Progressive Web Application Development For Use In The TCNJ Library Application

William Roche

Submitted in partial fulfillment of the requirements of CSC298 Mentored Research
12/4/2023

Abstract:
This mentored research project furthered the development of a Progressive Web Application to augment the TCNJ Library mobile application, specifically focusing on refining study room motion sensors and implementing a printer and computer finder feature. The PWA continued to integrate real-time data communication with a backend database as it had in the first semester regarding study room availability. However, a significant problem we faced was inaccuracies within the study room data, mainly due to various issues with the Raspberry Pi motion sensors. After fixing these problems, we developed more features within the PWA, explicitly implementing the computer and printer finder features. This consisted of displaying a map of a given floor as a canvas, with printers and real-time computer availability displayed. The languages used in these features include HTML, CSS, and Javascript. Appendices are attached below for context.

Problem Statement:
The goal of this mentored research project was to continue the development of a Progressive Web Application, or PWA, that integrates real-time data and communication with a backend database to enhance services used in the TCNJ Library mobile application for study room, printer, and computer availability. This platform-independent application used tools such as Capacitor, the Ionic framework, and Angular during the development process. Work with the Ionic framework includes using Javascript, CSS, and HTML. This semester also saw the Raspberry pi zero 2 w use, coupled with a motion sensor. This project seeked to enhance the functionality of the TCNJ Library App for student and faculty use and also serves as an academic exploration of PWA technology.

Background:
The R. Barbara Gitenstein TCNJ Library Application was published on the app store 11 years ago. Since its creation, an expansive list of students has contributed to its development. The application was first developed by Art Malarcyzk and Jon Stoeber as a capstone experience at TCNJ. Updates were implemented in the Spring of 2013 by Frank Corradi, Brandon Gottlob, and Ken Whittaker. These updates were furthered by work done in the Fall of 2014 by Warren Seto and Fall of 2016 by Madison Martin. Dual-platform implementation was implemented during a capstone experience by David Shull, assisted by Sophia Goldberg in the Spring of 2017. Additionally, Sophia Goldberg implemented study room occupancy detection from Fall of 2017 to the Spring of 2018, assisted by Randell Corrido and Victoria Swartz. From 2018 to 2019, Ryan Bogutz performed work on backend development and data filtering. During this time, Jenna Oak implemented accessibility, room scheduling algorithms, module design, and database enhancements, and Gabrielle Curcio implemented room scheduling integration. From 2019 to 2020, production rollout, server support, the accuracy of occupancy detection, and computer availability upgrade were completed by Mark Middleton, assisted by Allison Russell in 2019, Lisa Walker from 2019 to 2020, and Kenna Stiesi in 2020. Application WiFi stabilization, updated computer mappings, additional backend development including serviced computers and computer hostnames error reporting, study room searching, Mac occupancy detection, VPN information, and other minor frontend features were implemented by Payton Shaltis from Fall 2021 to Spring 2022, and Mila Manzano in Spring of 2022.Parvathi Krishnan and Will Roche implemented PWA Implementation with a study room availability feature in Fall of 2023.

Methodology:

The majority of my contribution within this research project entailed learning about the concept of PWA's, with most of my work being related to the functionality and development of the study room availability feature of the app. Prior to my participation in this research project, my exposure to the technologies being used was limited, and I had not previously acquired knowledge pertaining to their intricacies. Therefore, in the beginning stages of the research, a large portion of my time was centered around familiarizing myself with these tools.
A Progressive Web App (PWA) is a type of application delivered through the web, employing languages such as Javascript, HTML, and CSS, making it versatile across various platforms with web browsers. Notable examples include Starbucks, Spotify, and Pinterest, widely adopted by major companies. PWAs present cost-effective solutions to web development issues, requiring only one version for all platforms. While lacking certain features present within Native Apps, like using personal contacts, Bluetooth, and NFC integration, PWAs benefit from search engine visibility and are launched easily without the complications of an app store. While Native Apps demand a full development team, incurring higher costs and adding complexity to the developmental process, PWA's can be developed with a much smaller team that doesn't need to take into account the complexities of individual development for individual platforms. Ultimately, the choice between PWA and Native App depends on specific project requirements, hence, a progressive web application was a perfect choice for this project. As a small development team with relatively limited funding, progressive web applications served as a promising avenue to explore for the TCNJ Library App.
As previously mentioned, Javascript, HTML, and CSS were used throughout the process.
JS, or JavaScript is a versatile programming language typically used within web development by enabling dynamic, interactive content. Employed in conjunction with HTML and CSS, it facilitates client-side scripting, creating responsive user experiences.
HTML, or HyperText Markup Language, is the foundational language for web development. Employed to structure content, HTML uses tags to define elements like headings, paragraphs, and links, ensuring proper presentation across browsers.
CSS, or Cascading Style Sheets, is another important language in web design. It complements HTML by styling elements, controlling layout, and enhancing visual aesthetics. CSS enables consistent presentation across various devices.
This project employed the use of Materialize CSS, a front-end framework which is based on Google's Material Design guidelines. Materialize was designed by a group of students from Carnegie Mellon University: Alvin Wang, Alan Chang, Alex Mark, and Kevin Louie.

Implementation:
Along with shadowing the developmental process of this research program, I was given the opportunity to assist with the development of the study room finder function within the Library App PWA. To begin this process, I had spent some time monitoring the motion sensors in the study rooms and tracking the given data within a spreadsheet in order to survey their functionality. In order to do this, I learned how to access the backend files. This knowledge of the backend comes into play with the functionality of the study room availability checker. The

process of accessing this live information entailed logging into the Knox Lab Server (appendix E), accessing the mySQL server and selecting the library app database (appendix F), and displaying a table of the current status of the respective study room's motion sensor (appendix G). The process of returning a live response of the availability of the study rooms within a program is tantamount to manually viewing the data through the backend server, therefore, the logical process is similar when writing the code for the program. This section of the program begins by connecting to the server and using the fetch API to make a request to the server (appendix H). Afterwards, the same API fetches a PHP file of the live data within the library app database for study room availability (appendix I). Following this, the application takes the php file and fetches the XML data (appendix J). Using this XML data, we can interpret and output the current availability of the TCNJ Library study rooms. Using this process, we can create an HTML table that outputs the available rooms based on a user given input. Within the HTML, I created a function called 'HandleInput', in which I store user given values for the number of students and floor that the user was searching for (appendix K). Using these values, I wrote a loop that checked the availability status of every room on the user given floor, and displayed the rooms that were open for use (appendix L). Following this, I altered the table within the CSS such that the visual aesthetics would match that of the rest of the library app (appendix M). When the user submits a form, the values given are passed through the function, and a corresponding table showing a live representation of the currently available study rooms is displayed.

