Software Engineering
Software Requirements Specification (SRS) Document

Plant-Hydrate

09/20/2022

0.1

By: Kayla Abreu, Laura Love, Aimade Yacouba

I HAVE ABIDED BY THE UNCG *Academic Integrity Policy* ON THIS ASSIGNMENT

Ay, LL , KA

Table of Contents

1. Introduction

1.1 Purpose:

The goal of our project is to help users keep their plants alive and healthy by sending them reminders on the specific days that their plants need to be watered.

1.2 Document conventions:

The purpose of this Software Requirements Document (SRD) is to illustrate the requirements for the Plant-Hydrate (P-H20) system. In it, we will clarify the requirements from both the client and developers sides. The requirements from the client side will describe the types of user that will use the system and how they will interact with the user interface. From the software developer's side, the SRD will elaborate on the necessary software and hardware requirements needed to construct, use, and maintain the system at optimal levels.

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition. Acronym, Abbreviation
CSS	Cascading Style Sheets (CSS) works alongside HTML to format content on web browsers.
HTML	The HyperText Markup Language (HTML) is used to display content on web browsers.
Java	A popular object-oriented programming language that we will use to construct our application.
MySQL	An open-source database management system.
P-H20	An abbreviation for the Plant-Hydrate system.
Spring	A framework that makes the creation of Java applications more efficient.

1.4 Intended audience:

The entire SRD is intended to assist the software development team and the instructor of the CSC 340 class. Users of the application may benefit from reading the general description and the operating system and compatibility sections to see if they are interested in using the application and to confirm the application will run on their personal systems.

1.5 Project Scope:

The completion of the P-H20 system coincides with the goals of the CSC 340 class since it is a requirement of the class.

1.6 Technology Challenges:

1.7 References:

2. General Description

2.1 Product perspective:

The context and origin of the product is a group project that consists of three members. Members are to invent an application that saves data in a persistent format and must incorporate a third party API for data retrieval.

2.2 Product features:

When a user logs in, they will see a dashboard where they can add new plants. Then they will be able to schedule a reminder for that specific plant to be watered.

2.3 User class and characteristics:

Our web application will expect users to be able to access a web browser and to be self-educated on how often their plants need to be watered to set an accurate reminder time.

2.4 Operating environment:

The software is to be operated as a web app, running on traditional computers and phones.

2.5 Constraints:

- Use of specific 3rd party APIs such as the ‘google calendar api’

2.6 Assumptions and dependencies:

- Possible use of reactJS
- Possible use of Bootstrap
- Possible use of a Google calendar api

3. Functional Requirements

3.1 Primary

- FR0: The system will allow the user to login into their own account to set up their own personalized water planting system. (LL)
- FR1: The system will allow the user to enter their own specific plant into a plant-storage database. (KA)
- FR2: The system will allow the user to determine how frequently they want to be reminded and how, e.g. email. (Ay)
- FR3: The system will allow the user to change remindement times.(Ay)
- FR4: The system will allow the user to add more plants or remove them. (KA)
- FR5: The system will hold the users plant information.(LL)

3.2 Secondary:

Each member will be responsible for two functional requirements.

4. Technical Requirements

4.1 Operating System & Compatibility

4.2 Interface requirements

4.2.1 User Interfaces

First screen would be the login page with a create account link to sign up, if not a user. Then the next screen would take the user to the dashboard to add a new plant. Then the user can set up a reminder time to water plants.

4.2.2 Hardware Interfaces

Any device that supports JVM and can connect to a web browser will be able to run the P-H20 system. The system will use HTTP protocols.

4.2.3 Communications Interfaces

HTTPS will be used for the communication standards by the software as part of the project.

4.2.4 Software Interfaces

HTML/CSS will be used for the frontend to create a stylized user interface. Java will be used for the backend development with a Spring framework and Maven dependencies to make the application function. MySQL will be used for the database management system to maintain records of user logins, plant lists, and reminder schedules.

5. Non-Functional Requirements

Constraints on the services or functions offered by the system (e.g., timing constraints, constraints on the development process, standards, etc.). Often apply to the system as a whole rather than individual features or services.

5.1 Performance requirements

- NFR1(R): The beginner user will be able to set up their own account in less than a minute.
- NFR2(R): The beginner user will be able to set up their own reminder and add their plant in less than 5 minutes.
- NFR3(R): The novice user will be able to set up their own reminder and add their plant in less than 2 minutes.

5.2 Safety requirements

- Safe storage of passwords

5.3 Security requirements

Privacy and data protection regulations that need to be adhered to while designing the product. For Example:

- NFR4(R): Login credentials from user.

5.4 Software quality attributes

5.4.1. Availability
5.4.2. Correctness
5.4.3. Maintainability
5.4.4. Reusability
5.4.5. Portability

5.5 Process Requirements

5.5.1. Development Process Used

We will use the incremental development process for the P-H20 System.

5.5.2. Time Constraints

A prototype will need to be prepared by 10/20/2022 and a functional version of the application will need to be presentable by 11/22/2022.

5.5.3. Cost and Delivery Date

There should be no additional costs in the development of the P-H20 system and it will be ready by 11/22/2022.

5.6 Other requirements

All SRS/SRD should be:

- **Correct:** A method of analysis that ensures that the software meets the requirements identified.
- **Unambiguous:** There is only one interpretation of what the software will be used for and it is communicated in a common language.
- **Complete:** There is a representation for all requirements for functionality, performance, design constraints, attributes, or external interfaces.
- **Consistent:** Must be in agreement with other documentation, including a systems requirements specification and other documents.
- **Ranked for Importance and/or Stability:** Since all requirements are not of equal weight, you should employ a method to appropriately rank requirements.
- **Verifiable:** Use measurable elements and defined terminology to avoid ambiguity.
- **Modifiable:** A well-defined organizational structure of the SRS document that avoids redundancies can allow easy adaptation.
- **Traceable:** Ability to trace back to the origin of development and move forward to the documents produced from the SRS.
- **Legible and Professionally Presented**: Must use a consistent font and style. Must have proper formatting of tables and charts. Must be grammatically correct. Use active tense and concise sentences.