Sri Lanka Institute of Information Technology

IT2080 - Information Technology Project

Year 2, Semester 2 -2025

Project Proposal

ITP25_B7.2_168

Project Title: Veterinary Management System (PetIQ)

Group Details

	IT Number	Student Name	Student E-mail Address	Contact Number
1	IT23640948	Gamlath K. G. V. K. D	it23640948@my.sliit.lk	0702943487
2	IT23631106	Kavindu J. M. R	it23631106@my.sliit.lk	0703274142
3	IT23631274	Abeysinghe A. M. B. N	it23631274@my.sliit.lk	0761260885
4	IT23594722	Wanasinghe W. M. D. T	it23594722@my.sliit.lk	0774376338
5	IT23413474	Rathnayake W. P. D. D. W	it23413474@my.sliit.lk	0702979383

Table of Contents

1. Background

Our client is **PetIQ Animal Hospital,** a well-known veterinary clinic located in Athurugiriya, in the Western Province of Sri Lanka. With over 8 years of dedicated service, the hospital has grown steadily from a small startup clinic into a reliable and trusted animal care center for the local community.

Initially operating with limited resources and no laboratory facilities, the hospital has progressed through several stages of development to now feature modern facilities and fully functional labs. Its primary mission is to provide high-quality care for pets while ensuring their owners feel supported and comfortable. Services are delivered on time and at affordable rates, earning the trust of many pet owners over the years.

The head veterinarian aims to expand and improve services across the Western Province. However, managing daily operations, organizing appointments, maintaining patient records, and handling payments have become increasingly challenging due to the manual system in place. To address these issues, we propose the development of a Veterinary Management System, a digital platform that will streamline operations, improve data accuracy, and enhance overall service delivery for both staff and pet owners.

2. Problems and Motivation

2.a Problem Statement

The head veterinarian currently manages all hospital operations manually, a process that has become increasingly labor-intensive and inefficient. Although he has expressed a strong desire to shift to a digital system, his limited technical knowledge and lack of IT resources have prevented this transition.

At present, the following critical information is recorded manually:

- Staff details
- Pharmacy records
- Medical records
- Payment details
- Stock and inventory records
- Patient history and details

In addition to data storage, several important calculations are also performed manually, including:

- Payments of patients
- Stock and inventory evaluations
- Employee salary calculations

The client noted that this manual approach has created numerous challenges in his daily workflow. The lack of integration between processes means that even a small mistake can lead to serious disruptions, affecting his limited time and adding unnecessary workload. Furthermore, documenting daily activities using registers and daybooks is not only time-consuming but also prone to error, making it difficult to manage and retrieve information efficiently. These challenges have significantly impacted on the overall productivity and operational effectiveness of the hospital.

2.b Motivation (Benefits of This System)

Implementing a Veterinary Management System will provide several key benefits:

- **Readily Available**
  Accessible anytime, from anywhere (24/7), improving responsiveness and service availability.

- **Real-Time Data Access**
  Enables accurate, up-to-date insights that support timely and informed decision-making.

- **High Accuracy and Convenience**
  Ensures precise record-keeping and easy data retrieval, minimizing human error.

- **Cost Reduction**
  Streamlines processes, reducing the need for excess staff and associated overhead costs.

- **Secure Data Management**
  Protects sensitive data with controlled access, regular backups, and secure storage mechanisms.

- **Proper Operational Management**
  Replaces manual records with a web-based platform that simplifies administrative tasks.

- **Enhanced Efficiency and Productivity**
  Improves the workflow of staff and supports more efficient management of time and resources.

3. Aim and Objectives

3.a Aim

To design and develop a comprehensive Veterinary Management System (VMS) that digitizes and automates daily operations in veterinary clinics and animal care centers, improving efficiency, data accuracy, and healthcare delivery through seamless integration and a user-friendly interface.

3.b Objectives

- **Automate Core Operations**
  Develop modules for appointment scheduling, patient registration, EHR, billing, and inventory to reduce manual work.

- **Enhance Data Accuracy & Security**
  Use secure, centralized databases with role-based access, encryption, and backups.

