**Software Requirements Specification**

**for**

Web CMS

**Version 0.1.0 approved**

**Prepared by Group 3**

**4F00 Web CMS Analysis & Design**

**February 7th, 2020**

# Table of Contents

***Section # and Description Page #***

[**Table of Contents**](#_heading=h.hpyormsc6yks) **2**

[**Revision History**](#_heading=h.searzb23dlfh) **3**

**1.** [**Introduction**](#_heading=h.aqur0p5x6v5i) **4**

**1.1** [Purpose](#_heading=h.3znysh7) 4

**1.2** [Document Conventions](#_heading=h.11ir5434s5f0) 4

**1.3** [Intended Audience and Reading Suggestions](#_heading=h.dqetizv8nt23) 4

**1.4** [Product Scope](#_heading=h.3n3swhavt0qj) 4

**1.5** [References](#_heading=h.9w6qajqoyzyf) 4

**2.** [**Overall Description**](#_heading=h.k44otblyw8zm) **5**

**2.1** [Product Perspective](#_heading=h.2s8eyo1) 5

**2.2** [Product Functions](#_heading=h.o6ufc8xs6k5p) 5

**2.3** [User Classes and Characteristics](#_heading=h.puiohbaf6lvy) 5

**2.4** [Operating Environment](#_heading=h.5fjm8udqcls1) 5

**2.5** [Design and Implementation Constraints](#_heading=h.3zagupcopca4) 5

**2.6** [User Documentation](#_heading=h.44jduafzwrr0) 6

**2.7** [Assumptions and Dependencies](#_heading=h.ws1hqire9d79) 6

**3.** [**External Interface Requirements**](#_heading=h.ulb5h9p9nduq) **6**

**3.1** [User Interfaces](#_heading=h.2jxsxqh) 6

**3.2** [Hardware Interfaces](#_heading=h.p2qvkm95doni) 6

**3.3** [Software Interfaces](#_heading=h.bxg5d441c2fc) 6

**3.4** [Communications Interfaces](#_heading=h.trly52gp55a6) 7

**4.** [**System Features**](#_heading=h.lq2rayn4oaog) **7**

**4.1** [Drag and Drop Website Creation](#_heading=h.d6lptt99mb1j) 7

**4.2** [Templates](#_heading=h.6mf1kqyomokb) 8

**4.3** Payment System 8

**5.** [**Other Nonfunctional Requirements**](#_heading=h.3whwml4) **9**

**5.1** [Performance Requirements](#_heading=h.2bn6wsx) 9

**5.2** [Safety Requirements](#_heading=h.o73kz79jhjku) 9

**5.3** [Security Requirements](#_heading=h.kku706qellic) 9

**5.4** [Software Quality Attributes](#_heading=h.8yme3m4khndg) 9

**5.5** [Business Rules](#_heading=h.t76ure1s9u44) 10

**6.** [**Other Requirements**](#_heading=h.b0qcyh8serzl) **10**

[**Appendix A: Glossary**](#_heading=h.fat525m2xxqi) **10**

[**Appendix B: Analysis Models**](#_heading=h.2k3xwph1lmac) **10**

[**Appendix C: To Be Determined List**](#_heading=h.uurr9zk0phxk) **10**

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Ahmed Mohamed | 2020-01-16 | Starting draft of 4. System Features | 0.1.0 |
| Matthew Pierrobon | 2020-01-18 | Formatting, starting the intro (rough notes) | 0.1.0 |
| Jonathon Kruis | 2020-01-20 | Starting Section 2 (rough notes) | 0.1.0 |
| Tokunbo Oyebolu | 2020-01-21 | Safety & Performance Requirements (rough notes) | 0.1.0 |
| Steven Bubonja | 2020-01-21 | Added numbering on pg2 and section 4.3 | 0.1.0 |
| Matthew Alunni | 2020-01-21 | Addition to Section 2 | 0.1.0 |
| Ahmed Mohamed | 2020-01-22 | Continue work on System Features (#1 and #2) | 0.1.0 |
| Ahmed Mohamed | 2020-1-23 | General editing and added system features. | 0.1.0 |
| Greg Pogue | 2020-1-24 | Security requirements | 0.1.0 |
| Matthew Alunni | 2020-01-24 | Addition of class diagram | 0.1.0 |
| Casey Morgado | 2020-01-24 | Developer software quality attributes | 0.1.0 |
| Casey Morgado | 2020-01-25 | System reliability and availability attributes | 0.1.0 |
| Matthew Pierobon | 2020-01-26 |  |  |
| Casey Morgado | 2020-01-26 | Started system usability (rough notes) | 0.1.0 |

ex

* increase the version number if you feel a major update has been made
* version convention per the Semantic Versioning Standard

