Joe Fodera, Felipe Alves, Alex Kuang

ITWS1100

Professor Plotka

4/19/2023

**College Exam Hub Project Writeup**

**Proposal Summary**

College Exam Hub is a website meant to be a central hub for back exams at RPI, with the possibility of expansion to other colleges given more in-depth development. Its purpose is to facilitate the search and access to back exams for courses at RPI by storing exams on file along with ones uploaded by the users. This application would be centered around, but not limited to, college students at universities. However, it would not be locked to people who do not fall into that category. Users have the option to confirm their university email, which gives them additional functionalities such as uploading exams, commenting, and rating them. We were inspired to create this website due to the difficulty of finding relevant back exams as curriculum changes make older exams irrelevant. Searching on the web is time-consuming and usually unsuccessful, and the only alternative to searching for back exams online at RPI is by going to APO at the Union and going through a file cabinet of exams and finding a physical copy of an exam.

**Project Description**

RPI students can view and download back exams through this website. They have the option to log into their university email, and if they do so, they can also rate exams already on the website, as well as upload their own exam files to the course page that they specify during the uploading process. A search bar at the top of the home page, the page designated for uploading exams, and each course landing page allows users to search the course they wish to find exams of. The search bar takes the user to a course landing page; each course has one, and is where users can access the course’s available exams, as well as see some information about them, such as their average rating, the number of users who left a rating, as well as the year the back exam was given. From here, clicking on an exam takes the user to the exam’s exam page, where the user can leave a rating on a scale of one to five, along with viewing the information associated with that exam.

**Notable Features And How They Work**

One notable feature in our application is the search bar. It was implemented using a standard JavaScript file that takes advantage of absolute links. This is to ensure the back-end code can be included onto multiple pages without issues. Within this file, our 4 prototype courses are defined as possible destinations for the user. To help the user type these queries in exactly, autocomplete functionality was included in the same file. Every time the user edits the value of the input element (the search bar) a function is called that compares the value to the possible options. Overlapping characters are bolded. When the user clicks on the suggestion, it populates the value of the search bar with that suggestion. All of the aforementioned functionalities are defined within the *headSearch.js* file.

Another feature we would like to talk about is our implementation for uploading exams. The feature to upload their exam is only available and visible to users who verify their email and it’s located on the find exams page. There, they will be able to drag and drop a file into the target box due to event listeners monitoring for user interaction and executing functions based on what the user does with the file. Once the user hovers over the target box, the borders around the target box and the background color for the target box change to a different color/style, which shows that the target box is active and ready for the user to drop the file in. While the raw file is not currently stored anywhere, using the .dataTransfer Javascript method, the name of the file dropped is displayed to the user after the event listeners change the target box’s original contents to the contents related to the dropped file. From there, the user can select the course the exam is for from the drop-down and press the upload button to simulate the upload of the exam to the database. If the user would like to remove the file dropped, they would just click on the file itself and it would allow them to drop in another file. Just like common drag and drop boxes, the user also has the option to select a file from their own file explorer instead of dragging and dropping in files, and this was done by a simple input tag that has event listeners attached to it, monitoring change to the input value and changing the contents of the target box to the relevant information about the file.

Another notable feature we implemented is every course landing page’s usage of JSON, AJAX, and Javascript that allows for users to filter and sort the exams they see. Each course landing page has a JSON file containing information about each exam - the name, title, type (ex. exam 1, midterm, etc.), year it was given, overall rating, number of ratings, and the path to the exam’s landing page. All the course pages use a common Javascript file, courseHome.js, that reads their respective JSON file and makes an array with each element representing an exam, named examArr. The Javascript file then takes the values of the <select> elements that create the drop-down menus for filtering and sorting the exams, examType and sortBy, respectively. With examType, courseHome.js uses the filter function to remove elements from examArr that do not have a matching exam type, then calls the displayExams function, which is what writes HTML to the course page file that creates elements for each exam. With sortBy, courseHome uses the sort function based on the elements in examArr’s examYear value to rearrange examArr before calling the displayExams function.

**Planning And Execution**

We made a Gantt chart that laid out the tasks we had to do, such as create the web pages for the application with HTML, add functionality with Javascript, etc. When we made it, we did not have a good idea of when everything was due or how long these tasks were going to take. Although making the chart was helpful for grasping the scope of the whole project, we ultimately did not update or utilize it that much. Our primary methods of communication were an iMessage group chat, our team discord, and GitHub issue tracking. These along with regular group meetings for allocating tasks and discussing issues and ideas allowed us to stay on top of everything. One specific method we used to keep our separate work consistent was having an "overall goals'' issue on github. This had separate sections for different goals of our project and was something we constantly updated. It helped us greatly in ensuring we were all on the same page. Joe worked on the search bar, the home page, and the headers and footers, Felipe worked on the exam upload functionality and the page listing the courses, and Alex worked on the course pages and making them interactable. Joe and Alex worked together on the exam landing pages. For each of these features, each respective person working on them used HTML, CSS, and Javascript; we divided our workload based on tasks rather than front end versus back end.