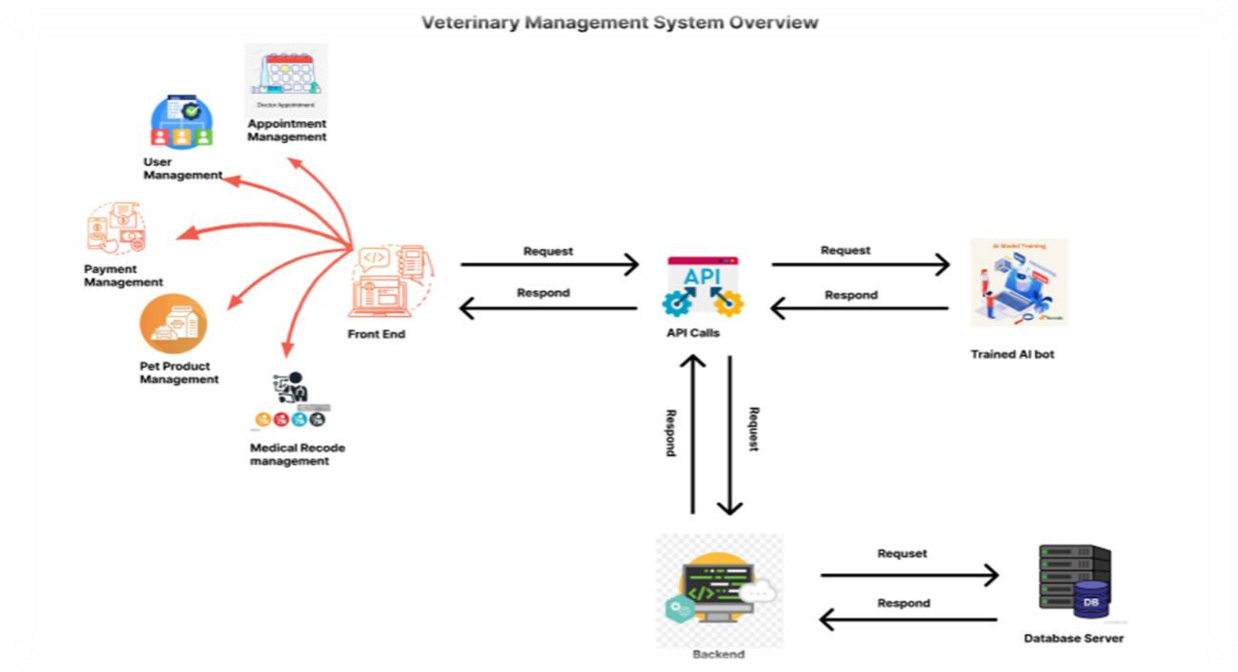
- **Streamline Inventory Management**
  Implement real-time stock tracking, alerts for low stock/expiry, and supplier tracking.

- **Enable Analytical Reporting**
  Provide dashboards and reports on treatments, finances, and inventory trends.

- **Ensure Scalability & Accessibility**
  Build a web/mobile-friendly system for clinics of all sizes with remote access.

- **Support Animal Health Monitoring**
  Track vaccination schedules, treatment history, and diagnostic follow-ups.

4. System Overview



4.1 Appointment Management

The Appointment Management module enables efficient booking and coordination of appointments within the veterinary clinic. Pet owners can easily schedule consultations, vaccinations, and treatment sessions, while veterinarians and staff can view and manage their schedules to reduce waiting times and ensure timely care. This module is tightly integrated with medical history and user profiles, contributing to a seamless and effective clinical workflow. Key features include real-time availability and support for managing multiple services. The system enhances operational efficiency, improves client satisfaction, and ensures pets receive timely medical attention.

Functional Requirements

- **Create Appointment**
  - Users (staff or pet owners) can schedule new appointments by entering:
    - Pet name
    - Owner name
    - Appointment date and time
    - Service type (e.g., consultation, vaccination)

- **Read/View Appointments**
  - Veterinarians and authorized staff can view all upcoming appointments.
  - Support filtering appointments by:
    - Date
    - Pet name
    - Owner name
  - Check availability for specific time slots.

- **Update Appointment**
  - Allow changes to:
    - Appointment date and time
    - Assigned veterinarian
    - Linked health records or services

- **Delete Appointment**
  - Enable cancellation of appointments with:

Non-Functional Requirements

- **Reliability & Availability**
  - The system must maintain high time and ensure appointments can be booked and retrieved without failure.
- **Usability**
  - Simple and intuitive appointment forms with calendar/date-picker integration for selecting time slots.
- **Security**
  - Appointment data must be protected using secure authentication and role-based access.

Technical Requirements

Access Control

- Only authenticated users (pet owners and authorized staff) can create, view, or modify appointments.
- Role-based permissions determine who can edit or cancel appointments.

Data Quality

- Validations on:
  - Time conflicts
  - Past date/time restrictions
  - Required fields (e.g., pet name, service type)
- Clear error messages for invalid inputs.

4.2 User Management

The User Management module is a foundational component of the Veterinary Management System (VMS). It handles onboarding, authentication, and role-based access control for all system users including Pet Owners, Receptionists, Nurses, Veterinarians, and Admins. This module ensures that user information is securely stored and easily maintained, supporting efficient communication and coordination across the system.
It is closely integrated with other modules such as Appointments, Billing, and Medical Records. This centralization of user data helps streamline operations and enhances both administrative control and patient care.

Functional Requirements

- **Create User**
  - New users (Pet Owners or Staff) can register by providing:
    - Name
    - Email
    - Phone number
    - Address
    - Username and password
    - Role (Pet Owner, Receptionist, Nurse, Veterinarian, Admin)
  - Admins can manually create staff user accounts.
- **Read/View Users**
  - Admins can view all user profiles along with roles and account statuses.
  - Staff can access relevant user data (e.g., assigned pet owners or patients).
  - Users can view their own profile and access a role-specific dashboard.
- **Update User**
  - Users can update personal details including:
    - Email
    - Phone
    - Address
    - Password
  - Admins can modify:
    - User role
    - Account status (active/deactivated)
    - Contact information

- **Delete User**
    - Admins can deactivate or permanently delete user accounts in cases of misuse, inactivity, or closure.
    - Deleted users should lose access to the system, while their linked data (e.g., appointments, records) remains stored for reference.