# **Introduction**

## **Purpose**

*<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>*

The purpose of this document is to describe the implementation of an online system for the management of websites. This is draft version 0.1.0 of this document. The overall goal of this product is to deliver a Content Management System (CMS herein) that enables users to create and edit web pages using a GUI editor for a more streamlined and intuitive web development experience. This document describes the entire CMS. This document will explain the main system features, the interfaces the system uses, and the constraints that exist on the system. This also includes the user requirements for the page editor GUI, as well as the functional and non-functional requirements for the database, queries, backend functions, and web page templates. This CMS makes use of various markup templates along with a default stylesheet that users may choose to customize or simply use to structure the content on their website. This document is primarily for developers and stakeholders, but also for users of the system as well.

## **Document Conventions**

*<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>*

***TBD***

***Note:*** *we should include any common terms or acronyms found in the document that the reader should understand without having to search the glossary, this list can be added to over time*

|  |  |
| --- | --- |
| **Term** | **Description** |
| Live mode | The front-facing website that a visitor would see |
| Edit mode | The backend web pages containing the page editing features that are only available to logged in users with administrator privileges |

## **Intended Audience and Reading Suggestions**

*<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>*

There are three different audiences that have been taken into consideration when preparing this document. The first of these is the development team responsible for designing and building the system. This includes the developers, testers, and project managers. This document will serve as an up-to-date record of the proposed system design that must be adhered to during development. Any changes made during the planning phase of the development process will be recorded here so the most recent system requirements are available to the team. System support staff can also use this document to understand why certain design decisions were made and the constraints that come alongside them. This is to assist in maintaining the system during operation and to aid in the possibility of extending/enhancing the product in the future.

The other two main audiences targeted by this requirements specification are project stakeholders, as well as users of the CMS that create and manage their website. The stakeholders will find this document useful for tracking the progress that is being made on the project and ensuring that all the requirements needed by the business are defined. It is crucial that any missing or incomplete requirements are brought to the attention of the development team as soon as possible. Users of the system can gain a better understanding of how the system works via this document. They can also read about the system features that are available to website administrators in greater detail to create better content for their website.

## **Product Scope**

*<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>*

This system is to be deployed on a web server to manage the content and layout of the website. The main feature of this product is the ability to create, edit, save, and update web pages using an editor to drag and drop various elements on the page. Users are able to create web pages easier since they do not have to manipulate the underlying HTML when building the page. Additionally, the system allows for the creation of users profiles with varying levels of account privileges. Website admins are able to view the administrator section of the site where pages can be modified, while regular users are only able to see the live version of the website that exists after an admin creates and uploads their webpages.

*Additional features?*

* *media library (for pictures, documents, videos)*
* *WYSIWYG editor (for templates)*

## **References**

*<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>*

***TBD***

* *Semantic Versioning 2.0.0 (* [*https://semver.org*](https://semver.org) *)*
* *Reference 2 (URL)*
* *Reference 3 (URL)*

