React 2 α

Due Oct 25 by 11:59pm **Points** 6 **Submitting** a text entry box
Available Oct 12 at 4pm - Oct 27 at 11:59pm 15 days

This assignment was locked Oct 27 at 11:59pm.

Homework: React 2 α (6 Points Total)

PLEASE READ THIS FIRST: If you have not submitted your React 1 α yet, DO NOT accept the GitHub Classroom invitation for React 2 α. Your React 1 α submission will be rejected if it is turned in after you accept the invitation below. Please submit your React 1 α before accepting the GitHub Classroom invitation.

This assignment is meant to introduce you to more features of React. This is the second α, feature based, assignment in which you will extend the features of a course guide application. This application uses a limited quantity of modified data from the UW-Madison course information database. With this assignment, you will extend the features of React 1 α to create a course recommendation system.

You may either use your code from React 1 α/β, or you may use the starter provided in the GitHub Classroom repository. To start from your React 1 submission, remove all non-hidden files from your **React 2 α** repository except for README.md, then copy all the non-hidden files from your React 1 repository into the top level directory of the React 2 α repository. Finally, add and commit your changes.

When your assignment is ready for grading, please submit to Canvas your repository name and latest commit hash from GitHub Classroom, e.g. `react2-alpha-osori, c13de63`.

Starter Code: React 2 α on GitHub Classroom (https://classroom.github.com/a/qzCS2wwX) (Do NOT open unless you submitted your React 1 α.)

Course data

The course data is being fetched from `http://cs571.cs.wisc.edu:53706/api/react/classes` and is formatted as follows:

```
[
    {
        "credits": <number of credits for the course>,
        "description": <course description>,
        "keywords": <1D list of string keywords>,
        "name": <course name>,
        "number": <unique course number>,
        "requisites": <2D list of course requisites>,
        "sections": [
            {
                "instructor": <instructor name>,
                "location": <section location>,
                "subsections": [
                    {
                        "location": <subsection location>,
                        "time": {
                            <weekday>: <time range>, ...
                        },
                                        "number": <subsection number>
                    }
                ],
                "time": {
                <weekday>: <time range>, ...
                },
                                "number": <section number>
            }, ...
        ],
        "subject": <course subject>
    }, ...
]
```

- The list of course requisites consists of 1D lists with AND operations between them. Each 1D list has OR operations between elements. For example: `[[A, B], [C, D, E], [F]]` means that the requisites are `(A OR B) AND (C OR D OR E) AND (F)`. The requisites will be represented as the course's alpha-numeric key used in the outermost object.
- Sections and subsections can have any number of times. Each time's key is a weekday in all lowercase ("monday", "tuesday", "wednesday", ...). Each time's value is a string with the following format: `"<12 hour time><am or pm> - <12 hour time><am or pm>"`. An example of this would be `"11:45am - 12:35pm"`.
- Each course has exactly one subject
- Your project must be able to accept any data with the same format as above and the data located at `http://cs571.cs.wisc.edu:53706/api/react/classes`

Completed Course Data

The course data is being fetched from `http://cs571.cs.wisc.edu:53706/api/react/students/5022025924/classes/completed` and is formatted as follows:

```
[
    {
        "data": <1D list of course numbers>
    }
]
```

- The completed course `data` returns only the course numbers. (e.g. ["COMP SCI 200", "CHEM 104"])
- If a user has not completed any course, `data` will have an empty 1D list as its value.

