Silver Surfer Project

Design Document #1

Silver Foxes

Faith Haiss, Amanda Hegidus, Ben Schroth, Marisa Stover

Revision History

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Faith Haiss, Amanda Hegidus, Marisa Stover, Ben Schroth	Initial Draft of Design Document #1	3/1/2021
1.1	Faith Haiss, Amanda Hegidus, Marisa Stover, Ben Schroth	Final Draft of Design Document #1	3/3/2021

Table of Contents

Introduction

Overview

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 software will automatically style it accordingly in line with the HTML tags and CSS styling.

Deliverables

The following provides a timetable for all of our anticipated deliverables.

January

1/13 Project Selection Due

1/27 SRS Rough Draft

February

2/3 Project Blog / Website up and running

2/8 SRS Final Draft

2/10 Press Release #1 & Schedule Rough Drafts

2/15 Press Release #1 Final Draft

2/17 Schedule Final Draft

2/22 Intro Presentation

March

3/1 Design Document #1 Rough Draft

3/8 #1 Design Doc Final Draft

3/22 Design Doc #2 Rough Draft

3/29 Design Doc #2 Final Draft

3/31 Status Meeting/Presentation (prototype due)

April

4/7 Press Release #2 Rough Draft

4/12 Press Release #2 Final Draft

4/14 User's Manual Rough Draft

4/21 User's Manual Final Draft

4/26 Project due

4/28 Final Report Due

Project Schedule Reviews

February 10th, 2021
February 17th, 2021
March 31st, 2021
April 12th, 2021
April 26th, 2021

Definition 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.

Project Organization

Software Life Cycle Model

The SLCM that the Silver Foxes team has chosen for this project is Kanban. 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.

Team Responsibilities

As the team leader, Amanda Hegidus will be in charge of setting up meetings and ensuring the team stays on schedule for the deliverables.

As the configuration management specialist, Marisa Stover will be in charge of both contributing and reviewing project documentation and keeping track of deliverables. She will also ensure that any progress to the project will be documented on the team blog.

As the project architect, Faith Haiss will be in charge of designing the user interface and other design oriented deliverables.

As the lead tester, Ben Schroth will be in charge of testing and documenting testing results. Ben will also be in charge of including test results in our final deliverables.

All team members will contribute to both code and deliverables.

Managerial Process

Objectives

The main goal for this project is to create a web application that will unify the Computer Science Department web pages and streamline how professors update and create new pages.

Priorities

Our main priority is to create a text editor that the Computer Science Department professors can use in order to update or create course web pages. This GUI will be made to be very user friendly and simple for the professors to use. We want them to easily be able to make changes so that they are more likely to use the department web pages in the future. Our second priority is to make sure that all of the web pages have a consistent look to them. This will enable users/students to access course information more easily than they can with the current Silver Server web pages.

Team Reporting and Monitoring Mechanisms

Our team meets on both Mondays and Wednesdays to discuss the schedule for the current week. The first topic of our discussion includes the status of our project and the progress that needs to be made during the current week. After our progress updates we also discuss any problems team members might be facing or that are preventing them from finalizing tasks. During these meetings we will also work on our deliverables and what should be included.

Technical Process

Tools

Silver Surfer is a full stack web application built using the Next.js React framework. Components and pages are built using React.js. Routing and API's are built using Next.js. For styling, layouts, and components Silver Foxes decided to use the Bootstrap framework. Because we are using React, we installed the react-bootstrap library which makes the experience of working with Bootstrap better because we can use react components instead of class based components. To use MongoDB in Silver Surfer we created a free database cluster through Mongo Atlas. A link is given to connect the database encoded with the account to the cluster. With the Next.js initialization done, we moved to connecting and querying the MongoDB database. We created a database named silver-mongo and a temporary collection named CoursePages. MongoDB is a document based database in contrast to relational databases like SQL. Instead of tables, columns, and rows, MongoDB uses collections and documents. Collections resemble tables and documents resemble rows and tables. We chose MongoDB because it can make queries to specific data instead of setting up complex schemas in a relational database. To connect our database to the app we installed the MongoDB Node.js driver npm i mongodb. We created a folder called util to hold the connectToDatabase() function. This keeps the project structure clean and well organized. Lastly, we added Storybook. Storybook is a UI tool used for designing and styling React components. Storybook is a developer dependency that allows edits and style components with its companion web GUI. This does not contribute to the final project, but it will make it easier to design components independently before they are added to the page.