Non-Functional Requirements

- **Usability**
    - Interfaces for registration, login, and profile editing must be user-friendly and responsive across devices.
- **Security**
    - All user data must be encrypted and secured using industry-standard authentication protocols.
    - Role-based access ensures users only see what is relevant to their role.
- **Recoverability**
    - The system should support recovery of inactive or accidentally deleted accounts through admin tools or data backups.
- **Enhanced Care Coordination**
    - Accurate role-based access helps streamline communication and task delegation, improving the quality of patient care.

Technical Requirements

- **Access Control**
    - Role-Based Access Control (RBAC) must enforce feature-level permissions.
    - Admins have unrestricted access to manage all user roles and actions.
- **Data Quality**
    - System must validate:
        - Email format
        - Password strength
        - Unique usernames
        - Valid role selection
- Backend logic must prevent duplicate user entries and ensure consistent user data.

4.3 Medical Record Management

The Medical Record Management module enables veterinarians to record, access, and manage comprehensive medical data for each pet. It supports full CRUD (Create, Read, Update, Delete) operations and is tightly integrated with modules such as Appointment, Prescription, and Doctor Management.
This module ensures secure, accurate, and accessible medical records, improving clinical decision-making, care quality, and overall efficiency in veterinary operations.

Functional Requirements

- **Create Medical Record**
  - Veterinarians can create new medical records during or after an appointment.
  - Inputs include:
    - Pet details (auto-filled from appointment or entered manually)
    - Diagnosis, symptoms, and vital signs
    - Treatment plan
    - Assigned veterinarian
  - The system allows uploads and attachments of:
    - Prescription documents
    - Lab reports (PDF, JPG, PNG)
    - Scans or X-rays
    - Notes on test results and follow-up steps
- **Read/View Medical Records**
  - Veterinarians and authorized staff can view:
    - Complete treatment history of any pet
    - Individual medical records with attached files
  - Records can be filtered by:
    - Date
    - Pet name
    - Owner name
    - Veterinarian
    - Diagnosis
    - Report type

- **Update Medical Records**
  - Authorized users can update medical entries by:
    - Modifying diagnosis or treatment details
    - Editing prescriptions
    - Uploading or replacing lab reports and scans
    - Adding progress notes or updates

- **Delete Medical Records**
  - Admins or senior veterinarians can delete:
    - Full medical records (in case of errors or test data)
    - Specific file attachments (e.g., outdated prescriptions or reports)
  - Deleted records must be soft-deleted or archived for audit and recovery purposes

Non-Functional Requirements

- **Usability**
  - The interface should support fast data entry, structured browsing, and file uploads with an intuitive design.

- **Enhanced Patient Care**
  - A complete history of each pet's health supports better diagnosis, continuous care, and treatment tracking.

- **Streamlined Communication**
  - Shared access to records improves collaboration and decision-making among the care team.

- **Recoverability**
  - Deleted or modified records should be restorable through version history or system backups.

- **Security**
  - Medical records must be encrypted and protected with role-based permission.
  - All access and modifications should be logged.

Technical Requirements

- **Access Control**
  - Only licensed veterinarians and authorized staff can create, update, or delete medical records.
  - Pet owners may only view their pet's medical history but cannot make changes.

- **Data Quality**
  - Mandatory field validation (e.g., date, diagnosis, veterinarian).
  - File uploads format and size restrictions (e.g.: PDF/JPG/PNG only, max 10MB).
  - Duplicate or incomplete entries should be prevented through backend validation.

4.4 Payment Management

The Payment Management module handles all financial transactions within the Veterinary Management System (VMS). It enables pet owners to pay securely for appointments, treatments, medications, and pet food orders. It also includes a complete Credit/Debit Card Management system that supports full CRUD operations. The system integrates with other key modules like Appointments, Billing, Medical Records, and Mart, ensuring all payments are linked to the correct service while complying with modern security standards.

Functional Requirements

1. Make a Payment

- **Pet owners can make payments for:**
  - Appointment bookings
  - Medical treatments
  - Prescriptions
  - Pet food orders

- **During checkout:**
  - Users can select a saved card or enter a new one
  - System processes the payment via a payment gateway

- **After payment:**
  - A receipt is generated and saved
  - Payment is linked to the relevant service or order

- **Users and admins can view past payment records**
- **Payments cannot be updated or deleted for audit and security reasons**

2. Credit/Debit Card Management

- **Add Card**
  - Pet owners can securely add a new card (cardholder name, number, expiry, CVV)
  - Fields are validated:
    - Card number (Luhn algorithm)
    - Valid expiry format
    - CVV (3-4 digits)

- **View Cards**
  - **Displays:**
    - Last 4 digits
    - Card type (Visa/Mastercard)
    - Expiry date

- **Update Card**
  - Pet owners can update their saved card details securely
  - All fields are re-validated

- **Delete Card**
  - Pet owners can delete any saved card at any time
  - Deleted cards are permanently removed and cannot be recovered

- **Use Card for Payment**
  - Cards can be used at checkout for faster transactions

Non-Functional Requirements

- **Usability**
  - Mobile-friendly user interface
  - Optional autofill (with consent)
  - Clear, simple forms for managing cards

- **Security**
  - **Complies with PCI-DSS standards:**
    - No sensitive card data stored in the local database
    - Tokenization used for card operations
    - All actions done over HTTPS
  - **Role-based access:**
    - Pet owners manage only their cards
    - Receptionists can use cards for payments (with consent)
    - Admins cannot view full card data