Recommender

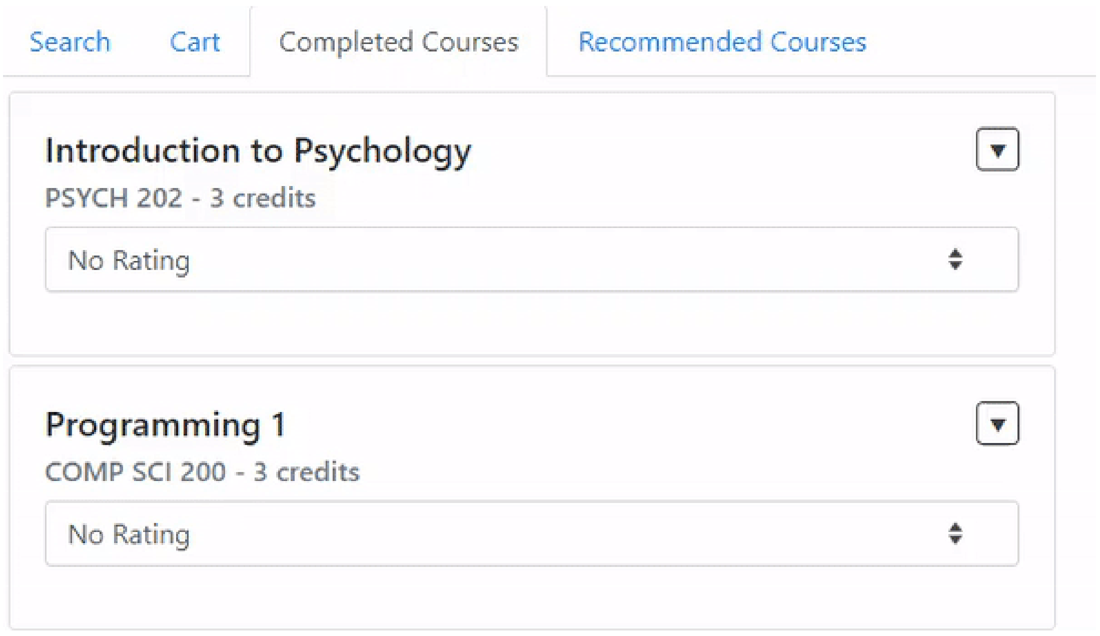
Problem 1 (1 point)

- Fetch data from server `http://cs571.cs.wisc.edu:53706/api/react/students/5022025924/classes/completed`. This data details which courses have already been completed.

- Create a new component to display a previously taken course. This component might look somewhat like the Course component, but it will be simpler and won't have options to add the course to the cart. In addition, it should not display information regarding sections/subsections since they vary between semesters. (You may also reuse the Course component if you can satisfy the aforementioned requirements with conditional rendering.)
- Create a new component to hold the previously taken course components. Make this component accessible as a new tab in the app. (Refer to Problem 2 and Problem 4 for example visualizations)

