Software Requirements Specification

for

Centralized Schedule and Reminder Management System

**Version 0.9**

**Prepared by:**

**Logan Gauchat, Dawson Henrichs, Andrew Hudson, Monikrishna Rayala, Colt Weiner**

**Group 12**

**02/25/2024**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1.** **Introduction 1**

1.1 Purpose

1.2 Project Scope 1

1.3 Definitions, Acronyms and Abbreviations 1

1.4 References 1

1.5 Overview 2

**2.** **Overall Description 2**

2.1 Product Functions 2

2.2 User Classes and Characteristics 2

2.3 Constraints 2

2.4 Assumptions and Dependencies 3

**3.** **Specific requirements 3**

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 4

3.1.4 Communications Interfaces 4

3.2 Functional requirements 5

3.2.1 Create Account Function 5

3.2.2 Create Category Function 5

3.2.3 Create Reminder Function 6

3.2.4 Edit Account Function 6

3.2.5 Create Task Function 6

3.2.6 Login Function 7

3.2.7 Edit Category Function 8

3.2.8 Edit Reminder Function 8

3.2.9 Edit Task Function 8

3.2.10View Category Function 8

3.2.11 View Reminder Function 8

3.2.12View Task Function 8

3.3 Performance Requirements 9

3.4 Software System Attributes 9

3.4.1 Availability 9

3.4.2 Security 9

3.5 Design constraints 9

3.5.1 Standards Compliance 9

3.5.2 Hardware Limitations 9

**Appendix A: Issues List 9**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Initial Creation | 02/25/2024 | Creation | 0.1 |
| Functional Reqs | 02/26/2024 | Addition of Functional Requirements | 0.4 |
| Remaining Reqs | 02/27/2024 | Addition of Remaining Requirements | 0.7 |
| Interfacing | 02/28/2024 | Addition of Interfaces | 0.9 |
| Finalization | 02/29/2024 | Final Revision | 1.0 |

# Introduction

* 1. **Purpose**

The purpose of this Software Requirement Specification (SRS) document is to outline the functional and nonfunctional requirements for the development of an application that is designed to streamline the management of schedules and reminders. The application will allow users to create reminders of various responsibilities and appointments, while remaining neat and organized. The application will also send the user notifications to alert them of their upcoming appointments.

## Scope

The scope of this project includes the following activities:

User creating and managing their account

User creating tasks

User setting reminders for tasks with various notification methods

User creating categories to organize their tasks

User searching their tasks via various search criteria

## Definitions, Acronyms and Abbreviations

* + 1. **Definitions.**

Admin: System administrator to manage and alter user accounts.

User: A potential user of the application.

* + 1. **Acronyms.**

API Application Programming Interface

CDN Content Delivery Network

CSRMS Centralized Schedule and Reminder Management System

CSS Cascading Style Sheets

GUI Graphical User Interface

SRS Software Requirements Specification

* + 1. **Abbreviations.**

No Abbreviations used.

## References

Find key project documents, including UI guidelines, contracts, and system specifications, in the CSRMS GitHub Repository.

Project GitHub Repository:

Title: "CSRMS GitHub Repository"

Name: “SE-4283-G12”

Author: Logan Gauchat, Dawson Henrichs, Andrew Hudson, Monikrishna Rayala, and Colt Weiner

Version Number: 938d2e0

Date: February 29th, 2024

Link: <https://github.com/SE-4283-G12>

## Overview

The remainder of the SRS includes the following:

* An overall description that includes general factors about CSRMS and its requirements. This provides a background for the requirements of the system.
* Specific functional requirements of CSRMS to a level of detail sufficient enough to enable designers to layout the system and testers to test the system to satisfy said requirements.
* Topics such as software and hardware interfaces, functions, performance, and standards compliance will be included.

# Overall Description

## Product Functions

CSRMS offers the functionality to create and edit custom task reminders. The user will be able to sort their tasks and merge them into folders. The user will be able to search through their tasks and set customer reminders of how and when they will receive notifications.

