Software Requirements Specification

for

<Pet Care Vet>

**Version 1.0 approved**

**Prepared by <Helisa Ormëni**

**Saeda Haka>**

**<Happy Paws>**

**<21.05.2024>**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to specify the software requirements for the Happy Paws Pet Care Vet Clinic website. This document covers the website's functional and non-functional requirements, including features such as schedule reservations, location information, and contact details. The scope includes the complete system required to manage pet care appointments, excluding payment processing and advanced medical consultations.

## Document Conventions

* **Bold**: Titles and important terms
* *Italics*: Emphasis on key points
* **Code**: Inline code or commands
* Requirements are numbered sequentially (e.g., REQ-1, REQ-2)
* Priorities are indicated as High, Medium, or Low

## Intended Audience and Reading Suggestions

This document is intended for:

* **Developers**: For implementation and understanding of functional and non-functional requirements
* **Project Managers**: For project planning and resource allocation
* **Testers**: For preparing test plans and cases
* **Documentation Writers**: For creating user manuals and help guides
* **Clients**: For verification that the requirements meet their needs

Suggested reading sequence:

1. Introduction
2. Overall Description
3. External Interface Requirements
4. System Features
5. Nonfunctional Requirements

## Product Scope

The Happy Paws Pet Care Vet Clinic website aims to provide a user-friendly platform for pet owners to schedule appointments, access clinic locations, and contact information, and view veterinarian profiles. The website will enhance the user experience by offering real-time availability checking and responsive design compatible with various devices. The project aligns with the business strategy of improving customer engagement and operational efficiency at the clinic.

## References

* Agile Methodology Guide
* PHP Documentation
* MySQL Documentation
* User Interface Style Guide
* Web Content Accessibility Guidelines (WCAG)

# Overall Description

## Product Perspective

Happy Paws is a new, self-contained product designed to streamline the appointment scheduling process for a veterinary clinic. It will replace the current manual booking system with an automated web-based system. The website consists of several components, including user management, vet profile management, and appointment management, all integrated to provide a seamless user experience.

## Product Functions

* User registration and authentication
* Viewing and managing vet profiles
* Booking, rescheduling, and canceling appointments
* Real-time availability checking
* Displaying clinic location and contact information
* Admin interface for managing vet profiles and appointments

## User Classes and Characteristics

* **Pet Owners**: Primary users who book appointments. Varying technical expertise needs an easy-to-use interface.
* **Admins**: Clinic staff managing appointments and vet profiles. Require secure access and advanced functionalities.
* **Vets**: View their schedules and manage their profiles. Need reliable and easy access to the system.

## Operating Environment

* **Hardware**: Standard web server, client devices including desktops, tablets, and smartphones
* **Operating System**: Cross-platform (Windows, MacOS, Linux, iOS, Android)
* **Software**: Web browsers (Chrome, Firefox, Safari, Edge)
* **Backend**: PHP, MySQL
* **Frontend**: HTML, CSS, JavaScript

## Design and Implementation Constraints

* Must comply with data privacy regulations (e.g., GDPR)
* Use of specific technologies (PHP, MySQL)
* Responsive design requirement for various screen sizes
* Limited to appointment scheduling, no online payment or advanced medical consultations

## User Documentation

* **User Manual**: Detailed guide for pet owners and admins
* **Online Help**: Integrated help sections within the website
* **Tutorials**: Step-by-step instructions and video guides

## Assumptions and Dependencies

* The project assumes the availability of a stable internet connection for all users.
* Dependencies include third-party libraries for frontend and backend development, and hosting services for deployment.
* The system assumes the availability of external map services for displaying clinic locations.

# External Interface Requirements

## User Interfaces

* **Home Page**: Branding, clinic information, quick access to booking
* **Booking Page**: Form to schedule appointments, real-time availability display
* **Vet Profile Page**: Details about veterinarians, qualifications
* **Admin Panel**: Manage appointments and vet profiles

## Hardware Interfaces

* **Server**: The system will interact with a web server hosting the backend services.
* **Client Devices**: Desktop computers, tablets, and smartphones will access the website through standard web browsers.

## Software Interfaces

* **Database**: MySQL for storing user data, appointments, and vet profiles.
* **APIs**: RESTful APIs for communication between frontend and backend.
* **Map Services**: Integration with external services (e.g., Google Maps) for location display.

## Communications Interfaces

* **HTTP/HTTPS**: For secure web communication.
* **Email**: For sending appointment confirmations and reminders.

# System Features

*<This template illustrates organizing the functional requirements for the product by system features, and the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## System Feature 1

#### 4.1.1Description and Priority

Enable users to register, log in, and authenticate. High priority for security and user management.

#### 4.1.2 Stimulus/Response Sequences

* User registers with email and password.
* System sends a confirmation email.
* User logs in with credentials.
* System authenticates and redirects to the dashboard.

#### 4.1.3 Functional Requirements

* **REQ-1**: The system shall allow users to register with an email and password.
* **REQ-2**: The system shall send a confirmation email upon registration.
* **REQ-3**: The system shall authenticate users using email and password.
* **REQ-4**: The system shall redirect authenticated users to their dashboard.

## System Feature 2 (and so on)

#### 4.2.1 Description and Priority

Allows users to book, reschedule, and cancel appointments. High priority for core functionality.

