# Practice React's useState hook with 8 Interactive Exercises

## 1. Counter:

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

function Counter() {
  const [count,setCount] = useState(0);

  return (
    <div>
      <button onClick = {() => setCount(count+1)}>Increment</button>
      <p>Count: {count}</p>
    </div>
  );
}

export default Counter;

////////////////////////////////////////////////////////////////////////////

// using handleIncrement function

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

function Counter() {
  const [count,setCount] = useState(0);
  function handleIncrement(){
    setCount(count +1);
  }
  return (
    <div>
      <button onClick = {handleIncrement}>Increment</button>
      <p>Count: {count}</p>
    </div>
  );
}

export default Counter;
```

## 2. Controlled Input Field:

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

function InputField() {
  const [text, setText] = useState("");

  function handleInputChange(event) {
    setText(event.target.value);
  }

  return (
    <div>
      <input type="text" value={text} onChange={handleInputChange} />
      <p>Input text: {text}</p>
    </div>
  );
}

export default InputField;
```

## 3. Toggle Visibility

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

function ToggleVisibility() {
  const [isVisible, setISVisible] = useState(false);

  function handleToggleVisibility() {
    setISVisible(!isVisible);
  }

  return (
    <div>
      <button onClick={handleToggleVisibility}>Show/Hide Text</button>
      {isVisible && <p>Toggle me!</p>}
    </div>
  );
}

export default ToggleVisibility;
```

### 4. Character Counter

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

function CharacterCounter() {
  const [text ,  setText] = useState("");

  function handleTextareaChange(event) {
    setText(event.target.value);
  }

  return (
    <div>
      <textarea value={text} onChange={handleTextareaChange} />
      <p>Character count: {text.length}</p>
    </div>
  );
}

export default CharacterCounter;
```

### 5. Todo List

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

function TodoList() {
  const [todos, setTodos] = useState([]);
  const [inputValue, setInputValue] = useState('');

  function handleInputChange(event) {
    setInputValue(event.target.value);
  }

  function handleSubmit() {
    if (inputValue.trim()) {
      setTodos([...todos, inputValue.trim()]);
      setInputValue('');
    }
  }

  function handleDelete(index) {
    setTodos(todos.filter((_, i) => i !== index));
  }
```

```
  return (
    <div>
      <input type="text" value={inputValue} onChange={handleInputChange} />
      <button onClick={handleSubmit}>Add Todo</button>
      <ul>
        {todos.map((todo, index) => (
          <li key={index}>
            {todo}
            <button onClick={() => handleDelete(index)}>Delete</button>
          </li>
        ))}
      </ul>
    </div>
  );
}
export default TodoList;
```

### 6. Color Switcher

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

function ColorSwitcher() {
  const [bgColor, setBgColor] = useState('');

  function handleColorChange(event) {
    setBgColor(event.target.value);
  }
  return (
    <div>
      <select onChange={handleColorChange}>
          <option value="">Select a color</option>
          <option value="red">Red</option>
          <option value="blue">Blue</option>
          <option value="green">Green</option>
          <option value="yellow">Yellow</option>
        </select>
        <div style={{ backgroundColor: bgColor, width: '100px', height: '100px'
}}></div>
      </div>
  );
}

export default ColorSwitcher;
```

### 7. Search Filter

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

const items = ['Apple', 'Banana', 'Carrot', 'Date', 'Eggplant', 'Fig', 'Grape'];

function SearchFilter() {
  const [search, setSearch] = useState(''); // Declare search state

  const filteredItems = items.filter(item =>
    item.toLowerCase().includes(search.toLowerCase())
  );

  return (
    <div>
      <input
        type="text"
        value={search}
        onChange={(e) => setSearch(e.target.value)} // Update search state on
input change
      />
      <ul>
        {filteredItems.map((item, index) => (
          <li key={index}>{item}</li>
        ))}
      </ul>
    </div>
  );
}

export default SearchFilter;
```

### 8. Drag and Drop List

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

const initialItems = ['Item 1', 'Item 2', 'Item 3', 'Item 4', 'Item 5'];

function DragDropList() {
  const [items, setItems] = useState(initialItems);
  const [draggingItem, setDraggingItem] = useState(null);

  function handleDragStart(index) {
```

```
    setDraggingItem(index);
  }

  function handleDragOver(index) {
    if (draggingItem === null || draggingItem === index) return;

    const newItems = [...items];
    const draggingItemValue = newItems[draggingItem];
    newItems.splice(draggingItem, 1);
    newItems.splice(index, 0, draggingItemValue);
    setDraggingItem(index);
    setItems(newItems);
  }

  function handleDragEnd() {
    setDraggingItem(null);
  }
  return (
    <div>
      <ul>
        {items.map((item, index) => (
          <li
            key={index}
            draggable
            onDragStart={() => handleDragStart(index)}
            onDragOver={(e) => {
              e.preventDefault(); // Needed to allow dropping
              handleDragOver(index);
            }}
            onDragEnd={handleDragEnd}
            style={{
              padding: '8px',
              border: '1px solid #ccc',
              margin: '4px',
              cursor: 'move',
              backgroundColor: draggingItem === index ? '#f0f0f0' : 'white',
            }}>
            {item}
          </li>
        ))}
      </ul>
    </div>
);}
export default DragDropList;
```