React-Redux Calculator Assignment With React Router

Deadline: Friday, JUL 31 2020, 09:00 AM

GitHub Classroom Link:

Introduction

This assignment is meant to challenge your knowledge of state, prop and element rendering in React while also using a store in redux to store information. React is currently one of the most popular and sought after front end technology at the moment, and having skills with it will greatly increase your employability. Redux is often used as a companion to React, and is favored for storing information over using state. React Router is used to create multiple pages in React, and is often used to keep applications and web pages organized.

Requirements

- ❑ Create two input fields that can take in numeric values
- ❑ Create a select (drop down box) which allows the user to choose what operation to apply to the two numeric fields - your operations must include addition, subtraction, multiplication, and division
- ❑ Output to the user the result of the calculation
- ❑ Store and output a list of prior calculations the user has done in a Redux store
- ❑ Use React Router to have a page for the calculator and a page for prior calculations

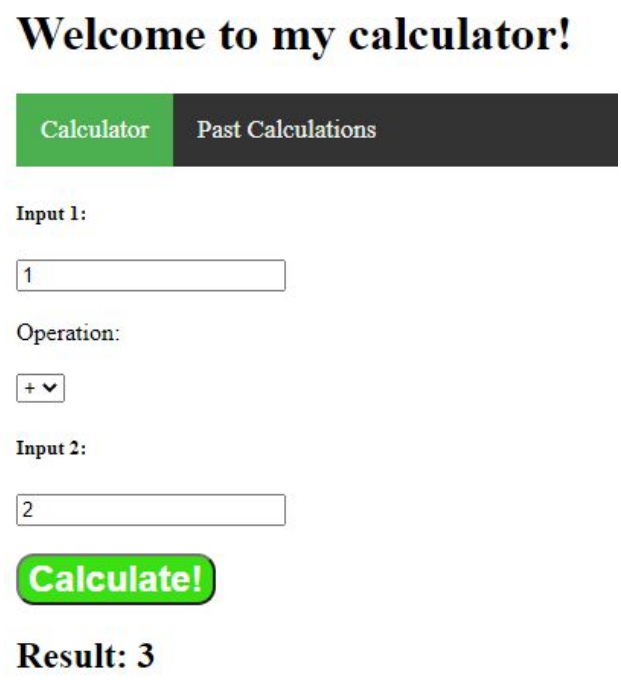
Challenges

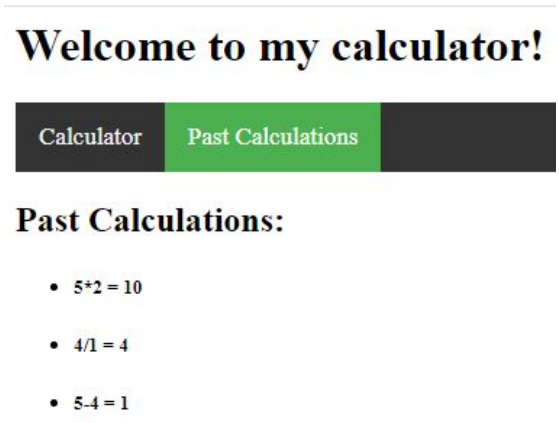
- ❑ Make it look nice with some CSS!
- ❑ Protect data input - return a message to the user if the user inputs a non-numeric character
- ❑ Have a single input field which can read the operations

Hints

- **You are welcome to build upon your React-Redux calculator project to fulfill the first three requirements**
- Look over what was done in class! It should contain all the tools necessary to get started on this assignment
- Take advantage of online resources! Most of the tasks we assign are very standard examples
- Focus on the requirements before moving on to the challenges! Challenges are extra
- Comment your code and commit/push your work regularly

Sample Screenshot





Rubric

You will be evaluated on the following points. You must get all 5 of the Mandatory points to pass:

Requirement	**Points**
MANDATORY: ● Create two input fields that can take in numeric values ● Create a select (drop down box) which allows the user to choose what operation to apply to the two numeric fields - your operations must include addition, subtraction, multiplication, and division ● Output to the user the result of the calculation ● Store and output a list of prior calculations the user has done in a Redux store ● Use React Router to have a page for the calculator and a page for prior calculations	5
CHALLENGE: ● Make it look nice with some CSS! ● Protect data input - return a message to the user if the user inputs a non-numeric character ● Have a single input field which can read the operations	3
Total:	5

Citation Guide

Whenever you borrow code, the following information must be included:

- ❏ Comments to indicate both where the borrowed code begins and ends.
- ❏ A source linking to where you found the code.
- ❏ Your reason for adding the code to your assignment/project instead of writing it out yourself
- ❏ How it works. Explain to us how the code is supposed to work, include links to documentation/articles you read to help you understand.
- ❏ A small demonstration to prove you understand how the code works.

```
const inputArr = [5,1,3,4,2];

/*Borrowed code for bubbleSort starts*/
let bubbleSort = (inputArr) => {
    let len = inputArr.length;
    for (let i = 0; i < len; i++) {
        for (let j = 0; j < len; j++) {
            if (inputArr[j] > inputArr[j + 1]) {
                let tmp = inputArr[j];
                inputArr[j] = inputArr[j + 1];
                inputArr[j + 1] = tmp;
            }
        }
    }
    return inputArr;
};

/*Borrowed code from bubbleSort ends*/

//Source: bubbleSort function obtained from https://medium.com/javascript-algorithms/javascript-algorithms-bubble-sort-3d27f285c3b2
//Reason to add: implementing bubble sort can be tedious and bug prone, it would be better to use a proven version than to write my ow
//How it works: I read the following article to understand how bubble sorts work (http://www.pkirs.utep.edu/CIS3355/Tutorials/chapter9
//Demonstration of understanding:
//Example array: [3,1,2]
//Step 1: Compare 3 and 1. Since 1 is smaller, swap places.
//Array: [1,3,2]
//Step 2: Compare 3 and 2. Since 2 is smaller, swap places.
//Array: [1,2,3]
//Step 3: Compare 1 and 2. No need to swap.
//Array: [1,2,3]
//Step 4: Compare 2 and 3. No need to swap.
//Array: [1,2,3]
//Function complete.
console.log(bubbleSort(inputArr));
```