

Silver Surfer Project

Software Requirements Specification (SRS) Document

Final Draft

February 2021

Silver Foxes

Faith Haiss, Amanda Hegidus, Ben Schroth, Marisa Stover



Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Rough draft (v1)	Amanda Hegidus Faith Haiss Marisa Stover Ben Schroth	Rough draft first version. First time adding content to the document.	1/27/2021
Final draft (v2)	Amanda Hegidus Faith Haiss Marisa Stover Ben Schroth	Final version. Making edits per Professor Cindric's suggestions and recommendations.	2/7/2021

Contents

1 INTRODUCTION	1
1.1 PURPOSE	1
1.2 DESCRIPTION	1
1.3 OVERVIEW	1
1.4 GLOSSARY OF TERMS	1
1.5 BUSINESS CONTEXT	2
2 OVERALL DESCRIPTION	3
2.1 PRODUCT PERSPECTIVE	3
2.1.1 System Interfaces	3
2.1.2 Memory Constraints	3
2.2 PRODUCT FUNCTIONS	3
2.3 SIMILAR SYSTEMS	3
2.4 USER CHARACTERISTICS	3
2.5 USER OBJECTIVES	4
2.6 CONSTRAINTS	4
2.7 FUNCTIONAL REQUIREMENTS	4
2.7.1 User Interface: Graphical (GUI)	9
2.7.2 Application Programming Interface (API)	9
2.7.3 Diagnostics (Error Reporting and Usage Logs)	9
2.8 SYSTEM REQUIREMENTS	9
2.8.1 Hardware Interfaces	9
2.8.2 Communications Interfaces	9
2.8.3 Software Interfaces	9
2.9 DOMAIN REQUIREMENTS/CONSTRAINTS	9
2.10 NON-FUNCTIONAL REQUIREMENTS	10
2.10.1 Reliability	10
2.10.2 Availability	10
2.10.3 Security	10
2.10.4 Maintainability	10
2.11 LOGICAL DATABASE REQUIREMENTS	10
3 SOFTWARE LIFE CYCLE MODEL	11
3.1 CHOICE OF SOFTWARE LIFE CYCLE MODEL	11
3.2 JUSTIFICATION FOR CHOICE OF MODEL	11

1 Introduction

1.1 Purpose

The purpose of this document is to outline the requirements and functions for the Silver Surfer Project. The intended audience is the Silver Foxes group and the CS department faculty, who may be involved in the future.

1.2 Description

Silver Surfer Project is a web application aimed to provide professors with an interface to create custom, consistent class pages in line with the Silver Surfer design style. This will make it easier for students to find what they need, and it will be easier for professors to communicate with students. The template will include a text area for professors to fill in the website's information and the template will automatically style it accordingly in line with the HTML tags and CSS styling.

1.3 Overview

This document will provide all the initial requirements and conceptualization for the Silver Surfer Project. After this initial description there will be specific details on the requirements that the team plans to include. Finally, the document will conclude with our planned Software Life Cycle Model and the reasoning behind this choice.

1.4 Glossary of Terms

Terms:

Silver Server – The server used by the University of Mount Union's CS department.

Discord – A popular social network consisting of “servers” that allow users to talk to other users via text and voice/video calls.

Abbreviations/Acronyms:

SLCM – (Software Life Cycle Model) A conceptual framework that allows the team to work on meeting project requirements efficiently and working on the maintenance of the project.

GUI – (Graphical User Interface) An interface that allows the user to interact with different visual elements.

UI – (User Interface) Includes the combination of human-computer interaction and the communication with a device.


CS – Computer Science

IT – Information Technology

UMU – University of Mount Union

1.5 Business Context

There is no direct external sponsor for this project. Upon completion, the Silver Surfer Project may be utilized by the UMU CS department.



2 Overall Description

2.1 Product Perspective

This program will be a web-based interface that allows UMU CS students to check on class resources or communicate with their professors.

2.1.1 System Interfaces

Silver Surfer Project is a web application so it will be made available to any device with a browser.

- Application Hosting - The final application will be bundled into a Docker container for shipping Silver Surfer to production.
- Next.js - The routing and API layer of Silver Surfer to handle and serve requests to the application.
- MongoDB - The database layer containing page information and authentication for the admin page.

2.1.2 Memory Constraints

The Silver Surfer Project may be subject to disk space and loading capacity if it is implemented in the future.

2.2 Product Functions

Silver Surfer Project will provide a more user-friendly interface for both the students and professors in the CS department. It will allow students to look at course syllabi, contact professors, and find helpful resources. This program will also help professors easily update course information and connect with students.

