Software Requirements Specification

for

<VetCare Online Veterinary Clinic Management System >

Version 2.0.1 approved

Prepared by <Shreyas Shah, Abdullah Abdosh, Idris Aklan, Farahan Wazer, Yifan Shen, Mohammed Ahtesh>

<VetCare>

<16/08/2024>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Shreyas Shah	16/08/2024	Completion of SRS document introduction, non-functional requirements and other requirements	1.0
Shreyas Shah	21/08/2024	Completion of SRS document functional requirements	1.1.0

Abdullah Abdosh	23/08/2024	Completion of architectural diagram and landing page user interface design	1.2.0
Shreyas Shah	23/08/2024	Completion of Overall Description section	1.3.0
Idris Aklan	24/08/2024	Completion of Medical Records Access interface design	1.4.0
Farahan Wazer	24/08/2024	Completion of Prescription Management interface design	1.5.0
Yifan Shen	24/08/2024	Completion of Educational Resources interface design	1.6.0
Mohammed Ahtesh	24/08/2024	Completion of Appointment Scheduling interface design	1.7.0
Shreyas Shah, Mohammed Ahtesh, Idris Aklan,Farahan	22/09/2024	Add all changes/updates made for Sprint 1/Milestone 2 to the SRS document and the project's design choices	1.8.0
Shreyas Shah, Idris Aklan, Mohammed Ahtesh	04/10/2024	Add all the changes/updates made for Sprint 2/Milestone 3 to the SRS document	1.9.0
Farahan	04/10/2024	Completion of Vet Dashboard Management Interface	2.0
Farahan	11/10/2024	Update Version of the Prescription Page	2.0.1

1. Introduction

1.1 Purpose

This SRS document specifies the software requirements for the VetCare Online Veterinary Clinic Management System, version 1.8.0. This SRS covers the full scope of the VetCare system, detailing all of its functional and non-functional requirements that will be used to develop the platform. VetCare provides pet owners with a convenient and efficient way of managing their pets' healthcare needs, including the ability to schedule appointments, access medical records, refill prescriptions and discover educational resources.

1.2 Document Conventions

This SRS document follows standard conventions that ensure precision and clarity. Subheadings are displayed in a different colour and font from the text to distinguish between both elements. All of the sections within this document follow the same standard conventions to maintain uniformity throughout.

1.3 Intended Audience and Reading Suggestions

This document is intended for the following users:

Developers: To understand the technical aspects required for the system.

Project Managers: To plan the development cycle, ensuring everything runs smoothly.

Marketing Staff: To note trending features of similar products to ensure it meets the users needs.

Testers: To create and execute test cases based on the requirements outlined in this document.

Documentation Writers: To prepare user documentation that aligns with the system's features.

It is recommended that readers start from sections 1 and 2 to gain context of the project's scope and it's capabilities. Developers are recommended to focus on sections 3, 4 and 5 which details all of the functional, non-functional and any other requirements for this product. Project Managers and Marketing Staff are recommended to focus on sections 6 and 7 which outline the architecture and design of the product.

1.4 Product Scope

The VetCare platform is an innovative platform designed to enhance the process of veterinary care for pet owners by seamlessly incorporating appointment scheduling, access to medical records, prescription management and education resources into a single, user-friendly platform. The main goal is to develop a solution that enhances the quality and accessibility of veterinary care through the use of modern-day technology. VetCare always take their customers as their number one priority and are determined to improve customer satisfaction by striving to release a platform that serves as a central community for all pet owners, veterinarians and external services.

1.5 References

<Attach Links To Documents Here>

2. Overall Description

2.1 Product Perspective

The VetCare platform is a new product, designed to provide pet owners with sophisticated online veterinary services. It is not apart of an existing product family, nor is it a replacement of any existing system. It integrates with external systems, however, including payment services such as PayPal. The primary goal of the platform is to create a user-friendly environment that offers convenience and accessibility to pet owners so that they can manage the healthcare of their pets from any location.

<Please see **Section 6** in this document for a diagram showing the major components of the overall system>

User Interface: Web-Based Interface (ReactJS, CSS/Bootstrap, Java)

Business Logic: Appointment Scheduling, Medical Records Access, Prescription Management, Educational Resources, Profile Management, Login/SignUp

Data Storage: SQLite

External Services: Payment Service

2.2 Product Functions

The VetCare platform offers various key functionalities to its users. All of these functions are used to create the best experience for managing pet healthcare:

- **User Management:** Registration, Login, Profile Management, including CRUD operations for user details.
- **Pet Profile Management:** Ability to create, read, update and delete (CRUD) pet details, accessing pet medical records history
- **Appointment Scheduling:** Booking, rescheduling and cancelling appointments
- **Medical Records Access:** Viewing, downloading, adding, editing and sharing medical records of pets
- **Prescription Management:** Requesting prescription refills and accessing detailed information about pet's medications and dosage instructions
- **Educational Resources:** Accessing articles, videos and guides related to pet healthcare
- **Payment Service**: Secure payments for appointment booking and products, with a way to view transaction history for a pet owner.
- **Login/SignUp Functionality:** Both pet owners and vets will be able to login/signup to VetCare.

2.3 User Classes and Characteristics

The VetCare platform will be utilised by various different user classes, each with unique features:

Pet Owners:

- Characteristics: Varying levels of technical experience, most frequent users on the platform, primary objective is to cater their needs of effective healthcare for their pets.
- Functions: Registration, Login, Appointment Scheduling, Medical Records Access, Prescription Management, Viewing Educational Resources, Payments

Veterinarians:

- Characteristics: Usually moderate to high technical experience, primary objective is to manage and attend appointments, view pet medical records and approve prescriptions
- Functions: Access to and managing pet medical records and prescriptions, appointment management

System Moderators:

- Characteristics: High technical experience, responsible for maintaining the platform, the users and the security of the platform.

- Functions: All moderator functions, including being able to flag inappropriate content, suspend user accounts for inappropriate behaviour on the platform, delete irrelevant or inappropriate articles, videos and guides on the educational resources section

Administrators:

- Characteristics: High technical experience, responsible for managing the operations of the platform
- Functions: All administrative functions, including managing user accounts by manually deleting inactive user accounts and suspending user accounts, managing appointments by manually adding, updating and deleting appointments, viewing, updating and deleting pet medical records manually, generating various reports including how many users were active on the platform today, the uptime and downtime of the platform's server each month and the how many payments were made and how many products were sold during a week

Importance: Pet Owners are the most significant user class to satisfy as they are the primary users on the VetCare platform.

2.4 Operating Environment

The VetCare platform will operate in the following environment:

Hardware Platform:

- Server: Cloud-based server with standard server configurations
- Client: Any device that can run a modern web browser including laptops, computer, tablets and smartphones.

Operating System:

- Any operating system capable of running a modern web browser including Windows, macOS, iOS, Android and Linux.

Web Browsers:

- Compatible with the latest versions of modern web browsers including Google Chrome, Firefox, Safari, Microsoft Edge.