Results:

The tools used during this process are standard for the development of progressive web applications, and as explained in the methodology of this paper, a PWA as an avenue of development is a fitting approach to explore based on our project needs and circumstances. My work in this project took place over the course of the first semester of my first year at TCNJ, specifically from August 31st 2023 to today. I began working on the PWA side of the research project on October 4th, following unfortunate issues with the progression of the backend. I shadowed Parvathi Krishnan's work until the second week of November, where I began my work on outputting availability based on user given input within the study room finder feature. This feature was completed by the end of November. The current state of the application is shown in appendixes A through D. The TCNJ Library App as a progressive web application has immense potential. The work completed this semester successfully constructed a strong foundation for the future development of this application.

Conclusions:

This mentored research project was successful in enhancing the TCNJ Library mobile application through the development of a progressive web application with real-time data integration and communication with a back-end database. Throughout this project, I was exposed to various new technologies as well as working as a part of a research team. Both of these concepts will undoubtedly aid me throughout my academic career. While overall successful, the work completed this semester leaves the application open to further development, which will come to follow next semester.

Future Work:

TCNJ Library App PWA will be built upon during the Spring 2024 semester through the implementation of other functionalities and expansion upon current functionalities. Additionally, various issues with motion sensor functionality within the library may be addressed. One possible advancement within pre-existing functions is the concept of introducing an interactive map within the study room availability feature, as well as the possibility of advancements with the format and user interface of the application.

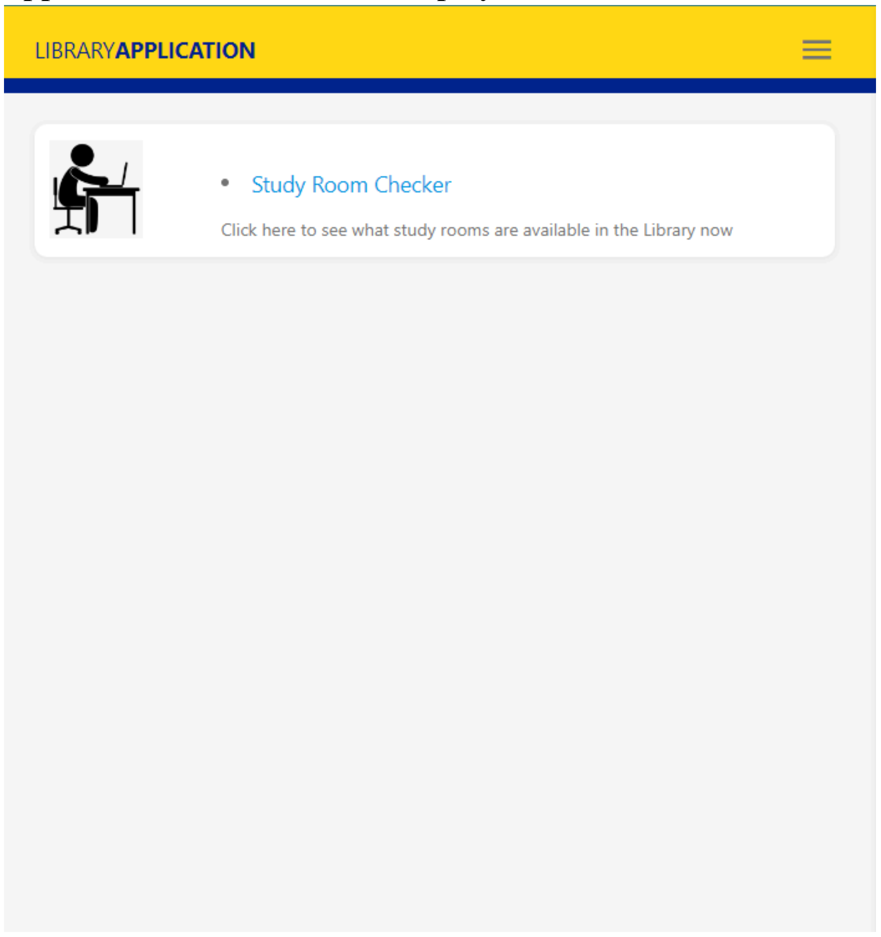
References:

CodeCrux. 6 best platforms to develop a web application in 2022. let's check. CodeCrux Web Technologies pvt ltd. https://codecrux.com/blog/what-is-the-best-platform-to-develop-a-web-application.html.
Grimm, S. 2021. Start, build &amp; deploy your first capacitor PWA with Ionic. Ionic Blog. https://ionic.io/blog/start-build-deploy-your-first-capacitor-pwa-with-ionic.
Hoang, M. 2023. The future of PWA for web development: What makes it promising. Magenest. https://magenest.com/en/future-of-pwa/.
Jelvix. 2023. Battle of the apps: Native app vs Pwa - who will win in 2023? YouTube. https://www.youtube.com/watch?v=DV-vVTkLgIU.
John, L.S., John, L.S., Vivek, J., and Vivek, J. 2023. What are the best PWA frameworks to develop Progressive Web Apps? Zuci Systems. https://www.zucisystems.com/blog/what-are-the-top-pwa-frameworks-to-develop-progressive-web-apps/.
Mehta, A. 2022. Progressive web apps vs native apps: Who is winning? Appinventiv. https://appinventiv.com/blog/native-vs-progressive-web-apps/.

