Software Requirements Specification

for

WebMail MSU

Version 1.0 approved

Prepared by Joshua Moore, Rushma Parajuli, and Khem Dhami

Intro-to-SE-Lab-Fall-21-Section-2

September 2nd, 2021

1. Introduction

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The aim of this document is to gather and analyze and give an in-depth insight of the complete **WebMail MSU**, a web based email client system, by defining the problem statement in detail. Nevertheless, it also concentrates on the capabilities required by stakeholders and their needs while defining high-level product features. The detailed requirements of the **WebMail MSU** are provided in this document.

1.1 Purpose

The main purpose of this document is to provide a Software requirement specification (SRS) for **WebMail MSU** version 1.0 (v.1.0.0) , which is a web based email client. The scope of the product that is covered by this SRS involves the implementation and design of the email client in its entirety.

In this document, we collect and analyze all assorted ideas from our team to define the system, its requirements with respect to the end user. The main purpose of this document is to define how our client, team and audience see the product and its functionality. This SRS document provides a detailed overview of **Webmail MSU**, its functions and goals.

1.2 Document Conventions

Convention	**Description**
Major Headings	Major headings are numbered starting from 1, 2, 3, etc. These major headings separate the document into relevant sections, allowing the reader to more easily find what he or she is looking for. These headings are 18 point Times New Roman font and bold to stand out more easily. example: **1. Introduction**

Sub Headings	Sub headings are numbered starting from the **Major Heading** number and incrementing by .1 . It is also bolded to stand out and uses 14 point Times New Roman font. example: **1.2 Document Conventions**
Reference to section	References for the reader to refer to another section appear in-line and in bold. example: *WebMail-Msu looks to only provide functions noted in section **2.2 Product Functions.***
Acronyms placed in parenthesis	All abbreviations or acronyms will be placed in parentheses after the initial word. example: Software Requirements Specification **(SRS)**

1.3 Intended Audience and Reading

This document serves as a technical description for use by developers, users, and testers, and it will also be the basis for validating the final system. It provides a technical overview of the application within its scope, as well as the underlying technologies used to construct it.

For developers looking to extend functionality, or learn from the code refer most closely to sections 2 and 3. For testers refer to sections 2,3, and mostly 4. For users please refer to section 1 and 2.

Section 1 serves as the introduction for the WebMail-Msu application by defining a clear purpose for the application, the scope of the project, and subheadings describing how to use the SRS effectively for different purposes.Section 2 is where the technical description of the application occurs with sub headings being dedicated to certain aspects of the application.Section 3 organizes the functional features of the application, giving detail to each feature. Section 4 describes all requirements such as performance and safety of the product. Section 5 provides all other information such as the glossary.

1.4 Product Scope

Webmail-MSU hopes to provide a web based email client where users can compose,edit and send an email to other users. A user will be able to search and forward emails to other users as well as send attachments (images,files, etc) with an email. The benefit to using WebMail-MSU is that it is web based allowing the user to log in on virtually any device with a browser, and it is simple by design focusing on fast sending and retrieval of email. The goal is to provide an easily modifiable product by using well documented technologies such as python, flask, smtp, and imap.

1.5 References

NOT APPLICABLE

2. Overall Description

2.1 Product Perspective

Webmail-MSU is a mail or message transfer agent. The product could work as a replacement for email systems that the students or employees at MSU currently use i.e., Outlook, GMAIL etc. While clients like Outlook incorporate many features such as contact organization, calendar syncing, meeting information, etc. WebMail-Msu looks to only provide functions noted in section **2.2 Product Functions.**

2.2 Product Functions

In MSU, every student and employee has a mailbox. To this mailbox any number of mails could be sent. The sole purpose of our mailbox would be to act as a virtual post office. Webmail-MSU stores incoming mails for distribution which can be accessed by the user by logging into the account. The server also sends out user composed outgoing messages which are easily editable. The outgoing messages may include attachments such as image or file.
To summarize, our Webmail-MSU has the following functionalities:

- A user should be able to login into his account to access his emails.
- A user should be able to compose,edit and send an email to other users.
- A user should be able to search and forward emails to other users.
- A user should be able to send attachments (image, files, etc.) along with an email