Software Components:

- Frontend: React.js, HTML, CSS/Bootstrap
- Backend: Spring Boot framework with Java
- Database: SQLite

2.5 Design and Implementation Constraints

Several constraints will assist in the design and implementation of the VetCare platform:

- **Performance:** The system must support up to 1,000 concurrent users on the platform with minimal latency.

- **Integration with external APIs:** The platform must integrate with third-party services such as payment services for appointment bookings and products.
- **Legal Compliance:** The platform must comply with standard data and privacy protection laws such as GDPR (General Data Protection Regulation) and The Health Records Act 2001 (Vic).
- **Time:** The platform must be fully delivered by the set deadline and major milestones should not be delayed what so ever.
- **Technology:** The use of the Spring Boot framework with Java, React.js for the frontend and SQLite for the database is compulsory.
- **Language:** The platform's default language must be English, with the option to translate to other languages, including French, Spanish and Chinese.
- **Security:** The platform must use TLS 1.3 or higher for standard data encryption and ensure Payment Card Industry (PCI) compliance for the payment services. Additionally, a hashing library (Bcrypt) must be used to hash user passwords for an additional layer of security for the users on the platform.
- **Programming Standards:** The codebase of the platform must be thoroughly commented and must follow standard coding conventional guidelines.

2.6 User Documentation

The following material will be delivered alongside the VetCare platform:

- **User Manual:** A detailed guide that showcases how to use the platform with accompanying screenshots and step-by-step processes.
- **Online Help Section:** A section integrated within the platform that provides assistance based on a specific query or concern a user might have.

The user documentation will be delivered through multiple options including PDF and web-based formats.

2.7 Assumptions and Dependencies

The development of the VetCare platform will be based on the following assumptions and dependencies:

- **Security Assumptions:** It is assumed that most of the users will follow the best security practices including enabling multi-factor authentication and using strong passwords and that the system will not be subject to critical cyberattacks and threats due to it's rigorous security measures.
- **Third-Party APIs:** It is assumed that third-party services for the payment functionality of the platform including PayPal will be available and reliable most of the time.
- **Technology Assumptions:** It is assumed that the chosen technology for this platform, including Spring Boot, React.js and SQLite, will be available and stable most of the time throughout the development and testing phases of the platform.
- **Users' Internet Access:** It is assumed that all of the users on the platform will have access to a stable internet connection since the platform requires online connectivity to perform it's functionalities.
- **Users' Device Compatibility:** It is assumed that all of the users on the platform will utilise a modern web browser running the latest version so that they all can interact

with the platform without any major performance issues. Any browsers running older versions may have degraded performance.

3. External Interface Requirements

Note: The functional requirements for the VetCare platform are integrated within this section as they relate to the system's design and interactions.

3.1 User Interfaces

The VetCare platform must be implemented with a user-friendly user interface which is designed to meet the requirements of casual users, pet owners, veterinarians, and admins/moderators. Several key characteristics will be defined here that describe the user interface.

Common Buttons and Functions: Common buttons such as 'Edit', 'Submit', 'Cancel', 'Back', 'Help' and 'Confirm' should appear on specific pages of the platform. The 'VetCare' logo at the top left of the website must navigate the user to the home page whenever clicked on and must be fixed to the top of the interface even when the user navigates between different pages of the platform. The 'Help' button should open up a small section including relevant guides and areas for online support.

Error Messages: Error messages must be displayed at the top of each relevant area in a red background, with text at the front displaying the reason/s for the error. An example would be when a user tries to login and input an incorrect email or password, then an error message will be displayed at the top of the login form with accompanying text displaying, "Incorrect email/password".

Screen Layout: The user interface must be compatible with all screen sizes and devices including computers and mobile phones. The layout must consist of a navigation bar at the top of the page for each accessibility to different pages of the platform, including appointment scheduling, prescription refills, medical records and educational resources.

<u>User Interface Components</u>

Landing Page Section: The home page of the VetCare platform must display information about the platform and a navigation bar that allows for quick access to features including appointment scheduling, viewing medical records, managing prescriptions and exploring educational resources.

Appointment Scheduling Section: Pet owners should be allowed to view upcoming appointments for their pets, book, reschedule and cancel appointments.

Medical Records Section: Pet owners should be allowed to securely view and manage their pet's medical history, vaccination records, and treatment plans. Also, records can be downloaded and shared with veterinary professionals.

Prescription Management Section: Pet owners should be allowed to request prescription refills and access detailed information about their pet's medications and dosage instructions.

Educational Resources Section: Users must be allowed to explore a range of articles, videos and guides on pet healthcare and wellness and can keep them informed about the latest trends and the best practices in veterinary medicine.

Payment Section: The checkout process must allow users to pay for their appointment bookings and add products and then select a payment method from either credit/debit card or PayPal and complete the transaction securely. Pet Owners can also view their transaction history from their profile page.

Login/SignUp Section: A dedicated page for pet owners and veterinarians to login/signup to VetCare, where they can choose their role and a user interface tailored to that specific rule will be available for them once logging in or signing up.

Profile Management Section: A profile page for both pet owners and veterinarians where they can view their profile details, edit their profile, delete their profile and for pet owners, add their pet profiles, which will be used for other features on the VetCare platform.

3.2 Hardware Interfaces

The VetCare platform is primarily a web-based application and only interacts with standard client devices including computers, tablets and smartphones. The following hardware components are related to the platform's functionality with just standard client devices.

Client Devices: The platform must only be functional on standard client devices including computers, tablets and smartphones and must support all of these devices by incorporating unique interfaces depending on the device. For instance, a mobile phone's navigation bar should be displayed as a drop down where each tab will be shown when the user taps on the side bar icon. Additionally, the software should be compatible with standard input devices such as keyboards, mice and touchscreens.

Server Hardware: The platform must be compatible with the host's hardware specifications, including CPU, memory and storage so that on the backend, the platform can manage data processing, storage and communications. Additionally, the hardware should account for higher user loads on the platform and should recover from any server failures within a short period.

Peripheral Devices: The platform should be compatible with peripheral devices including printers which may be required to print out medical records or prescriptions so that a hard copy can be sent to veterinarians and pharmacies.

Communication Protocols: The platform must utilise standard communication protocols including Wi-Fi and Ethernet for network connectivity.

3.3 Software Interfaces

The VetCare platform will account for several software interface requirements so that it delivers all of the intended requirements listed in this document.

Web Browsers: The VetCare platform should be compatible with a range of the latest versions of browsers, including Google Chrome, Firefox, Safari and Microsoft Edge. Regular testing must be conducted to ensure all compatible browsers are functional with the platform.

Operating Systems: The platform must be compatible with multiple operating systems including Windows, macOS, iOS, Android and Linux. Regular testing must be conducted to ensure these operating systems always function correctly with the platform and to account for chances of operating system failures.

APIs: The platform should integrate external APIs such as payment handling for the purchase of medication, prescriptions and for booking appointments. Additionally, educational resources about pet health care and treatment must be fetched through external APIs and displayed on the platform's educational resources section.

