Library Room Availability Project

Jaiminkumar Desai, Vanshika Chopra,

Samantha Inneo, Divyamshu Mandadi

SSW 540: Fundamentals of Software Engineering

Final Report

18 April 2022

[Executive Summary](#_gk753vy95gkr) 2

[List of Figures](#_bav87lq8jgdr) 2

[List of Tables](#_fin1lb62hroh) 2

[1. Introduction](#_8nblfai2thcf) 2

[1.1. Purpose and Scope](#_stoiri3dnrbu) 2

[1.2. Product Overview (including capabilities, scenarios for using the product, etc.)](#_29m26jqlelit) 2

[1.3. Terms, Acronyms, and Abbreviations](#_4zdbyoltw3cq) 3

[2. Project Management Plan](#_24xm2ll3sury) 3

[2.1. Project Organization](#_btczv2gg3w5b) 3

[2.2. Lifecycle Model Used](#_glutqwx4qey2) 3

[2.3. Risk Analysis](#_lycbwr5a2bgw) 3

[2.4. Software Resource Requirements](#_w6xgeb17tr5a) 4

[2.5. Deliverables and Schedule](#_2nblnz6szuzm) 5

[2.6. Monitoring, Reporting](#_e9op6cm620so) 5

[2.7. Professional Standards](#_drysmbg3kg8j) 6

[2.8. Impact](#_2u8qomayfwti) 6

[3. Requirement Specifications](#_48s81n8ky5jt) 6

[3.1. Stakeholders](#_o4ks027lv6ul) 6

[3.2. Use case model](#_c9gdxrjcokkp) 7

[3.2.1. Graphic use case model](#_r9ap23km9ul2) 7

[3.2.2. Textual Description for each use case](#_52uywoeepouy) 7

[3.3. Rationale for your use case model](#_fhv8f410ip4x) 8

[3.4. Non-functional requirements](#_rnc82r3jf1nb) 8

[4. Architecture](#_s2mz4ov1cl36) 9

[4.1. Architectural model](#_aol91w8cseht) 9

[4.3. Technology, software, and hardware used](#_pdqbnvc3omqa) 9

[4.4. Rationale for your architectural model](#_uy9uuq3lzqde) 9

[5. Design](#_eswwdfelclhl) 10

[5.1. GUI (Graphical User Interface) design](#_6t7pqt0piso) 10

[5.2. Static model – class diagrams](#_gdk92ik9huqv) 10

[5.3. Dynamic model – sequence diagrams](#_ryauzluf90lv) 11

[5.4. Rationale for your detailed design model](#_f8un8zw811fu) 11

[6. Test Plan](#_bysqzjjsmwv8) 12

[6.1. Requirements/specifications-based system level test cases](#_l5uvpo6c81u) 12

[References](#_uhkpzeq3ybnh) 13

## Executive Summary

The Samuel C. Williams Library on the campus of Stevens Institute of Technology is always looking for ways to improve the students’ lives through innovative programs and technological improvements. The librarians have noticed other colleges have better systems for booking resources in their libraries and are looking for a similar technical solution. Currently, our library offers bookings of three group study spaces and six private study spaces on its website. However, the system for booking makes it difficult to see if there is currently available space without going through multiple steps of the booking process. The goal of this project is to provide a software solution similar to the ones implemented at other universities for viewing in real-time how many bookable spaces are available.

## List of Figures

Figure 1- Initial GUI design, page 10

Figure 2- Final HTML design, page 10

## List of Tables

Risk Analysis- Risk analysis matrix detailing likelihood, impact, and mitigation strategy of the associated risks, page 3

## 1. Introduction

### 1.1. Purpose and Scope

The essential purpose of this project is to improve the Samuel C. Williams Library site so it presents to the users the availability of studying spaces at any given point of time. This helps the librarians in managing those spaces better and helps students to plan their time accordingly. This project will only take into consideration the group study rooms, since there is a smaller number of these, making them in higher demand. The single study rooms are not included in the project, though with this code, the librarians can more easily implement this functionality later.