Acknowledgements

Collaborators:
Mateo Calderone [Research Partner]
Parvathi Krishnan [Research Partner]
Deborah Knox [Research Mentor]
Previous students:
Art Malarcyzk, Jon Stoeber, Frank Corradi, Brandon Gottlob, Ken Whittaker, Warren Seto, Madison Martin, David Shull, Sophia Goldberg, Randell Corrido, Victoria Swartz, Ryan Bogutz, Jenna Oak, Gabrielle Curcio, Mark Meddleton, Allison Russell, Lisa Walker, Kenna Sties, Payton Shaltis, Mila Manzano

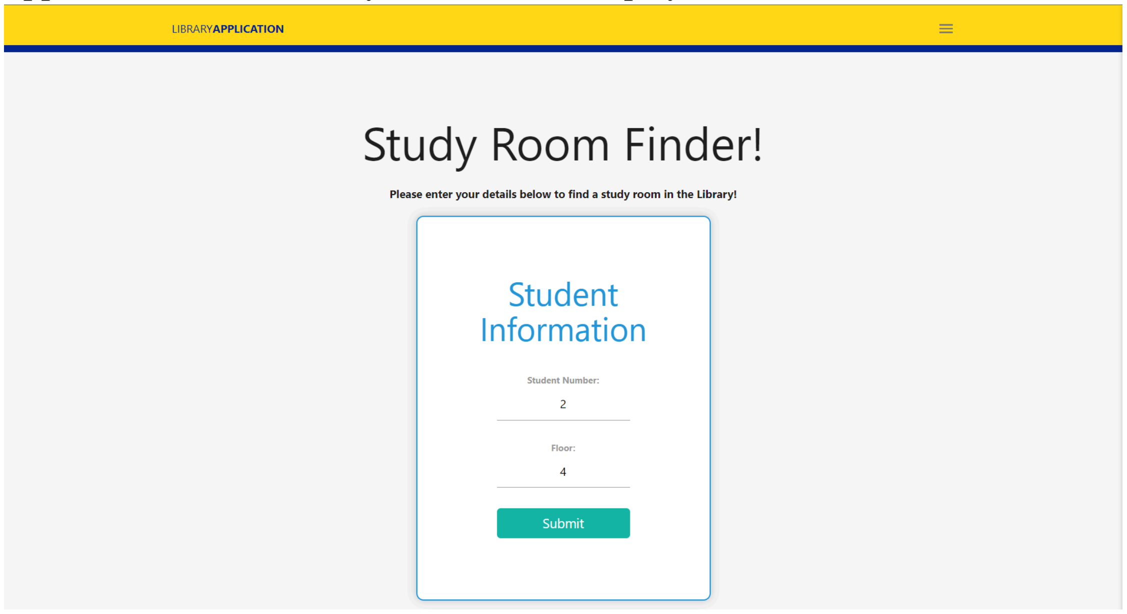
Appendix A: Current Home Display



- Study Room Checker

Click here to see what study rooms are available in the Library now

Appendix B: Current Study Room Finder Display


LIBRARYAPPLICATION

Study Room Finder!

Please enter your details below to find a study room in the Library!