Database: The platform must utilise a relational database to store user information, medical records, appointment details and product information. The preferred database is SQLite, where SQL queries will be used to store, fetch and update data.

Libraries and Tools: The platform should utilise the Spring Boot framework with Java for development and the Bootstrap library for styling the user interface which ensures a clean and professional look for the platform.

Software Components:

- **Authentication:** Module that communications with the backend to manage the signup and login functionalities and user sessions.
- **Payments:** Communicates with third-party services to manage payments of appointment bookings and products in a secure manner.

3.4 Communications Interfaces

The VetCare platform will utilise multiple communication interfaces for interactions with third-party and external systems.

Internet Protocols: The platform must utilise HTTP for all communications through the web. HTTPS should be utilised to ensure payments are being made securely and for data transmissions via encryption.

Security Protocols: All communications made on the platform must be encrypted with standard TLS encryption.

Network Server Communications: The platform's backend should communicate with the database using standard TCP protocols.

Data Transfer Rates: The platform must have a steady data transfer rate that will support downloads and user inputs, ensuring a smooth experience is given to all users.

Synchronisation: The platform must implement an automated synchronisation mechanism to sync data including pet medical records between the platform and the database.

4. Nonfunctional Requirements

4.1 Performance Requirements

The VetCare platform should be able to handle 1,000 concurrent users with minimal latency. The platform must have an uptime of 99.9% to ensure high availability for all users. For real time features, including virtual appointments and appointment scheduling, response times must be immediate and not exceed 1000 milliseconds unless due to the event of server issues. The navigation between different pages of the platform should have a minimal response delay and must have the pages loaded within 3 seconds under ordinary conditions. Regular testing of the performance must be conducted to ensure the performance requirements defined are regularly met.

4.2 Safety Requirements

To ensure the safety of the VetCare platform's data, regular backups will be administered not just at a singular area but multiple scattered areas to ensure no data is lost possibly due to power outages, natural disasters and physical intrusions of devices that monitor the platform. Safeguards will be integrated within the VetCare platform to ensure data confidentiality and integrity during transactions for prescription refills and appointment scheduling. All user and pet information will be kept private and secure within the system through the use of authentication and any modifications made to important documentation or events by users such as medical records and virtual emergency appointments will be logged to ensure unauthorised changes are prevented.

4.3 Security Requirements

Security is a paramount consideration for the VetCare platform. Users will be prompted to turn on the multi-factor authentication setting but are not forced to do so. This is done for the security of all users, ensuring unauthorised access is prevented. Additionally, all data that is communicated between the platform's frontend and the server will be encrypted using the latest version of standard TLS (Transport Layer Security). In terms of access to specific parts of the platform, highly confidential information such as medical records of pets and pet details will be restricted based on the level of authority a user group has. For instance, both pet owners and veterinarians are the only user groups that can view medical records and prescriptions of pets. Standard data protection and privacy laws will be followed including the GDPR (General Data Protection Regulation) and The Health Records Act 2001 (Vic) to ensure standard industry legislation is complied with. PCI (Payment Card Industry) compliance will be followed for the payment services, used for appointment bookings and product purchases. The Bcrypt library will be used for hashing the passwords of the users before being stored in the database.

4.4 Software Quality Attributes

Interoperability: The VetCare platform must integrate third-party services such as additional veterinary clinic information with minimal issues.

Maintainability: The VetCare platform's code should be thoroughly commented and documented in the case of future updates, testing or fixing bugs.

Reliability: The VetCare platform should perform its intended functionalities with minimal issues.

Scalability: The VetCare platform must be created in a way such that it can handle an increase in user activity without affecting the performance of the platform.

Usability: The VetCare platform's user interface must be designed in a way that is easy to navigate and has well-known design elements including symbols, colours, buttons and links to different pages so that it account for the diversity of users on the platform.

4.5 Business Rules

The VetCare platform will enforce a list of operating principles including:

- Users must provide valid payment information to book appointments or refill prescriptions.
- Only registered users/pet owners and veterinarians can access medical records and prescriptions.
- Only registered users/pet owners can request a refill of a prescription.
- Only registered users/pet owners can book appointments and have consultations with a verified veterinarian/s.
- Only registered users/pet owners have a profile page.
- Only pet owners can view their transaction history.
- Veterinarians must provide users/pet owners with the specific prescription/s and must also approve prescription refills before it is sent to the pharmacy.
- Any content that is available in the educational resources section can be accessible to anyone including unregistered users.

5. Other Requirements

Internationalisation Requirement: The VetCare platform should support multiple languages including English, French, Spanish and Chinese.

Database Requirement: The VetCare platform must utilise a relational database to store user data, appointment details, medical records, prescriptions and transaction data.

Legal Requirement: The VetCare platform must comply with local health, user privacy and veterinary laws and regulations.

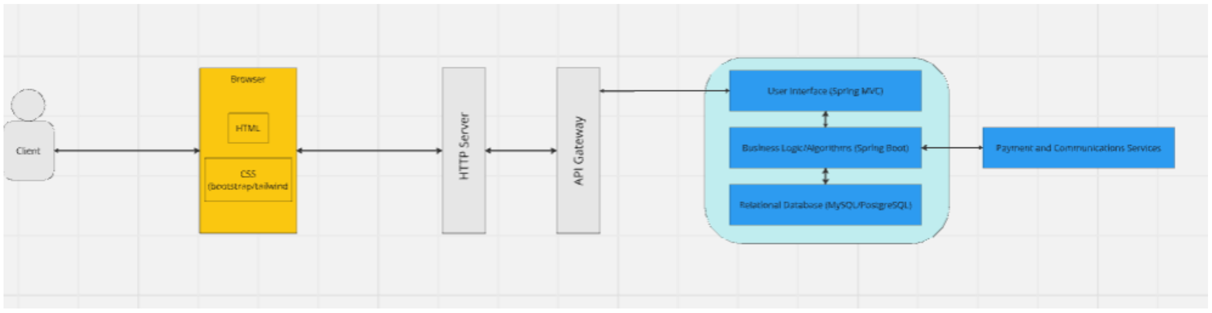
Reuse Objective: Third-party services integration, user authentication and encryption protocols should be reused for future updates to the VetCare platform or projects related to the platform.

6. System Architecture

Summary

The architecture of VetCare, a comprehensive online vet clinic management system, is designed using Java and Spring Boot to provide a robust, scalable, and maintainable platform. The architecture aims to ensure high availability, security, and performance while offering seamless integration with external veterinary clinic and store databases. Modular design principles and RESTful API integration form the core of the system's design to support extensibility and ease of future enhancements.

6.1 Architecture Overview


Client
Browser
HTML
CSS (bootstrap/tailwind
HTTP Server
API Gateway
User Interface (Spring MVC)
Business Logic/Algorithms (Spring Boot)
Relational Database (MySQL/PostgreSQL)
Payment and Communications Services

6.2 Architectural Decisions

1. Use of Java and Spring Boot

- Decision: The system is developed using Java and Spring Boot.

