**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Software Requirements Specification**

**PetLife**

Prepared by Sarah Richards

Team F Society

Wednesday, October 2, 2019

# Table of Contents

Table of Contents 2

1. Introduction 3

1.2 Purpose 3

1.2 Scope 3

1.3 Definitions 3

1.4 Overview 3

2. Overall Description 3

2.1 Product Perspective 3

2.2 Product Functions 4

2.3 User Characteristics 4

2.4 Constraints 4

2.5 Assumptions and Dependencies 4

3. External Interfaces 5

3.1 User Interface 5

3.2 Hardware Interface 5

3.3 Software Interface 5

3.4 Communication Interface 5

4. Functional Requirements 5

4.1 Household members (all users) 5

4.1.1 Log-in Page 5

4.1.2 Pets Page 6

4.1.2 Schedule page 6

4.1.3 Owners page 6

4.1.4 Settings page 6

4.1.5 Push Notifications 6

5. Non-Functional Requirements 7

5.1 Performance Requirements 7

5.2 Design Constraints 7

6. Appendix 7

6.1 Glossary 7

# Introduction

# 1.2 Purpose

The purpose of the app PetLife is to help a household more efficiently take care of their pets. They will be able to keep track of their pet’s needs, coordinate a feeding schedule with fellow housemates, and remind them when to perform tasks.

# 1.2 Scope

PetLife is a household pet management web application that will help pet owners manage the day to day requirements of owning a pet. The software that will be used for this web app will be Angular JS, Node JS, Python, and MongoDB.

This application will create a schedule for the pet owners on when they will complete certain tasks for each pet. The schedule will help owners keep track of their responsibilities, as well as give them peace of mind when they see other members check off responsibilities/needs. It’s easy to get busy and forget what time it is which is why our application will also send push notifications as needed for reminders.

# 1.3 Definitions

PetLife: the name of the application.

# 1.4 Overview

The following Software Requirements Specification document details the following sections:

* Overall description of the application
* Functional and nonfunctional requirements of the application
* An appendix

# 2. Overall Description

# 2.1 Product Perspective

Users only require a web browser to access our application on a public domain.

# 

# 2.2 Product Functions

The product has a household where users can join and schedule when to feed, water, walk, and take medical care of the pets. Each member of the house can do a task for a pet and when a task is completed the application will notify the group of the task’s completion. The application will also form a schedule based on the household member’s availability where users can select shifts they want to feed/walk the pet during. If there are no members of the household signed up for a given task when the time comes, an alert will be sent out notifying the household. The user will be able to add pets to the household and customize all their needs including what time certain actions need to be performed. Users can also enter in veterinarian information and set up alerts for upcoming visits. Alerts can also be set up for playtime, feeding time, and more.

# 2.3 User Characteristics

The users of this application will be pet owners that reside in a household with multiple people taking care of the pet(s) or by themselves and need more help managing their pet’s needs. Users oversee adding and removing pets, entering in pet information, and setting up their schedule for when they can perform tasks for the pets. It is up to the user if they want to be notified of events/tasks being completed and what tasks they complete.

# 2.4 Constraints

The system will only work on web browsers.

Browser Versions supported by Angular:

* Chrome (latest)
* Firefox (latest)
* Edge (2 most recent major versions)
* IE (11, 10, 9 (“compatibility view” mode not supported))
* IE Mobile (11)
* Safari (2 most recent major versions

# 2.5 Assumptions and Dependencies

We assume that there are no bugs in the MongoDB server that will cause issues. We depend on the MongoDB community keeping it up-to-date and stable.

# 3. External Interfaces

# 3.1 User Interface

The user interface is GUI-based, allowing the user to interact via icons and indicators. The user interface is then rendered in the internet browser. When the user first opens the application, there will be a login screen. Once the user logs in or signs up, there will be a main dashboard with a checklist of upcoming tasks with a sidebar. The sidebar will contain a pet, household, schedule, and settings tabs. When you click on a tab, it will navigate to the designated page with functions pertaining to the category.

# 3.2 Hardware Interface

The hardware interface will reside on the computer or electronic device with an online browser.

# 3.3 Software Interface

The software interface is an internet browser that is connected to our web application.

# 3.4 Communication Interface

The Web app will work on web browsers (PC and mobile) and communicate with the database service (MongoDB).

# 4. Functional Requirements

# 4.1 Household members (all users)

## 4.1.1 Log-in Page

* First-time users will be prompted to create an account or log in to an existing account.
* Once logged in, the system will check if the user is a member of a household.
* If the user does not belong to a household, they will be prompted to create a “household”, view invites to households, or request to join an existing household.
* On the “Create a household” page, the user must provide a name for the household. The user will be prompted to invite other users to be members of the household.

## 

## 4.1.2 Pets Page

* Members can view pets and their needs
* Members of a household will automatically become “owners” of that household’s pets.
* Members of a household will be able to add/remove pets from the household.
* Members of a household will be able to edit pets’ profiles of their household. Editing includes the pet’s name, needs (food, water, walking, play-with, shower, heat lamps, clean tank, etc.), frequency of needs, age.
* When an owner marks a pet’s need as “fulfilled”, all owners of that pet will have access to the update.

## 4.1.2 Schedule page

* Each user is prompted to enter their schedule for walking the pets, feeding, and any other needs. Users can then sign up for shifts for when they want to feed/walk/etc.
* The schedule will fill up with the different users and who is doing what when and allow other users to view other users’ schedules.
* If a shift is empty when the time comes, a push notification will be sent to all users alerting them of the need.

## 4.1.3 Owners page

- Members can remove themselves as an owner of a pet if they don’t want to get notifications about or do tasks for a pet.

- Displays all owners in the household and what pets they take care of

## 4.1.4 Settings page

- Allows member to leave the household they’re in (they’ll then be prompt for a new household)

- Allows members to edit their contact information

## 4.1.5 Push Notifications

* Push notifications will be sent out after a designated time has passed without any users marking the animal as fed after their designated feeding time.
* Push notifications will be sent out after designated amounts of time to remind users of visits to the vet, applying medication, or anything else.
* Users will have the opportunity to opt-out or into certain types of push notifications

# 

# 5. Non-Functional Requirements

# 5.1 Performance Requirements

- The system will support up to 10 users logged in to a household at a time

- The application’s pages must load in under 3 seconds

- The application will up and available 24/7

- UI is consistent across application

# 5.2 Design Constraints

- The system will not have a limit on how many pets a household can have

- There is no system in place for removing members from the household, so if someone got ahold of your household ID, they could join and mess up the schedule

- There are not any requirements for security, so the app may not be fully secure

# 6. Appendix

# 6.1 Glossary

Angular JS – Angular JS is a front-end, structural framework used for dynamic web

applications. Uses HTML as its template language as well as leverages CSS and

JavaScript.

MongoDB - A document-oriented, open source database management system. Made up of

collections and documents rather than tables and rows in the database.

Node JS – Platform on Chrome’s JavaScript runtime that is used for network applications.

Used for non-blocking with its event-driven servers.