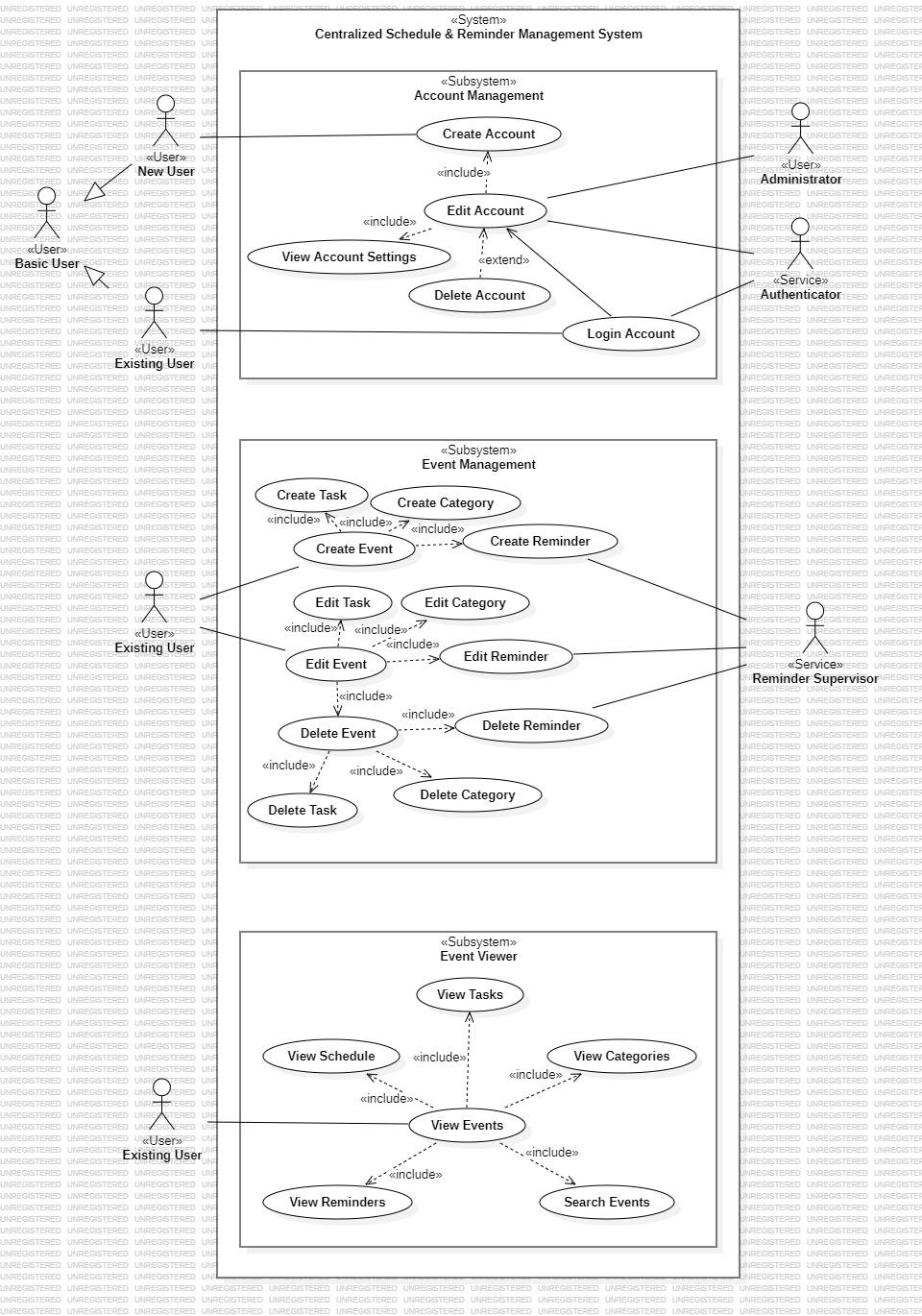
## User Classes and Characteristics

The **User** is able to create and manage their own user account; create and manage tasks, categories, and appointments; search through their tasks via user defined search criteria; and set custom reminder notifications for their tasks.

The **Administrator** is responsible for the management of user accounts, this includes the deletion of user accounts.

The **Authenticator** is responsible for ensuring all accounts have valid credentials to log in to the system.

The **Reminder Supervisor** is responsible for the sending notifications to each user.



## Constraints

The project is completed within the allocated timeline to be delivered by the end of the semester.

The project should include all the functionality laid out in the scope of the project, while adding in additional functionality if time permits.

## Assumptions and Dependencies

The CSRMS relies on seamless API integration for interacting with external applications, streamlining task execution without starting from scratch. Additionally, Content Delivery Networks (CDNs) ensure swift resource delivery, enhancing user experience with consistent, visually appealing interfaces across devices and browsers.