- Rationale: Java provides robustness and portability across platforms, which is essential for the backend services of VetCare. Spring Boot facilitates rapid development, easy deployment, and out-of-the-box configurations which are crucial for managing diverse modules of VetCare.

2. Microservices Architecture

- Decision: Opting for a microservices architecture over a monolithic approach.

- Rationale: This approach allows individual service scalability and easier maintenance. It also enhances the system's resilience by isolating failures to individual services rather than affecting the entire application.

3. Database Selection

- Decision: Use of SQLite for transactional data and NoSQL databases (e.g., MongoDB) for unstructured data like medical records and educational content.

- Rationale: SQLite ensures data integrity and facilitate complex queries with relationships. NoSQL databases offer flexibility and better performance for large volumes of data that do not require complex relationships.

4.Security Implementation

- Decision: Implement comprehensive security measures using Spring Security.
- Rationale: Ensuring data protection and secure access are paramount. Spring Security provides robust authentication and authorization mechanisms that are critical for protecting sensitive medical and personal information.

5.API Gateway

- Decision: Implementation of an API Gateway to manage, authenticate, and route API requests.
- Rationale: An API Gateway simplifies the client interface, provides security at the entry point, and allows easier monitoring and management of traffic between clients and services.

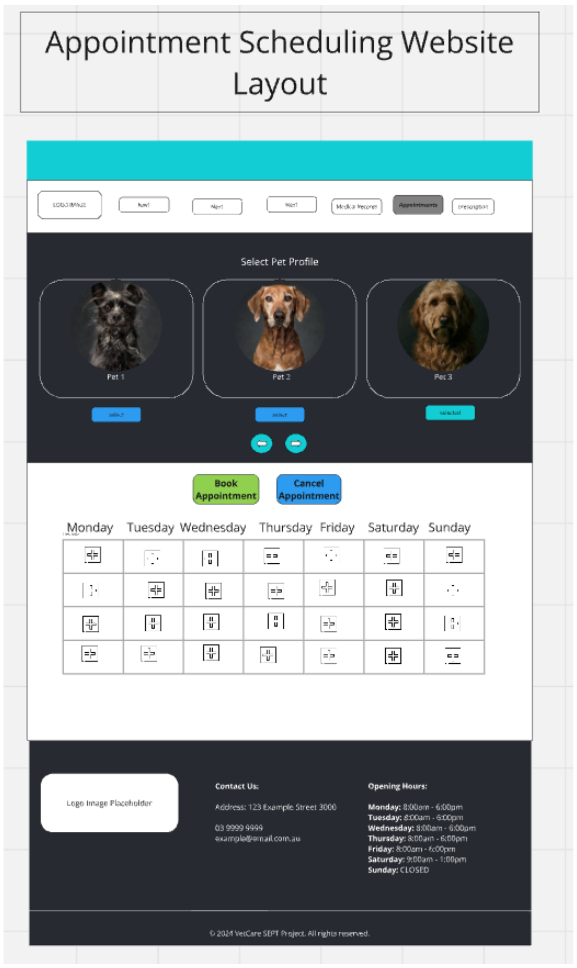
6. Client-Side Technology

- Decision: Development of a responsive client-side application using Angular.
- Rationale: Angular provides a structured framework for developing dynamic and responsive web applications, which enhances user experience and accessibility across different devices and platforms.

These decisions are aimed at creating a reliable, scalable, and user-friendly platform that meets the current and future needs of pet owners and vet clinics. The architecture supports the primary objectives of the VetCare system, focusing on usability, security, and comprehensive care facilitation.

7. User Interface Design

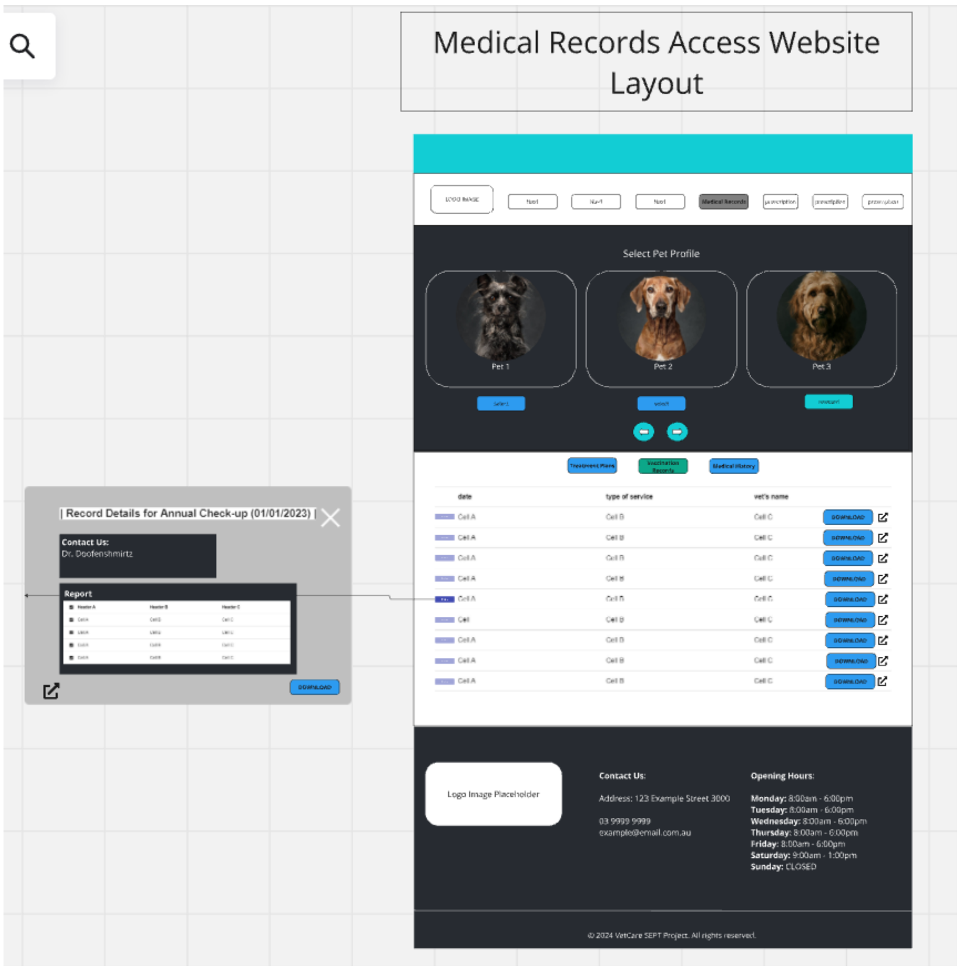
1. Appointment Scheduling: Easily book, reschedule, or cancel appointments with just a few clicks.



