**- What were some of the challenges you encountered?**

The only challenges I encountered was adding additional calculator features of not letting the user input an operation without at least one operand already being entered and not allowing operations to be entered back to back. I ended up using a variable (boolean) with useState to determine if an operand has been entered already. However, at first I tried using the variable (firstEntry) without useState that was declared initially as false in the panel component and was changed to true when an operand was entered. This didn’t work as I was using useState for other variables in the same component which re-renders the component when those variables are changed. Thus reinitializing the firstEntry variable back to false even though an operand had already been entered. So, I used useState for firstEntry too, but I know it isn’t ideal as it’ll cause unnecessary renders of the component. I tried using useRef, but I wasn’t able to get it working. Other than that, I had no other problems implementing the calculator or the game since I have some experience with React Native, which is very similar to React.

**- Differentiate between Real DOM and Virtual DOM. List out a few differences.**

Differences:

* Virtual DOM is just a virtual representation, like a soft copy, of the Real DOM which is the real structure of the webpage
* Making changes to the Virtual DOM is easier and faster than making changes to the the Real DOM
* Changes to the Virtual DOM show faster than changes to the Real DOM
* Upon a change to an element in the Real DOM, a new DOM is made to accommodate for it
* Real DOM can update HTML directly, while Virtual DOM cannot
* Virtual DOM increases performance

**- What are some problems that React can solve for us?**

It helps with making changes to the DOM efficiently which increases the performance of the product. The use of components helps with making parts of the user interface reusable if needed. I also found that seeing the UI as a bunch of individual components helps with planning and breaking down what needs to be done. Components promote modularity as well and makes maintaining, testing, and changing parts of the UI much easier without having to worry about messing up the other parts. This also makes finding what part needs to be changed easier since it’ll be in its own file (decreases the amount of time finding the specific part that needs to be fixed). Overall, I think using React makes the development process and maintenance much easier. Additionally, due to how efficient it is at updating the UI, the user experience also improves.