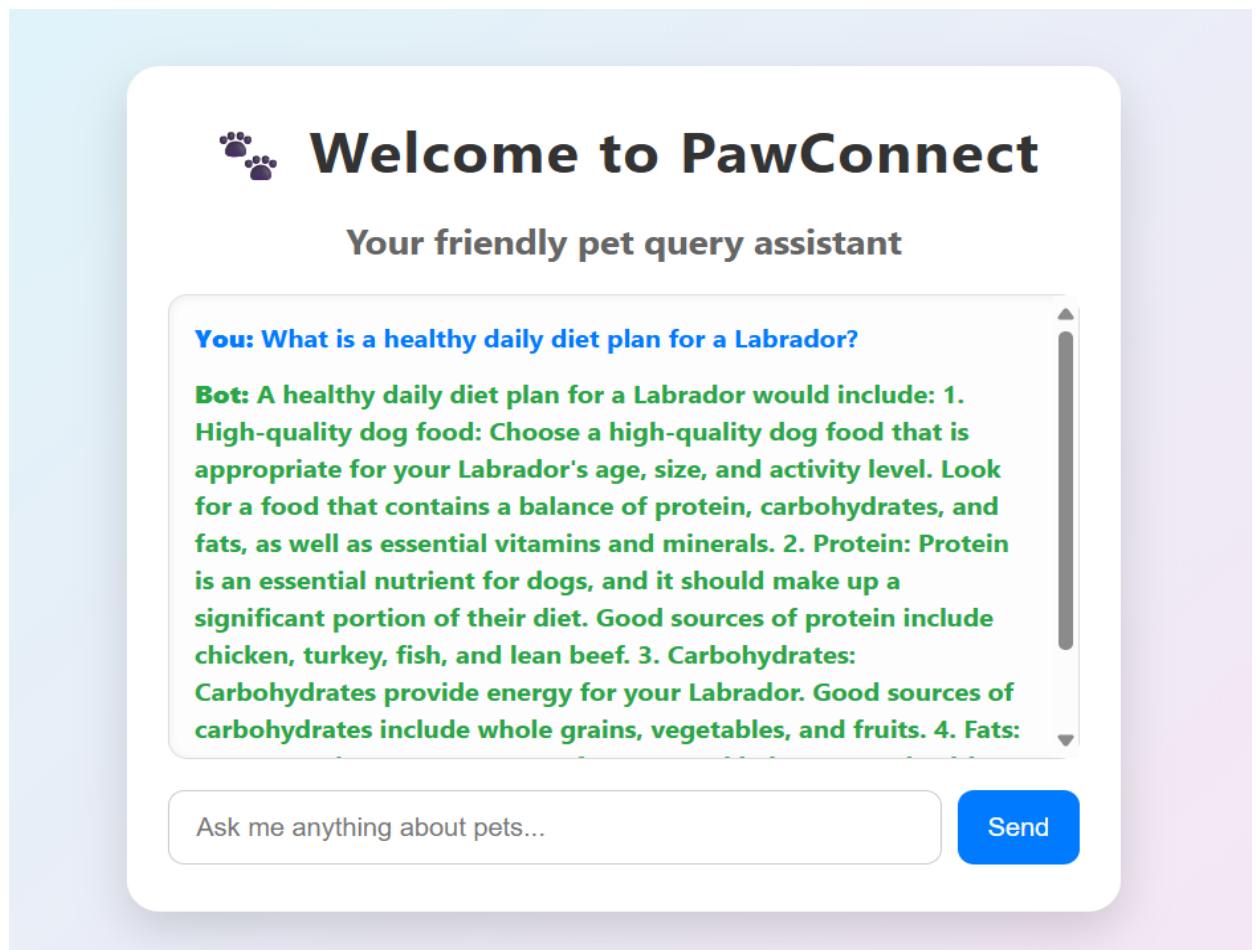


Figure 17: Chatbot Application for User

- **Admin Panel**

Figure 18 shows the Admin Panel Dashboard, where administrators can manage users, products, appointments, and other site activities. It provides a centralized and organized interface for efficient backend operations.

