CS1300 - Development (v1.1)

Release date: Thursday, November 14, 2019
Gearup: Monday, November 18th from 7:00-8:00pm in CIT 269
(Optional) Due date for Intermediate Step: Tuesday, November 26, 2019 at 6:00pm
Due date for Final Submission: Thursday, December 5, 2019 at 6:00pm
Late due date: Sunday, December 8, 2019 at 6:00pm

You may work alone or with 1 other person.

Overview

Have you ever been annoyed trying to filter through 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!

The TAs will also host a gear-up session for this project. We strongly encourage that you attend this session from 7:00-8:00pm on November 18th (Mon) at CIT 269, especially if you are new to React!

Suggested Timeline

<table>
<thead>
<tr>
<th>Task</th>
<th>Finished by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decide on a list of 12 items filter categories, and sorting fields.</td>
<td>Saturday, 11/16</td>
</tr>
<tr>
<td>Have React file structure planned out (component, props, states you want to store)</td>
<td>Tuesday, 11/19</td>
</tr>
<tr>
<td>Have basic (unstyled) architecture of page laid out</td>
<td>Friday, 11/22</td>
</tr>
<tr>
<td>Complete intermediate step - have sort and/or filter finished; if pair programming, get started on favoriting feature!</td>
<td>Tuesday, 11/26</td>
</tr>
<tr>
<td>HAPPY THANKSGIVING</td>
<td>Sunday, 12/1</td>
</tr>
</tbody>
</table>
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/](https://yarai.github.io/mentorship-page/)
Warby Parker: [https://www.warbyparker.com/eyeglasses/women](https://www.warbyparker.com/eyeglasses/women)
Airbnb: [https://www.airbnb.com/s/Paris--France](https://www.airbnb.com/s/Paris--France)
Yelp: [https://www.yelp.com/search?find_desc=&find_loc=Providence%2C+RI&ns=1](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 and tags)
- **At least 1 method of sorting the content** (e.g. Amazon’s search results can be sorted by price)
- **At least 12 list 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
- **Ability to handle any combination of filters and all filters should work in tandem with your sorting method**
- **Some way to revert back to the original state of the list** without refreshing the page (e.g. through the use of a button or through the use of your filters)
- **At least two different React components in addition to your filtering list**
  - 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!](https://www.yelp.com/search?find_desc=&find_loc=Providence%2C+RI&ns=1)
  - Components should be useful additions to your code (that is, you should not be making a component with a single line of text, as that would be more work than just adding the text). Each of your two required components must have **state** and **prop**!
Note: **App does not count towards your two additional required components**

- **An intuitive and easy to use** design (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.

- **If you’re working with another person:** In addition to the two React components, sorting, and filtering functionality, you must include the ability to “Favorite” items in your list. When adding this functionality to your project, take inspiration from Instagram’s “Saved” feature or Urban Outfitters’ “Wish List” feature. Your Favoriting feature should:
  - Add an icon to each list item that visibly changes when “favorited”
  - Add a Favorites toggle that displays only your favorites. You should still be able to sort/filter on this list!

Note: Keep in mind that if you’ve already completed the React lab, you’ve already created quite a few different components! A couple of these components are very applicable to this assignment (and would count towards this two component requirement). You are more than welcome to use code directly from your React lab in this project.

**Intermediate Step**

**If you are working alone:**
In order to receive a late pass, you must have **either filter or sort functionality working on your page**. Submit your React app directory (the one that contains the ‘public’ directory, ‘src’ directory, and other files) as zipped file in Gradescope by November 26, 2019 at 6:00pm. Remember that this step is optional and you will not be penalized for not completing it.

**If you are working as a pair:**
In order to receive a late pass, you must have **filter and sort functionality working on your page**. Submit your react app directory (the one that contains the ‘public’ directory, ‘src’ directory, and other files) as zipped file in Gradescope by November 26, 2019 at 6:00pm. Remember this step is optional and you will not be penalized for not completing it.

**README & Handing In**

This project doesn’t require a separate write-up. Instead, you will write a brief README file that you will turn in along with your code. In your README, please include:

- **A link to your deployed app** or **instructions on how to run your code** if you submit your local files. (Instructions for how to deploy your project using Heroku can be found [here](#)).
- **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.
• A sentence or two on the high-level goal of your application and why it might be valuable to a user.
• A clear explanation of the purpose of your webpage to someone who is not familiar with the project, just as it states in the style guide.

When you’re ready to hand in, submit a zipped file including both your README and the following files / directories from your code:

• src folder
• public folder
• package.json

Note: Do not submit the node_modules folder -- this will cause problems in Gradescope!

Grading and Requirements (22 points)

14 Points - Functionality
• 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.).

2 Points - React Components
• 2 points — Implements at least 2 different React components using state and props in addition to your filtering list.

2 Points - Usability
• 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.

2 Points - README
• 1 point — Includes link to page (if deployed) or instructions to run code locally.
• 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.

2 Points - Style
• 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.