CSCI 1300 - Development

Release Date: November 20, 2018
Due Date: December 6, 2018 at 12:00PM (noon)

Overview

Have you ever been annoyed trying to filter products in old ecommerce websites? Ever appreciated how easy it is to browse movies on Netflix? With so much content and information these days, live filtering is a very crucial feature on websites and a good filtering system can considerably improve user experience. Many of these websites' filters are made using React, a modern JavaScript library for building scalable, component-based user interfaces.

With this in mind, **you will be using React.js for this assignment.** We highly recommend starting with the React lab, as completing the tasks outlined there will be a huge help in implementing the tasks below!

Task

Your task is to **create your own live filtering page for a content type of your choice**! We encourage you to be creative and choose something that interests you. Your live filtering page can be a list of movies that can be filtered by ratings, a list of books that can be filtered by page numbers, or even list of Pokémon filtered by their strengths!

Here are some possibly inspirational example pages:

Hack@Brown Mentorship Page: https://yarai.github.io/mentorship-page/
Congress Search Tool: https://yarai.github.io/congress-example/
Airbnb: https://www.airbnb.com/s/Paris--France
StockPhoto: http://www.istockphoto.com/photos/jeff?phrase=Jeff&sort=best
Yelp: https://www.yelp.com/search?find_desc=&find_loc=Providence%2C+RI&ns=1

Requirements

Your live filtering page must have **at least 2 categories for filtering the content** (e.g. the Hack@Brown mentorship page can be filtered by company, tags, and skills) and **at least 1 method of sorting the content** (e.g. Amazon's search results can be sorted by price). Your page should also have **at least 12 items,** each of which should display: **an image of your item, the 2 categories that your filters use, and the 1 field that your sorting uses**. Your page should be able to **handle any combination of filters and all filters should work in tandem with your sorting method**.

Additionally, **there should be some way to revert back to the original state of the list without refreshing the page** (e.g.through use of a button or through use of your filters).

We require you to create **three different React components** to be used in your filtering program. React components function very similarly to how classes work in Javascript - if you'd like more information on what React components are and how they work, we encourage you to come to the Development gear-up (date and time below) and to run through the React lab!

Keep in mind that if you've already completed the React lab, you've also already created quite a few different components! A couple of these components are very applicable to this assignment (and would count towards this three component requirement).

Finally, please design your page to be **intuitive and easy to use** (using design principles learned in class). A user should be able to find the filtering buttons and sorting fields and easily understand their purpose. Additionally, a user should also be able to easily see how the displayed items change depending on the buttons/filters selected.

TAs Alainey, Kat, and Hannah will be holding a gear-up session on Tuesday, 11/27, from 8pm in CIT 269. Also, feel free to ask questions at TA hours and on Piazza!

Additional Resources

- Facebook's official React documentation
- React video tutorial
- Facebook talk explaining the rationale behind using React
- React Developer Tools (Chrome, Firefox)

Handing in your Assignment

In a README, include a **link to your page or instructions on how to run your code**.
Instructions for how to deploy your project can be found in the React Lab handout.

Additionally, **include a paragraph** about how your interface relates to the user interface design principles we have learned in class, how data is passed down through your components and how user interactions can trigger changes in the state of components. Include a sentence or two on the high-level goal of your application and why it might be valuable to a user. Create a **[LastName-FirstName].zip** that includes both **your code and the README** and submit it on Gradescope.

Grading Rubric (22 points)

Functionality (14 points)

- 3 points — There are at least 2 categories to filter content, and both of these filters work.
- 3 points — There is at least 1 method of sorting items, and this method of sorting works.
- 3 points — The program can correctly handle multiple filters and sorting working together.
- 1 point — There are at least 12 items on the page.
- 1 points — Each item has a picture, at least 2 text fields, and at least 1 sortable field.
- 2 points — Can revert to original state.
- 1 point — The page does not crash (e.g. the page does not freeze, crash the browser, terminate unexpectedly, etc.).

React Components (2 points)

- 2 points — Implements at least 3 different React components.

Usability (2 points)

- 2 points — Interface is intuitive and user-friendly, demonstrating interaction and navigation principles learned in class; a user is able to easily understand and interact with your webpage, and feedback is easily understandable based on the actions the user takes.

README (2 points)

- 1 point — Includes link to page and/or instructions to run code.
- 1 point — Explains how interface relates to the design principles learned in class, how data is passed down through your components, how user interactions can trigger changes in the state of components, and the overall goal and value of the application to a user.

Style (2 points)

- 2 points — Project is explained well and compelling to an outside audience. Would someone unfamiliar with the assignment understand it and know what its purpose is? Review the style guide for more details.