- **Recoverability**
  - Deleted cards cannot be restored
  - All payment transactions are logged permanently
  - Admins can trace transactions for audits

- **Availability**
  - The payment module is available 24/7 for online bookings and purchases

Technical Requirements

- **Access Control**
  - Pet Owners: Add, view, update, and delete their cards; make payments
  - Receptionists: Use cards to process payments (with consent)
  - Admins: View payment history/logs (no card details)

- **Payment Gateway Integration**

- **Secure connection with payment gateway**

- **Gateway must:**
  - Return success/failure status
  - Tokenize card data
  - Log transactions securely

- **Data Validation**
  - Card number: Validated using Luhn algorithm
  - Expiry date: Must be in the future
  - CVV: Must be 3-4 digits
  - Only valid cards can be saved or used

4.5 Pet Product Management

The **Pet Product Management** module enables veterinary clinics to manage and sell pet related products through the Veterinary Management System (VMS). It provides an **admin interface** for authorized staff to manage product listings (CRUD operations) and a **customer interface** for pet owners to browse, search, and purchase products.
Products may include food, medication, accessories, supplements, and grooming items, supporting both clinic operations and customer convenience.

Functional Requirements

1. Manage Products

- **Create:** Add new products with details such as:
  - Product name
  - Description
  - Category (food, medicine, accessories, etc.)
  - Price
  - Stock quantity
  - Product image

- **Read:** View a list of all products with sorting and filtering options.

- **Update:** Edit product details, change stock levels, update prices, or replace image.

- **Delete:** Remove products that are discontinued or out of stock.

2. Product Catalog

- Display available products with:
  - Product name
  - Description
  - Price
  - Quantity available
  - Product images

3. Search and Filter Products

- Search by product name or category.
- Filter by:
  - Product type (food, toys, supplements, etc.)
  - Price range
  - Availability status (in stock / out of stock)

4. Add to Cart

- Add multiple products to a shopping cart.
- Select quantities per product.
- Persist in cart data during the session until checkout.

Non-Functional Requirements

- **Usability:** Simple, responsive user interface for browsing, searching, purchasing, and managing products.
- **Accuracy:** Ensure product details (price, stock, description) are always up to date.
- **Security:** Secure product, cart, and order data, especially during checkout, using encrypted protocols.
- **Availability:** Product catalog and cart should be accessible 24/7 with minimal downtime.

Technical Requirements

- **Access Control:**
  - Only admins and authorized staff can add, update, or delete products.
  - Pet owners can view, search, and purchase products but cannot modify listings.
- **Data Quality:**
  - Prevent duplicate product entries.
  - Validate product inputs:
    - Price must be numeric.
    - Stock count must be non-negative.
    - Product name and category cannot be empty.
- **Integration with Payment Module:**
  - Link cart items to the payment system during checkout.
  - Generate invoices and store transaction details after payment confirmation.

5. Literature Review

We decided to closely examine these two solutions.

1. **Medical Records on Paper**

This is the basic traditional method that can be taken as an existing solution. It involves manually attending to and noting down each patient's requirements, prescriptions, arrival dates, and information about patients who have already received treatment as well as details regarding the patients who still need to be treated.

Paper based medical records are created by recording all the necessary data and information about patients into logbooks, daybooks, files, physical charts, and manually drawn diagrams.

Additionally, financial issues and payments, such as past-due loans, patient payments, and deferred payments, will all be individually recorded in paper-based materials.

Pros and cons of using medical records on paper

Pros	Cons
Cheaper medium to use.	Highly prone to been damaged.
Easy access to the medium.	Higher chances of records have been misplaced.
Readily available.	Lack of proper security.
	Time consuming.
	Problems will arise regarding storage space.

Stacks of paper-based records are very vulnerable to mutilation and damage. Books and files containing information have been lost or are unavailable in an emergency. Time allotted for more crucial duties will be disrupted by inadequate security and the costs associated with locating suitable storage.

2. Utilizing Excel spreadsheets

Excel spreadsheets are effective tools for keeping a certain quantity of data; they can be used to document things like salary, inventories, suppliers, restocking schedules, payments, available medications, and future prescription drug needs.
The benefits and drawbacks that the clinic may experience are listed below. Should this solution be applied.

Pros and cons of using Excel spreadsheets

Pros	Cons
Data can be stored well organized.	Highly subjected to human error and mistakes.
Not necessary to obtain technical support.	Not a suitable method in doing collaborative work.
Easily available anytime.	No proper security.
	Slow and less reliability.
	Manual entering can consume a lot of time.

Medical records provide data and information about the patient. If this data were lost or corrupted because of unforeseen system failures, it would be a huge catastrophe.

Excel spreadsheets are vulnerable to unauthorized access, thus there is no assurance that patient privacy will be protected.

Therefore, a more dependable and effective solution must be offered to handle and arrange all the data belonging to all internal and external stakeholders appropriately, as the two options that have already been highlighted contain too many hazards and inaccuracies.

Therefore, we stress the importance of having a more technologically based and successful solution (the system that the customer has asked us to design) in order to fully handle all of the jobs efficiently.