2. Medical Records Access:

a. Securely view and manage the pet's medical history, vaccination records, and treatment plans.

b. Download and share records with other veterinary professionals as needed.



date	type of service	vet's name
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C

| Record Details for Annual Check-up (01/01/2023) |

Contact Us:
Dr. Doofenshmirtz

Report

Header A	Header B	Header C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C
Cell A	Cell B	Cell C

Contact Us:

Address: 123 Example Street 3000

03 9999 9999
example@email.com.au

Opening Hours:

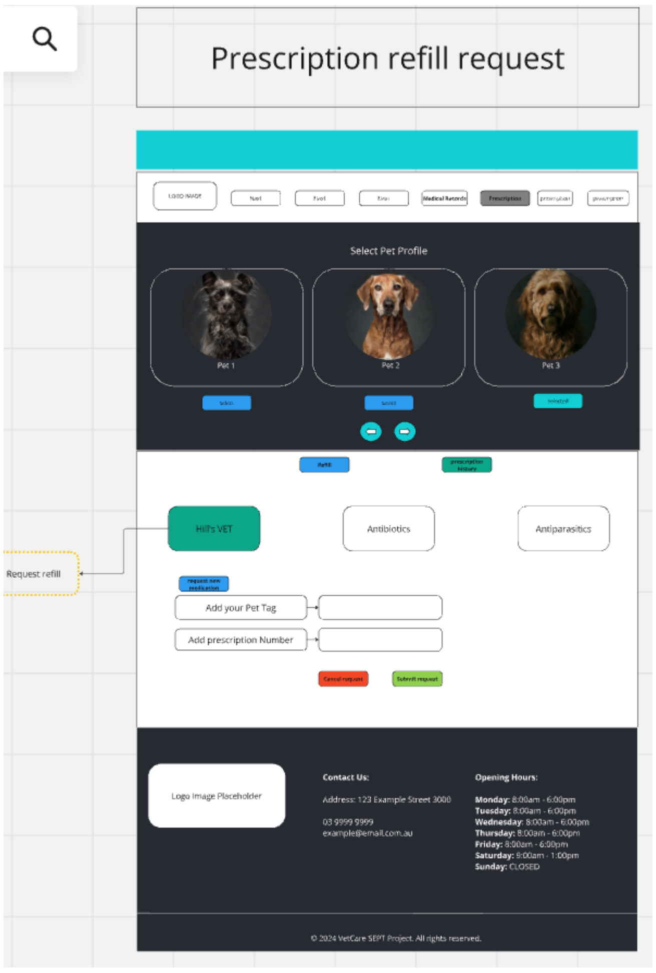
Monday: 8:00am - 6:00pm
Tuesday: 8:00am - 6:00pm
Wednesday: 8:00am - 6:00pm
Thursday: 8:00am - 6:00pm
Friday: 8:00am - 6:00pm
Saturday: 9:00am - 1:00pm
Sunday: CLOSED

© 2024 VetCare SEPT Project. All rights reserved.

3. Prescription Management:

a. Request prescription refills and have them delivered to your doorstep.

b. Access detailed information about your pet’s medications and dosage instructions.



Contact Us:

Address: 123 Example Street 3000

03 9999 9999
example@email.com.au

Opening Hours:

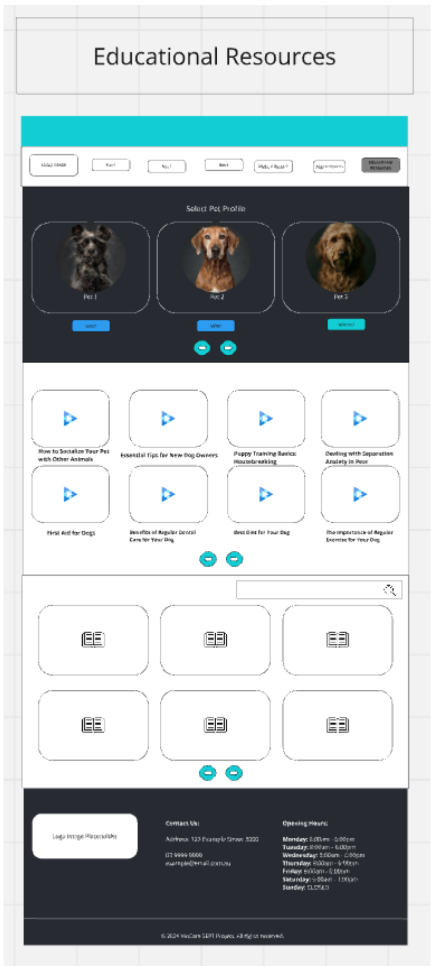
Monday: 8:00am - 6:00pm
Tuesday: 8:00am - 6:00pm
Wednesday: 8:00am - 6:00pm
Thursday: 8:00am - 6:00pm
Friday: 8:00am - 6:00pm
Saturday: 8:00am - 1:00pm
Sunday: CLOSED

© 2024 VetCare SEPT Project. All rights reserved.

4. Educational Resources:

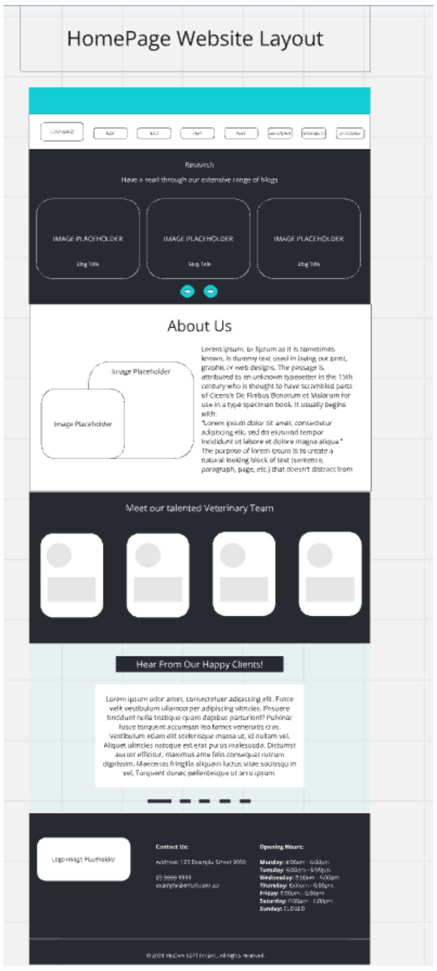
a. Explore our extensive library of articles, videos, and guides on pet care and wellness.

b. Stay informed about the latest trends and best practices in veterinary medicine.



Educational Resources
Select Pet Profile
Pet 1
Pet 2
Pet 3
How to Socialize Your Pet with Other Animals
Essential Tips for New Dog Owners
Puppy Training Basics: Housebreaking
Dealing with Separation Anxiety in Pets
First Aid for Dogs
Benefits of Regular Dental Care for Your Dog
Best Diet for Your Dog
The Importance of Regular Exercise for Your Dog
Logo Image Placeholder
Contact Us:
Address: 123 Example Street 3000
03 9999 9999
example@email.com.au
Opening Hours:
Monday: 8:00am - 6:00pm
Tuesday: 8:00am - 6:00pm
Wednesday: 8:00am - 6:00pm
Thursday: 8:00am - 6:00pm
Friday: 8:00am - 6:00pm
Saturday: 9:00am - 1:00pm
Sunday: CLOSED
© 2024 VetCare SEPT Project. All rights reserved.