These essential APIs and CDNs include:

* Notification APIs: These APIs, which may include offerings from Microsoft, Google, or Amazon, enable the CSRMS to efficiently manage and dispatch notifications to users.
* CSS Framework CDN: Leveraging a CDN for CSS frameworks ensures consistent and visually appealing interfaces across devices and browsers, enhancing user experience.
* jQuery CDN: Utilizing a CDN for jQuery libraries enhances the CSRMS with dynamic and interactive web functionalities, simplifying tasks like DOM manipulation and event handling.

The CSRMS depends on essential development tools for operation:

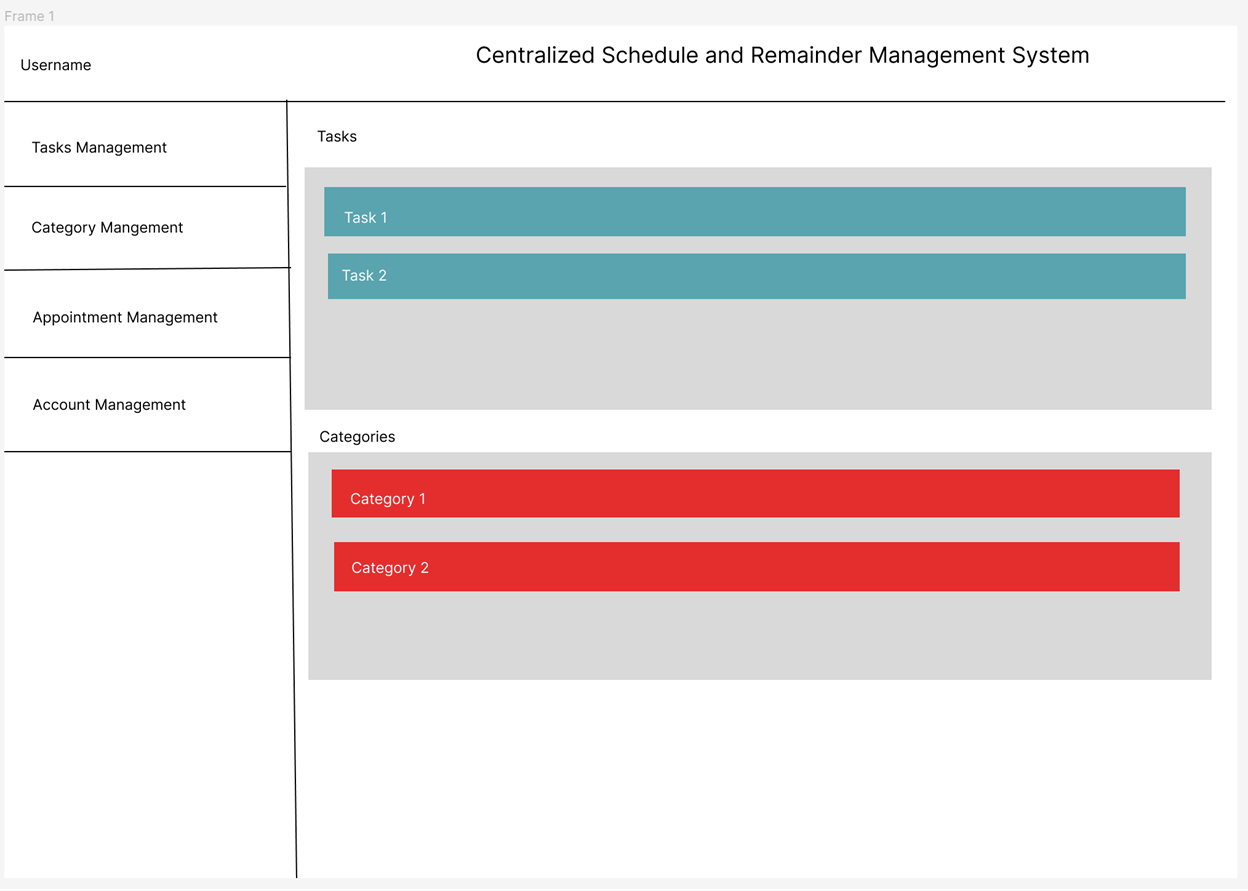
* Node.js: Node.js serves as a dependency by providing the runtime environment necessary for executing server-side JavaScript, enabling efficient backend operations and integration with APIs.
* Electron Framework: Electron is a dependency for cross-platform desktop application development, ensuring consistent user experience across operating systems and facilitating seamless integration with APIs to enhance CSRMS capabilities.