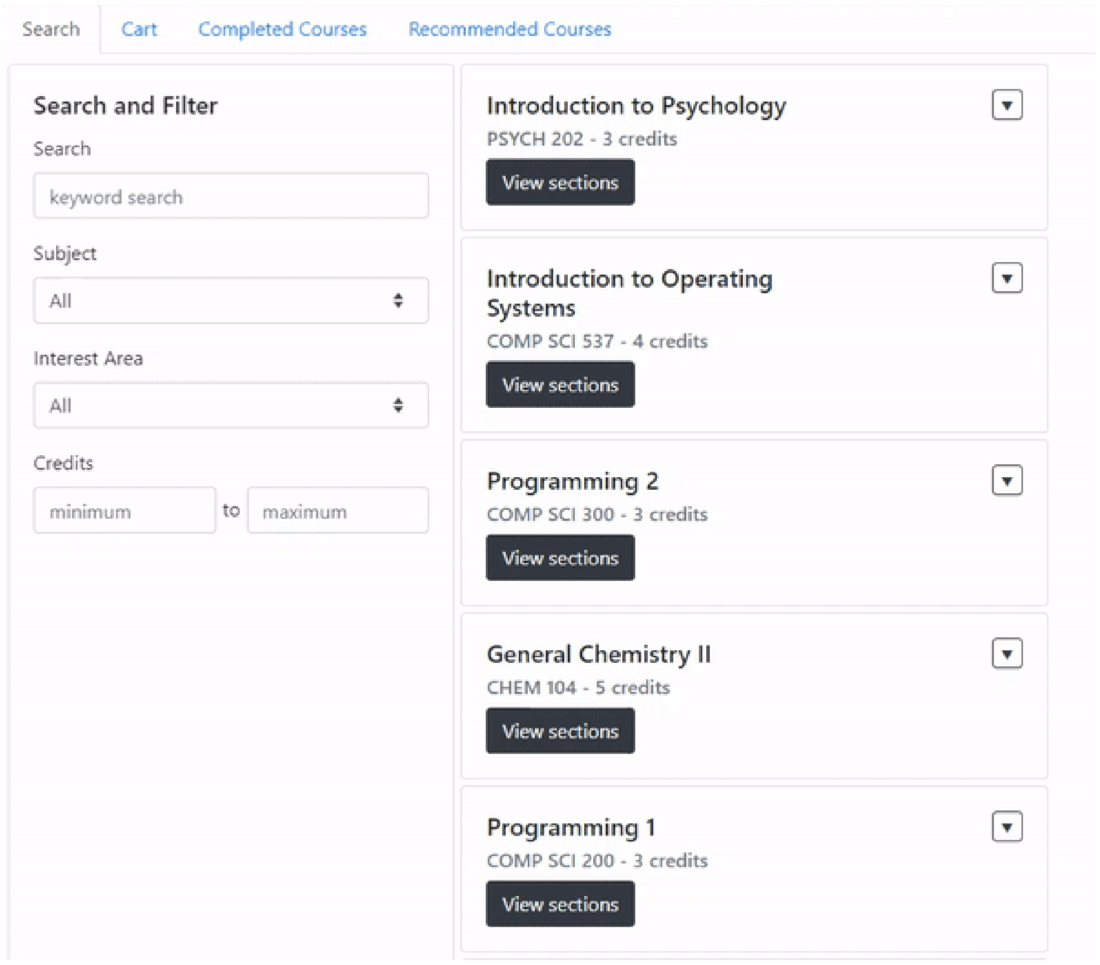
Problem 2 (1 point)

- Create a component for rating a specific course. For example, the sample implementation below created the rating component as a child of the completed course component.
- Allow the user to rate courses they have already completed.



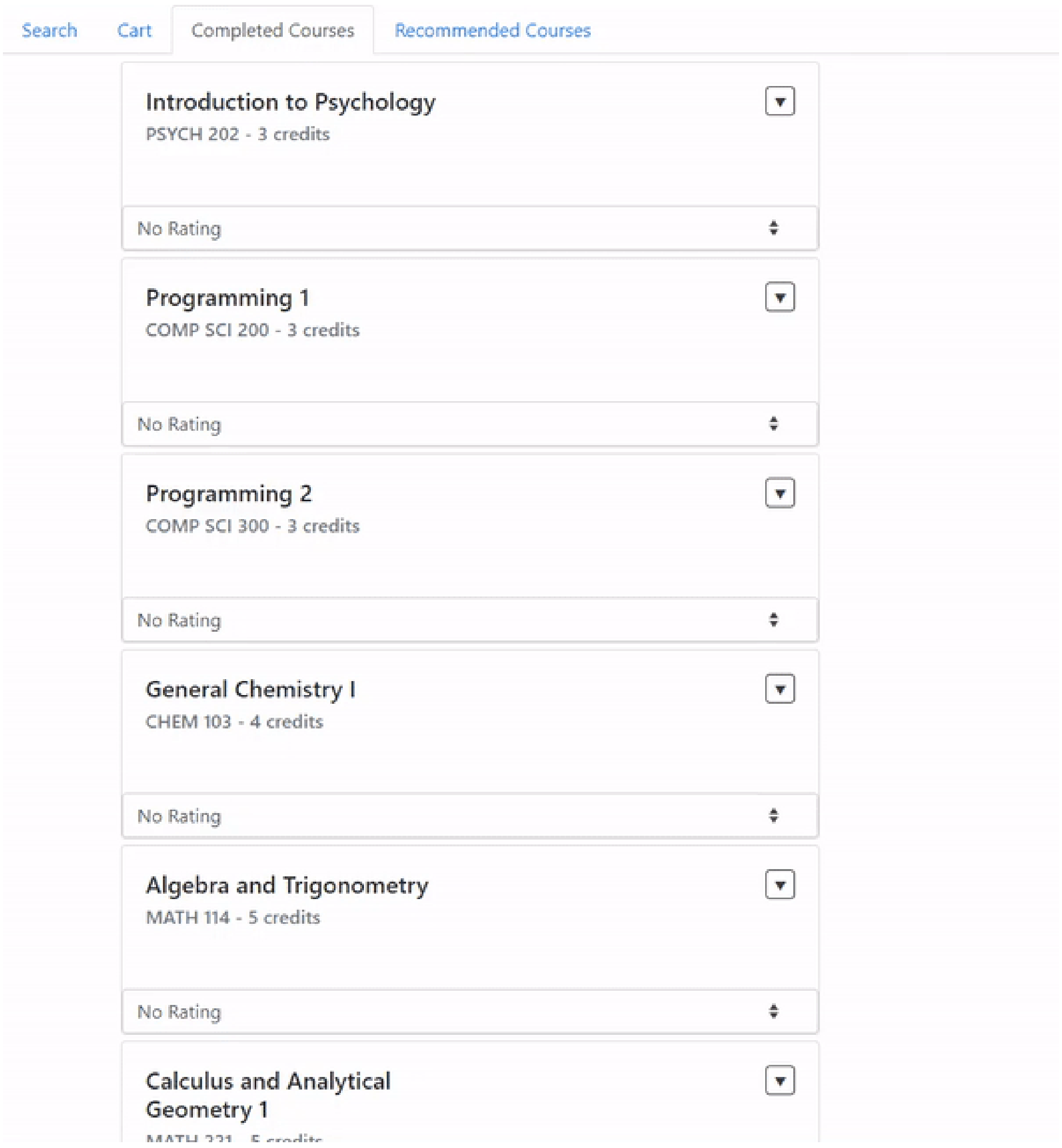
Problem 3 (1 point)

