React Practical Experiments

1 Create a react application that tracks and renders the count of mouse clicks inside an element using use- state hook. 2 Create a react application with JSX simple and nested components using props and render it to the dom in vs code 3 Create a react application to handle user form input submission and save to a JSON file 4 Create a react application with JSX components based on conditional rendering from an input value 5 Create a react application that renders and adds to a list of items dynamically using the map function 6 Create a react application that renders a dynamic slider of images with configurable props.

give this 6 experiment code for sem practical exam

Here are 6 React experiment codes for your semester practical exam, each covering a core concept like state handling, props, form submission, conditional rendering, dynamic lists, and configurable components.

---

1. Mouse Click Counter Using `useState`

```jsx
// App.jsx
import React, { useState } from 'react';

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

  return (
    <div onClick={() => setCount(count + 1)} style={{ padding: '50px', border: '2px solid black' }}>
      <h2>Click inside this box</h2>
      <p>Click Count: {count}</p>
    </div>
  );
}

export default ClickCounter;
```

---

2. JSX Simple and Nested Components with Props

```jsx
// App.jsx
import React from 'react';

function Welcome(props) {
  return <h2>Hello, {props.name}</h2>;
}

function Greeting() {
  return (
    <div>
      <Welcome name="Alice" />
      <Welcome name="Bob" />
    </div>
```

```
  );
}

export default Greeting;
```

---

3. Form Input Submission and Save to JSON File (Simulated)

Note: In-browser apps can't directly write files. Use this mock "save" simulation.

```
jsx


// App.jsx
import React, { useState } from 'react';

function UserForm() {
  const [formData, setFormData] = useState({ name: '', email: '' });
  const [jsonData, setJsonData] = useState(null);

  const handleChange = (e) => {
    setFormData({...formData, [e.target.name]: e.target.value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    setJsonData(JSON.stringify(formData, null, 2));
  };

  return (
    <div>
      <h2>User Form</h2>
      <form onSubmit={handleSubmit}>
        <input name="name" placeholder="Name" onChange={handleChange} />
        <input name="email" placeholder="Email" onChange={handleChange} />
        <button type="submit">Submit</button>
      </form>
      {jsonData && <pre>{jsonData}</pre>}
    </div>
  );
}

export default UserForm;
```

---

4. Conditional Rendering Based on Input Value

```
jsx


// App.jsx
import React, { useState } from 'react';

function AgeChecker() {
  const [age, setAge] = useState('');

  return (
    <div>
      <input
        type="number"
        placeholder="Enter your age"
        onChange={(e) => setAge(e.target.value)}
      />
      {age && (
        <h2>{age >= 18 ? "You are an adult" : "You are a minor"}</h2>
      )}
    </div>
  );
}
```

```
export default AgeChecker;
```

---

5. Dynamic List Rendering with `map()`

```
jsx

// App.jsx
import React, { useState } from 'react';

function ItemList() {
  const [items, setItems] = useState([]);
  const [input, setInput] = useState('');

  const addItem = () => {
    if (input) {
      setItems([...items, input]);
      setInput('');
    }
  };

  return (
    <div>
      <h2>Item List</h2>
      <input value={input} onChange={(e) => setInput(e.target.value)} />
      <button onClick={addItem}>Add</button>
      <ul>
        {items.map((item, index) => <li key={index}>{item}</li>)}
      </ul>
    </div>
  );
}

export default ItemList;
```

---

6. Dynamic Image Slider with Props

```
jsx

// App.jsx
import React, { useState } from 'react';

function ImageSlider({ images }) {
  const [index, setIndex] = useState(0);

  const next = () => setIndex((index + 1) % images.length);
  const prev = () => setIndex((index - 1 + images.length) % images.length);

  return (
    <div>
      <button onClick={prev}>(</button>
      <img src={images[index]} alt="slide" style={{ width: '300px', height: '200px' }} />
      <button onClick={next}>)</button>
    </div>
  );
}

const images = [
  'https://via.placeholder.com/300x200?text=1',
  'https://via.placeholder.com/300x200?text=2',
  'https://via.placeholder.com/300x200?text=3',
];

export default function App() {
```

```
    return <ImageSlider images={images} />;
}
```

Let me know if you want all of these packaged in one project or converted into practical lab manual format (with aim, procedure, etc.).