React Assignments:-

Assig 1:-

State and Props:--

Task 1: Data Manipulation in ParentComponent

Extend the ParentComponent to include an array of objects, where each object represents a person with properties like id, name, and age. The ParentComponent should pass this array as a prop to ChildComponent.

Task 2: ChildComponent Data Display

Modify the ChildComponent to receive the array of persons as a prop. Display the names of all persons in an unordered list (<ul>). Also, provide a button in each list item (<li>) to toggle the display of the person's age. Initially, the age should be hidden.

Task 3: Conditional Rendering in ChildComponent

Implement conditional rendering in the ChildComponent to show additional information about a person when their age is toggled to be displayed. For example, display a message like "Underage" or "Adult" based on a predefined age threshold (e.g., 18 years).

Task 4: Bonus Challenge

Implement a feature in the ParentComponent that allows adding new persons to the array. This should trigger a re-render in the ChildComponent to reflect the updated list.

Assig 2:-

Assignment: Product Functional Component

1. Create a new React functional component called Product.
2. The Product component should receive the following props from its parent component:

productName (string)

price (number)

brandName (string)

quantity (number)

1. Inside the Product component, use the useState hook to initialize a state variable called quantityInCart with the initial value set to the quantity prop received from the parent.
2. Display the product information including productName, price, brandName, and the current quantityInCart in the component.
3. Add two buttons:

* One button to simulate the purchase of the product. When clicked, it should decrease the quantityInCart by 1.
* One button to undo the purchase. When clicked, it should reset quantityInCart back to its original value.

Assig 3:

Assignment: Employee Details with State Management

1. Create a new React functional component called EmployeeDetails.
2. Inside the EmployeeDetails component, use the useState hook to initialize a state object called employee with the following properties:

id (number)

name (string)

position (string)

salary (number)

1. Display the current details of the employee in the component.
2. Add buttons to update the employee details:

* One button to change the employee's position.
* Another button to increase the employee's salary by 1000.
* Use the prevState parameter in the setState function to ensure that state updates are based on the previous state.

Assig 4: (useState and useEffect Hook)

Fetching Data from an API

1. Create a React functional component called UserList.
2. Inside the UserList component, use the useState hook to create a state variable called users with an initial value of an empty array.
3. Use the useEffect hook to fetch a list of users from a mock API (you can use a JSONPlaceholder or any other mock API). Update the users state with the fetched data.
4. Display the list of users in the component.

Assig 5:

Real-time Clock

1. Create a React functional component called Clock.
2. Inside the Clock component, use the useState hook to create a state variable called currentTime with an initial value of the current time.
3. Use the useEffect hook to update the currentTime state every second, creating a real-time clock.
4. Display the current time in the component.

Auto-Saving Form

1. Create a React functional component called RegisterComponent.
2. Inside the RegisterComponent component, use the useState hook to create state variables for form data (e.g., formData—name/email/address).
3. Use the useEffect hook to simulate auto-saving whenever the form data changes. Log a message to the console indicating that the form data has been auto-saved.
4. Create a form with input fields that allow users to modify the form data.