### 1.2. Product Overview (including capabilities, scenarios for using the product, etc.)

In the existing system for making a reservation, the user needs to click on the link “Book a room” on the library’s website, which takes them to a new page. Here, after filling out the given particulars, the availability of different rooms are shown. WIth the proposed system, the number of rooms available is displayed on the homepage of the website in real time, thereby letting the user know it at the beginning what is available, rather than after filling out the particulars. This helps the users by saving them time when the bookings are unavailable. It will also allow the librarians to see how many of the rooms are in use, which can be helpful for their daily tasks.

### 1.3. Terms, Acronyms, and Abbreviations

HTML - Hypertext Markup Language

Node Js- It is a server side platform that executes JavaScript code outside web browser

VS Code- Visual studio code

API - Application Programming Interface

## 2. Project Management Plan

### 2.1. Project Organization

For this project, we created a loose organization during planning, but due to different competencies, this organization was changed throughout development. Initially, Samantha was going to work on the fronted, Vanshika and Jaimin were going to work on python, and Divyamshu was to be the API admin. Due to availability, Samantha ended up as API admin after completing the HTML. Since Vanshika had more knowledge about using APIs, she ended up writing the NodeJS that functions as the backend of the project.

### 2.2. Lifecycle Model Used

For this project, we functioned under a waterfall model. We did not have frequently changing requirements or the need for prototyping, so agile’s biggest benefits were not as important to us. We still met multiple times with Romel to ensure our understanding of the library’s needs, but did not conduct sprints around these meetings.

### 2.3. Risk Analysis

| Title | Likelihood | Impact | Mitigation |
| --- | --- | --- | --- |
| Impact on other LibCal entities | Low | Moderate | When setting up with LibCal, we assigned the project to only have Read access from the APIs to avoid overwriting data by accident. |
| Lack of experience | High | High | Most of the group has little to no experience making software like this, so the learning curve on some aspects may be steep and derail progress. We will work together to determine who is best suited for different tasks and use resources external to the group to fill in the gaps. |
| Final Product Incompatible with Library Website | Low | High | We are using technologies that have previously been used on the library’s website to reduce the chance of integration issues. |

### 2.4. Software Resource Requirements

1. HTML- HyperText Markup Language also known as HTML is the main markup language for creating web pages and other information that can be displayed in a web browser. HTML is written in the form of tags enclosed in angle brackets(like<html>), within the web page content. HTML tags mostly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example<img>. The first tag in a pair is the start tag, and the second tag is the end tag. In between these tags the web designers can add text, further tags, comments and other types of text-based content. The purpose of a web browser is to read HTML documents and compose them into visible or audible webpages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. HTML allows images and objects to be embedded and can be used to create interactive forms. It can embed scripts written in languages such as javascript which affect the behavior of HTML web pages
   1. This was learned by Samantha.
2. Node.JS- Node.JS is a cross platform and open source javascript run time environment that allows the javascript to run on the server side. The usual way to run node.js is to run the globally available node command and pass the name of the file that needs to be executed. In this NPM plays a function as in it provides repositories for node.js modules and npm usually comes bundles with node.js. There is a simple syntax for installing modules using npm is $ npm install <module name>. We can use the npm ls command to list down all the locally installed modules. Package.json is used to define the properties of the package and are present in the root directory of any node module. Similarly in node.js there are various commands through which we can uninstall, update, change and create a module.
   1. This was learned by Vanshika
3. LibCal APIs- LibCal is a web platform designed to handle libraries calendaring needs. The four modules present work together to provide an integrated solution for location management, room reservation, event registrations and appointment scheduler. In our case we use the LibCal APIs to retrieve the booking data on the availability of spaces in real time.
   1. This was learned by Samantha and Vanshika.
4. Postman- Postman is a web platform that allows users to test their API calls before implementing them in code. This allows for developers to quickly check the results of an API call, which helps remove potential errors coming from the code. For this project, we used it to understand the results of the LibCal APIs, so we could properly process and display the availability of the group rooms on campus.
   1. This was learned by Samantha and Vanshika.