**Information Architecture**

In terms of our file organization, we have a folder called CEH where we have the index.html file (which is the home page), a resources folder, and a pages folder containing all the other pages for the website. Inside the resources folder, we have images, and a Javascript file enabling jQuery, and CSS and Javascript files for the header and footer, and homepage. Everywhere in our project, we made sure to keep the names of corresponding files the same, only differing in the file extensions. This made it easier for us to track changes and easily find each corresponding file for each page. Speaking of pages, in our pages folder we have the findexams folder that holds the page where the user can either search for an exam or drag and drop a file from their computer to upload one (only if the user verified their university email). Then, we have the coursePages folder that contains the landing page for each course, and inside this folder, there are also folders for each course so it would make it easier for us to find the landing page for each course. A detailed map of these individual course folders is located [here](https://drive.google.com/file/d/1rgux3qVKnk6zG8gdAMj3ORbIatCc3tfA/view?usp=sharing). Each course-specific folder has a folder that contains the exams for that course; we decided to do this instead of having a folder with all the exams in the main CEH folder because we wanted to organize our files by the courses they are for. Going back to our pages folder, we initially had an exam-specific folder that was supposed to hold the course page and its contents of each exam page, but we chose to get rid of this folder, and just chose to store everything in the folders for each course page that is inside the coursePages folder. We decided to do this because every exam page would have the same format with the only difference being the exams being shown. In this folder, we were also planning to have a database file containing our database with the exam, but we chose to remove it as we changed our focus area and weren’t going to focus too much on storing the exams. Finally, we included resources folders in most folders which made our lives much easier when it came to organizing the files for each HTML page inside the folders that make up the pages folder.

**Challenges**

We were a bit too ambitious with the features in our mockup application we wanted to add, and ultimately had to put aside several features, such as the comments section and the user login page, to ensure that we could finish every task we wanted to complete and have a presentable mockup. We also were unsure about our two focus areas in the early stages of making our project - our two focus areas initially were pulling real data from a database (Area 4) and using HTML and CSS (Area 1), but as we made progress, it made more sense to switch from Area 4 to Area 2 - using Javascript to improve the user’s experience.

Each team member ran into their own challenges when making the notable features mentioned above. A challenge Alex ran into was figuring out how to make the course pages interactable with filtering and sorting exams. The process of taking in JSON data, taking in values from the HTML file to properly manipulate it with Javascript, and displaying it correctly was difficult due to having to learn to use several new Javascript and jQuery functions. Both Alex and Joe also ran into problems with the page footer not being at the bottom of the screen or covering content.

A challenge that Felipe ran was to make the actual drag and drop somewhat functional. Many event listeners had to be added to the target box to handle every single user interaction and since we couldn’t really store the file, he had to find a way to get the file dropped inside the target box to stay there without opening another tab. Also, there wasn’t much information on the events that had to be added to the event listeners to simulate the drag and drop feature and it was very confusing to decide on which events to use, as there weren't many examples online. Another challenge was to figure out how to get information on the file such as the file name and other specifics but after some trial and error, I figured it out.

Lastly, Joe ran into many errors while attempting to implement the autocomplete functionalities of the search bar. Initially, when the user would edit the value of the search bar, new suggestion divs would be added on top of the already existing ones. This was an issue because each time the user edits the value of the search bar, it should delete all previous suggestions and create new ones. Joe fixed this issue by calling the closeAllLists function at the beginning of each event listener call (when the value of the search bar was changed). Another issue Joe faced was trying to make the headSearch.js file work with all of the pages we wanted it on. Joe then remembered absolute links from in class and that solved the issue pretty efficiently.

**Summary**

College Exam Hub is a website where users can view, download, rate, and upload back exams, and it was designed to help students, mentors, or simply anyone struggling, quickly find good study material. Through making this website, we became more familiar with HTML, CSS, Javascript, jQuery, AJAX, and GitHub.

**What’s Next?**

We plan to add a comments section in the exam landing pages so that users can help each other by leaving questions and having discussions. We also want to add real back-end functionality to user ratings so that users’ ratings are saved and affect the ratings of exams, as well as the exam upload page such that when a user drags and drops an exam file, it is stored and a landing page is created for it. We plan to make these things possible through parsing and modifying JSON files, along with implementing a database. We also intend to add proper functionality to the “confirm email” button by actually having the user enter their email and confirm it with us. Additionally, we would like to make this application usable with other universities and implement bots or a system that gives users moderator status to keep the website organized by getting rid of unrelated content and moving exams to the right course page if someone uploads it in the wrong place.