Software Engineering Software Requirements Specification (SRS) Document

The Little Helper

09-20-2022

<Version 1.0>

By: Chinwendu Afiemo, Gary Elam, Dahao Huang

<Honor Code>

Table of Contents

1. Introduction

1.1 Purpose: The Little Helper is a web-based Software application that helps the user create and maintain a to-do list. The proposed features include The ability to create a list, add items to the list, check off items in the list, and finally archive a list.

1.2 Document conventions: "The purpose of this Software Requirements Document (SRD) is to describe the client-view and developer-view requirements for the The Little Helper (TLH). Client-oriented requirements describe the system from the client's perspective. These requirements include a description of the different types of users served by the system. Developer-oriented requirements describe the system from a software developer's perspective. These requirements include a detailed description of functional, data, performance, and other important requirements."

1.3 Definitions, Acronyms, and Abbreviations

This section Include specialized terminology dictated by the application area or the product area.
For example:

THL	An abbreviation for The Little Helper. This is the name of the web application that is being built.
Java	A programming language created by Oracle. We will be using this language to build the back end (API) of THL
API	An Application Programming Interface
DB	An abbreviation for Database.
JS	An Abbreviation for JavaScript. A programming language that will be used for the frontend development of THL.

1.4 Intended audience: This entire document is worded such that any stakeholder or viewer should be able to understand THL, its use cases, its user interface, and its function. Abbreviations are used for the sake of clarity and brevity when possible.

2 Project Scope: The primary goal of every business is to make a profit. By charging users a small fee for using this app we accomplish that goal. Additionally there is the added sense of fulfillment from helping people be a better version of themselves by being more effective in their day-to-day activity.

2.1 Technology Challenges:

2.2 **References:**

H2 Database Engine. (n.d.). Retrieved September 20, 2022, from https://www.h2database.com/html/main.html

Spring Boot. (n.d.). Spring. Retrieved September 20, 2022, from https://spring.io/projects/spring-boot

JavaScript | MDN. (2022, September 13). Retrieved September 20, 2022, from https://developer.mozilla.org/en-US/docs/Web/JavaScript.

2. General Description

2.1 **Product perspective:** This project was assigned as a requirement of the CSC-340 class provided by the computer science department of the University of North Carolina Greensboro.

2.2 Product features: The Little Helper is a list and task organizer. list It feature which helps users create a list of tasks or make a shopping list. The item crossing feature is a line that cancels out a completed task or list. It helps the users keep track of their accomplishments.

2.3 User class and characteristics: Our website application does not expect our users to have any prior knowledge of a computer, apart from using a web browser. Anybody with basic knowledge of computers can use our website application, because it's made easy and interactive.

2.4 Operating environment: TLH is a web application and can be accessed on cell phones, and on any operating system etc.

2.5 Constraints:

Due to the use of a 3d engine, we had to limit the web browsers supported. To limit user error when entering the user's address, we implemented a drop-down AJAX country, state, and city selection.

2.6 Assumptions and dependencies:

Due to the nature of web applications we can assume certain limitations will be placed on what our app can do, especially in the graphics range.

3. Functional Requirements

Functional requirements
Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.

2.1 Primary

The system will allow the user to add, edit and delete their events.
The system will allow the user to use their own code to sign in the application.
The system will order the events by date which users choose.

All the requirements within the system or subsystem in order to determine the output that the software is expected to give in relation to the given input. These consist of the design requirements, graphics requirements, operating system requirements and constraints if any.

For Example:

- FR0: The Little Helper will allow the user to check created lists or tasks. This information will be accessed by clicking on MyList button
- FR1: The Little Helper will allow the user to add a new events
- FR2: The Little helper will allow the user to archive an existing event
- FR3: The Little Helper will allow the user to delete events.
- FR4: The system will keep the user's organized list database synchronized to within 24 hours.

2.2 **Secondary:** Some functions that are used to support the primary requirements.
The application will have a database to store users' information and their events.

3. Technical Requirements

3.1 Operating System & Compatibility

3.2 Interface requirements

3.2.1 User Interfaces

When users enter the application, they will view the and welcome, and they can follow the guide to enter their username and password to sign in the database. After log in, there are users' events and some buttons. Users can view and edit their events. They also can click buttons to add different types of events.

3.2.2 Hardware Interfaces

A user must be able to run an internet browser to access this application client side. Additionally a stable network connection is required if this application is being accessed in the case it is being run on a server.

Server Side the volume of disk space you may need is dependent on the amount of users you plan on managing with TLH. At least two gigabytes of ram are required to run this application.

3.2.3 Communications Interfaces

Data pertaining to date and time will be obtained via external API.

3.2.4 Software Interfaces

Spring is used as a server management and the primary framework behind the web based application THL. A time-getting api is used to get information about the time and timezone of the user to properly push notifications to the user.

4. Non-Functional Requirements

4.1 Performance requirements.

- NFR2(R): The list of stored events should take no more than 20 megabytes.
- NFR3(R): The application will take more than two gigabytes of ram while running

4.2 Safety requirements

User information should not be stored anywhere within the application user-side or server-side.

4.3 Security requirements

- NFR7(R): The system will only be usable by authorized users.
- NFR8(R): The system will not store any data pertaining to the users

4.4 Software quality attributes

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

Detailing on the additional qualities that need to be incorporated within the software like maintainability, adaptability, flexibility, usability, reliability, portability etc.

4.5 Process Requirements

5.5.1. Development Process Used
5.5.2. Time Constraints
5.5.3. Cost and Delivery Date

4.6 Other requirements

- NFR4(R): The system will conform to FERPA guidelines to maintain student privacy.

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.