6. Methodology

6.1 Tools and Technologies

Category	Tools/Technologies
Programming Environment	Visual Studio Code
Programming Language & Stack	MERN Stack (MongoDB, Express.js, React.js, Node.js)
Frontend Development	React.js, HTML5, CSS3, JavaScript (ES6+), Tailwind CSS
Backend & APIs	Node.js, Express.js
Database	MongoDB (hosted on MongoDB Atlas)
UI/UX Design	Figma (wireframing, prototyping)
Diagramming	Draw.io (UML, system flows), StarUML (detailed modeling)
Version Control	Git, GitHub
API Testing	Postman
Deployment	Vercel/Netlify (Frontend), Render/Railway (Backend)

In our Veterinary Management System project, we utilized a combination of visual modeling tools and project management software to support the system's planning and development. We structured our project in modules, allowing individual team members to work independently on different components while following a unified development process.

For diagrams and visual representations, we used tools like **Draw.io**, **StarUML**, and **MockFlow** to design system flows, user interfaces, and UML models. Each member delivered their assigned parts on time. Presentations for initial and second-stage evaluations were created using **Microsoft PowerPoint**, including the Model 1 demonstration and the Scrum based presentation.

All project documentation, including the proposal and final report, was completed using **Microsoft Word**, which helped us format and organize the content clearly. Team members contributed collaboratively and combined their writing into a single well-organized document.

To maintain clarity and creativity in interface design, we used **Figma**, a vector based design tool ideal for web and application UI development. We also used **Draw.io** for browser based diagramming. This tool's library of shapes and symbols allowed us to map out system architecture visually. Our system structure and logic were documented using **StarUML**, a popular modeling tool for creating standard UML diagrams.

For modeling our MongoDB database, we incorporated **Hackolade** to visually represent collections and relationships.

For task scheduling and project tracking, we made use of **Microsoft Project (MS Project)**, which helped us manage deadlines and deliverables effectively.

Our core programming was done in **JavaScript**, executed in the **Node.js** environment, which leverages the V8 JavaScript engine. **Node.js** was chosen for its ability to build scalable, high-performance web apps. The system was developed using **Visual Studio Code**, with **Sublime Text** used as a lightweight alternative editor.

On the backend, **Express.js** enabled us to rapidly develop APIs, while **React.js** on the frontend allowed us to create responsive and interactive single-page interfaces. These tools, together with **MongoDB**, formed the backend and frontend foundation of our MERN stack system.

6.2 Requirements Engineering Methods

- **Stakeholder Interviews**: We gathered requirements by speaking directly with users, including pet owners, store owners, and veterinary staff.
- **Observation of Existing Workflows**: The current manual processes were analyzed to identify inefficiencies and determine how automation could improve them.
- **Use Case Modeling**: We developed Use case diagrams to map out how different user roles such as pet owners, receptionist, veterinarian, and admins interact with the system.

6.3 Design Methods

- **UML Diagrams**: We used tools like **Draw.io** to design UML models including activity diagrams, sequence diagrams, and class diagrams to represent system logic and user flows.
- **Wireframing**: Using **Figma** we sketched out wireframes before development to visualize screen layouts and enhance usability.

6.4 Development Tools and Technologies

1. Frontend Development

- React.js was used as the primary frontend framework for building dynamic and component-based UIs.
- HTML5 and CSS3 were used to structure and style pages.
- JavaScript (ES6+) added interactivity and functionality to the frontend.
- CSS Frameworks:
  - Tailwind CSS enabled utility-first styling for custom, mobile-friendly designs.

2. Backend Development

- Node.js powered the backend using an event-driven, non-blocking I/O model.
- Express.js was used as the web application framework to build RESTful APIs and handle server-side logic.

3. Frameworks & Libraries

- Express.js - Backend framework for handling routes and middleware.
- Mongoose - ODM (Object Data Modeling) library used to interact with MongoDB.
- Axios - For making HTTP requests from the frontend to the backend.

4. Database

- MongoDB was used as the NoSQL database for flexible, JSON-like document storage.
- Hosted via MongoDB Atlas for cloud scalability, security, and backups.

5. Version Control

- Git was used for code versioning and source control.
- GitHub supported team collaboration, code reviews, and repository management.

6. Development Environment

- Visual Studio Code was the primary IDE for coding, debugging, and extension based enhancements.
- Postman was used for testing and verifying REST API endpoints.

7. Hosting & Deployment

- MongoDB Atlas hosted the cloud database.
- Render or Vercel/Netlify for frontend deployment.
- Node.js server deployed using Render, Railway, or similar platforms.

6.5 Testing Methods

- Functionality Testing
- Usability Testing
- Web UI Testing
- Compatibility Testing
- Performance Testing
- Security Testing

6.6 Integration Methods

- **Presentation Level Integration** - Frontend and UI integration.
- **Business Process Integration** - Connecting modules like appointment, treatment, and payment.
- **Data Integration** - Synchronizing and managing information across systems.
- **Communications Level Integration** - Ensuring smooth data transfer through APIs.

6.7 Alternatives and Justifications

- We considered using Adobe XD, but selected Figma and wireframes due to our team's experience and the tool's efficiency.
- While Django was an option for backend development, we opted for Node.js with Express.js for better compatibility with React and easier API integration.
- The Agile methodology was chosen for its flexible, iterative nature, which allowed us to make improvements throughout the project based on user feedback.