The data flow diagram for displaying the functions of the system is attached below:

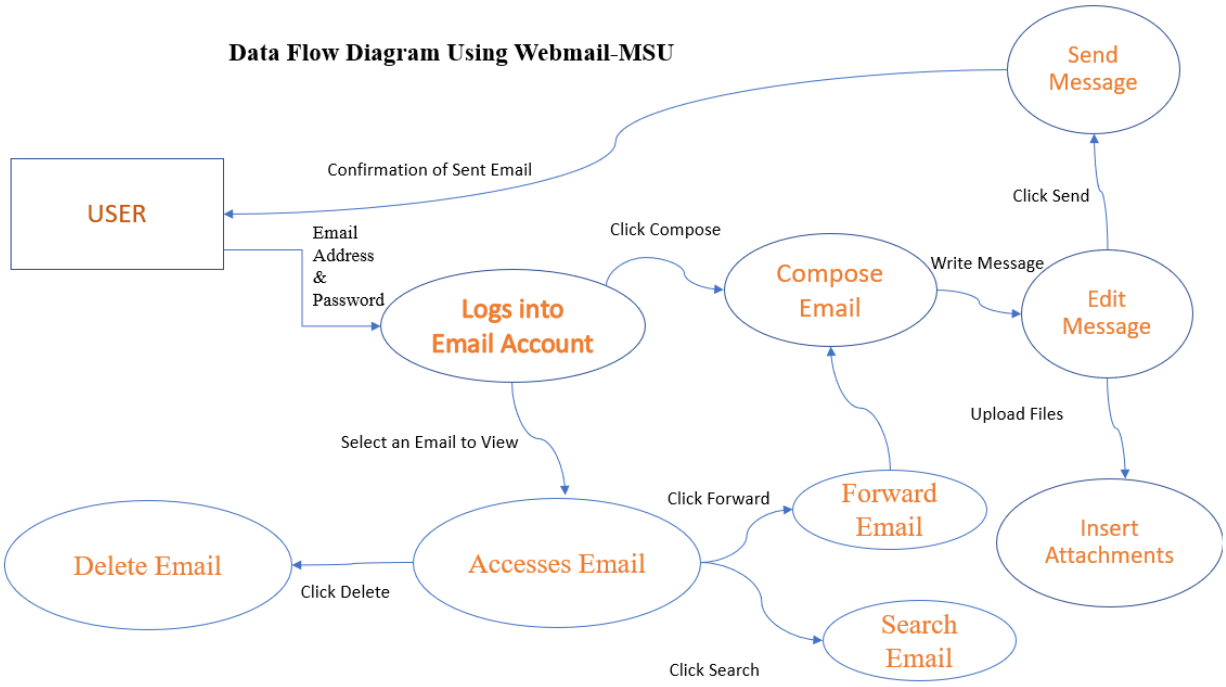

Fig. Data Flow Diagram WebMail-MSU

2.3 User Classes and Characteristics

The user class proposed for this project is any MSU student that is looking for a simple and portable way to use their email account without all of the extra features of competing clients. This way the student can focus more on a given task, and not worry about distractions. Less important users are those outside of MSU which might benefit from added functionality not in scope of this project.

2.4 Operating Environment

The software will operate in a google chrome or firefox browser on Windows, Mac, or Linux operating systems. In this way, portability will be an added feature of the application due to the interoperability of modern web browsers.

2.5 Design and Implementation Constraints

Constraints may arise from the use of limiting technologies such as the limitations of flask's "flask-mail" or further extension of functionality not within the initial scope of the project.This could include but is not limited to adding functionality akin to those found in competitor's projects such as the previously mentioned Outlook in **section 2.1.** Other constraints could arise due to the nature of the protocols being used such as IMAP and SMTP which often are difficult for end users to understand and configure without some knowledge of the underlying technologies. The product will be open source with "NO WARRANTY", so the end user will be responsible for any issues found that are not addressed in future releases.

3. System Features

- A user should be able to login into his account to access his emails.
- A user should be able to compose,edit and send an email to other users.
- A user should be able to search and forward emails to other users.
- A user should be able to send attachments (image, files, etc.) along with an email

3.1 Access emails

3.1.1 Description and Priority

This is the core feature of the application, and is crucial to get right. priority: 9

3.1.2 Stimulus/Response Sequences

The user should navigate to a website where he or she can then log into their account giving them access to their email.