# **Overall Description**

## **Product Perspective**

*<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>*

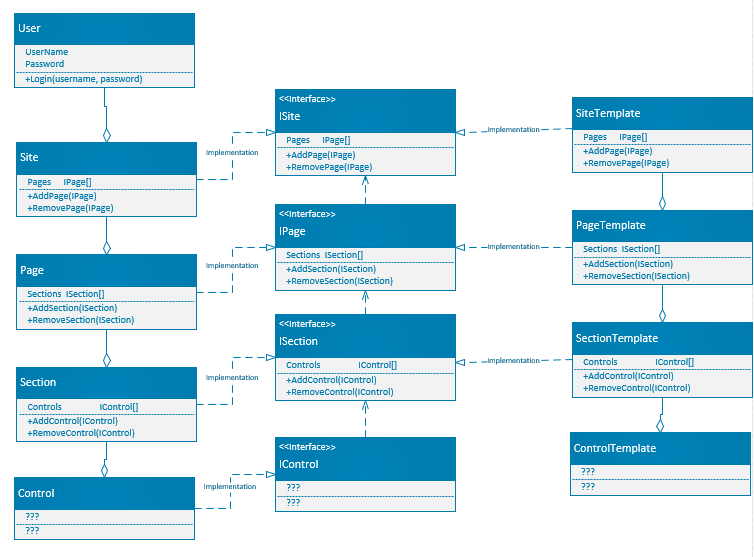
The CMS is a new, self-contained system that is used to create, manage and modify content on a website without having to manually edit any source code. The site will have two different interfaces, one for end-users, and one for content managers. The end-user site will provide information about the editor section and will have the option to be taken to a login/purchase screen. The end-user section of the application is also where users may view content uploaded by other users. The content manager interface will require login authentication and will contain the editor which allows the user to modify the layout and/or content of their website using the drag and drop functionality of the editor.

*//Major Component diagram here*

## 2.2 Product Functions

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

* Pre-Made Templates: Starting points for users when creating a website. Users should be able to choose from a collection of pre-made templates. Users should be able to see a preview of what templates look like, and then should be able to apply a template to their page.
* Drag-and-Drop Editor: The application requires a way for users to upload and format content. This feature should be highly customizable. The editor should be user-friendly so that users are encouraged to upload custom web pages. Users should also be able to upload external content to the editor. This could be in the form of images.
* Viewing Pages: While the application is used in uploading and creating web pages, there should also be a “View Mode”. The user should be able to access web pages in a “read-only” mode. In this mode, the user should not be required to log in. The user should only be consuming content in this view, not creating it.
* Payment Processing: The user should be able to register and purchase access to uploading content to the website.



## 2.**3** **User Classes and Characteristics**

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

|  |  |
| --- | --- |
| Website End-User | The End Users are the people who are browsing the front page of the site, with no login or need for any privileges or technical expertise. Optionally, they may switch user classes by logging in the editor, or purchasing access to the editor. To purchase access, they are taken to a secure 3rd party application to process payment (PayPal). |
| Content Manager (Important) | The Content Managers are the people who are using the editor and must log in to access it. They will use the editor to create, modify or remove the content from their website using the drag and drop system provided in the editor. Within the editor, the managers can use templates and modify them to suit their needs. They also have the option to switch between live mode and edit mode, where live mode is how the site looks to end-users. Within edit mode, while the manager is working on a page, a draft is created of that page and is saved so managers can edit the content without directly editing the live site, and the changes only take effect after the changes are committed. |
| System Administrator (Important) | The System Administrator(s) will have access to all the features of the site along with access to all other user accounts. They will maintain user accounts and will have the access into each user’s information, aside from payment info. System Admins will have a different login process than content managers will, as the privileges of the account are different, and require a different security clearance. |

## 2.4 **Operating Environment**

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

* The system will run in the newest version of most browsers (Chrome, Edge, Safari, Firefox).
* The system must operate and store all data on sandcastle.
* The system should be available at all times.
* Minor service interruptions are tolerable, but no data shall be lost.
* Personal information must be protected.
* One user cannot see another user’s information within the database.
* All changes to a webpage are stored in a draft until changes are committed.

## 2.5 **Design and Implementation Constraints**

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).*

* All system data must be stored on sandcastle.
* The user interface will be created using HTML, CSS, JS, PHP, Ajax.
* HTTPS used for secure login.
* Browser that supports HTML, CSS, and Javascript.
* This software will be used on a web browser. This means that the user will have no read or write access to the application when offline.
* Users will upload their content to a specified Brock domain since the website is hosted on sandcastle. Users will not be able to use their own domain names.

## 2.6 **User Documentation**

*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

A user manual will be produced along with the system, and the manual will provide users with information on how to use the editor and all the features that are contained within the editor function. The user will be able to access the manual through the help section of the application. The application will also feature a “frequently asked questions” page. This page will feature common user questions, as well as tutorials.

## 2.7 **Assumptions and Dependencies**

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

Assumptions:

* Users have a PayPal account that can be used for purchase.
* Users have at least minimal experience using drag-and-drop editors.
* Users will know how to find our web application.
* Users have basic computing knowledge, including operation of a web browser
* Sandcastle will be operating functionally at all times.
* Sandcastle will be able to adequately handle the number of users on the system.