### 2.5. Deliverables and Schedule

| Deliverable | Date |
| --- | --- |
| Problem Identification | 02/22/2022 |
| Project Proposal | 02/26/2022 |
| Requirement Gathering | 03/10/2022 |
| Initial HTML | 03/28/2022 |
| API Functionality | 04/07/2022 |
| Testing | 04/15/2022 |
| Final Changes | 04/18/2022 |

### 2.6. Monitoring, Reporting

2.6.1. Monitoring: In monitoring there are two aspects in front end and back end. In the front end the monitoring service will try to ping the site using curl requests and see if it gets any response( i.e success code return). For the backend we would want to make sure API calls are successful. We will create mysql table where API calls return statuses to mysql table.

2.6.2. Reporting: We rely on data to get to know how many times the webpage is up and how frequently we are able to get return codes from the backend. This will be handled by the librarians and the IT department as they study this for the rest of the existing website.

### 2.7. Professional Standards

In this project, we used multiple best practices in order to keep our code clean. First, we utilized Github for version control and code distribution. We used multiple branches to protect the main branch, and always pushed our changes to individual branches before merging them with the main. Next, we tried to implement comments where necessary to improve the understandability of the code. The IEEE code of ethics were followed and maintained throughout the process.

### 2.8. Impact

The entire design ultimately benefits the students and staff of the library. Not only does it make the entire process more efficient, which in turn increases the productivity of the students, but also by using this ethos it could be expanded into solving any other inefficiencies that are being faced by the user. Any student who frequently studies in the library will benefit from the ease this application adds to the booking process.

