React Buttons Counter Assignment – props and state

Objective:

Develop a React application with two functional components: **App** (parent) and **ButtonCounter** (child). The main goals of the assignment are:

1. Display 9 buttons, each with a unique label.
2. Track and display the number of times each button is clicked.
3. Calculate and show the sum of the numbers of all clicked buttons.

Requirements:

1. Use functional components and the **useState** hook for state management.
2. Integrate Bootstrap for styling. Ensure that three buttons are displayed in each row.

Detailed Instructions:

1. Setup:

- Create a new React project using the Create React App command-line tool:

```
npx create-react-app button-counter-app
```

- Install Bootstrap:

```
npm install bootstrap
```

2. ButtonCounter Component:

- Create **ButtonCounter.js** inside the **src** directory.
- Design the component to receive two props:
    1. **number** - The label for the button.
    2. **onButtonClick** - A callback function to notify the parent component when a button is clicked.
- Use the **useState** hook to maintain a local state (**hitCount**) representing the number of times the button has been clicked.
- Render a button that, when clicked, increases the **hitCount** and calls the **onButtonClick** callback with the button's number.

```
    // Sudo code for ButtonCounter.js
    import necessary libraries and hooks
function ButtonCounter(props) {
        initialize hitCount with useState

        function handleClick() {
          increase hitCount
          call onButtonClick with the button's number
        }

        return button with number and hitCount
}
export ButtonCounter
```

3. App Component:

- Modify **App.js** to import **ButtonCounter** and Bootstrap's CSS.
- Initialize a state (**totalSum**) to keep track of the cumulative sum of the clicked button numbers.
- Create a function (**handleButtonClick**) that will update the **totalSum** state.
- Render 9 **ButtonCounter** components. Pass down the unique number and the **handleButtonClick** function as props to each.
- Display the total sum below the grid of buttons.

```
// Sudo code for App.js
import necessary libraries, hooks, and ButtonCounter
function App() {
  initialize totalSum with useState

  function handleButtonClick(clickedNumber) {
    update totalSum by adding clickedNumber
  }
  return (
    container div {
      row div {
        loop from 1 to 9 {
          column div containing ButtonCounter component
        }
      }
      display totalSum
    }
  )
}
export App
```

4. Styling:

- Make use of Bootstrap classes to style your components. Each row should contain 3 buttons, so you'd utilize the Bootstrap grid system with **col-4** for each button.
- You can enhance the styling by adding custom CSS or using more Bootstrap utilities.

5. Testing:

- Start the React development server:

  npm start

- Ensure that each button's click count updates correctly.

- Verify that the total sum updates appropriately when any button is clicked.

EVALUATION CRITERIA:

- Correct implementation and functionality.
- Proper management of state (**props** and **state**).
- Effective integration of Bootstrap for responsive design.
- Code organization, readability, and consistency.

BONUS POINTS:

1. Extend the application to save the state in the browser's local storage.
2. Implement unit tests for your components using a library like Jest.
3. Enhance the UI/UX with smooth animations or transitions.

By the end of this assignment, you should have a deeper understanding of state management in React, component communication, and integration of external libraries like Bootstrap for styling. Happy coding!