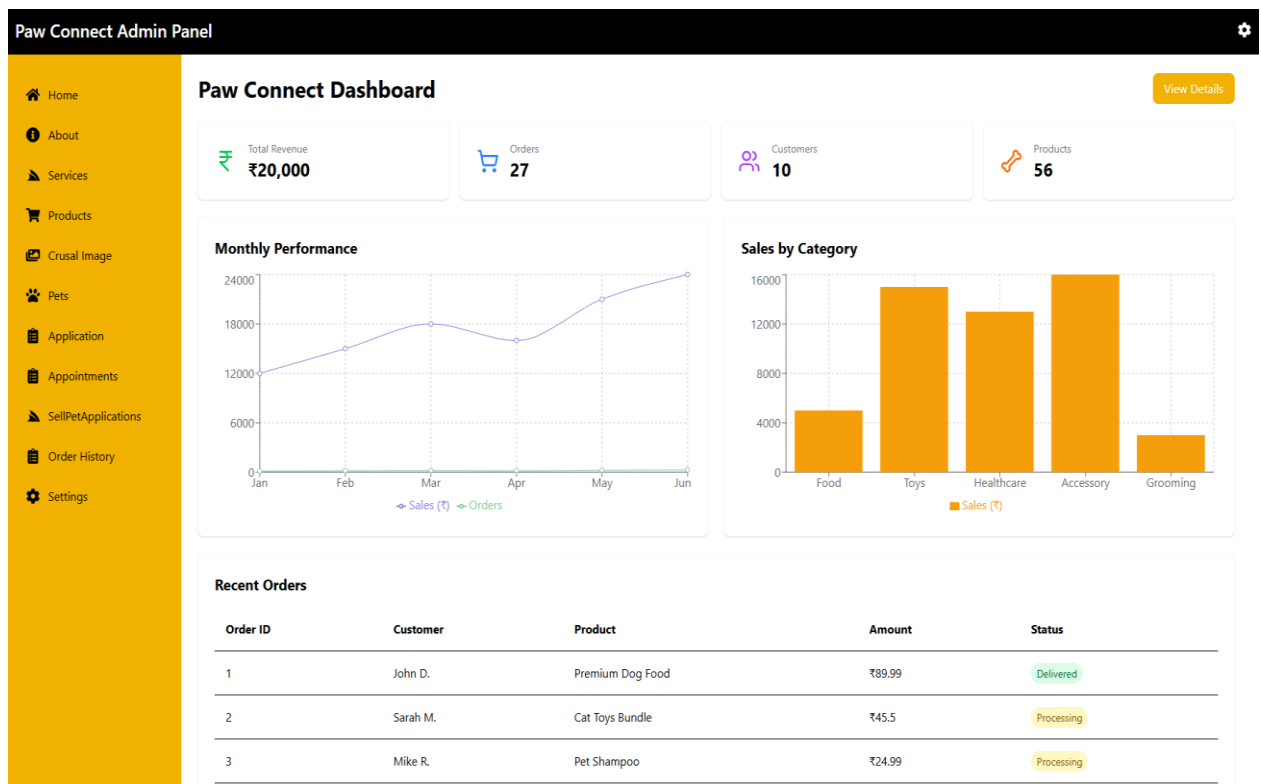


Figure 18: Admin Panel Dashboard

Figure 19 displays the interface where the admin reviews appointment requests, with options to accept or reject them. This feature ensures proper management and scheduling of appointments within the system.