3.1.3 Functional Requirements

- REQ-1:*Website hosting the email application should be working and available*
- REQ-2:*Flask-mail should be functionally operable*
- REQ-3:*Bugs that would render the hosted application unusable should be ironed out*
- REQ-4:*There should be an authorization process for validating user accounts*

3.2 Compose, edit, and send email

3.1.1 Description and Priority

This is another core feature of the application, and is crucial to get right. priority: 9

3.1.2 Stimulus/Response Sequences

The user should navigate to a website where he or she can then log into their account giving them access to their email. The user should then be able to navigate a user interface that gives them the ability to compose, edit, and send emails.

3.1.3 Functional Requirements

- REQ-1:*Website hosting the email application should be working and available with all backend technologies such as IMAP and SMTP functioning properly*
- REQ-2:*Flask-mail should be functionally operable as well as any plugins that would enable the user to perform composing, editing, and sending mail properly*
- REQ-3:*Bugs that would render the hosted application unusable should be ironed out in this context specifically bugs that would not allow the user to perform composing, editing, or sending emails.*
- REQ-4:*There should be an authorization process for validating user accounts so that one user cannot send, compose, or edit another user account's emails.*

3.3 search and forward emails

3.1.1 Description and Priority

While this is still an important part of the application, it is not as important as the user being able to compose, send, receive, edit email etc. The core system should still function without this added functionality and this should be considered after other more important components have been added and tested. Priority 3

3.1.2 Stimulus/Response Sequences

The user should navigate to a website where he or she can then log into their account giving them access to their email. From there the user should be able to click a search bar which will search all available emails that they have and return results based on the keyword they gave. There should also be a forward arrow beside the reply button in INBOX emails that the user can then forward the email to others.

3.1.3 Functional Requirements

- REQ-1:*Website hosting the email application should be working and available with all backend technologies such as IMAP and SMTP functioning properly. The applications prioritized functionalities should also be functioning properly for this feature to function. This functionality will depend on the ability to parse the text of the emails of the application for keywords.*
- REQ-2:*Flask-mail should be functionally operable as well as any plugins that would enable the user to perform composing, editing, and sending mail properly. If any plugin specifically related to searching or forwarding emails this should also be functionally operable.*
- REQ-3:*Bugs that would render the hosted application unusable should be ironed out in this context specifically bugs that would not allow the user to perform composing, editing, or sending emails. There could also be bugs specifically related to the parsing of the text in emails that should be tested for its functionality.*

3.4 send attachments (image, files, etc) along with an email

3.5

3.1.1 Description and Priority

These days users expect their email applications to do a lot for them. In regard to having an email application that will support the media that will be used for MSU students this feature set should be considered a priority. Priority 8

3.1.2 Stimulus/Response Sequences

The user should navigate to a website where he or she can then log into their account giving them access to their email.Once all other core functionality has been implemented the ability to send attachments in email should be considered.

3.1.3 Functional Requirements

- REQ-1:*Website hosting the email application should be working and available with all backend technologies such as IMAP and SMTP functioning properly. The applications prioritized functionalities should also be functioning properly for this feature to function. This feature set will depend on a stable core, and some storage functionality on the backend to enable the sending and receiving of media such as pictures or files.*
- REQ-2:*Flask-mail should be functionally operable as well as any plugins that would enable the user to perform composing, editing, and sending mail properly. If any plugin specifically related to sending attachments exist it should be explored before other solutions are considered.*
- REQ-3:*Bugs that would render the hosted application unusable should be ironed out in this context specifically bugs that would not allow the user to perform composing, editing, or sending emails. There could also be bugs specifically related to adding media to emails either going or receiving and these should be addressed before final verdict is rendered on adding the functionality.*

4. Other Nonfunctional Requirements

4.1 Performance Requirements

- The product shall be based on the web and has to be run from a web server.
- The initial load time of the Webmail-MSU would depend on the internet connection strength.
- The internet connection strength would depend on the media from which we run the product.

- The performance shall depend upon hardware components of the client.