Dependencies:

* PayPal being available when someone tries making a purchase.
* Sandcastle running functionally.
* Sandcastle being able to handle an appropriate number of users on the system.

# **External Interface Requirements**

## **User Interfaces**

Frontend: HTML, Javascript, CSS, PHP, Ajax

Backend: SQL

The frontend user interface is a client-served copy of the website, which is dynamic in nature. The pages are loaded from the backend server when requested by the browser as the current, live copy of the site. The site shall include buttons and features that are easy to locate, distinguish, and where applicable, redundant across all pages of the site. The data specific to users and their content shall be located on an SQL Database running on the Sandcastle system. When a site is loaded, the proper authentication means shall take place to allow the user to receive and edit authorized sections of the database and site. The user shall not interact directly with a database or the values within; all updates and views must be handled by the website’s frontend, with proper authorization.

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

## **Hardware Interfaces**

A browser which supports HTML, CSS and Javascript (moved to 3.3)

The system interfaces with the Sandcastle environment at Brock University. The hardware to access the system must support access to an operating system, a supported web browser (as outlined in 3.3), electricity to power the equipment, and an internet connection. The communication protocol would be HTTP.

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

## **Software Interfaces**

Operating System:

Database:

Visual Studio Code?:

The software interfaces required to access the website include an operating system which can run a supported web browser.

* The web browser must support the following protocols:
  + HTML, CSS and Javascript.

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

## **Communications Interfaces**

Web browser which supports HTML, CSS and Javascript.

Ability for HTTPS connections for secure login.

Electronic forms user sign up and payment.

Network server communication protocols to retrieve, upload, update and delete page templates, and update user data.

Data transfer rates:?

Synchronization Mechanisms:?

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

# **System Features**

*<This template illustrates organizing the functional requirements for the product by system features, 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.>*

## Drag and Drop Editing

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

4.1.1 Description and Priority

The main feature of the service is an easy to use website builder, facilitated through drag and drop. The drag and drop feature is of High priority, since it is the focal feature that provides high useability. Emphasis will be placed on this feature.

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Value** | **Rational** |
| Benefit | High | The drag and drop feature provides the most benefit to non technical users |
| Penalty | High |  |
| Cost | High | Developing the drag and drop feature requires extensive developed. |
| Risk | High | The feature must work as intended to avoid loss of user’s work or cumbersome use. |

*Figure x: Tabular analysis of the Drag and Drop feature.*

4.1.2 Stimulus/Response Sequences

The user will be able to select the object type from a list of objects that are listed and displayed on the left side of the interface. The user may then drag and drop an object onto their canvas. The canvas will then reflect the new object that the user has added. The user cannot drag an object onto another object.

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1: Object types shall be clear, accessible and easily selected.

REQ-2: Objects shall be clickable, then dragged onto the canvas, with an immediate response and reflection on the canvas.

REQ-3: Objects shall not overlap. The system shall not allow objects to be placed on top of each other.

REQ-4: Objects shall be deleted with either the “Delete” key or by right clicking and selecting “Delete Object”.

REQ-5: Object values shall be optionally modified, including: object size, object dimensions, object colour and mo6

## Templates

4.2.1 Description and Priority

Templates help the user create a site faster than dragging and dropping from scratch. The templates will provide common designs that the user can further customize to suit their needs.

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Value** | **Rational** |
| Benefit | Medium | The template feature provides faster site creation, and provides ideas; however, it is not a critical feature. |
| Penalty |  |  |
| Cost | Low | After developing the drag and drop feature, creating templates for the user will have a relatively low cost. |
| Risk | Low | The feature should work as intended as to increase efficiency and customer satisfaction; however, should it fail, it would be an inconvenience, rather than catastrophic. |

*Figure x: Tabular analysis of the Template feature.*

4.2.2 Stimulus/Response Sequences

The user will be able to select from a list of templates inside the templates pane on the left side of the editor page. Templates will be named and can be clicked on for a preview of their appearance. Once the user selects their template, an editable web page will be shown with that template. The user can then edit and manipulate that web page through drag and drop, and other features as needed.