5. User-Friendly Interface: Our intuitive and easy-to-navigate platform ensures a seamless experience for all users.



Lorem ipsum, or lipsum as it is sometimes known, is dummy text used in laying out print, graphic or web designs. The passage is attributed to an unknown typesetter in the 15th century who is thought to have scrambled parts of Cicero's De Finibus Bonorum et Malorum for use in a type specimen book. It usually begins with:
"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua."
The purpose of lorem ipsum is to create a natural looking block of text (sentence, paragraph, page, etc.) that doesn't distract from

Lorem ipsum odor amet, consectetuer adipiscing elit. Fusce velit vestibulum ullamcorper adipiscing ultricies. Posuere tincidunt nulla tristique quam dapibus parturient? Pulvinar fusce torquent accumsan leo fames venenatis cras. Vestibulum etiam elit scelerisque massa ut, id nullam vel. Aliquet ultricies natoque est erat purus malesuada. Dictumst auctor efficitur, maximus ante felis consequat rutrum dignissim. Maecenas fringilla aliquam luctus vitae sociosqu in vel. Torquent donec pellentesque ut arcu ipsum.

Appendix A: Glossary

Glossary for the VetCare SRS Document

API (Application Programming Interface)

A set of protocols and tools for building software applications, specifying how software components should interact. APIs are used to allow an easy way for different parts of a program to communicate and are essential for integrating third-party services.

Architecture

The fundamental organization of a system embodied in its components, their relationships to each other, and the environment, and the principles guiding its design and evolution. This includes the structure of software or hardware.

Business Rules

Explicit regulations, conditions, or policies that govern how a business process operates. In software terms, these rules define or constrain some aspect of business and always assert business structure or influence the behavior of the business.

Communications Interfaces

The pathways or protocols through which data is transmitted between different systems or components within a system. This includes specifying the methods and data formats for exchanging information between devices and software.

External Interface

The defined boundaries where a system interfaces with external systems, users, or other entities outside the system. This includes hardware, software, and human interfaces.

GDPR (General Data Protection Regulation)

A regulation in EU law on data protection and privacy in the European Union and the European Economic Area. It also addresses the transfer of personal data outside the EU and EEA areas.

Microservices Architecture

A design approach to build a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. Each service is built around a specific business capability.

NoSQL Database

A type of database that provides a mechanism for storage and retrieval of data that is modeled in ways other than the tabular relations used in relational databases. These databases are often used to store large volumes of data and offer flexible schemas.

PostgreSQL

An open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance. It is known for its robustness, scalability, and support for advanced data types.

Relational Database

A database structured to recognize relations among stored items of information according to a relational model of data.

SRS (Software Requirements Specification)

A comprehensive description of the intended purpose and environment for software under development. The SRS fully describes both functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide.

Spring Boot

A project that uses the Spring platform and third-party libraries so you can get started with minimum fuss. It simplifies the bootstrapping and development of new Spring applications.

TLS (Transport Layer Security)

A cryptographic protocol that provides end-to-end security of data sent across insecure networks such as the internet. It is the basis for secure communication on the web, including HTTPS.

User Interface (UI)

The means by which the user and a computer system interact, particularly the use of input devices and software. The UI provides controls for the user to manage software functions and data.

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

Appendix Milestone 2

SRS Document Changes

- Added the following user interface components under section 3.1:

  Payment Section: *The checkout process must allow users to pay for their appointment bookings and add products and then select a payment method from either credit/debit card or PayPal and complete the transaction securely.*

  Login/SignUp Section: *A dedicated page for pet owners and veterinarians to login/signup to VetCare, where they can choose their role and a user interface tailored to that specific rule will be available for them once logging in or signing up.*

  Profile Management Section: *A profile page for both pet owners and veterinarians where they can view their profile details, edit their profile, delete their profile and for pet owners, add their pet profiles, which will be used for other features on the VetCare platform.*

- Changed the database to SQLite whenever the previously planned database to use (PostgreSQL or MySQL) occurred within the document.

New/Changed Functionality

- Used CSS and Bootstrap only for styling instead of also planning to use Tailwind as well.
- Ensured that vets use an email ending with '@vetcare.com' when signing up to the platform.
- Changed Stripe 3rd Party App for Payment Functionality to PayPal since a sandbox account was needed for the stimulation of payments without actually deducting money from a user's account.

- Shopping Cart, Digital Receipt and Transaction history all removed from features of the system due to a finding of a more efficient method for payments. The method is to have a modal for payments and summary after the user books an appointment or orders a prescription for their pet(s).
- Appointments cost $50.00 to book.
- The database used was changed from MySQL to SQLite.

Code Files and Structure

- Created two new folders, one for the frontend, named 'frontend', and the other for the backend, named 'backend'.
- Within the frontend folder, all of the React.js files are stored for all features implemented, including landing page, appointment booking page, medical records page, prescription refill page, educational resource page, user and pet profile page and the login and signup page. The additional feature implemented was payments for appointment bookings and prescription refills, having the option to choose from credit/debit card and PayPal.
- Within the backend folder, the Junit test cases were placed in the 'tests' folder within the 'main' application folder and all tests for all the features of the VetCare platform were placed within that folder. Additionally, the database used for this project is SQLite, where the database's file is stored in the backend folder.
- It was Decided that Back end functionality for the appointment module was postponed to sprint 2. This is because, it was not deemed necessary for sprint 1 deployment.

Test Files

- Acceptance Test Cases for each feature were created and the results were documented within the GitHub Project Board.
- Added "AppointmentTests.java" file and respective Junit tests within the file for the appointment scheduling module.
- Added Junit tests for login (LoginTest.java), signup (SignUpTest.java) in the 'test' folder within the 'backend folder'.

User Stories

- Removed all Epics from the GitHub Project Board.
- Added a user story for the signup functionality and one for the login functionality of the VetCare platform.
- Refined user stories to be in the expected format (as per discussion on canvas - https://rmit.instructure.com/courses/125192/discussion_topics/2525447)
- Removed User stories related to Virtual veterinary consultations, as it has been deemed not needed for the application of the appointment section.

- Removed Appointment Schedule Management for Veterinary Professionals on VetCare Platform User Story, as it was deemed not needed for the application of the appointment section.
- Added the user story "Selecting Pet Profiles" as this is a main feature in the appointment scheduling section.
- Added the User Story #18 - #22

Removed User Stories:

User Story #3: Virtual Veterinary Consultations on VetCare Platform
User Story #6: Notifications for Appointments and Prescription Refills on VetCare Platform
User Story #7: Health Maintenance Reminders on VetCare Platform
User Story #11: Service Feedback and Rating on VetCare Platform
User Story #12: Community Forum for Pet Owners on VetCare Platform
User Story #14: Clinic Profile and Service Management on VetCare Platform
User Story #15: Wellness Goal Setting and Tracking on VetCare Platform

Added User Stories:
User Story #26: Veterinarian Dashboard for Managing Daily Activities

Appendix Milestone 3

SRS Document Changes

- Added the following line into **Section 4.3 (Security Requirements)**:

The Bcrypt library will be used for hashing the passwords of the users before being stored in the database.