- Generate a list of interest areas based on the course data. Note that the interest areas should not be hard-coded but should be based on some values of the course data. For example, the sample implementation below dynamically generates interest areas using *keywords* from the courses.
- Create a component for the user to filter course results by interest area, using your list of interest areas. Make this component available to the user. The sample implementation put this component in the Sidebar Component.



Problem 4 (2 points)

- Create your own recommender algorithm that takes in **the rated courses and interest areas**. Use the interest areas of highly rated courses to recommend courses **which have not yet been taken**.
- Create a new tab which displays the recommended courses to the user. For instance, the sample implementation below shows several recommended courses in the interest areas of rated courses, sorted by the rating scores user gave for each interest area. You may also choose to show only several top recommendations instead of showing all possible recommendations.
- Note: As long as your algorithm considers user's recommendation score and interest areas you defined in Problem 3, and does not include previously taken courses, *the rest of the specifics is up to you.*



Introduction to Psychology
PSYCH 202 - 3 credits
No Rating

Programming 1
COMP SCI 200 - 3 credits
No Rating

Programming 2
COMP SCI 300 - 3 credits
No Rating

General Chemistry I
CHEM 103 - 4 credits
No Rating

Algebra and Trigonometry
MATH 114 - 5 credits
No Rating

Calculus and Analytical Geometry 1

Problem 5 (1 point)

- When adding a course to the cart, design a way to let the user know if they are not able to take the course based off of the requisites and the user's previously taken courses. Even if a student does not meet the requisites to enroll in a course, they should still be able to add it to the cart.
- Anytime a course, section, or subsection is added, the user should be notified in some way if they don't meet a requisite. As long as you adhere to this basic requirement, *the way to achieve this is completely up to you.* (You may use one of the components from Bootstrap, or even a simple JavaScript function.)

Problem 6 (0.5 points extra credit)

- If the user is not able to take a course in the cart because the user does not meet the requisites, design a way to show the user the possible course paths to take to be able to take the desired course.