4.2.3 Functional Requirements

REQ-1: Template names shall be concise, unique and descriptive.

REQ-2: Templates may be previewed before selection.

REQ-3: Upon selection, a warning of the template overriding the current site will be given.

REQ-4: Sites created with templates may be edited with the same features as sites created without templates.

## 4.3 Payment System

4.3.1 Description and Priority

*The payment system allows a user to access the editor to their site. It is presented when the user navigates to this part of the site when they have not previously set up a payment method, or when their subscription is not active (Expired subscription or declined payment). The payment system is of High priority, since it allows/restricts users to edit their site, and has direct ties to the usability of the CMS for the users of the system. It also has direct ties to the functionality and purpose of the site, so it shall be available at all times.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Benefit** | **Penalty** | **Cost** | **Risk** |
| Availability | High | High | High | Medium |
| Updates to card data | High | Medium | Medium | Low |
| Card verification | High | High | High | High |
| Personal data stored | High | High | Medium | High |

***\*\*\*Note:Possibly define benefit, penalty, cost, risk and their levels of severity?\*\*\****

*Figure x: Tabular analysis of the Payment System feature.*

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.3.2 Stimulus/Response Sequences

The user shall be given visual cues to where to enter their card details. Ie. field that denotes where a card number shall be entered vs where an expiry date shall be entered.

The user shall be given visual feedback as immediate as possible when entering incorrect data that can be checked locally, such as missing fields, invalid month or date, or an incorrect credit card checksum.

The “Pay” button which submits the payment request shall lock when clicked to ensure that:

1. User has feedback that they have indeed “clicked” the button, and
2. Avoidance of double-billing a user.

4.3.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

REQ-1: The system shall accept payment cards through the appointed payment provider; PayPal.

REQ-2: Payment cards shall be validated by the system using billing address verification of the cardholder to mitigate fraudulent transactions.

REQ-3: For subscriptions, the user shall be aware of the fees and intervals of payment (monthly payments) required to have access to the paid features of the site (editor).

REQ-4: Payment card data shall not be stored in the database; rather, the payment provider shall appropriately handle payment card data for charging the user, and remit limited data back to the corporation for tracking transactions, such as:

* Card Network (Visa / MasterCard, etc)
* Cardholder’s name
* Payment amount in $ CAD
* Date and time of the transaction in EST
* Last 4 digits of payment card
* Whether payment was approved or declined
* Transaction reference number

REQ-5: The payment system shall have a mechanism for a user to cancel their subscription to the subscription service. The user can continue to use the service until the end of the month they have paid for.

REQ-6: A “donation” transaction made by a user visiting a user’s CMS site shall be handled as either a one-time payment, or as a monthly cancellable “donation subscription”. Both shall follow the payment requirements laid out in this section.

REQ-7: The payment system shall have a mechanism for a user to update/change/delete their card data. Deleting the card shall trigger a message to ensure that the user intended to delete the card data. If deleted, the process to cancel the subscription will follow, similar to REQ-5 above.

REQ-8: Upon approved payment, the system shall allow the user to access the subscription subset of the site (editor). Access shall be instantaneous, or as soon as instant, when resources can allow them.

REQ-9: The system shall use the checksum to locally detect incorrect card numbers entered. The user shall be notified that a typo has been made.

REQ-10: The system shall allow a user to make payment with another card when one has been declined. This shall be limited to 10 attempts in a 1-hour window to reduce fraudulent use of the system.

REQ-11: All payment failure/success attempts shall be logged by the system, and measures shall be taken (IP blocking) when suspicions of fraudulent activity has occurred.

REQ-12: The system shall prevent a user to be double-billed. The form shall lock when submitted, and, through failure of the system, user, or network, the system shall fail-safe. The system shall ensure that in the case of similar payment details getting to the authorization system more than once in a defined timeframe (5 minutes), only the first attempt is charged and subsequent requests are refused.

## 4.4 Authentication and Privileges

4.3.1 Description and Priority

Users who are administrators of the websites hosted on the CMS must be authenticated securely, as they have full access to editing of their site. It is of utmost priority to ensure the security of users’ sites. This will be facilitated by accounts that these users create and manage. Non administrators will not have editing privileges, rather these are the users that will view the website only.