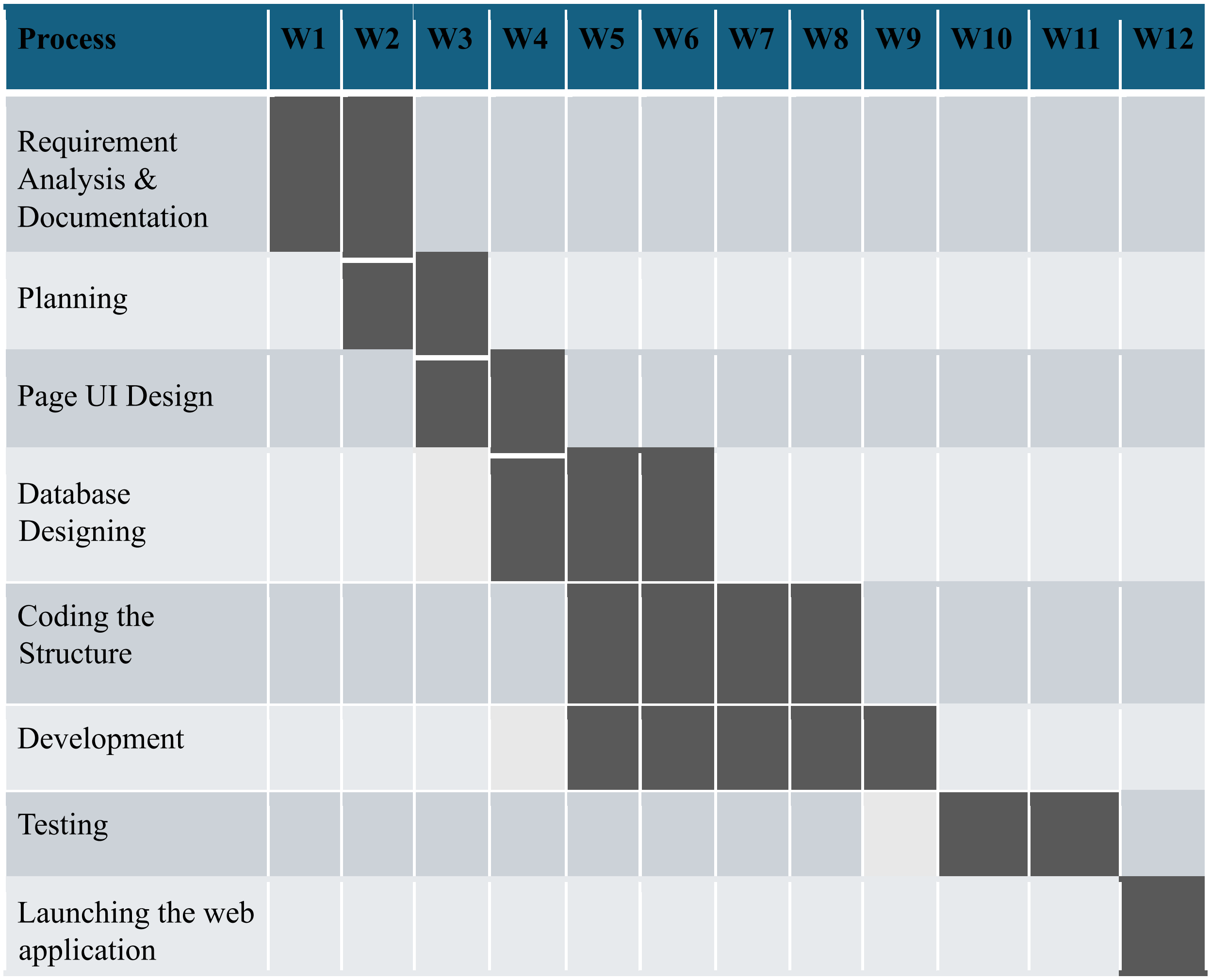
6.8 Work breakdown Structure

	Student ID & Name with initials	Brief Description of the Function
01	**Gamlath K. G. V. K. D (IT23640948)**	**Appointment management** Allow users (receptionists or pet owners) to schedule, view, reschedule, or cancel appointments with veterinarians. • **Create**: Book a new appointment. • **Read**: View upcoming or past appointments. • **Update**: Change appointment time, doctor, or date. • **Delete**: Cancel an appointment.
02	**Kavindu J. M. R (IT23631106)**	**User management** This module handles the registration, login, and management of system users including pet owners and staff (veterinarians, receptionists, nurses). • **Create**: Register new pet owners or staff. • **Read**: View user profiles and roles. • **Update**: Modify details like name, contact, role, or password. • **Delete**: Remove users who are no longer active (e.g., resigned staff).
03	**Abeysinghe A. M. B. N (IT23631274)**	**Payment management** Handles all secure financial transactions related to **appointments, treatments, prescriptions**, and **pet product orders**. • **Create**: Pet owners can securely add a new card (cardholder name, number, expiry, CVV) • **Read**: view card details. • **Update**: Pet owners can securely update their saved card details • **Delete**: Pet owners can delete any saved card at any time.

04	**Wanasinghe W. M. D. T (IT23594722)**	**Medical Record management** Stores and manages all medical records of pets including prescriptions, diagnosis history, lab reports, and treatment images (e.g., scans, X-rays). • **Create**: Add new medical records and prescriptions. • **Read**: View patient’s full medical and treatment history. • **Update**: Modify or append diagnosis, notes, or test results. • **Delete**: Remove incorrect or outdated records.
05	**Rathnayake W. P. D. D. W (IT23413474)**	**Pet Product management** Handles the management of pet related products such as food, accessories, and medicines available for sale in the clinic. • **Create**: Add new products to the catalog. • **Read**: View/search products in the catalog. • **Update**: Edit product details name, stock, or price. • **Delete**: Remove discontinued or expired products.

