React State Lab

New Attempt

Due No Due Date **Points** 1 **Submitting** a website url

FORK (https://github.com/learn-co-curriculum/react-hooks-state-and-events-lab/fork) (https://github.com/learn-co-curriculum/react-hooks-state-and-events-lab) (https://github.com/learn-co-curriculum/react-hooks-state-and-events-lab/issues/new)

Learning Goals

- Update state based on events
- Work with multiple state variables together

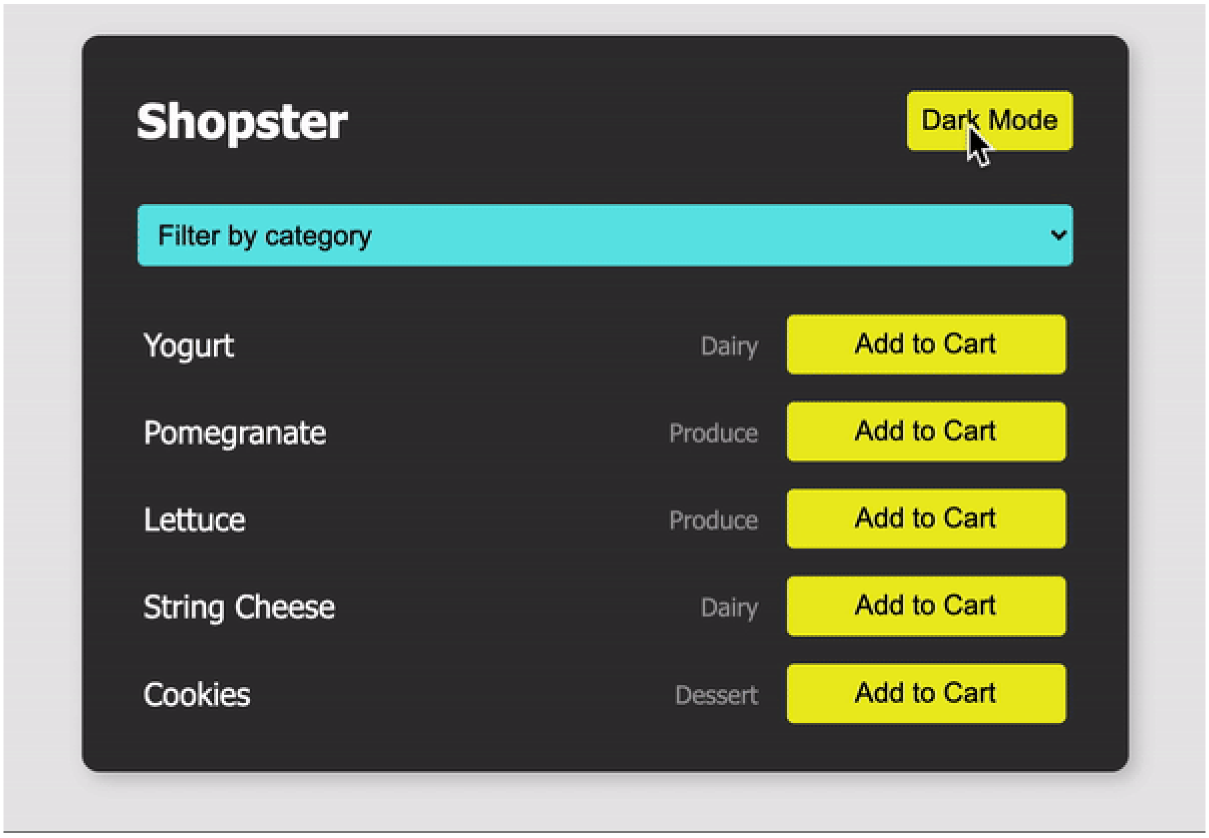
Introduction

In the labs for this section, we'll be working on a grocery list app. Some of this app is already built out using static versions of the components. We'll be using **state** and **events** to make our app dynamic.

The components you have to work with are in the `src/components` folder. Start by examining these components and draw out your component hierarchy to see how data can be passed from one component to another.

In this lab, you'll update state and get more practice with the `useState` hook.

It's recommended that you run `npm start` and work on this in the browser **before** running tests. Try to get your app to match the demo! Once you've built out the components below, run `npm test` or `learn test` to see if your code passes the tests.



Deliverables

Dark Mode Toggle

In the `App` component, there is a button for toggling between dark mode and light mode. Using the `useState` hook, create a state variable in the `App` component. Then, use that variable to determine if our app should be in dark mode or light mode.

You will also need to add an event handler to the dark mode button, and update state when the button is clicked.

The actual functionality of changing our app's theme is handled in CSS, so all you have to do is set up the code to update the `className` of the div based on your state variable:

```
<div className="App dark">
{/* for dark mode */}
</div>

<div className="App light">
{/* for light mode */}
</div>
```

Add to Cart

In the `Item` component, when the user clicks the `<button>` element, the item should be added to their virtual cart. The way we'll show the user that the item is in the cart is by changing the className on the `<li>` element:

```
<li className="in-cart">
{/* the item is in the cart */}
</li>

<li className="">
{/* the item is NOT in the cart */}
</li>
```

If the item *is not* in the cart, the `<button>` element's text should read "Add to Cart", and if the item *is* in the cart, the `<button>` element's text should read "Remove From Cart". Naturally, you'll also need to add state to the `Item` component to solve this deliverable!

Filter

In the `ShoppingList` component, there is a `<select>` element that will allow us to **filter** the list of items by category.

Use the `useState` hook to create a state variable called `selectedCategory` for keeping track of the selected value from this `<select>` element. When the value of the `<select>` element is **changed**, update state.

You should also use the `selectedCategory` variable to determine which items to display in the shopping list. You'll need some way of **filter**ing the array of items based on the `selectedCategory`.

Resources

- React Docs on `useState` (https://reactjs.org/docs/hooks-state.html)