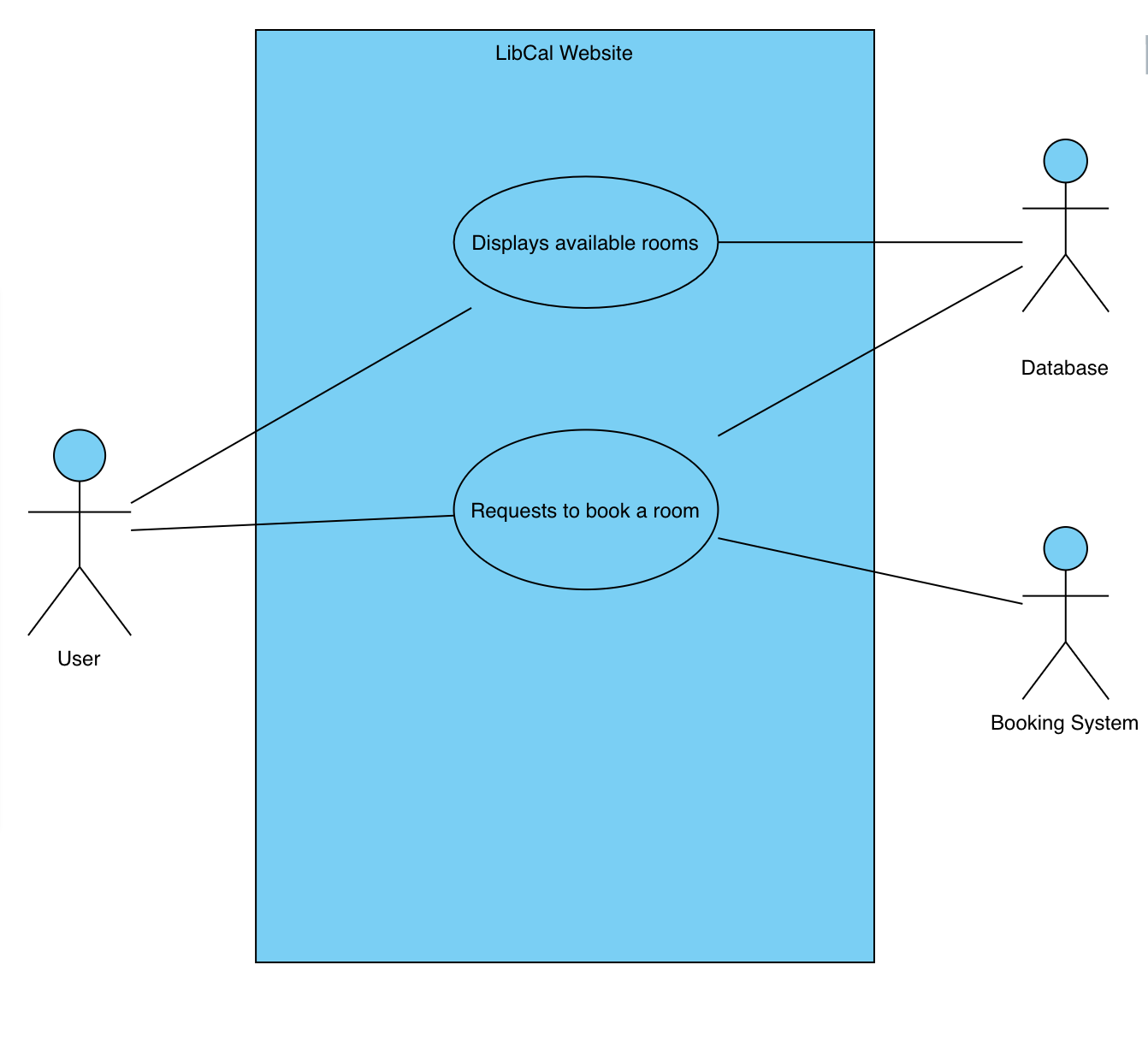
## 3. Requirement Specifications

### 3.1. Stakeholders

The first group of stakeholders is the librarians. This software is being made at their request in order to improve the library’s website. It will allow their library to function more smoothly as students will be able to begin the process of booking rooms straight from the library’s home page. The next group of stakeholders is the current and future student body at Stevens Institute of Technology. Students will be able to see how many of the campus spaces are available to be booked from the library’s homepage, while the current system requires them to click through multiple steps before even checking the availability. When no spaces are available, the students waste time going through these steps, whereas this new system will tell them immediately what is available.

### 3.2. Use case model

#### 3.2.1. Graphic use case model



#### 3.2.2. Textual Description for each use case

User - A user, who could be a student, professor, or librarian, enters and starts interacting with the system.

Database - The database is accessed through the LibCal APIs to receive real time data on the number of rooms available.

Display available rooms - It shows the exact number of group spaces available in real time.

Requests to book a room -The user can click to go to the portion of the library’s website so they can request an available space.

### 3.3. Rationale for your use case model

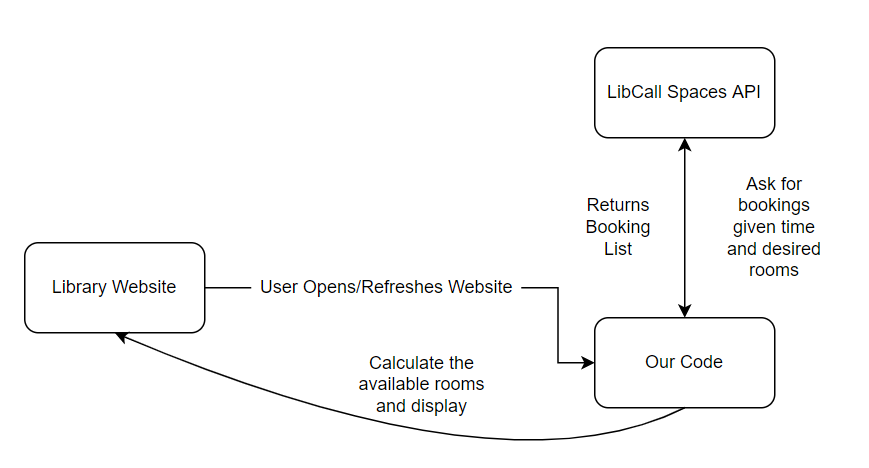
The functionalities of the system are captured in the use case diagram shown above. The actors here are the user, database and booking system. They are external to the system and interact with and perform use cases to accomplish tasks. The two use cases present have relationships and dependencies which are stated in the diagram. The user loads the page, which will cause our system to call APIs to retrieve the bookings from the database. We calculate the available rooms and display this back to the user. Our widget also links to the booking system, which allows users to efficiently book the rooms they can now see are available.