# Specific requirements

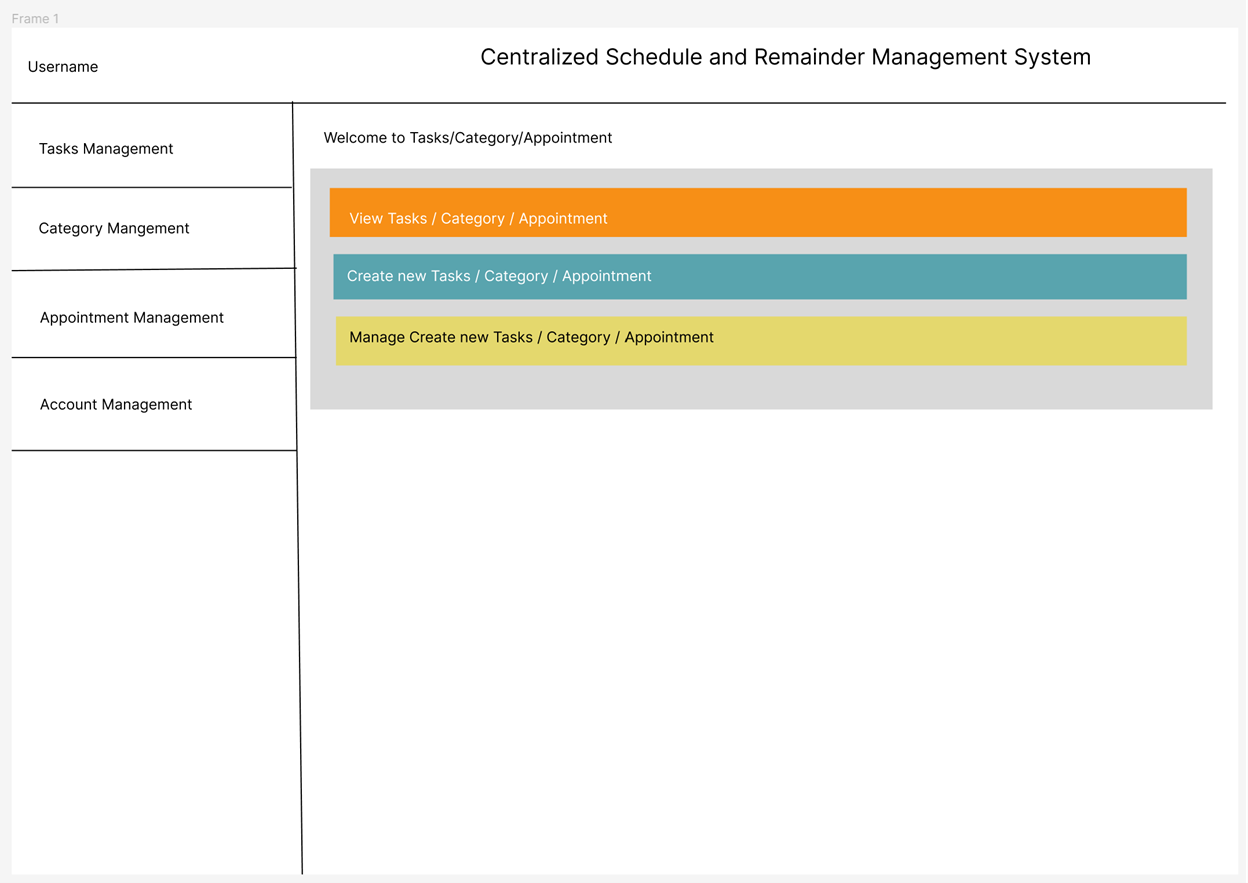
## External Interface Requirements

### User Interfaces

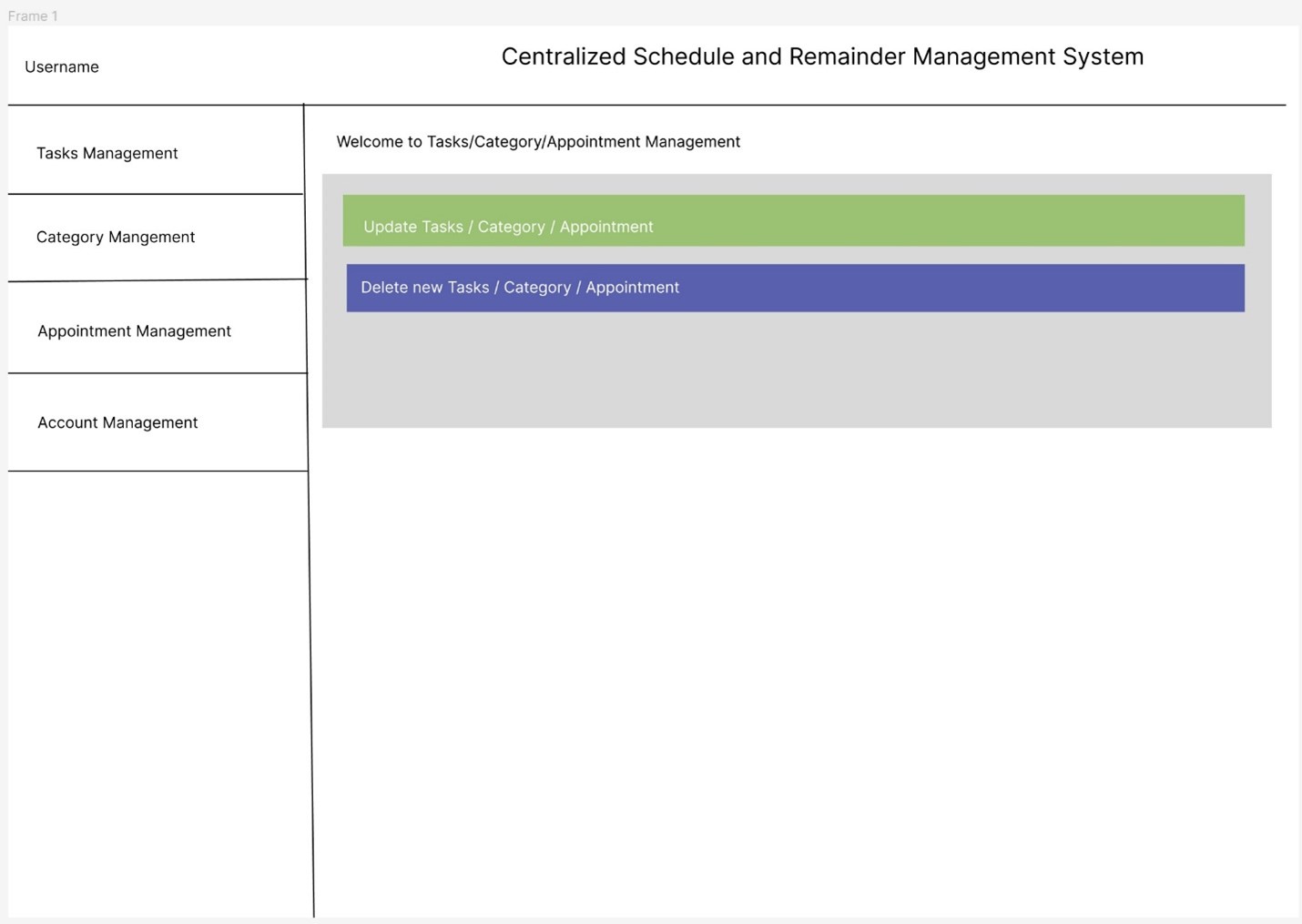
The application home page is populated with recent tasks and some lists of categories. On the side panel, a User will have navigation options for task management, category management, appointment management and account management. Upon selecting an option, the user will be able to see the general management page for each respective option. Additionally, each management page has three options to choose from: viewing existing Tasks / Categories / Appointments, creating new Tasks / Categories / Appointments, and managing existing tasks. The management option is provided to either update an existing event or delete it.



(Sample image of the home page)



(Sample image of the generic management page)



(Sample image of the event editing page)

### Hardware Interfaces

CSRMS will be supported on all operating systems that are capable of running major internet browsers (Google Chrome, Firefox, Safari, etc.).

### Software Interfaces

CSRMS works with different software components to function smoothly.

These components are:

1. Databases: The CSRMS stores and gets schedule data, user info, and notifications from databases like MySQL or MongoDB.
2. Operating Systems: It works on Windows, macOS, and Linux to be available on different devices.
3. Tools: Tools like Node.js and Electron Framework help build and manage the CSRMS.
4. Libraries: CSS frameworks and jQuery libraries are used for better-looking interfaces and easier web tasks like DOM manipulation.
5. Integrated Commercial Components: APIs from Microsoft, Google, or Amazon help manage notifications effectively.