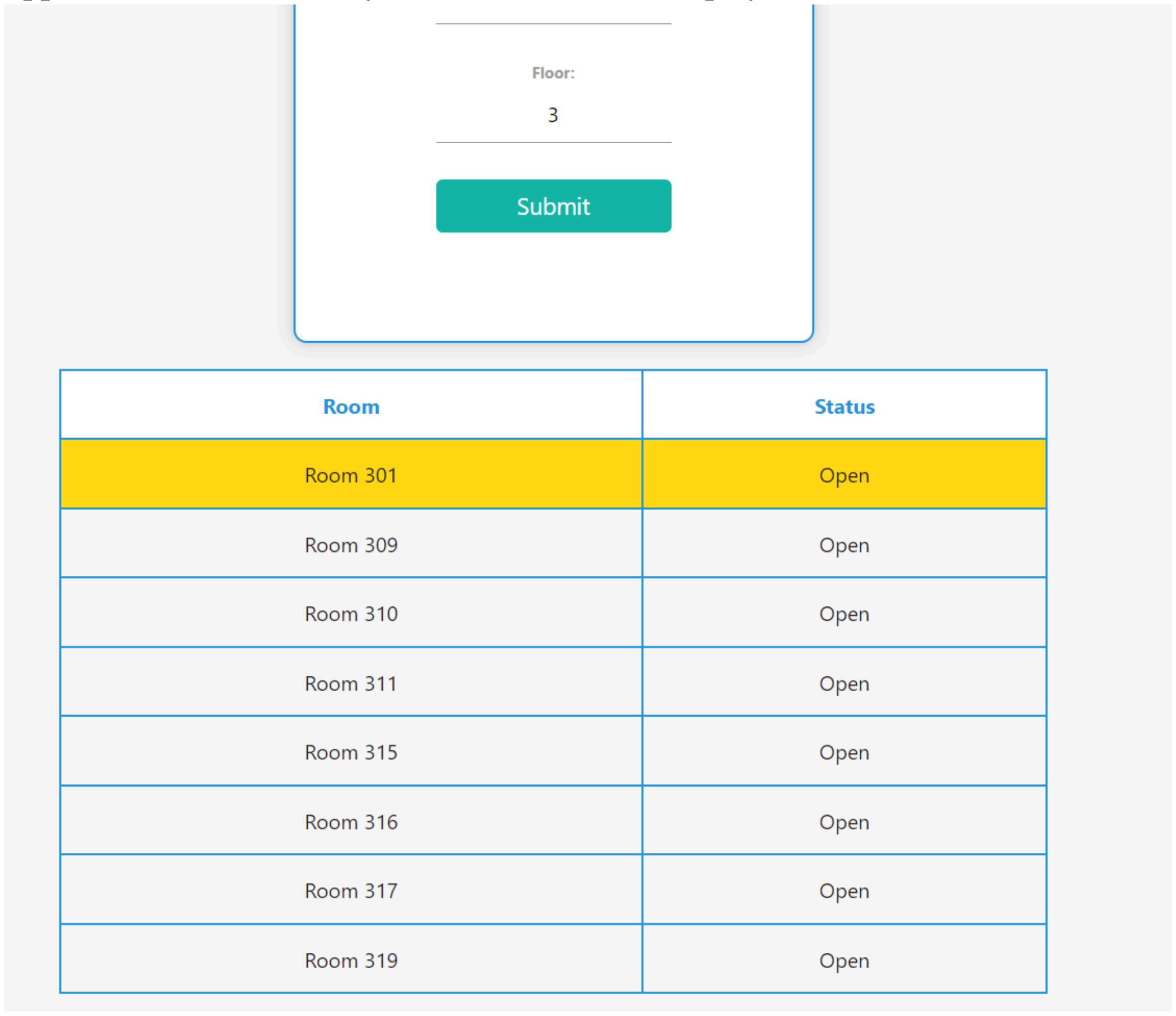
Student Information

Student Number: 2

Floor: 4

Submit

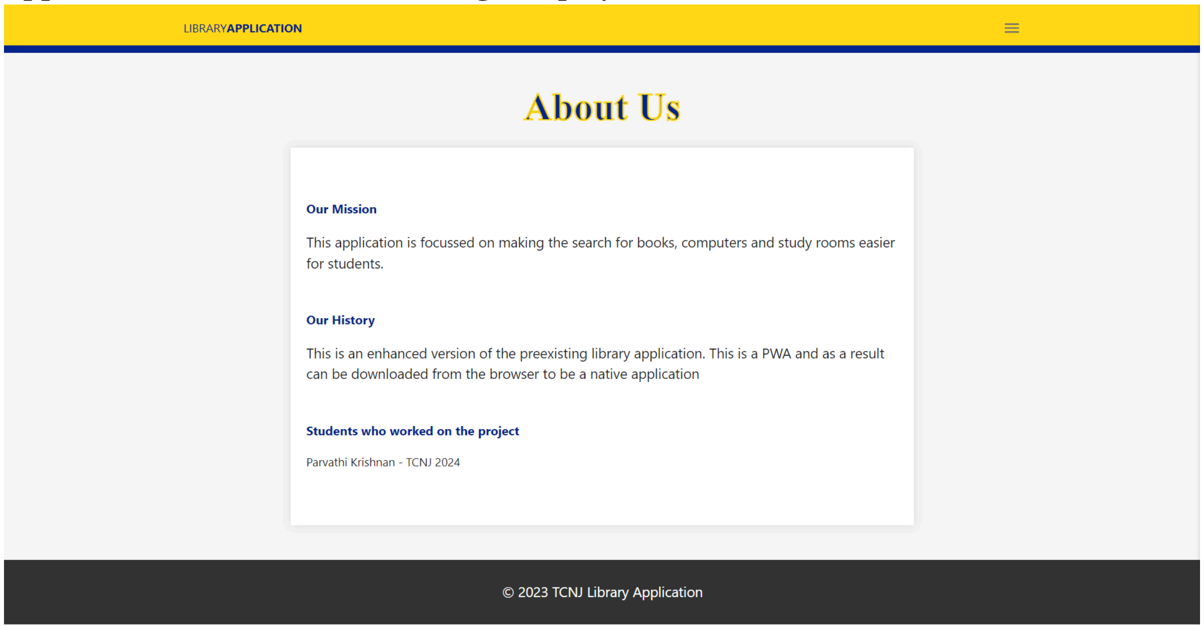
Appendix C: Current Study Room Finder Table Display


Floor: 3

Submit

Room	Status
Room 301	Open
Room 309	Open
Room 310	Open
Room 311	Open
Room 315	Open
Room 316	Open
Room 317	Open
Room 319	Open

Appendix D: Current About Us Page Display



About Us

Our Mission

This application is focussed on making the search for books, computers and study rooms easier for students.

Our History

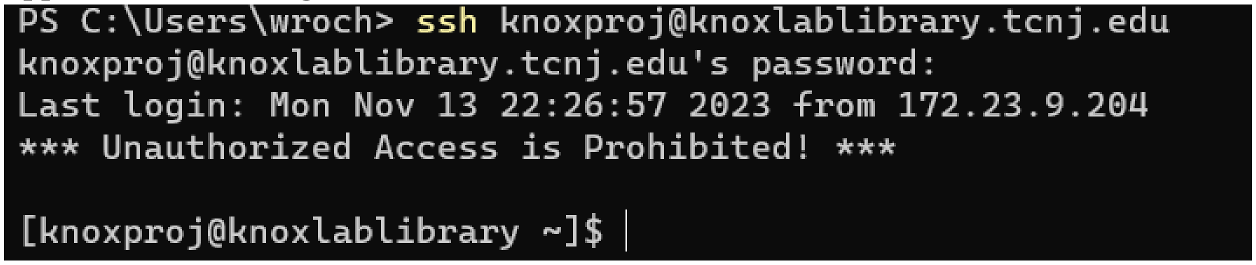
This is an enhanced version of the preexisting library application. This is a PWA and as a result can be downloaded from the browser to be a native application