2.3 Similar Systems

The existing Silver Server webpages currently exist providing ideas for content for the Silver Surfer Project. Inspiration for the layout was taken from systems like Discord, but the functionality is unique.

2.4 User Characteristics

The intended users of the Silver Surfer Project are the students and professors of the UMU CS department. Students will access the read-only pages to view information for their classes. Professors may have the ability to log into the application via a hidden route to add/edit pages, depending on the progress of the Silver Surfer Project.

2.5 User Objectives

- Provides an improved user-friendly interface for all clients to use.
- Provides course information for all classes offered in a given semester.
- Provides club information for any active CS department club websites/webpages.

- Provides information on professors in the CS department.

2.6 Constraints

- Scheduling/Time: With limited time, making time management and scheduling important.
- Administrative Login: Administrative access can be attacked with improper security. It may also be difficult to implement and throw the project off-schedule.

2.7 Functional Requirements

Priority Scale: Low (1) – Medium (2) – High (3)

1. Low: Items that can be eliminated should the need arise, without adversely affecting the product. These items are not urgent and not as important to the final product.

2. Medium: Items that are desired by the customer and/or users of the system, but that may be postponed until a future release. These items are not urgent and but are important parts of the final product.

3. High: Items that are mission critical and without which the system cannot function in a manner that is satisfactory to the customer. These items are urgently needed and important to the success of the final product.

1. The system has a responsive design

1.1. Description

The design layout should be designed with responsive web in mind. Web Pages should respond to window size so that all content is mobile and web friendly.

1.2. Priority

3

1.3. Risks

Some risks may include incorrect responsive design making some of the information hard to understand.

1.4. Dependencies with other requirements

The consistency of the layout is dependent on this because if the web page is not responsive then the layout will not remain consistent. The accessibility aspect is also dependent on this requirement because if the design is not responsive, it may pose accessibility problems.

2. The code is well-organized and documented

2.1. Description

The code should be organized logically with descriptive file names and variables to prevent confusion. Comments should be added to explain the

code and break it into sections. Spacing and layout of code will be used for easy readability.

2.2. Priority

3

2.3. Risks

Some risks would be someone misunderstanding the code when trying to make a change resulting in an error.

2.4. Dependencies with other requirements

Most of the requirements will depend on this because if the code is not well-organized many of the features may contain errors when running the program.

3. The layout is consistent and familiar to use

3.1. Description

The layout will not reinvent the wheel, but rather bring together the best features from other systems and adapt them. The layout will be simple, so it is easy for students to navigate the system.

3.2. Priority

3

3.3. Risks

If the layout is not consistent/familiar it could lose the accessibility aspect of it.

3.4. Dependencies with other requirements

This is dependent on responsive web design and the ability to remain consistent across all pages, as well as the accessibility aspect.

4. The layout to add information via text area is user friendly

4.1. Description

Silver Surfer users will be able to add information using text areas in a manner that is simple and time-saving.

4.2. Priority

3

4.3. Risks

Some risks may revolve around the idea of the admin user needing a password so that only they can make changes to a page or create one.

4.4. Dependencies with other requirements

This will be dependent on the accessibility requirement in order to be user friendly, as well as the requirement to allow for an administrative login.

5. The system contains info from the Silver Server**5.1. Description**

The Silver Surfer Project will contain content that already exists on Silver. Fake test data is not necessary.

5.2. Priority

3

5.3. Risks

There are no risks with this requirement.

5.4. Dependencies with other requirements

This will be dependent on storing information on a database.

6. The system is accessibility friendly**6.1. Description**

The system will be user friendly in the aspects of accessibility which includes, but is not limited to, picking web friendly fonts that are easy to read and colors that are easy on the eyes.

6.2. Priority

3

6.3. Risks

Some risks might be that the light/dark mode will affect it.

6.4. Dependencies with other requirements

This requirement is dependent on the layout remaining consistent among all pages and being familiar to the users.

7. Class page content & authentication stored in MongoDB**7.1. Description**

Class page content will be stored in MongoDB to be retrieved and presented on the Silver Surfer UI. Authentication will also be stored for professors to have access to creating and editing pages.

7.2. Priority

3

7.3. Risks

Storing authentication in a database requires safe security practices.

7.4. Dependencies with other requirements

Professors will have to update information from silver.mountunion.edu.

8. Continuous Integration/Deployment**8.1. Description**

Silver Surfer Project will have a CI/CD pipeline that will make it easier

to push code to production. Committed changes to the master branch will be packaged into a Docker file for deployment.