Data and messages like schedule updates and notifications go in and out of the CSRMS. The system uses services like database access and API integration for this. It follows specific rules for communication, and any shared data is handled securely.

### Communications Interfaces

Users will need a device that can access the internet, an email provider such as Gmail, Hotmail, and yahoo to create an account and receive reminders. They will need to have a standard web browser such as Google, Mozilla Firefox, Safari.

## Functional requirements

### “Create Account” Function

#### Introduction

The “Create Account” function will be present for a new user at the start of using the program for the first time

#### Inputs

The user will have to enter a valid email, valid password between eight and twelve characters including at least one special character and one number, and their first and last name.

#### Processing

Email.js will be used to check that a valid email address is added. The password will need to be verified that it contains at least one special character and one number as well as being between eight and twelve characters long. Any failure to verify credentials will result in the user being prompted to correct the invalid fields.

#### Outputs

Once verification is complete the user will be notified that their account has been successfully created.

### “Create Category” Function

#### Introduction

The “Create Category” Function will allow the user to create new categories to easily categorize reminders and tasks.

#### Input

The user will have a button that will take them to a page where they can define the category name, and set a color to represent the category.

#### Processing

Upon clicking the button and filling out the form, the user will click create category. The system will create a new category in the filesystem that will allow the user to use it going forward.

#### Outputs

The category is created and now available system wide.

### 3.2.3 “Create Reminder” Function

#### 3.2.3.1 Introduction

#### The “Create Reminder” Function will allow the user to create new reminders for themselves so they won’t forget any task that needs to be done.

#### 3.2.3.2 Input

The user will have a button that will take them to a new page where they can define what they’d like to be reminded of. If it is a recurring reminder or a one time reminder. If it’s recurring, how often they need to be reminded. They can also assign a category for the reminder.

#### 3.2.3.3 processing

Upon clicking the button and filling out the form, the user will click ‘create reminder’. The system will create a new reminder on the filesystem and create a notification event in the reminder system.

#### 3.2.3.4 Outputs

The reminder is created, the user is brought back to the home page, The user will receive a reminder on the set date.

### 3.2.4 “ Edit Account” Function

#### 3.2.4.1 Introduction

The “Edit Account” function will allow a user to make changes to their account, such as changing their email address, updating their password, or changing their name in the system.

#### 3.2.4.2 Input

The user will need to enter new information to replace the data field that they wish to replace.

#### 3.2.4.3 Processing

Email.js will be used to check that a valid email address is added. The password will need to be verified that it contains at least one special character and one number as well as being between eight and twelve characters long. Any failure to verify credentials will result in the user being prompted to correct the invalid fields.

#### 3.2.4.4 Outputs

Once verification is complete the system will display to the user that the field they requested to be changed has been updated.

### 3.2.5 “Create Task” Function

#### 3.2.5.1 Introduction

The “create Task” function will allow the user to create tasks they need to complete. These tasks can be tied to categories.

#### 3.2.5.2 Input

The user will be given a button where they will be brought to the create task page. They will then need to input the task name, details around the task needing to be completed, and assign a category to the task.

#### 3.2.5.3 processing

Upon clicking the button and filling out the form, the user will click ‘create task’. The system will create a new task on the filesystem.

#### 3.2.5.4 Output

The task is created, the user is brought back to the home page.

### 3.2.6 “Login” Function

#### 3.2.6.1 Introduction

The “Login” Function will allow the user to login to the system to begin creating tasks.

#### 3.2.6.2 Input

The user will need to enter their most recent email address and their most recent password.

#### 3.2.6.3 Processing

The system will add the salt stored in the system for the password and process the combined password through the hash function set up in the system, and compare the hashed password to the one stored in the system. The system will also compare the email address to the one stored in the system for that user. Any error in validating the user's login credentials will result in the system prompting the user to retry their login credentials. The user will have three attempts to login with their valid email address before the account becomes locked.

#### 3.2.6.4 Output

The user will be logged in and taken to the home page.

### 3.2.7 “Edit Category” Function

#### 3.2.7.1 introduction

The “Edit Category” Function will allow the user to edit existing categories.

#### 3.2.7.2 Input

The user will have a pencil Icon Button that will take them to a page where they can redefine the category name, and change the color representing the category.