7.Evaluation Method

7.1 Project plan (Gantt chart)



Process	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12
Requirement Analysis & Documentation	■	■										
Planning		■	■									
Page UI Design			■	■								
Database Designing				■	■	■						
Coding the Structure					■	■	■	■				
Development					■	■	■	■	■			
Testing										■	■	
Launching the web application												■

Figure - Grantt Chart

7.2 Project Timeline - PetIQ

We started gathering information for PetIQ from the beginning of the semester. It took us approximately two weeks from the start of the semester to complete the requirement analysis and documentation phase. We were able to gather all the information required within this time, setting a solid base for our project without any major problems.

In the second week, we focused on planning the project, defining milestones, and structuring our development process. In the third week, we began designing the UI/UX with Figma, ensuring that our platform is user-friendly and aligns with the major functionalities of PetIQ.

During the fourth week, we were working on the database design phase that took approximately three weeks. While doing this, we felt the need to implement additional features to make it more effective. We were able to create the basic database layers, but additional complexity in the second layer is required to achieve optimum data handling for user roles, transactions, and tracking waste.

Starting from the fifth week, we will begin building the platform, aiming for both backend and frontend development. This will consume approximately six weeks, and during this time, all the features required, such as user authentication, waste collection scheduling, and admin features, will be successfully developed.

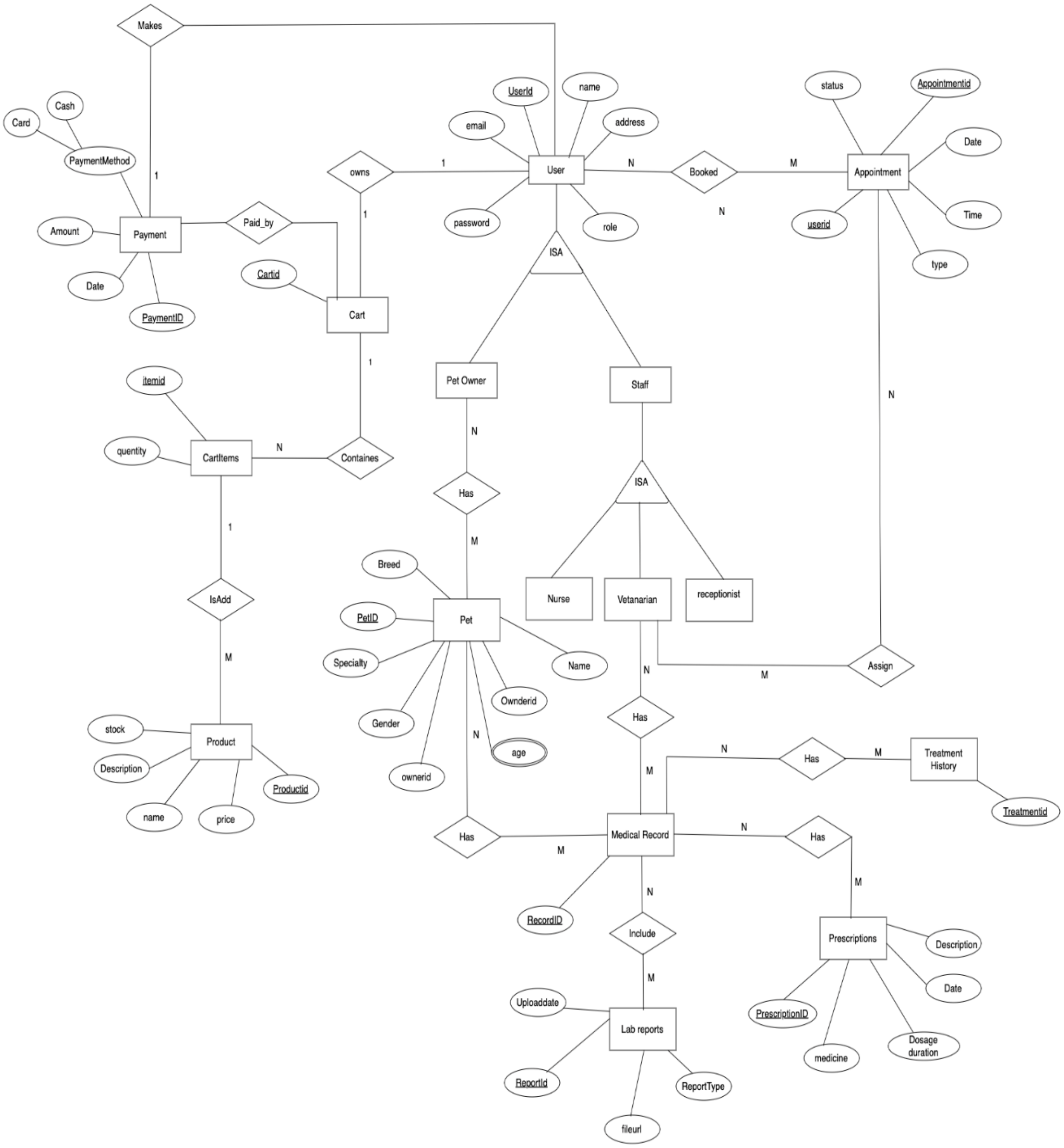
Going into the ninth week, we will intensify our development process, refine the code and integrate the various tools and frameworks required for PetIQ We will focus on optimizing performance, security, and usability without compromising the smooth user experience.