8.2. Priority

2

8.3. Risks

Automating tasks could cause side-effects of pushing code not staged for production.

8.4. Dependencies with other requirements

Having clean and well-documented code makes it easier to manage CI/CD and debug any issues in the pipeline.

9. The system has a landing page

9.1. Description

The Silver Surfer Project will have a landing page with content such as the UMU logo. It will have a side navigation bar with course numbers, professors, and an et cetera section. Once a section is selected, a main UI will display with content relating to the selection.

9.2. Priority

2

9.3. Risks

There are no risks with this requirement.

9.4. Dependencies with other requirements

This will be dependent on the accessibility requirement and the responsive design requirement.

10. There is Markdown Compatibility for content pages

10.1. Description

For professors to have the ability to easily share course syllabi and any other documents.

10.2. Priority

2

10.3. Risks

If the markdown capability does not work the users will not be able to view documents.

10.4. Dependencies with other requirements

This requirement is dependent on organized code and sharing information from Silver Server.

11. There is a light/dark mode

11.1. Description

The Silver Surfer Project will have the option of either a light mode or a dark mode for users.

11.2. Priority

1

11.3. Risks

May conflict with the accessibility requirement.

11.4. Dependencies with other requirements

Ties in with the aesthetically pleasing and accessibility requirements.

12. It is aesthetically pleasing**12.1. Description**

The Silver Surfer Project is aesthetically pleasing to the user's eye.

12.2. Priority

1

12.3. Risks

May conflict with the accessibility requirement.

12.4. Dependencies with other requirements

Ties in with the accessibility requirement, as well as the light/dark mode requirement.

13. There is an administrative login to add pages**13.1. Description**

There is the ability to log into the Silver Surfer Project via a password and add/edit pages.

13.2. Priority

1

13.3. Risks

This requirement may be difficult to implement at this point in time, and an attempt to implement it may throw the Project off-schedule.

13.4. Dependencies with other requirements

Ties in with the requirement that deals with the layout to add information.

2.7.1 User Interface: Graphical (GUI)

There will be a text area for professors to add/edit pages in the Silver Surfer Project. If the project schedule allows for its implementation, there will be an administrative login to access this ability.

2.7.2 Application Programming Interface (API)

API's that the Silver Surfer Project will offer (endpoints are examples):

/[class#]

Serves pages stored in the database per class.

/[class#]/[category]

Serves content such as syllabi, schedule, homework, and any course related content.

/admin

Password protected and hidden from main navigation. With access, it allows users to create and edit class pages.

2.7.3 Diagnostics (Error Reporting and Usage Logs)

Errors will be able to be viewed by users with the use of the alert boxes logging in the browser console and the use of the 401: Page not found error.

2.8 System Requirements

MongoDB will hold course information, administrative passwords, and website information.

2.8.1 Hardware Interfaces

There are not any designated hardware interfaces that need to be used for this project.

2.8.2 Communications Interfaces

In production, a local port will be exposed on the server for MongoDB database connections and the Next.js app server.

2.8.3 Software Interfaces

Students and administrators will need access to a computer or phone with a web browser.

2.9 Domain Requirements/Constraints

The Silver Surfer Project has no limitations.

2.10 Non-Functional Requirements

2.10.1 Reliability

The Silver Surfer Project web pages will be a reliable source of course information for only UMU CS students. These pages will also reliably connect to a database to provide stored

information to the users. In addition, the administrators will be able to reliably update information and create new pages.

2.10.2 Availability

The Silver Surfer Project's read-only pages are available to any user to view them, while the administrative pages are available to any user with the password.

2.10.3 Security

Silver Surfer Project will need authentication to gain access to the admin page to create/edit class pages. The rest of the app will be read-only.

2.10.4 Maintainability

To ship code to production, Docker containers will be made to package the app for production.

2.11 Logical Database Requirements

Each class page will be its own document that will contain all of the information provided by the professor. This can include homework, announcements, calendar, schedule, etc. Passwords will be stored in its own collection for administrative access.



3 Software Life Cycle Model

3.1 Choice of Software Life Cycle Model

The Software Life Cycle Model the Silver Foxes has chosen is Kanban.

3.2 Justification for Choice of Model

We chose this SLCM because it seemed the most logical for completing our list of requirements, and it makes the most sense for our current social context compared to other SLCMs such as SCRUM. Our schedules do not allow for daily meetings, so we need a SLCM that does not limit us with meeting requirements.