#### 3.2.7.3 Processing

Upon clicking the button and editing the form, the user will click “Update Category”. The system will edit the existing category in the filesystem and update the category system wide.

#### 3.2.7.4 Output

The category is edited and now updated system-wide.

### 3.2.8 “Edit Reminder” Function

#### 3.2.8.1 Introduction

The “Edit Reminder” Function will allow the user to edit existing reminders.

#### 3.2.8.2 Input

The user will have a pencil Icon Button that will take them to a page where they can redefine the Reminder details, such as Recurring options, reminder description, category related to it.

#### 3.2.8.3 Processing

Upon clicking the button and editing the form, the user will click “Update Reminder”. The system will edit the existing Reminder in the filesystem.

#### 3.2.8.4 Output

The Reminder is edited and now updated accordingly.

**3.2.9 “Edit Task” Function**

#### 3.2.9.1 Introduction

The “Edit Task” Function will allow the user to edit existing Tasks.

#### 3.2.9.2 Input

The user will have a pencil Icon Button that will take them to a page where they can redefine the Task details, such description, category associated.

#### 3.2.9.3 Processing

Upon clicking the button and editing the form, the user will click “Update Task”. The system will edit the existing Task in the filesystem.

#### 3.2.9.4 Output

The Task is edited and now updated accordingly.

### 3.2.10 “View Categories” Function

#### 3.2.10.1 Introduction

The “View Task” Function will allow the user to view all the details of their categories.

#### 3.2.10.2 Input

This function will take in the category ID to get all the details about the category to display it to the user.

#### 3.2.10.3 Processing

The user will click into the category they want to see then it will display a more detailed view of the category.

#### 3.10.4 Output

The user is able to view all the details of the category.

### 3.2.11 “View Reminder” Function

#### 3.2.11.1 Introduction

The “View Reminder” Function will allow the user to view the details of their reminder.

#### 3.2.11.2 Input

This function will take in the reminder ID to gather all the details about the reminder.

#### 3.2.11.3 Processing

The user will see a list of reminders and upon selecting one to view the system will display all the details of the reminder to the user.

#### 3.2.11.4 Output

The user is able to view all the details of the category.

### 3.2.12 “View Task” Function

#### 3.2.12.1 Introduction

The “View Task” Function will allow the user to view the details of their Task.

#### 3.2.12.2 Input

The user will see a list of tasks and upon selecting one to view the system will display all the details of the task to the user.

#### 3.2.12.3 Processing

The user will see a list of tasks and upon selecting one to view the system will display all the details of the task to the user.

#### 3.2.12.4 Output

The user is able to view all the details of the task.

## Performance Requirements

A computer with Windows 10 or later, and an Intel core i3 processor, and 4GB of RAM.

## Software System Attributes

### Availability

### CSRMS is built to make sure it’s always up and running when you need it. We will keep an eye on the system at all times and handle any problems as they arise. Our aim is to have the system work smoothly with uptime around 95%. When we have scheduled maintenance that brings the system down we will give the users a week's notice and redirect them to a status page.

### Security

Password salting and hashing will be used to keep users login credentials as secure as possible to avoid having plain text user credentials stored within the system.

## Constraints

### Standards Compliance

CSRMS will adhere to industry standards and guidelines, ensuring that it meets the same-high quality requirements as other programs. By following these standards, we’re committed to delivering a reliable and smooth user experience every time.

### Hardware Limitations

*CSRMS is made to adapt to different hardware environments. However, certain limitations should be considered to optimize software performance:*

*Memory Requirements:*

*CSRMS operates efficiently with a minimum of 4GB of RAM. Users with lower memory may experience slower response times.*

*Processor Speed:*

*Optimized for Intel Pentium 4 core processors or higher. Lower-grade processors may lead to performance bottlenecks during complex operations.*

*Storage Capacity:*

*CSRMS has minimal storage requirements, but users with limited capacity should manage data efficiently to prevent performance issues.*

*Network Connectivity:*

*Stable and high-speed internet is recommended for seamless CSRMS usage, as it relies on an internet connection.*

**Appendix A: Issues List**

*This section highlights ongoing issues that need attention during the development of the CSRMS.*

*UI Consistency*

*Issue: Team has not fully decided on a user interface design.*

*Reminder Notifications*

*Issue: Team has not decided on what methods and APIs to use for handling reminder notifications.*

*Password Security*

*Issue: Team has not documented password salting and hashing details.*

*Error Handling*

*Issue: Team has not established an approach for handling errors.*