Milestone 3: High-Fidelity Prototyping

November 14 2024

Introduction:

The prototype was built using the languages HTML, CSS and JavaScript. The JavaScript library React was used to create interactivity and functionality, while HTML and CSS were used for design and layout of the website. We collaborated using github and utilized separate branches to handle conflicts in our repository.

The major features of our interface are viewing program requirements, searching for available courses, building a plan of courses for each year of university, and building a weekly schedule of courses to enroll in.

Horizontal and Vertical Prototypes:

Major Features

The three pages of the interface are the Planner page, the Home page and the Schedule page. We chose these pages following feedback of our paper prototype, which used a similar structure. We will now go into detail on each page and the major features they provide.

The Planner page is the most unique part of the prototype. During the Investigation phase of the user-centered design process (UCD), research showed that many students struggled to easily find the proper courses needed to graduate. Adding this page to the website is an attempt to fix this issue. The Planner page corresponds to the task of planning courses from Milestone 2, and will be the first vertical prototype of this Milestone. This task is performed by first navigating to the Planner page using the “Academic Planner” button on the left. Then, the user can use the Requirements dropdown tree to view their requirements to graduate.

Next, the user can search for classes to add to the planner by using the search bar and corresponding filters. These filters are “All” (which does not filter out any courses), “Required” (which only displays required courses when searched), and “Planned” (which only displays courses that have been previously planned when searched). When a user clicks on a course that has been searched, the interface presents additional course details, and the option to select which year of the planner the course will be added to.

After pressing the big green button, the course is added to the planner for that year. Planned courses can be viewed all at once, or on a yearly basis.

Figure 1: A view of the planner with all years visible

In Figure 1, the "All" button has been selected, which means planned courses for all years will be displayed. The user can press the "+" and "–" buttons to add and remove years as they choose. The total number of years has a maximum of six and a minimum of one. Courses can be removed from the planner by clicking on them (a trashcan icon appears on hover).

The next two pages comprise the second vertical prototype which corresponds to the task of enrolling in a course from Milestone 2. This vertical prototype is implemented as a combination of the Home page and the Schedule page. The search component of the Home page is nearly identical to that of the Planner page, with the only difference being that a searched course will be added to the course shortlist for the current term. The shortlist is displayed on the right side of the page. The user can click the trashcan icon to remove a course from the shortlist. Once they are satisfied with their shortlist, they can proceed to the Schedule page by clicking the "Proceed to Schedule" button. Once on the Schedule page, the user can see their shortlist from the previous page, along with a weekly schedule using the courses from the shortlist. The user is allowed to add as many courses as they choose to the shortlist, however the weekly schedule is hardcoded to only display the first five. On this page, the shortlist can be clicked, displaying additional information about the course, including the professor and the number of seats available.

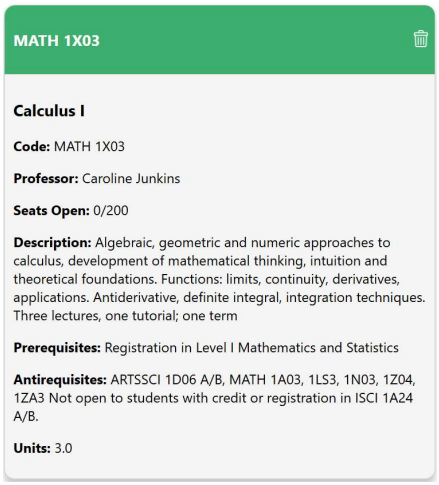

Figure 2: Course details for a class that is full

On the right side of the page, the weekly schedule is displayed with one possible combination of the courses in the shortlist.

Horizontal Features

There are several assets that are not functional in the prototype by choice. They are included as the horizontal prototype and together create a full page layout.

The course data hardcoded in the prototype assumes the user is a Math and Computer Science student. Thus, we have only added course information for a select few Compsci and Math courses. There are also two elective courses for Astronomy and Music Therapy (these are filtered out when the "Required" filter is chosen).

Following feedback from Milestone 2, we added a "Sign Out" button to the header to indicate the student is signed in. If this website were used in production, the user would have to sign in, and their requirements would change based on their program. However, for the scope of this prototype we just added a hardcoded button. Additionally, a help button was added to the layout per our user requirements.

Figure 3: Sign Out and help buttons which have no functionality

On the schedule page, we have omitted the ability to view many schedules built from the selected courses. This has been modified from our paper prototype following feedback we received. Before, you could only shuffle to view a random schedule, now there is a forward and a back button. We also have not implemented what comes after pressing "Enroll" for the purposes of this prototype. The buttons for these features are still part of the layout with no functionality.

Figure 4: Change schedule and enroll buttons which have no functionality

Strengths and Coding Solutions:

We are proud of the many difficult solutions we found to implement strong features of our interface.