From week twelve, we will enter the testing phase, during which we will conduct unit testing, integration testing, and user acceptance testing to verify the functionality and performance of the platform. With the end of the testing phase, PetIQ will live as a complete web application, and our project will be successfully concluded.

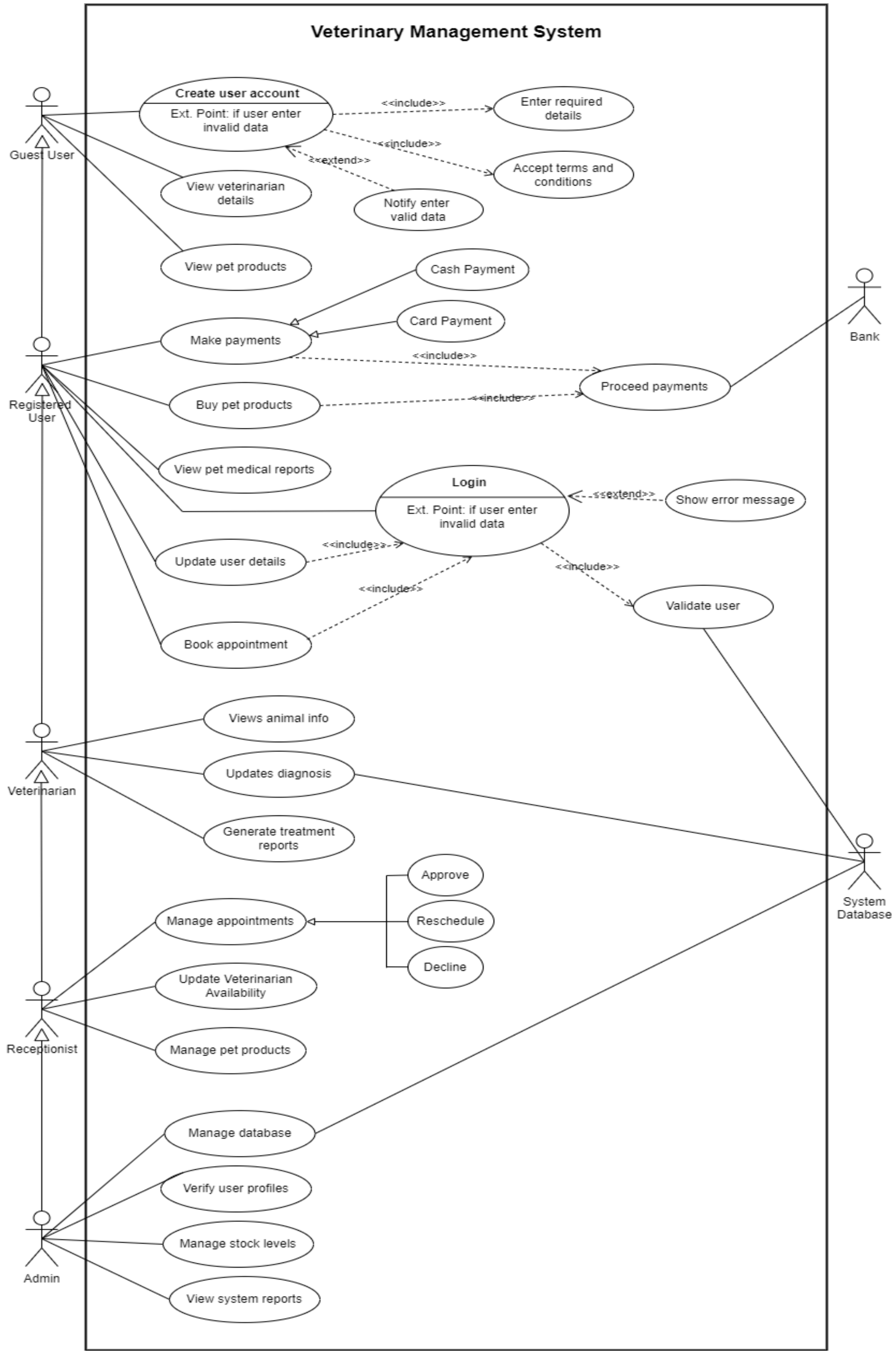
This structured timeline ensures that each component of the platform requirement gathering, UI/UX designing, database structuring, development, and testing will be executed in a proper way, resulting in a high-quality and user-friendly web solution for recycling and waste management.

8.Appendix

8.a VMS-ER Diagram



8.b Use Case Diagram



9.References

- https://www.avma.org
- https://stripe.com/docs/security/pci-compliance
- https://docs.oracle.com/javaee/
- https://cheatsheetseries.owasp.org
- https://www.who.int
- https://developer.mozilla.org/
- https://www.pcisecuritystandards.org
- https://stackoverflow.com
- https://m.mu.edu.sa/sites/default/files/content/2020/09/final%20report%20Veterinary-RE.pdf
- https://www.geeksforgeeks.org/mern-stack/
- https://www.figma.com/