Documentation Strategy

Silver Foxes is utilizing our project blog site to keep updates tracked on a weekly basis and to serve as a reference of when something was worked on. We are utilizing two Github repositories - one for our project code and one for our project blog site. This allows the changes to easily be tracked per person and per item completed. The separate repositories allow us to keep things separate and avoid confusion. The project code repository utilizes separate branches for each person's work and each person is to create a pull request that the other team members are to approve or deny before that work is merged into the master branch.
For project task documentation, Silver Foxes is utilizing a Kanban board in Github. There is a section for tasks that need prioritized and sections for each priority level - high, medium, and low. There is a section for in progress tasks and for completed tasks where items will be moved to when necessary.

Risk Analysis

Risk Identification

- Team lack of familiarity with React
- Some dependency on one team member
- Well designed API
- Courseload of all Team Members
- Security for Admin Page
- Backend and Frontend in sync
- If the final product is used by professors
- Cost (Hosting, Database, etc.)
- Hosting (Accessibility to site)
- Client loses internet connection
- Conflicting schedules preventing team members from attending weekly meetings
- Missing test/use cases
- Scheduling pitfalls

Risk Analysis

- Team lack of familiarity with React
    - Likelihood: High
    - Impact: High
- Some dependency on one team member
    - Likelihood: Medium
    - Impact: High
- Well Designed API
    - Likelihood: Medium
    - Impact: High
- Courseload of all Team Members
    - Likelihood: Medium
    - Impact: High
- Security for Admin Page
    - Likelihood: Medium
    - Impact: Medium
- Backend and Frontend in sync
    - Likelihood: Medium
    - Impact: Medium
- If the final product is used by professors
    - Likelihood: Medium
    - Impact: Low
- Missing test/use cases
    - Likelihood: Low
    - Impact: Medium

- Cost (Hosting, Database, etc.)
    - Likelihood: Low
    - Impact: Low
- Hosting (Accessibility to site)
    - Likelihood: Medium
    - Impact: Low
- Client loses internet connection
    - Likelihood: Low
    - Impact: Medium
- Conflicting schedules preventing team members from attending weekly meetings
    - Likelihood: Medium
    - Impact: Low
- Scheduling pitfalls
    - Likelihood: Medium
    - Impact: High

Risk Planning

- Team lack of familiarity with React
    - Detailed documentation with components
    - Discord with Robert Ranallo to ask questions
- Some dependency on one team member
    - Other members take time and read documentation
    - Ask questions when needed to develop further understanding
- Well Designed API
    - Design a database schema that makes pulling data consistent and explanatory
    - Endpoints should provide most useful information for faster load times
- Courseload of all Team Members
    - Focus on time management
    - Schedule meetings when everyone is available, ensuring that everyone can attend and not miss on information
- Security for Admin Page
    - Use authentication involving the professor's Mount Union Email
- Backend and Frontend in sync
    - Both the backend and frontend need to be updated at the same time to ensure the user is able to properly use the web application
- If the final product is used by professors
    - Provide documentation for professors to see the benefit of using this new web application
    - Explain why this could be better for students
- Missing test/use cases
    - As a group think of all possible issues facing the project
    - Think of all of the different reasons as to why the application is being used
- Cost (Hosting, Database, etc.)
    - Using cloud database to prevent the team from incurring a cost

- Hosting (Accessibility to site)
    - The site will not be hosted on Silver so that a VPN will not be needed to access the site
        - Commuters sometimes have issues using a VPN to access Silver, this eliminates that concern
- Client loses internet connection
    - The client will need an internet connection in order to access the website
- Conflicting schedules preventing team members from attending weekly meetings
    - Only schedule meetings during times when everyone is available
    - Team members will need to notify the team if they cannot attend a meeting
- Scheduling pitfalls
    - When creating schedule allow for padding in case some tasks take longer than anticipated
    - Keep track of how much time the tasks take

Design Architecture

Data Flow Diagram

Data Model

CoursePage = {

course_name: String

course_number: Integer

sections: [

syllabus: [CDN Link to PDF],

homework: [CDN Links],

schedule: [ ],

resources: [ ],

lecture_notes: [ ],

grades: []

students: [ ],

}

DepartmentPage = {

pages: [ ],

}

Professor = {

name: String

office: String

office_hours: String

courses: String

email: String

}