Software Requirements Specification

for

EC Email Client™

Version 1.0 approved

Prepared by Group 1
Patrick Bell
Hunter Nenneau
James Watters
David Mentgen

Group 1 - Section 3

August 26th, 2020

1. Introduction

1.1 Purpose

The purpose of this SRS document is to provide a thorough understanding of the requirements for our EC Email Client v1.0 from Group 1. This SRS covers the requirements of the email client in its entirety as seen from the beginning of development.

1.2 Document Conventions

This document has a series of basic conventions that will be utilized to increase comprehension. It can be assumed that similar type functional requirements can inherit basic attributes from others. For instance, the forwarding email capability can inherit commonly used features from the send email functional requirement.

1.3 Intended Audience and Reading

The intended audience of this SRS document is developers, project managers, testers, and stakeholders. It should be used by developers for reference when designing and coding the application, project managers when reviewing what has and hasn't been accomplished, testers to see if features are acting in an appropriate manner, and stakeholders to see if the desired features are known and being implemented. Readers should begin by reading the introduction, and then proceed to the overall description if not familiar with the project. If they are familiar with the project, then they should skip to sections 3 and 4 in order to see the more detailed descriptions of the features of the application.

1.4 Product Scope

This software is to be used as a web-based email client for signing in to one or more email services. The user should be able to send and receive emails, search emails, and send attachments. Other similar features can be seen in the sections below. This software aims to fill the need of our company (Into to SE Class) of creating an email application for users.

1.5 References

Gmail Rest API - https://developers.google.com/gmail/api
ReactJS Homepage - https://reactjs.org/
Amazon Webservices Lightsail Documentation - https://lightsail.aws.amazon.com/ls/docs/en_us/all

2. Overall Description

2.1 Product Perspective

This product is intended to function as the User's replacement for an email client. Using this extremely unique, state-of-the-art technology the User can expect to be met with an experience like no other. The user can access the web facing email client that will allow them to manage their emails in ways that just makes sense. This product will be minimal and will have no bloatware. It's an email client and nothing more!

2.2 Product Functions

Using this product, users can expect to be able to…

- Login to their personal email account(s)
- Compose, edit, and send emails to other users
- Search and forward emails to other users
- Provide attachments (images, files, etc) to their emails

2.3 User Classes and Characteristics

- Average users are likely to desire a minimalistic email client.
- Avid Gamers are always checking their emails for notifications and would likely desire the appeal of a minimalistic email client to provide them with all of their basic functions
- School students need email clients to stay up-to-date with their classes.
- Professional workers need email clients to stay up-to-date with their coworkers.

2.4 Operating Environment

This email client will be entirely web based, therefore it should be able to run in basic web browsers like Google Chrome and Firefox. The client will be written in Javascript and use the ReactJS framework to handle UI views and logic. The client will need to be able to operate using the Gmail API in order to communicate with the gmail servers for basic functionality. The webapp itself will be hosted on a basic Amazon Web Services Lightsail Server running Linux.

2.5 Design and Implementation Constraints

One of our biggest constraints is time and familiarity with email applications. Since we are limited on time, we are focusing on using just one type of email service which is Gmail. We will be limited to whatever features the API offers.

3. System Features

3.1 Login

3.1.1 Description and Priority

Priority: HIGH
Users should be able to provide email credentials to access their email.

3.1.2 Stimulus/Response Sequences

User accesses web application home page -> Home page presents user with "Username/Password/Login" section -> User should provide legitimate login information that will be verified -> If account credentials are verified, user will be logged into the client with access to their email

3.1.3 Functional Requirements

- REQ-1: (LoginValidator) To be used to verify the provided user credentials are correct. If invalid credentials are provided, the user will be notified that their credentials are incorrect and be prompted to try again. If user credentials are valid, the LoginValidator will proceed to retrieve the user's information so that the user may access their account.

- REQ-2: (HomePage) Must be created on the frontend for the user to access the login page.

3.2 Compose, Edit, and Send Emails

3.2.1 Description and Priority

Priority: HIGH

Users will be able to write emails, with the ability to edit them, and send them to other users or any other email addresses.