4.2 Safety Requirements

- Opening some email message may try to cause some harm to ot access the system.
- The safety of the systems and computers may depend on whether the user has used antivirus software on the computer.
- The safety of the system also depends on the user's internet habits.
- The user/client's age would also be one of the safety concerns on the product.
- There would be some age restrictions for clients to be able to access/ sign up for some contents, and also for web purchases using the WebMail MSU.

4.3 Security Requirements

Security in matters of data, privacy issues is a must. So, WebMail has a data security concern with the proper handling of the data which is via user's consent, notice or regulatory obligations. Webmail consists of authentication requirements every time a user wants to log in; this way it will be easier for WebMail to ensure data privacy. Nowadays, every company collects customer's data by passive location tracking, secretly absorbing the user's personal address book or recording every site that the customer visits. So, to ensure that WebMail users get their privacy and keep their data hidden, several tests will be created to confirm the existence of new functionality or disapproving the previous insecure option. Since, security requirements need implementing new ideas, every feature or function within the application, will go through selecting, documenting and confirming proper implementation of security attributes.

4.4 Software Quality Attributes

Additional quality characteristics that would be important for the customers or the developers using WebMail MSU are as follows:

adaptability: Any browser or any platform should be able to adapt our WebMail MSU and run it smoothly no matter which platform is being used.

availability: WebMail MSU is able to execute the tasks that are assigned to it from every corner.

correctness: Webmail MSU will be correct and fully proofed with respect to the specification.

flexibility: It will be able to adapt external changes depending on the value delivery, in a timely and effective manner. It will have the ability to run the system automatically depending on the user's choice. Example: running Webmail MSU in a laptop, phone, tablet, etc.

maintainability: It is necessary to keep the WebMail MSU from errors, bugs, glitches etc, so WebMail MSU will be easier to maintain by the developers or the end users. WebMail will be easily maintainable to correct defects and their causes, make changes, meet new requirements, maximize safety and to make it efficient.

portability: WebMail is easier, effective and efficient to transfer from one environment to another.

reliability: Webmail will be reliable to operate under any conditions, to handle complexity of data and it will be able to avoid patterns that will lead to unexpected problems and behaviours.

reusability: WebMail's assets which are software components, codes, designs, documentation will be able to reuse in case MSU decides to use it again for any purpose.

robustness: WebMail has the ability to cope up with the incorrect input and to cope with errors during execution.

testability: As a part of the regular testing process, it will be easier to find and isolate faults for the WebMail.

usability: WebMail is more efficient to use which means that it takes less time to complete particular tasks, easier to learn or adapt and more satisfying to use.

sustainability: WebMail is designed in a way that it is capable of leading the software development process which will sustain or survive for a long time period.

timeliness: WebMail ensures the ability to collect, process, transfer and present given data to the user in real time.

efficiency: WebMail will be efficient for the intended task. It will ensure centralization of client's request to reduce network traffic.

5. Other Requirements

5.1 Database Requirements:

WebMail supports data integrity, consistent data in a concurrent surrounding, configuration management, tool development, data access by the end users, and traceability for consistency and completeness checking.

5.2 Internalization Requirements:

WebMail adapts the computer software to different languages and technical requirements of a target locale.

5.3 Legal Requirements:

WebMail MSU should display the disclaimers, copyright, and terms and conditions for the use of user data to the client before they agree on using the email-client server.

Reuse Objectives:

WebMail consists of systematic software reuse to embrace a design process which makes the software faster and at a cheaper cost;

Appendix A: Glossary

Email-client System: An email client system is a program that lives on your computer or browser and lets you send and receive emails.

Configuration: It means a product which is available.

IMAP(Internet Message Access Protocol): It's the open standard that describes how to access messages in an email mailbox.

SMTP(Simple Mail Transfer Protocol): SMTP stands **for Simple Mail Transfer Protocol**, and it's an application used by mail servers to send, receive, and/or relay outgoing mail between email senders and receivers.

Portability: Portability is a **characteristic attributed to a computer program** if it can be used in an operating system other than the one in which it was created without requiring major rework.

Appendix B: Analysis Models

Data Flow Diagram is shown above on 2.2 Product Functions.

Appendix C: To Be Determined List

Not Applicable