Sleeping Duck tech test

This test is designed to test your fundamental React and CSS ability. There is no time limit. If you have **any** questions, please email me (Matt) at matt@sleepingduck.com

Getting started

Download this self contained HTML file. The HTML file contains all the code to run the test and should open in any modern browser.

Your response should just be an updated version of this file. No need to create multiple files.

Things to keep in mind:

- Make sure it stays responsive after your changes - whatever way you see fit
- You can use the latest JS/React/CSS, don't worry about the browser it's running in

Things to do:

- Move the buttons in the nav bar to always be on the right
- Add another button to the nav bar with the label 'user' - it doesn't have to do anything
- Turn the list of to-dos into a table like view with:
  - 1 column on screen sizes under 600px
  - 2 evenly spaced columns on screen sizes under 750px
  - 3 evenly spaced columns on all other screen sizes
- Create a *bulk edit* function
  - Add a checkbox to each to-do item
  - Once at least one to-do item is selected, a floating menu at the bottom of the screen should show up
  - The menu should give the option to bulk *mark as complete*, or *mark as incomplete*
  - Once one of the options are complete the appropriate action should be taken and all items should be unselected
  - **Bonus points:** Only show buttons that are relevant to the selected options. For example, if only completed items are selected, only show the *mark as incomplete* button. Or if the there is a mix of items, show both.

Questions

Answering these questions aren't required, but correct answers will help.

Part 1

When setting the state within the `toggleItem` function, I used this code to modify the array:

```
this.setState(({ items }) => ({
  items: items.map((item, i) => (index === i ? { ...item, complete: !item.complete } : item)),
}));
```

Questions

1. Why would I have passed a function to the `setState` call, rather then modifying `this.state`?
2. What could be the reason for me mapping over the entire array rather than modifying the array directly (IE. `items[index].complete = !item[index].complete`)? And is there any possible reasons for using the object spread syntax within the loop?

Part 2

Currently, we use the array index as the way to know what to-do item we are toggling, as well as using them for the `key` prop when rendering the list.

Questions

1. Generally, what drawbacks—if any—does this have and how would you solve them?