Students who worked on the project

Parvathi Krishnan - TCNJ 2024

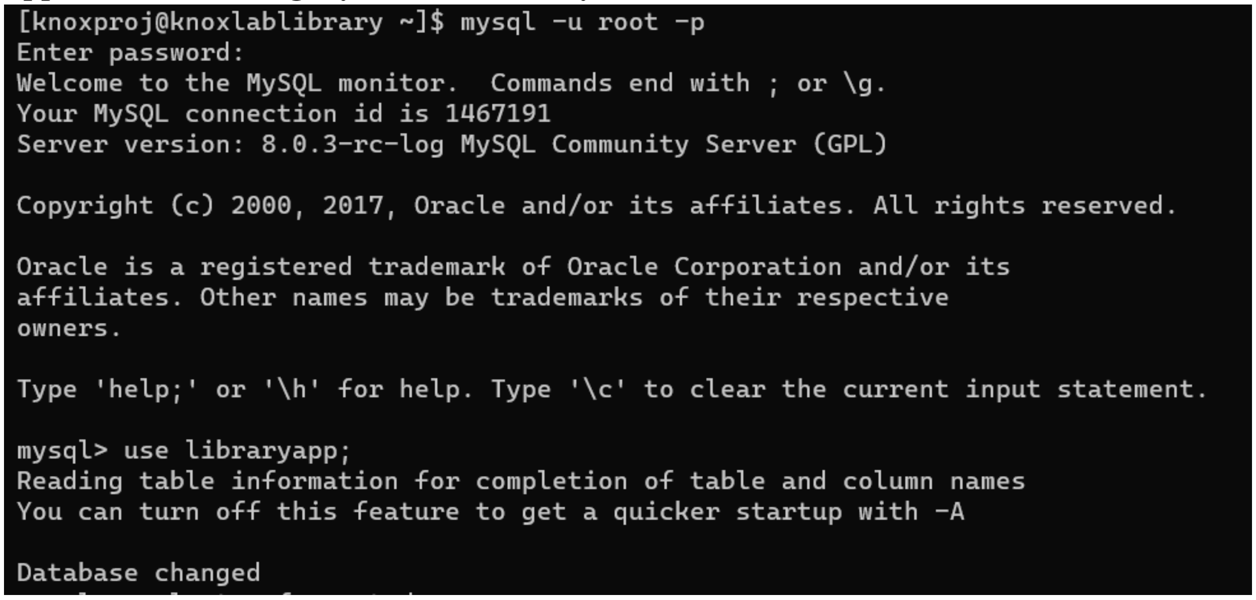
© 2023 TCNJ Library Application

Appendix E: Accessing Server



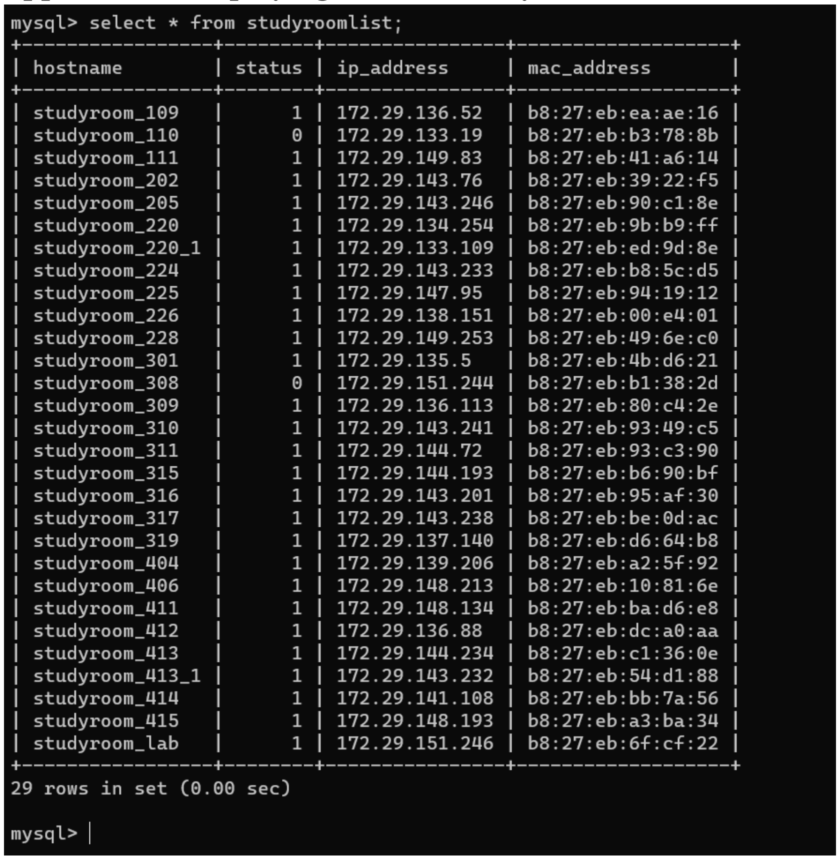
```
PS C:\Users\wroch> ssh knoxproj@knoxlablibrary.tcnj.edu
knoxproj@knoxlablibrary.tcnj.edu's password:
Last login: Mon Nov 13 22:26:57 2023 from 172.23.9.204
*** Unauthorized Access is Prohibited! ***

[knoxproj@knoxlablibrary ~]$
```