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Value** | **Rational** |
| Benefit | High | The template feature provides faster site creation, and provides ideas; however, it is not a critical feature. |
| Penalty |  |  |
| Cost | Medium | The main costs associated with this feature are: encrypting the connection, encrypting user credentials on the server and developing the login page. |
| Risk | High | The feature is extremely important because should unauthorized users gain access to a site, a breach of data and control would occur. |

4.3.2 Stimulus/Response Sequences

An administrator must log-in before they are able to manage and edit their site. To do this, they may click “log-in” and enter their credentials. The CMS will then securely push their credentials to the server. If it matches a valid account, the administrator will now be logged in, via cookies, and able to manage their site.

A user who does not or fails to log in will not be able to edit any sites on the CMS. They may view publicly viewable websites in a view-only fashion.

4.3.3 Functional Requirements

REQ-1: When a logged off user is browsing the website, every page shall have a log-in button.

REQ-2: Login buttons shall be consistently and conspicuously placed among all pages.

REQ-3: Users that are logged out cannot edit any site whatsoever.

REQ-4: Users that are logged in may only edit any site that they own and have administrative privileges for.

REQ-5: Clicking of the log-in button shall direct the user to a log-in page where they can enter their credentials. Upon the pushing of valid credentials, the user shall now be logged in.

# **Other Nonfunctional Requirements**

## **Performance Requirements**

**Editor Mode:**

* The site(s) must be available in Live Mode by all users while the site(s) is/are being edited. This will allow users to still access data needed even if changes are being made to the data.
* All button features (change colour, add menu, remove heading, etc.) must have response times under 1 second.
* No ads will be allowed to inhibit editing if user has chosen the free package
* Editor will only be required to work on Desktop machines. If user tries to log in to editor from mobile, the system shall display a message asking the user to log in via desktop.
* If user selects the action to delete a page or site, the system should ask for confirmation of this action before proceeding to remove page(s) or site(s) via a pop-up.
* User must be able to upgrade their package to make more features of the system available to them without losing any work done on any of their sites so far. This will allow for consistency and increase safety of system.
* System should be able to revert to an old (published) version of the site. This will increase resilience and maintainability of system should corruption occur.
* Ads must not be editable in the free version.

**Live Mode:**

* Response time for selecting buttons and options must be under 1 second.
* Users must only be able to view the most recently published version of the site. Once site is published again, the user will not see the newly published site until they move to another page or refresh the site.
* Interaction time between user and forms must be under .3 seconds.
* When interacting with a form at the same time, it is being re-published, submitted form content must be saved even after the site is re-published. This increases reliability of the system.

## **Safety Requirements**

* Log-in portal may only allow three (3) attempts to log in. After every attempt, the particular key that is missing (email, ID or password) will not be told to the user. This increases safety and security and safeguards against access via brute force.
* If editor is open on another desktop, system must close editor on that desktop before opening it on new desktop upon request. This will increase dependability of the CMS so that any changes are always saved. It also helps system to avoid faults.
* Changes must be automatically saved every 2 minutes with options to undo up to \_\_\_\_\_ changes. However, the changes will not go live automatically. Administrators must choose to publish the site themselves by selecting the “Publish” option and changes must take effect on the live site within 5 seconds.
* The WYSIWYG editor must have response times under 1 second. This avoids confusion between the user and the system as to what change has taken effect in editor mode and what change has not.
* Upon leaving a page, the system must display a pop-up that asks the user to save changes before leaving. The user may decline, undoing all changes (including those automatically saved) or may accept and all automatically saved and manually saved changes will be saved.
* Passwords will be encrypted and saved in database of CMS.
* Data and files uploaded to the database via one user will not be available to any other users of other systems. Meaning, no two administrators of different sites will have access to each other’s data or files.
* There will only be 1 administrator per site but an administrator can manage multiple sites depending on the package of the CMS chosen.