Problem 7 (0.5 points extra credit)

- Create a way for user to select courses they would like to take in the future from the courses they are currently unable to take in the cart (because of requisites). Factor these courses into the recommendation algorithm, giving a larger bias to the courses needed for the selected interest courses.

Styling and npm packages

You are allowed and encouraged to use **react-bootstrap** (https://react-bootstrap.github.io/) for styling, and it is already installed in the React project for your use. You may alternatively use **Bootstrap** (https://getbootstrap.com/) or CSS for styling if desired.

If you would like to use additional npm packages, ask one of the TAs or Peer Mentors for permission.

You will be graded on the content you display and the style in which you display it, as well as your code quality.

Run `npm install` in the terminal after cloning to automatically install needed npm packages such as react-bootstrap

This project was bootstrapped with **Create React App** (https://github.com/facebook/create-react-app).

Available Scripts

In the project directory, you can run:

`npm start`

Runs the app in the development mode.
Open **http://localhost:3000** (http://localhost:3000/) to view it in the browser.

The page will reload if you make edits.
You will also see any lint errors in the console.

React 2 *α* Rubric

Criteria	Ratings			Pts
Completed Course Display Fetch and display of completed courses via components.	**1 to >0.5 pts** **Full Marks** The submission displays a component containing components displaying each of the courses a student has completed. The component containing all of the components corresponding to individual courses is available in a dedicated tab. The completed courses are correctly fetched from the API.	**0.5 to >0.0 pts** **Partial Marks** The Course display is partially correct, but either does not fetch completed courses from the API, is not in a new tab, or does not have the correct component structure.	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.	1 pts
Rating Completed Courses There is a way for users to rate completed courses.	**1 to >0.0 pts** **Full Marks** Users are able to leave a rating for completed courses. The rating should be numeric or categorical.	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.		1 pts

Criteria	Ratings			Pts
Filter by Interest Area A list of interest areas is generated from course data, and users are able to filter their search to match by interest areas from this list. Note that this means associating each course with a subset of the total interest areas.	**1 to >0.5 pts** **Full Marks** A list of interest areas is generated, and users are able to filter their course search results to only contain courses within a particular interest area.	**0.5 to >0.0 pts** **Partial Marks** A list of interest areas is generated, but filtering by interest area is incompletely implemented or missing.	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.	1 pts
Recommender Algorithm and Recommendation Display Course recommendations for not-yet-taken courses sharing an interest area with highly-rated completed courses should be displayed in a new tab.	**2 to >1.0 pts** **Full Marks** A list of course recommendations is generated by selecting not-yet-taken courses sharing one or more interest areas with highly rated completed courses, and these recommendations are displayed to the user in a dedicated tab.	**1 to >0.0 pts** **Partial Marks** Some issues with recommendation display or generation	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.	2 pts

Criteria	Ratings		Pts
Requisite Status Alert When adding a course to the cart, if there are requisites for that course which are not satisfied by the completed courses, the user should be alerted that they do not meet the requisites. The course should still be added to the cart.	**1 to >0.0 pts** **Full Marks** User is correctly notified when adding a course to the cart when requisites are not satisfied by completed courses.	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.	1 pts
Possible Requisite Paths Display (0.5 points Extra Credit) If a user is not able to take a course, a possible path through the requisites is displayed. Note that for this path to be correct, a requisite course can only be added to a path if its own requisites are satisfied by completed courses or previous requisites in the path.	**0 pts** **Full Marks** A correct requisite path is displayed for not-yet-taken courses which users currently lack the completed requisites.	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.	0 pts

Criteria	Ratings		Pts
Recommender Algorithm Biased in Favor of "Bookmarked" Courses (0.5 points Extra Credit) Users should be able to "bookmark" , "star," or otherwise notate not-yet-taken courses. The recommender algorithm should be biased toward recommending courses notated this way.	**0 pts** **Full Marks** Users are able to "bookmark" or otherwise notate not-yet-taken courses, and the recommender algorithm is biased towards recommending these courses.	**0 pts** **Incorrect** Not implemented, or minimal attempt at implementation.	0 pts
			Total Points: 6