Appendix F:Accessing mySQL and Library Database



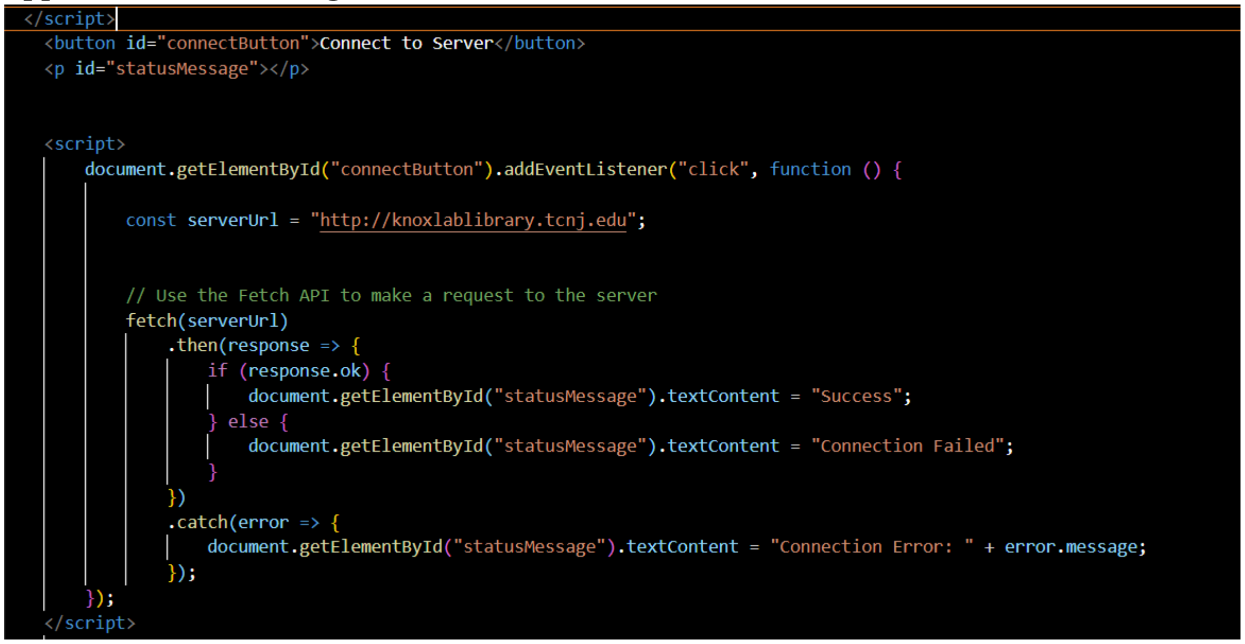
```
[knoxproj@knoxlablibrary ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1467191
Server version: 8.0.3-rc-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use libraryapp;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Appendix G: Displaying Table of Study Room Data



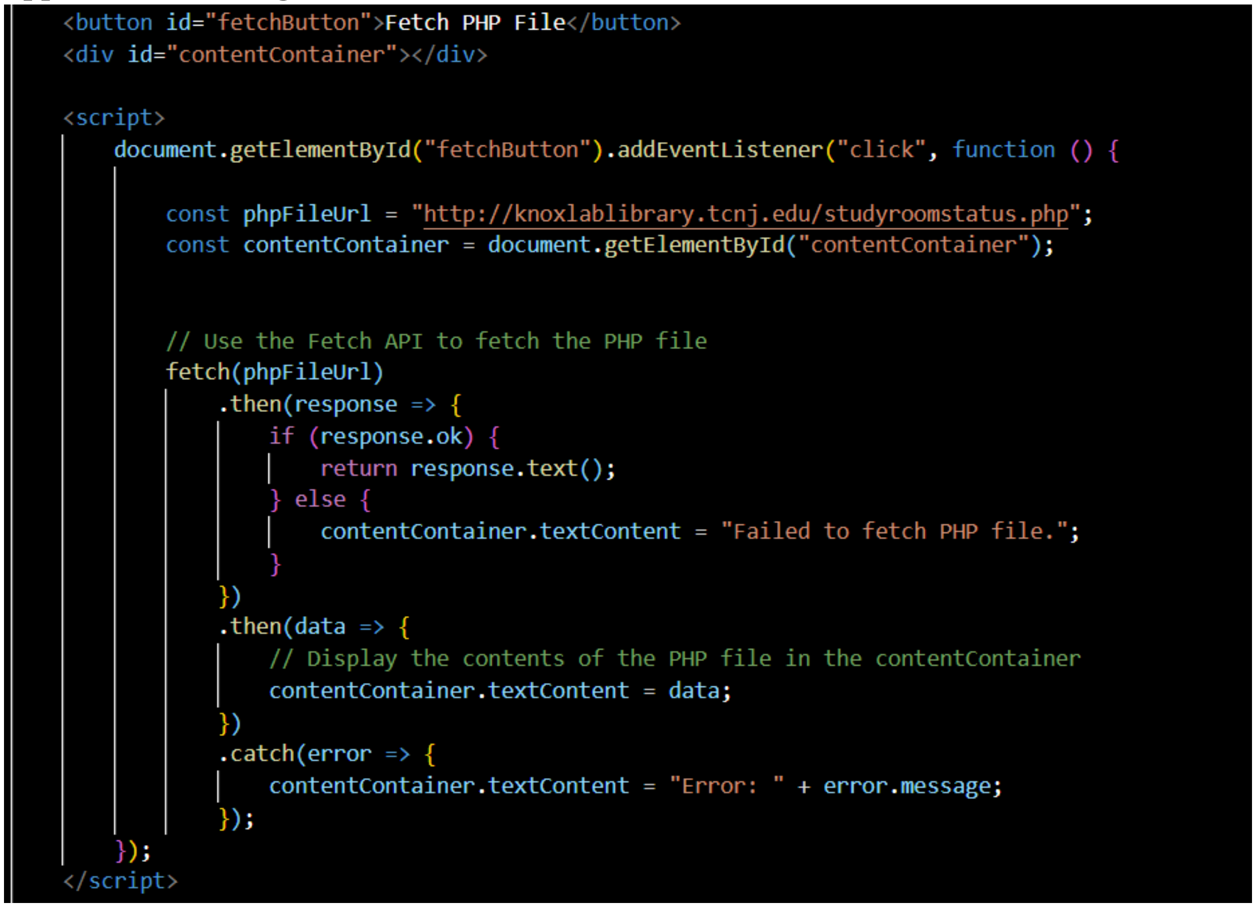
```
mysql> select * from studyroomlist;
+-----------------+--------+----------------+-------------------+
| hostname        | status | ip_address     | mac_address       |
+-----------------+--------+----------------+-------------------+
| studyroom_109   |      1 | 172.29.136.52  | b8:27:eb:ea:ae:16 |
| studyroom_110   |      0 | 172.29.133.19  | b8:27:eb:b3:78:8b |
| studyroom_111   |      1 | 172.29.149.83  | b8:27:eb:41:a6:14 |
| studyroom_202   |      1 | 172.29.143.76  | b8:27:eb:39:22:f5 |
| studyroom_205   |      1 | 172.29.143.246 | b8:27:eb:90:c1:8e |
| studyroom_220   |      1 | 172.29.134.254 | b8:27:eb:9b:b9:ff |
| studyroom_220_1 |      1 | 172.29.133.109 | b8:27:eb:ed:9d:8e |
| studyroom_224   |      1 | 172.29.143.233 | b8:27:eb:b8:5c:d5 |
| studyroom_225   |      1 | 172.29.147.95  | b8:27:eb:94:19:12 |
| studyroom_226   |      1 | 172.29.138.151 | b8:27:eb:00:e4:01 |
| studyroom_228   |      1 | 172.29.149.253 | b8:27:eb:49:6e:c0 |
| studyroom_301   |      1 | 172.29.135.5   | b8:27:eb:4b:d6:21 |
| studyroom_308   |      0 | 172.29.151.244 | b8:27:eb:b1:38:2d |
| studyroom_309   |      1 | 172.29.136.113 | b8:27:eb:80:c4:2e |
| studyroom_310   |      1 | 172.29.143.241 | b8:27:eb:93:49:c5 |
| studyroom_311   |      1 | 172.29.144.72  | b8:27:eb:93:c3:90 |
| studyroom_315   |      1 | 172.29.144.193 | b8:27:eb:b6:90:bf |
| studyroom_316   |      1 | 172.29.143.201 | b8:27:eb:95:af:30 |
| studyroom_317   |      1 | 172.29.143.238 | b8:27:eb:be:0d:ac |
| studyroom_319   |      1 | 172.29.137.140 | b8:27:eb:d6:64:b8 |
| studyroom_404   |      1 | 172.29.139.206 | b8:27:eb:a2:5f:92 |
| studyroom_406   |      1 | 172.29.148.213 | b8:27:eb:10:81:6e |
| studyroom_411   |      1 | 172.29.148.134 | b8:27:eb:ba:d6:e8 |
| studyroom_412   |      1 | 172.29.136.88  | b8:27:eb:dc:a0:aa |
| studyroom_413   |      1 | 172.29.144.234 | b8:27:eb:c1:36:0e |
| studyroom_413_1 |      1 | 172.29.143.232 | b8:27:eb:54:d1:88 |
| studyroom_414   |      1 | 172.29.141.108 | b8:27:eb:bb:7a:56 |
| studyroom_415   |      1 | 172.29.148.193 | b8:27:eb:a3:ba:34 |
| studyroom_lab   |      1 | 172.29.151.246 | b8:27:eb:6f:cf:22 |
+-----------------+--------+----------------+-------------------+
29 rows in set (0.00 sec)

