Experiment 3

Create a Counter Application using React that demonstrates state management with the useState hook. Display the current value of the counter prominently on the screen. Add buttons to increase and decrease the counter value. Ensure the counter updates dynamically when the buttons are clicked. Use the useState hook to manage the counter's state within the component. Prevent the counter from going below a specified minimum value (e.g., 0). Add a "Reset" button to set the counter back to its initial value. Include functionality to specify a custom increment or decrement step value.

App.js

```
import React, { useState } from 'react';

 import './App.css';

 function App() {

 // Initialize the state for counter, step, and minimum value

 const [counter, setCounter] = useState(0);

 const [step, setStep] = useState(1);

 const minValue = 0;

 // Handle increment

 const increment = () => {

       setCounter(counter + step);

   };

 // Handle decrement

 const decrement = () => {

if (counter - step >= minValue) {

          setCounter(counter - step);

       }

   };

 // Reset counter to initial value

 const resetCounter = () => {

       setCounter(0);

   };

 // Handle step change
```

```
const handleStepChange = (event) => {
const value = Number(event.target.value);
if (value > 0) {
        setStep(value);
     }
  };
return (
<div className="App">
<h1>Counter Application</h1>
<div className="counter-container">
<h2>{counter}</h2> {/* Display current counter value */}
<div className="controls">
{/* Buttons for Increment, Decrement, and Reset */}
<button onClick={increment}>Increase</button>
<button onClick={decrement}>Decrease</button>
<button onClick={resetCounter}>Reset</button>
</div>
<div className="step-container">
<label>
                 Step:
<input
type="number"
value={step}
onChange={handleStepChange}  // Set custom increment/decrement step value
min="1"
/>
</label>
</div>
</div>
```

```
 </div>
   );
 }
 export default App;
```

App.CSS

```
.App{
ext-align: center;
 font-family: Arial, sans-serif;
 }
 .counter-container {
 margin-top: 50px;
 }
 h2 {
 font-size: 3em;
 margin: 20px 0;
 }
 button {
 padding: 10px 20px;
 margin: 10px;
 font-size: 1.2em;
 cursor: pointer;
 }
 button:hover {
 background-color: #ddd;
   }
 .step-container {
 margin-top: 20px;
 }
 input {
```

```
font-size: 1.2em;
padding: 5px;
width: 60px;
margin-left: 10px;
}
```