- Added the following line into **Section 2.5 (Design and Implementation Constraints) - Security**:

Additionally, a hashing library (Bcrypt) must be used to hash user passwords for an additional layer of security for the users on the platform.

- Added the following into **Section 2.2 Product Functions - Payment Service**:

with a way to view transaction history for a pet owner.

- Added the following into **Section 3.1 User Interfaces - Payment Component**:

Pet Owners can also view their transaction history from their profile page.

New/Changed Functionality

- Added back transaction history implementation, since it was deemed appropriate to have, considering digital receipt feature was removed. This is so that users have a way of viewing past transactions in a more efficient way through their profile, instead of having to receive a receipt every time they make a purchase. Pagination was also used so that transactions are not just on one page but rather a certain number of transactions are on a single page. Users can also search for a specific transaction based on a search term they enter.
- Ensured passwords are hashed using the spring boot version of Bcrypt before they are stored in the database for security purposes.
- Removed the requirement of vets needing their emails to end with '@vetcare.com', since the medical records need to be shared with existing email addresses in the real-world.
- Added a vet dashboard for all vets to check their upcoming appointments and prescription requests.
- Added a table of all clinics and their respective vets, vets specialties and a price to book an appointment on the appointment. This was deemed necessary to have as a user must be able to compare the pricing of an appointment at a different clinic before booking one.
- Now when users share a medical record to a vet the vet can see this on their dashboard

Code Files

- Added files all the models, controllers, services, and repositories for each feature that was implemented in the 'backend' folder.
- Added a 'TransactionHistory.js' file in the 'frontend' folder which contains the frontend code for a user's transaction history.
- Sorted all the files of the features into separate folders to improve accessibility of files and to avoid confusion on where certain files may be located.

Test Files

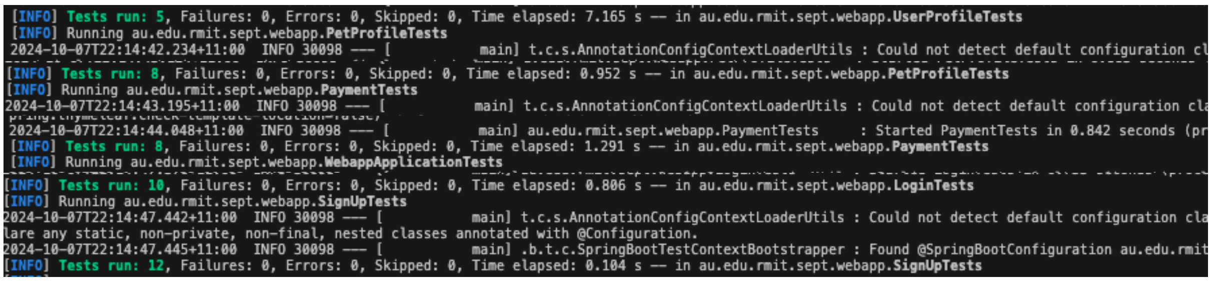
- Added the following test files: user profile tests (UserProfileTests.java), Medical Records tests (MedicalRecordTests.java), pet profile tests (PetProfileTests.java) and payment tests (PaymentTests.java) in the 'test' folder within the 'backend folder'.
- Added AppointmentTests.java test file in the 'test' folder within the 'backend folder'.

User Stories

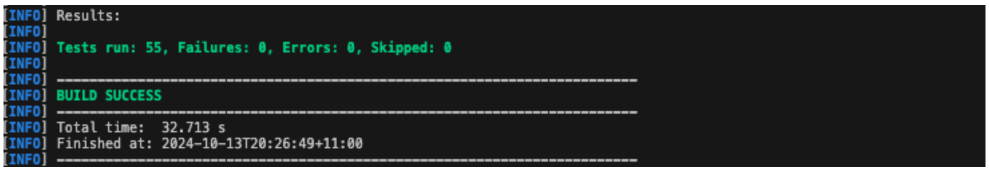
- Updated User Story #9 - Secure Online Payments on VetCare Platform with all the transaction history acceptance criteria and tasks. (Please see GitHub Project Board).
- Updated User Story #16 - Sign Up Functionality for Pet Owners and Veterinarians With Secure Password Hashing with the addition of Bcrypt for password hashing. (Please see GitHub Project Board).

- Added new User Stories - User Story #26: Medical Records Management on VetCare Platform and User Story #27: Appointment management on vet dashboard.

Evidence of Test Executions and Results

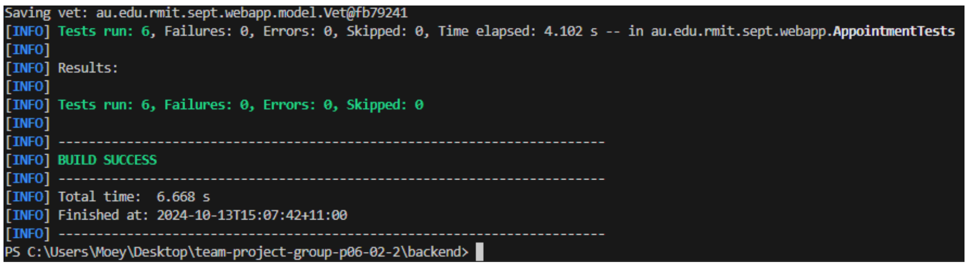


```
[INFO] Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 7.165 s -- in au.edu.rmit.sept.webapp.UserProfileTests
[INFO] Running au.edu.rmit.sept.webapp.PetProfileTests
2024-10-07T22:14:42.234+11:00  INFO 30098 --- [           main] t.c.s.AnnotationConfigContextLoaderUtils : Could not detect default configuration cl
[INFO] Tests run: 8, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.952 s -- in au.edu.rmit.sept.webapp.PetProfileTests
[INFO] Running au.edu.rmit.sept.webapp.PaymentTests
2024-10-07T22:14:43.195+11:00  INFO 30098 --- [           main] t.c.s.AnnotationConfigContextLoaderUtils : Could not detect default configuration cla
2024-10-07T22:14:44.048+11:00  INFO 30098 --- [           main] au.edu.rmit.sept.webapp.PaymentTests     : Started PaymentTests in 0.842 seconds (pr
[INFO] Tests run: 8, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.291 s -- in au.edu.rmit.sept.webapp.PaymentTests
[INFO] Running au.edu.rmit.sept.webapp.WebappApplicationTests
[INFO] Tests run: 10, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.806 s -- in au.edu.rmit.sept.webapp.LoginTests
[INFO] Running au.edu.rmit.sept.webapp.SignUpTests
2024-10-07T22:14:47.442+11:00  INFO 30098 --- [           main] t.c.s.AnnotationConfigContextLoaderUtils : Could not detect default configuration cla
lare any static, non-private, non-final, nested classes annotated with @Configuration.
2024-10-07T22:14:47.445+11:00  INFO 30098 --- [           main] .b.t.c.SpringBootTestContextBootstrapper : Found @SpringBootConfiguration au.edu.rmit
[INFO] Tests run: 12, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.104 s -- in au.edu.rmit.sept.webapp.SignUpTests
```