mysql>
```

Appendix H: Connecting to Server



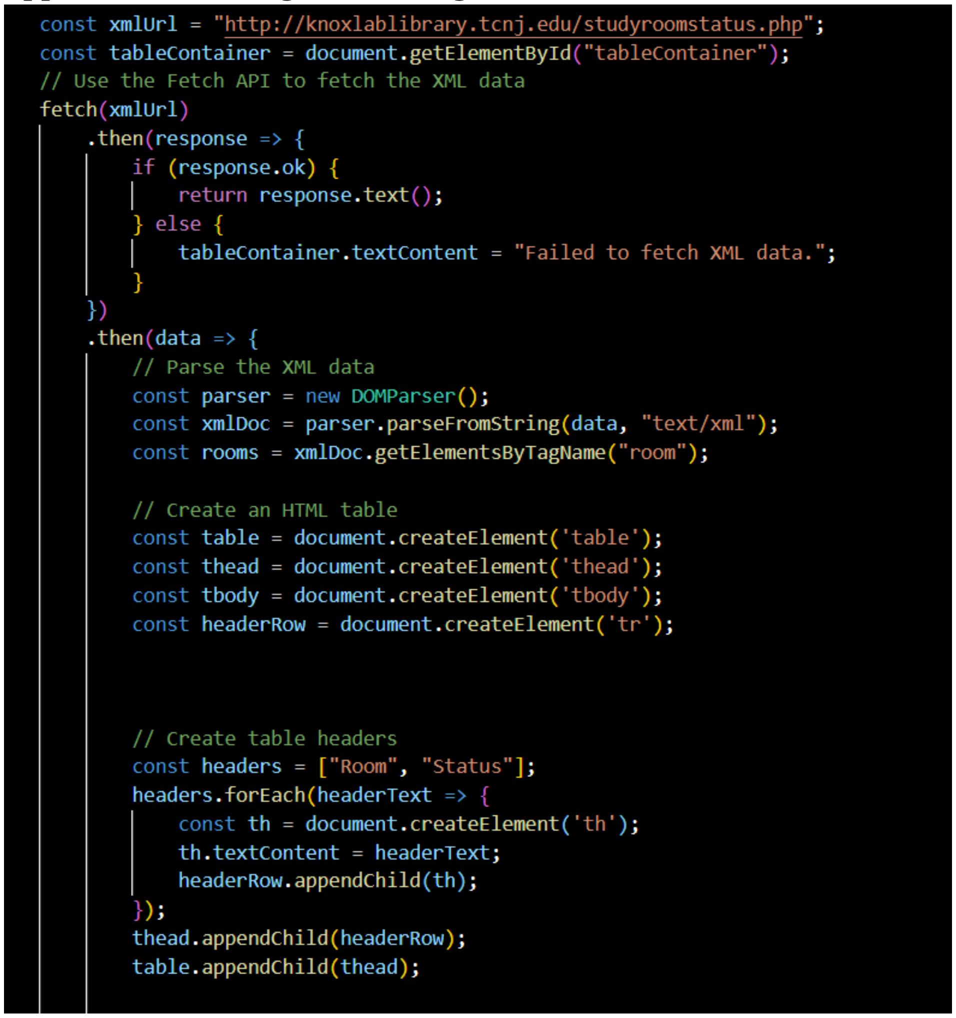
```
</script>
<button id="connectButton">Connect to Server</button>
<p id="statusMessage"></p>


<script>
    document.getElementById("connectButton").addEventListener("click", function () {

        const serverUrl = "http://knoxlablibrary.tcnj.edu";


        // Use the Fetch API to make a request to the server
        fetch(serverUrl)
            .then(response => {
                if (response.ok) {
                    document.getElementById("statusMessage").textContent = "Success";
                } else {
                    document.getElementById("statusMessage").textContent = "Connection Failed";
                }
            })
            .catch(error => {
                document.getElementById("statusMessage").textContent = "Connection Error: " + error.message;
            });
    });
</script>
```

Appendix I: Fetching PHP File



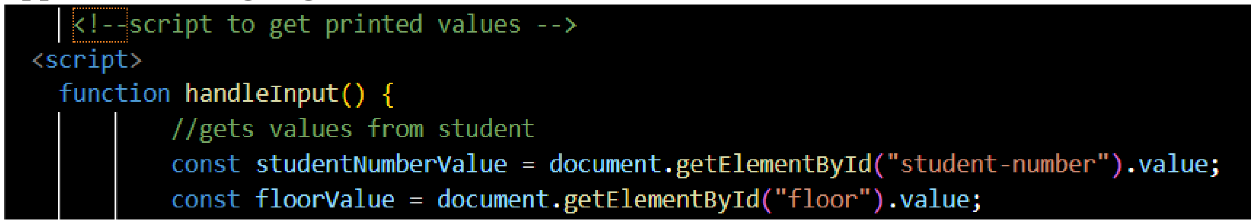
```
<button id="fetchButton">Fetch PHP File</button>
<div id="contentContainer"></div>

<script>
    document.getElementById("fetchButton").addEventListener("click", function () {

        const phpFileUrl = "http://knoxlablibrary.tcnj.edu/studyroomstatus.php";
        const contentContainer = document.getElementById("contentContainer");