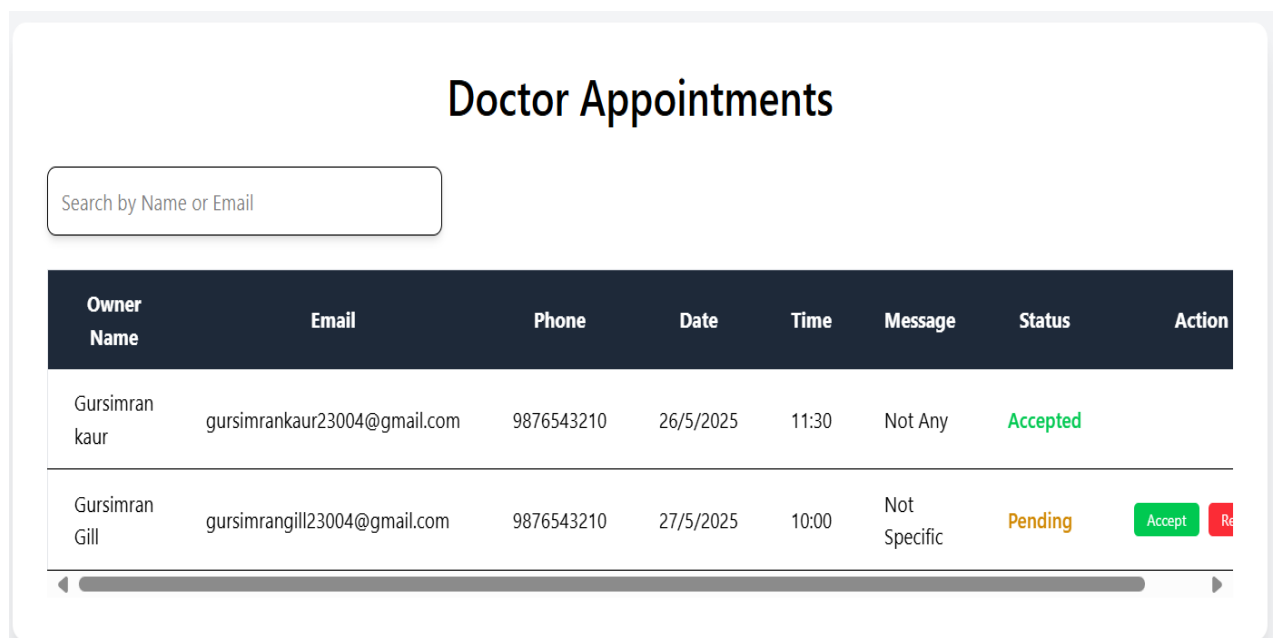


Figure 19: Appointment request to be accepted/rejected

Figure 20 shows the Order History section, where admin can view details of users past product purchases. It helps admin keep track of users transactions.

Order History


Filter by Email

Filter by Product ID

Order ID	Email	Product ID	Product Name	Amount
683338301fda47c929e69e43	harleenparmar158@gmail.com	prod_SN0tAdb7ELozxS	Nail Caps	₹450

Figure 20: Order History of Products

Figure 21 illustrates the admin interface for managing pet details, allowing administrators to add, update, or remove pet information efficiently. This helps keep the pet database accurate and up to date.

 **Pet Registry**

Manage your beloved pets


+ Add New Pet

Filter by Category

All

Cat


Dog



Simba

Cat

A curious and active kitten



Kitty

Cat

A calm brown Persian cat

Figure 21: Managing Pets Detials by admin

Figure 22 displays the admin panel for managing product details, where administrators can add, edit, or delete product information. This feature ensures that the product catalog remains current and well-organized.

Product Page

+ Add Product

All Categories ▾







Image	Name	Price	Category	Quantity	Actions
	Lamb and Rice	400	Dog	10	Edit Delete
	Nail Caps	450	Cat	12	Edit Delete
	Dog Shelter	300	Dog	8	Edit Delete
	PatPet	1000	Cat	5	Edit Delete
	Whiskas	300	Cat	12	Edit Delete
	Cat Harness	2000	Cat	8	Edit Delete

Figure 22: Managing Products Details by admin