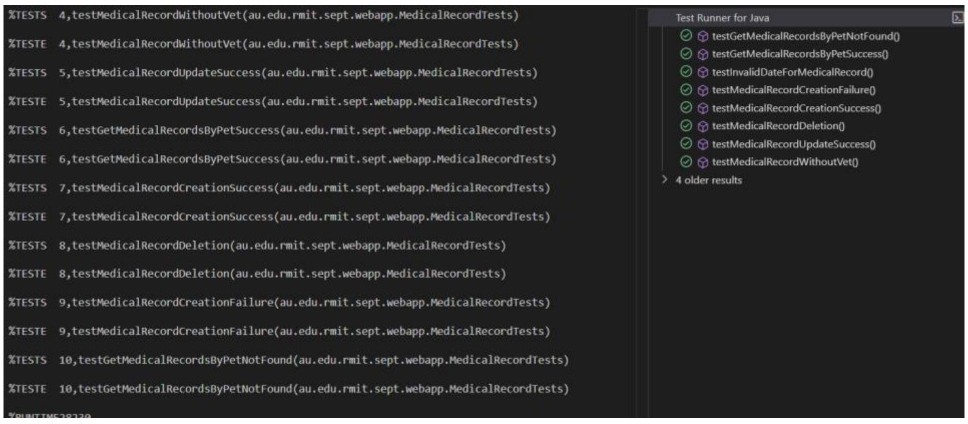
```
[INFO] Results:
[INFO]
[INFO] Tests run: 55, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  32.713 s
[INFO] Finished at: 2024-10-13T20:26:49+11:00
[INFO] ------------------------------------------------------------------------
```

Appointment Tests :



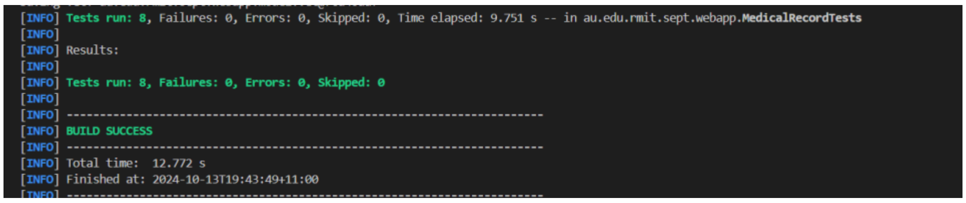
```
Saving vet: au.edu.rmit.sept.webapp.model.Vet@fb79241
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 4.102 s -- in au.edu.rmit.sept.webapp.AppointmentTests
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  6.668 s
[INFO] Finished at: 2024-10-13T15:07:42+11:00
[INFO] ------------------------------------------------------------------------
PS C:\Users\Moey\Desktop\team-project-group-p06-02-2\backend>
```

Medical Records Tests :



```
%TESTS  4,testMedicalRecordWithoutVet(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  4,testMedicalRecordWithoutVet(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTS  5,testMedicalRecordUpdateSuccess(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  5,testMedicalRecordUpdateSuccess(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTS  6,testGetMedicalRecordsByPetSuccess(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  6,testGetMedicalRecordsByPetSuccess(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTS  7,testMedicalRecordCreationSuccess(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  7,testMedicalRecordCreationSuccess(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTS  8,testMedicalRecordDeletion(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  8,testMedicalRecordDeletion(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTS  9,testMedicalRecordCreationFailure(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  9,testMedicalRecordCreationFailure(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTS  10,testGetMedicalRecordsByPetNotFound(au.edu.rmit.sept.webapp.MedicalRecordTests)

%TESTE  10,testGetMedicalRecordsByPetNotFound(au.edu.rmit.sept.webapp.MedicalRecordTests)
```

```
Test Runner for Java
  ✓ testGetMedicalRecordsByPetNotFound()
  ✓ testGetMedicalRecordsByPetSuccess()
  ✓ testInvalidDateForMedicalRecord()
  ✓ testMedicalRecordCreationFailure()
  ✓ testMedicalRecordCreationSuccess()
  ✓ testMedicalRecordDeletion()
  ✓ testMedicalRecordUpdateSuccess()
  ✓ testMedicalRecordWithoutVet()
> 4 older results
```



```
[INFO] Tests run: 8, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 9.751 s -- in au.edu.rmit.sept.webapp.MedicalRecordTests
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 8, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  12.772 s
[INFO] Finished at: 2024-10-13T19:43:49+11:00
[INFO] ------------------------------------------------------------------------
```

Extra Notes

System Architecture

1. Frontend Layer:

- Technologies: ReactJS or Angular for a dynamic and responsive interface.

- Components: Pages for appointment scheduling, medical records, prescription management, educational resources, and payment.

- Features: Implement components for user reviews, notifications, and a map for order tracking.

2. Backend Layer:

- Server: Node.js or Python Flask/Django for handling API requests.

- Database: MongoDB or PostgreSQL for storing user data, appointment details, medical records, and product information.

- APIs: RESTful APIs to facilitate communication between the frontend and backend, and integration APIs to connect with third-party services like payment gateways or external vet clinic databases.

3. Middleware:

- Authentication: Implement OAuth for secure login processes.

- Middleware Services: Services for data validation, error handling, and request routing.

4. External Integrations:

- Payment Gateway: Integration with Stripe or PayPal for handling transactions.

- SMS/Email Service: Twilio or SendGrid for sending notifications and reminders.

5. Cloud Infrastructure:

- Hosting: AWS or Azure for deploying the application.

- Storage: Use cloud storage solutions for backups and large data (like medical images).

User Interface Design

1. Homepage:

- Overview of services.

- Quick access buttons for appointment scheduling, consulting a vet, and accessing the educational center.

2. Appointment Scheduling Page:

- Calendar view for selecting dates.

- List of available time slots and veterinarians.

- Option to reschedule or cancel.

3. Medical Records Dashboard:

- Sections for viewing and downloading medical history, vaccination records, and treatment plans.

- Secure login and access controls.

4. Prescription Management Interface:

- List current medications with options to request refills.

- Detailed information on each medication.

5. Educational Resources Section:

- Searchable library of articles, videos, and guides.

- Featured content on pet care trends and tips.

6. Payment and Checkout Page:

- Secure payment form.

- Summary of charges, with options to apply promo codes and view detailed billing.

7. Custom Features (like a map for order tracking or user reviews):

- Interactive map showing the status of orders.

- User review system with ratings and comments.

Sketches and Wireframes

Using tools like Sketch, Figma, or Adobe XD, you can start creating wireframes based on the described UI elements. Focus on:

- Usability: Ensure the interface is intuitive and easy to navigate.

- Accessibility: Design keeping in mind accessibility standards to cater to all users.

- Responsive Design: Make sure the UI adapts well to different devices and screen sizes.