        // Use the Fetch API to fetch the PHP file
        fetch(phpFileUrl)
            .then(response => {
                if (response.ok) {
                    return response.text();
                } else {
                    contentContainer.textContent = "Failed to fetch PHP file.";
                }
            })
            .then(data => {
                // Display the contents of the PHP file in the contentContainer
                contentContainer.textContent = data;
            })
            .catch(error => {
                contentContainer.textContent = "Error: " + error.message;
            });
    });
</script>
```

Appendix J: Fetching and Parsing XML Data



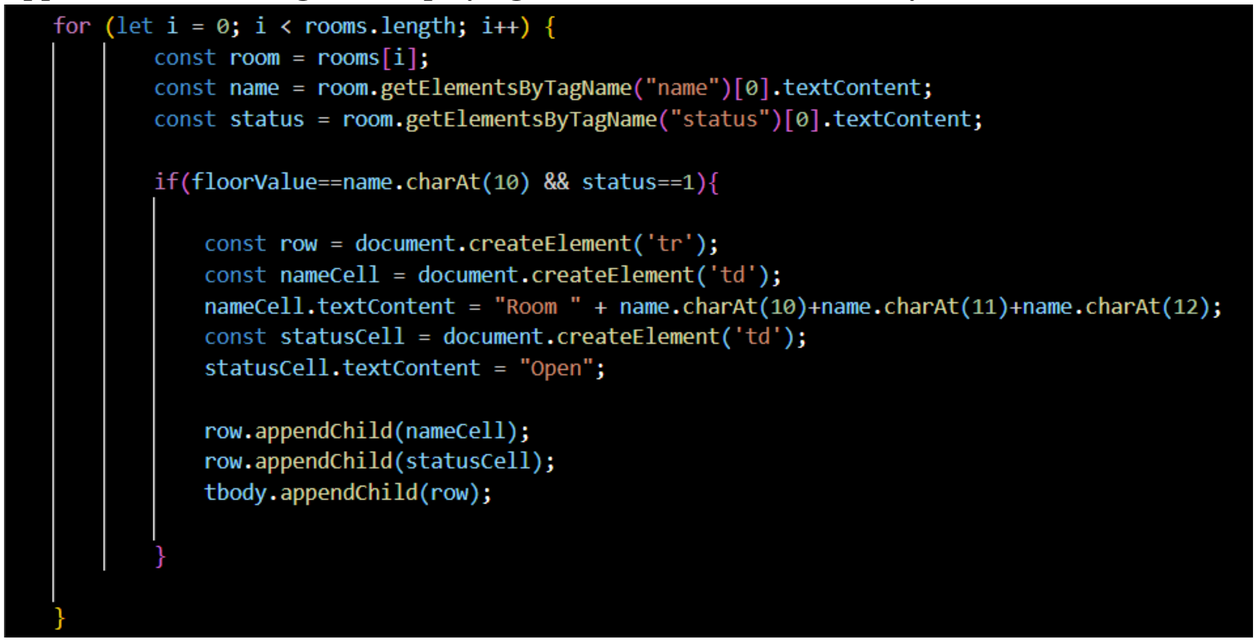
```
const xmlUrl = "http://knoxlablibrary.tcnj.edu/studyroomstatus.php";
const tableContainer = document.getElementById("tableContainer");
// Use the Fetch API to fetch the XML data
fetch(xmlUrl)
    .then(response => {
        if (response.ok) {
            return response.text();
        } else {
            tableContainer.textContent = "Failed to fetch XML data.";
        }
    })
    .then(data => {
        // Parse the XML data
        const parser = new DOMParser();
        const xmlDoc = parser.parseFromString(data, "text/xml");
        const rooms = xmlDoc.getElementsByTagName("room");

        // Create an HTML table
        const table = document.createElement('table');
        const thead = document.createElement('thead');
        const tbody = document.createElement('tbody');
        const headerRow = document.createElement('tr');



        // Create table headers
        const headers = ["Room", "Status"];
        headers.forEach(headerText => {
            const th = document.createElement('th');
            th.textContent = headerText;
            headerRow.appendChild(th);
        });
        thead.appendChild(headerRow);
        table.appendChild(thead);
```

Appendix K: Assigning Given Values To Variables



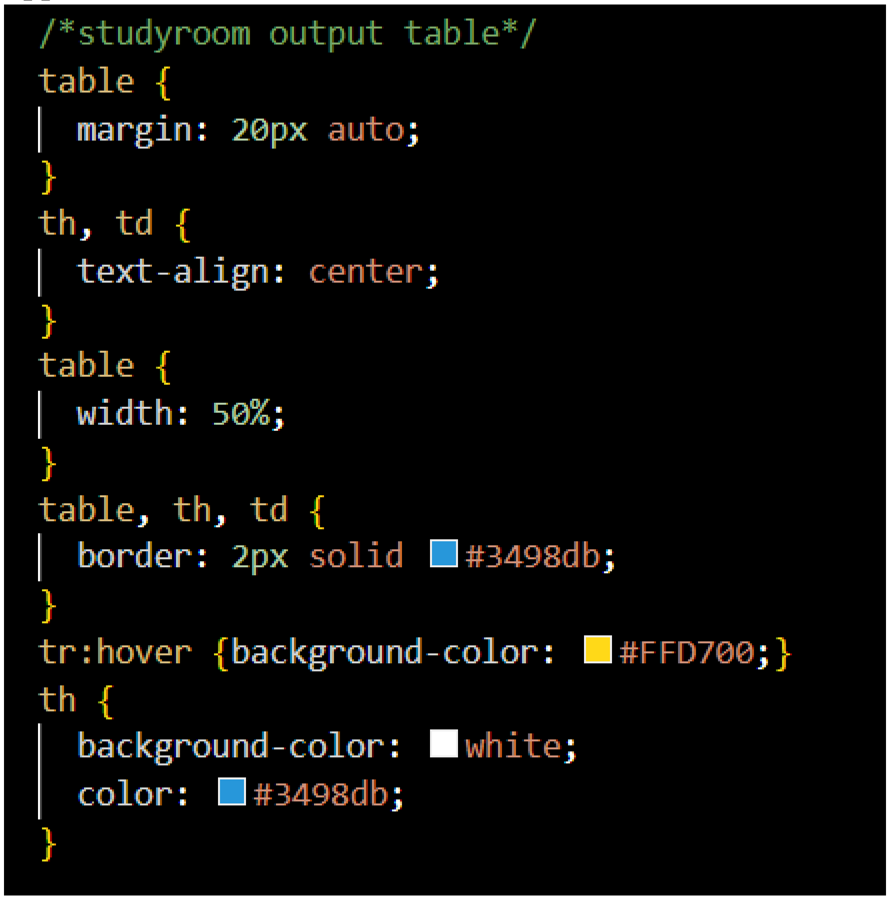
```
  <!--script to get printed values -->
<script>
  function handleInput() {
      //gets values from student
      const studentNumberValue = document.getElementById("student-number").value;
      const floorValue = document.getElementById("floor").value;
```

Appendix L: Checking and Displaying Rooms Based on Availability and Status



```
for (let i = 0; i < rooms.length; i++) {
        const room = rooms[i];
        const name = room.getElementsByTagName("name")[0].textContent;
        const status = room.getElementsByTagName("status")[0].textContent;

        if(floorValue==name.charAt(10) && status==1){

            const row = document.createElement('tr');
            const nameCell = document.createElement('td');
            nameCell.textContent = "Room " + name.charAt(10)+name.charAt(11)+name.charAt(12);
            const statusCell = document.createElement('td');
            statusCell.textContent = "Open";

            row.appendChild(nameCell);
            row.appendChild(statusCell);
            tbody.appendChild(row);

        }

}
```

Appendix M: Table CSS



```
/*studyroom output table*/
table {
  margin: 20px auto;
}
th, td {
  text-align: center;
}
table {
  width: 50%;
}
table, th, td {
  border: 2px solid #3498db;
}
tr:hover {background-color: #FFD700;}
th {
  background-color: white;
  color: #3498db;
}
```