3.2.2 Stimulus/Response Sequences

Users will be able to click a button that pops up a window with input values. The input values will include a recipient address, subject line, and the contents of the email. The user will be able to compose and edit any of the fields as desired. There will be another button that allows them to send email.

3.2.3 Functional Requirements

- REQ-1: (Email Address Validation): The input on the recipient input must be a valid email address. A series of valid characters, followed by '@' and then a valid domain. The user will be notified and unable to send the email if the address is not correct.
- REQ-2: (Subject Line Validator): The subject line cannot be blank and must contain some text or the user will get a warning with an attempt to send the email.
- REQ-3: (Email Body Validation): The input of the email body cannot be blank, the email body must contain text, an image, or a file. The user will be warned and unable to attempt to send the email.

3.3 Search and Forward Emails

3.3.1 Description and Priority

Priority: HIGH

A user should be able to search and forward emails to other users.

3.3.2 Stimulus/Response Sequences

Users will click the search bar and type keywords, and the system will filter the viewable emails to include only ones that contain keywords from the search bar. When viewing an email, users will click a forward button and type the desired email address to forward an email, and the system will gather the information for that specific email and send it to the desired recipient using the API.

3.3.3 Functional Requirements

- REQ-1 (Search Bar): The search bar should be located at the top of the page used to view all emails. The users can click on the search bar and type text. While typing, the search bar should automatically update the list of emails that contain the keywords. The system should have all the emails stored in RAM so that it can easily search without having to fetch the new emails. If no emails contain the keywords then it will be displayed that there were no matches.
- REQ-2 (Email Input Validator): Once the user clicks to forward an email, the user must give a recipient. The input on the recipient input must be a valid email address. A series of characters, followed by '@' and then a valid domain. The user will be notified and unable to send the email if the address is not correct. The subject line will also be checked, and if empty, the user will be asked if they want to send an email with no subject line.

3.4 Sending Attachments with Email

3.4.1 Description and Priority

Priority: HIGH
Users should be able to have the option to add attachments along with the composed email.

3.4.2 Stimulus/ Response Sequences

After users have opened a window to compose an email, they will be able to click a button that allows them to attach a non-text data item. This includes files, images, etc. and append this onto the email.

3.4.3 Functional Requirements

- REQ-1 (Attachment Validation): The attachment that is chosen must be a valid image, or file. The file must also be smaller than 25MB. If the file is larger, the user will receive a warning that the file must be compressed, or cannot be sent with the email.
- REQ-2 (Attachment Scan): The attachment will be scanned and must pass the same criteria defined by the Gmail API. Therefore, the attachment cannot be deemed malicious, or thought to have a virus.

4. Other Nonfunctional Requirements

4.1 Performance Requirements

4.1.1 System should be responsive and provide quick access to the functions and features of the software. System should be designed as lean as possible to make it a lightweight application and made so that the only limiting factor for the speed of the system is the internet connection of the client.

4.2 Safety Requirements

4.2.1 Since our system is simply a client to connect users to their gmail accounts and use Google servers, loss of user data should be kept to a minimum. Any deletion of data should be prefaced with a warning to the user and require a confirmation from the user that they are aware that the data is being deleted and will not be able to be recovered through our system.

4.3 Security Requirement

4.3.1 Our email client will use Google's authentication API to the link users gmail account with our email client. By using such, we are giving the email client the ability to not have to store any user information and use end to end encryption via the Google authentication API.

4.4 Software Quality Attributes

4.4.1 Our email client should be visually appealing to the end user, with elements that are readable and whose functions are easily identifiable by either icon or text. The layout of the client should be familiar enough so that users know exactly what they are using, but should be unique enough to not be a direct clone of existing email clients.

4.4.2 Bugs for the email client should be easily reported and should be worked on in a timely manner by the development team.

4.4.3 The codebase for the email client should be easily readable with code structure that is coherent and functional, as well as variable names and function names that accurately describe what they are.

5. Other Requirements

NA

Appendix A: Glossary

REST: Representational State Transfer
API: Application Programming Interface
JS: JavaScript
SE: Software Engineering

Appendix B: Analysis Models

N/A

Appendix C: To Be Determined List

N/A