First, maintaining user data throughout pages. Following feedback from our milestone 2, we wanted to give users the ability to see their planned courses when enrolling. However, being on different pages, keeping this data consistent wasn't easy. We implemented a routing system using react-router-dom. This allows us to pass the value of the planned courses to the Home page when the user switches pages. Additionally, following the same methods, we were able to pass the

value of the shortlist courses between the Home Page and the Schedule page. Therefore, this enabled us to build an innovative design and the user is able to move back and forth between the Home page and Schedule page while enrolling.

Second, creating an interactive tree hierarchy of requirements. We knew from our paper prototype this would be a must-have feature to provide the user with an easy and interactive view of their requirements. However, we underestimated the coding difficulty to create something that seems simple on paper. Our solution involves reading course data from a JSON file and storing each category and individual course in their own tree node. Each node has a parent node, children nodes, and a state of isOpen. From these nodes we created a recursive system to view the entire tree.

Finally, building search filters to properly filter the searchable courses was difficult. This required passing the selectedFilter variable and function to set the selectedFilter as props to the Filters component from the Home page (called CourseSearch) component. In the Home page component, the selectedFilter variable could be used to return courses in the JSON file with the same filter value. This worked for filtering by requirements and all courses. However, to filter by planned courses, as mentioned before, the planned courses had to be passed into the Home page component by the react-router-dom package. Then when selectedFilter was equal to “planned”, the component would return only the passed in planned courses.

Problems and Known Bugs:

Although there are many strengths of the system that we are proud of, there are also bugs that we are aware of. The main one that we know of is that the width of the system exceeds the width of a maximized window when a computer screen is set to scale beyond 100% in 1920x1080. This does not affect any other assets in terms of functionality, though it does allow the user to scroll horizontally. All three pages of the system are affected by this bug, so viewing on a screen set to 100% scale is recommended.

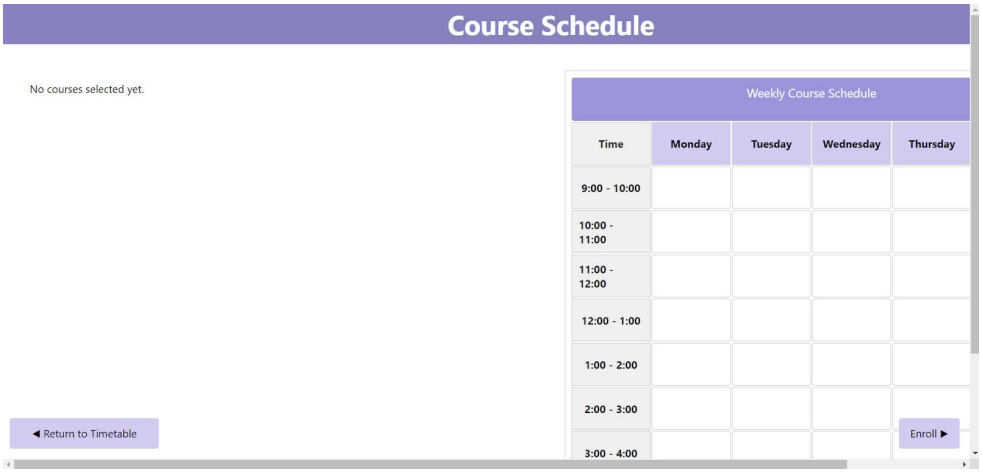

Figure 5: An example of this bug on the Schedule page: (Note: The height of this page is greater than the height of the window)

Another issue that we found is that accessing the Planner page resets all the data that should be saved. If you plan a course, you can search for it on the Home page and add it to the course shortlist. This course will show up in the Schedule page and on the weekly schedule. However, this course will disappear from the planner and course shortlist if the user accesses the Planner a second time.

A minor bug that we've found is that the filters cannot be applied to the search query in the middle of typing. For example, if a user selects the "All" filter and types 2m into the search box then switches to the "Required" filter, the following shows up:

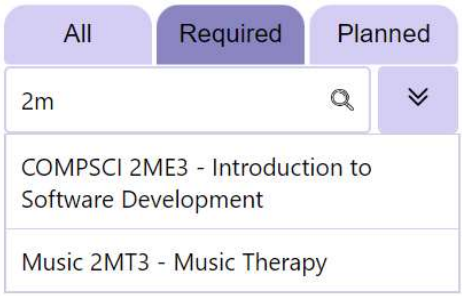

Figure 6: COMPSCI 2ME3 and MUSIC 2MT3 are shown as the results, even though only the first one is a required course

However, when the user continues to type–say, adding an "e"–the filter is applied.

Figure 7: The filter is properly applied (Note that MUSIC 2MT3 disappears but COMPSCI 2ME3 stays)