#### 4.2.2 Stimulus/Response Sequences

* User selects the date and time for an appointment.
* System checks availability and confirms booking.
* User receives a confirmation email.

#### 4.2.3 Functional Requirements

* **REQ-5**: The system shall display available appointment slots.
* **REQ-6**: The system shall allow users to book appointments.
* **REQ-7**: The system shall send a confirmation email upon successful booking.
* **REQ-8**: The system shall allow users to reschedule or cancel appointments.

## Vet Profile Management

#### 4.3.1 Description and Priority

Enable admins to manage vet profiles. Medium priority for admin functionalities.

#### 4.3.2 Stimulus/Response Sequences

* Admin adds or updates a vet profile.
* System saves the profile and updates the vet listing.

#### 4.3.3 Functional Requirements

* **REQ-9**: The system shall allow admins to create and update vet profiles.
* **REQ-10**: The system shall display vet profiles to users

# Other Nonfunctional Requirements

## Performance Requirements

* The system should handle up to 1000 concurrent users.
* The system should respond to user actions within 2 seconds.

## Safety Requirements

* The system must ensure data integrity and prevent data loss.
* Regular backups must be scheduled and maintained.

## Security Requirements

* User data must be encrypted both in transit and at rest.
* The system must implement strong password policies and authentication mechanisms.

## Software Quality Attributes

* **Usability**: The system must be easy to use and navigate.
* **Reliability**: The system must be reliable and available 99.9% of the time.
* **Maintainability**: The system should be easy to update and maintain.
* **Scalability**: The system should handle increasing loads smoothly.

## Business Rules

* Appointments can only be booked between 12:00 and 18:00.
* The system will not support online payments or advanced medical consultations.
* Reservations are anonymous, no user accounts or profiles are maintained.

# Other Requirements

### 6.1 Database Requirements

* **DB-1**: The database shall be a MySQL database.
* **DB-2**: The database must store user information, vet profiles, appointment details, and clinic information.
* **DB-3**: The database shall use appropriate indexing to ensure quick retrieval of appointment slots and vet profiles.
* **DB-4**: The database design should ensure data integrity through the use of foreign keys and constraints.

### 6.2 Internationalization Requirements

* **INT-1**: The system should support multiple languages, starting with English and Spanish.
* **INT-2**: All user-facing text should be stored in resource files to facilitate easy translation.
* **INT-3**: The system should handle date, time, and number formats based on user locale settings.

### 6.3 Legal Requirements

* **LEG-1**: The system must comply with GDPR (General Data Protection Regulation) for users within the EU.
* **LEG-2**: User data must be stored securely and not shared with third parties without user consent.
* **LEG-3**: The system must provide a clear privacy policy and terms of service to users.

### 6.4 Reuse Objectives

* **REU-1**: Code for user authentication and authorization should be modular to allow reuse in other projects.
* **REU-2**: UI components such as forms and buttons should be designed as reusable components.
* **REU-3**: Documentation and testing scripts should be written in a way that they can be reused for future projects.

### 6.5 Other Pertinent Sections

* **Maintenance Requirements**: The system should include comprehensive logs for error tracking and debugging purposes.
* **Backup and Recovery**: Regular backups should be scheduled and automated to prevent data loss.

## Appendix A: Glossary

* **Admin**: A user with special permissions to manage the website's data and settings.
* **API**: Application Programming Interface, used for communication between software components.
* **CSS**: Cascading Style Sheets, used for styling the website.
* **GDPR**: General Data Protection Regulation, a legal framework for data protection and privacy in the EU.
* **HTML**: HyperText Markup Language, used for creating web pages.
* **HTTP/HTTPS**: HyperText Transfer Protocol (Secure), used for communication over the web.
* **MySQL**: A relational database management system.
* **PHP**: Hypertext Preprocessor, a server-side scripting language.
* **UI**: User Interface, the space where interactions between humans and machines occur.
* **UX**: User Experience, the overall experience of a person using a product such as a website.

## Appendix B: Analysis Models

### Data Flow Diagram (DFD)

**Context Level DFD**:

* Show the interaction between users, the system, and external entities such as email services for notifications.

**Level 1 DFD**:

* Detailed view showing processes like user registration, booking management, and profile management.

### Entity-Relationship Diagram (ERD)

* **Entities**: Users, Appointments, Vets, Admins.
* **Relationships**: Users book Appointments, Admins manage Appointments and Vets, and Vets have Profiles.

### Class Diagram

* **Classes**: User, Admin, Vet, Appointment.
* **Attributes and Methods**: Details for each class including fields like username, password, appointment date, and methods like bookAppointment(), and cancelAppointment().

### State-Transition Diagram

* **States**: Registered, Logged In, Booking in Progress, Appointment Confirmed, Appointment Canceled.
* **Transitions**: Actions like logging in, booking an appointment, confirming a booking, or canceling an appointment.

## Appendix C: To Be Determined List

1. **TBD-1**: Final selection of the map service provider for displaying the clinic's location.
2. **TBD-2**: Detailed design for user authentication, including whether to use two-factor authentication.
3. **TBD-3**: Exact implementation details for internationalization, including the specific languages to support initially.
4. **TBD-4**: Decision on whether to implement a mobile app version in addition to the responsive website.
5. **TBD-5**: Detailed requirements for data backup frequency and retention policies.
6. **TBD-6**: Final decision on the external email service provider for sending appointment confirmations and reminders