### 3.4. Non-functional requirements

1. Integrate into existing library framework
2. Utilize LibCal APIs
3. Must match existing color scheme

## 4. Architecture

### 4.1. Architectural model



### 4.3. Technology, software, and hardware used

1. HTML- structuring visual aspects of the code
2. Node.JS- backend API service
3. LibCal- APIs provided to retrieve displayable data
4. Github- code distribution and version control
5. Postman- testing APIs before coding

### 4.4. Rationale for your architectural model

When the user loads the page, the software must automatically calculate the number of open rooms. To do this, we must call the spaces/seat bookings API provided by LibCal. We can filter the bookings by the current time and by the room identifiers to determine the number of group rooms currently occupied. Once we have this information, we must subtract the number of occupied rooms by the number of total rooms, which will give us the number of available rooms. This number is displayed in our HTML.

## 5. Design

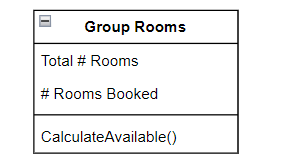
### 5.1. GUI (Graphical User Interface) design

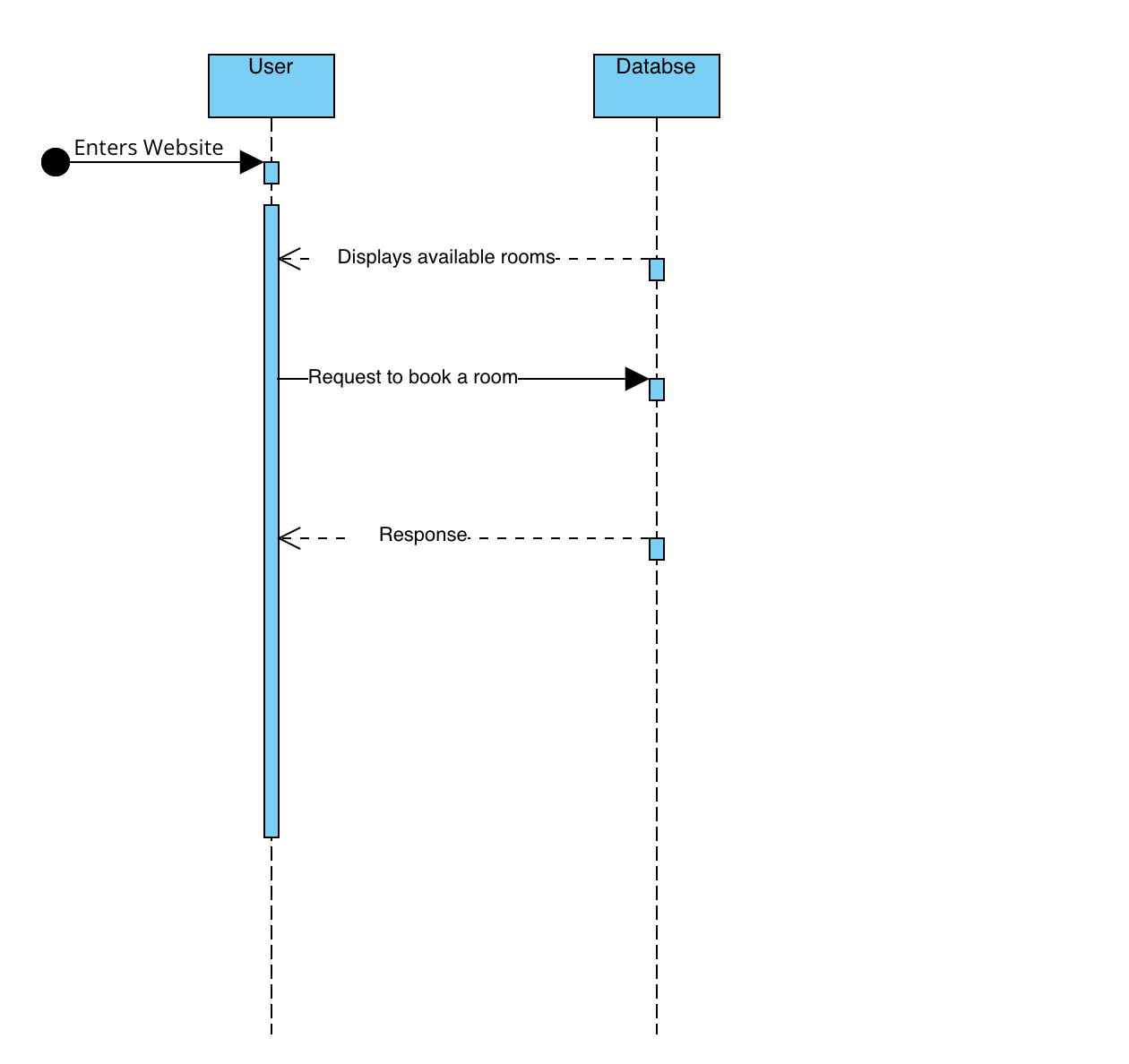
**Figure 1- Initial Design Figure 2- Final Design**