## **Security Requirements**

*<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

**User:**

* Users will require a password length of at least 8 characters with a requirement that at least one character is a letter and at least one character is a number.
* When a login attempt fails, the user will not be told where the failure occurred. That is, if the username and/or password was incorrect.
* An IP history will be recorded of the last used IP***(too strict?)***. When logging in from a new IP the user will be required to verify themselves with a code provided to the primary email contact. The last used IP will be available to the user under User Settings.
* An option for 2-factor authentication will be provided in the form of a secondary email. This will be used with cookies to authenticate a browser. Additionally, this email will be alerted when a password is changed through the Forgot Password feature.
* Magic Link

**Server:**

* Log-in and authentication shall follow industry security standards. Log-in credentials will be transmitted securely through SSL and stored securely on the server via encryption.
* Passwords are never shared/written to disk unencrypted.
* All input into the system shall be filtered to protect against SQL injection attacks
* All output from the system shall be properly escaped (E.g. HTML special characters, URL encoding)

## **Software Quality Attributes**

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

The quality of the software will be to the highest standard. The software will be

**Developers:**

* Maintainability
  + The system will be developed with maintainability from the start.
  + Code is developed in increments with reviews at every sprint, to ensure code is within the required parameters.
  + Code is to be consistently readable for all developers to easily understand.
  + Documentation will be created alongside code, to ensure system is understandable.
  + Refactoring will be done through each stage to maintain the understandability.
  + Automated tests are made to be continuously ran to ensure the code is maintained in a functional state.
  + Flexibility will be implemented into code to prevent expensive costs for future changes, evolution and bug fixes
  + Git is used for version control of code.
* Reusability
  + Reusability will be integrated into code developed where symmetry seems reasonable.
  + Components that are found in multiple areas of the system will be developed once and reused to reach high levels of efficiency in code.
  + Polymorphism will be used to in development patterns to aid in the reusability of objects.
* Portability
  + The system will be portabile on the newest version of all leading web browsers, including Chrome, Edge, Safari, Firefox
* Testability
  + Developers will perform their own testing on code they have developed
  + Testing of software will take place throughout the entire development stages of the product. At the end of development for each unit of the system, unit testing will take place to ensure all processes are functioning to the associated requirements. As development for component’s are complete, component testing will take place to ensure all units are working properly together. Finally, when the development of features are completed, end-to-end testing will occur to ensure all components are performing together correctly. The software will also include rainy day tests, release tests before each release to the production environment. All release tests must pass ahead of code moving from development environment to production.
* Correctness
  + Review checks, automated tests, project board, and standups are the several techniques used to ensure system correctness according to the software requirements specification
  + Review checks will take place at each sprint (weekly) as an overall checkup to make sure all processes are in correct order and following the correct requirements
  + Automated tests will be used to guarantee code is functioning as required
  + Project board will be used to manage and organize processes for correctness
  + Standups will be a daily gathering with the development team and project manager to evaluate the work being done for efficiency and correctness
* Robustness
  + Robustness will be incorporated into rainy day testing of the system. This encompasses testing code for invalid inputs, all possible user inputs, errors, unlikely input paths and exceptions
* Adaptability/Flexibility
  + Flexibility will be implemented into code on each stage of development.
  + Object oriented programming will be considered the foundation for building future additions to the system and incorporating any changes.
  + The maintainability will be possessed through proper flexibility and adaptability of the system.
* Resilience

**Users:**

* Reliability
  + Reliability is very important to the use of the system.Users are expecting a system that is seamless and easy to use. Therefore, the system will deliver accurately devoid of faults.
  + Systems will be put into place to handle faults in all stages (avoidance, detection, tolerance)
  + Security and safety requirements will address avoidance
  + Various reliability test requirements will address detection and statistics
  + Measures of redundancy and recovery will address tolerance
  + The required system target for POFOD will be 0.005
  + With the use of a protection system architecture we will be able to monitor and prevent any system errors
* Availability
  + Availability is a metric used to express reliability and dependability in the system.
  + Users expect to have their websites live at all times, unless otherwise altered by the users themselves.
  + The main component of the system that needs to be available 100% of the time is hosting of users content.
  + In case of repairs and restarts, the editor and content management is due for maintenance at certain times. The system requirement will be to limit the down time. Therefore, the system target is to have an availability of 0.955
* Usability
  + The system front-end design is built for maximum human interaction
  + Navigating through the website will be seamless and easy due to the uncrowded outline

## **Business Rules**

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

**5.5.1 Governance**

The development, maintenance and data collection of this project will follow the applicable laws of Ontario and Canada.

# **Other Requirements**

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

# Appendix A: Glossary

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

# Appendix B: Analysis Models

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

# Appendix C: To Be Determined List

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*