The most important aspects according to the client are that it is user-friendly and matches the rest of the existing website well. Our initial design is shown in figure 1, whereas the final design is shown in figure 2.

### 5.2. Static model – class diagrams



### 5.3. Dynamic model – sequence diagrams



### 5.4. Rationale for your detailed design model

The sequence model shows the interaction between the objects in a sequential order. A Found Message is used to represent a scenario and The Found message here is the “Enters website” and after the user enters the website the step by step sequence is shown. The reply messages are used to show the message being sent from sender to receiver in the form of a dotted line.There are two reply messages present in the system and one synchronous message. After the first synchronous message the response message is given out. The diagram overall helps in visualizing the logic behind the function and helps us understand the detailed functionality.

## 6. Test Plan

### 6.1. Requirements/specifications-based system level test cases

The main aim of testing is to determine the defects in our project. The program was subjected to a set of inputs and various observations were made and based on those observations the efficiency of the program was determined.

The project went through two levels of testing- Unit testing and Integration testing.

6.1.1. Unit testing: In this, each module is individually created, tested and reviewed. The need in this is also of a procedure to call the functions of the module under test with appropriate parameters. For our system the individual modules, the HTML code, as well as the node.js files were tested. The node.js file was tested at various times of the day to see if the availability of the rooms is changing accordingly. For the HTML file the check was conducted on whether the requirement is synchronizing properly in a web browser.

6.1.2. Integration Testing: In this type of testing we test with the various integration patterns of the project module. The primary objective is to test the various modules in order to ensure that no errors are occurring when one module invokes another. For integration testing we made a server and used it to call the API and the HTML rendering was done as well. The code ran according to the requirements.

## 

## References

Auth0, “Use access tokens,” *Auth0 Docs*. [Online]. Available: https://auth0.com/docs/secure/tokens/access-tokens/use-access-tokens. [Accessed: 17-Apr-2022].

B. Jones, J. Burns , and O. Anghel, “Creating layers on your web pages with divs,” *HTML Goodies*, 05-Aug-2021. [Online]. Available: https://www.htmlgoodies.com/css/creating-layers-on-your-web-pages-with-divs/#:~:text=When%20creating%20a%20web%20page,in%20one%20or%20more%20layers. [Accessed: 17-Apr-2022].

“Baruch College: Newman Library,” *Baruch College Newman Library*. [Online]. Available: https://library.baruch.cuny.edu/. [Accessed: 17-Apr-2022].

“HTML element reference,” *HTML Reference*. [Online]. Available: https://www.w3schools.com/tags/default.asp. [Accessed: 17-Apr-2022].

“IEEE code of Ethics,” *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 18-Apr-2022].

*Stevens Institute of Technology*. [Online]. Available: https://library.stevens.edu/. [Accessed: 17-Apr-2022].

Code can be found at: GitHub: https://github.com/samanthainneo99/SSW540Project