NAME – Vinit Yadav(23BCE1767)

Lab 15 Web programming

Code:

App.jsx

import React, { useState } from "react";

// Component Imports

import Header from "./components/Header";

import Content from "./components/Content";

import Footer from "./components/Footer";

import StyledButton from "./components/StyledButton";

import LifecycleDemo from "./components/LifecycleDemo";

import Parent from "./components/Parent";

import CounterState from "./components/CounterState";

import CounterReducer from "./components/CounterReducer";

import JokeFetcher from "./components/JokeFetcher";

import FocusInput from "./components/FocusInput";

import ThemeToggler from "./components/ThemeToggler";

import FormUseState from "./components/FormUseState";

import FormUseRef from "./components/FormUseRef";

// Component Menu Map

const componentMap = {

  "Header + Content + Footer": (

    <>

      <Header title="Welcome to the React App" />

      <Content />

      <Footer />

    </>

  ),

  "StyledButton": <StyledButton />,

  "LifecycleDemo": <LifecycleDemo />,

  "Props Example": <Parent />,

  "Counter (useState)": <CounterState />,

  "Counter (useReducer)": <CounterReducer />,

  "Joke Fetcher (useEffect)": <JokeFetcher />,

  "Focus Input (useRef)": <FocusInput />,

  "Theme Toggler (useContext)": <ThemeToggler />,

  "Form (useState)": <FormUseState />,

  "Form (useRef)": <FormUseRef />,

};

function App() {

  const [activeComponent, setActiveComponent] = useState("Header + Content + Footer");

  return (

    <div style={{ padding: "20px", fontFamily: "Arial" }}>

      <h1>React Lab Component Demo</h1>

      <div style={{ marginBottom: "20px" }}>

        {Object.keys(componentMap).map((key) => (

          <button

            key={key}

            onClick={() => setActiveComponent(key)}

            style={{

              margin: "5px",

              padding: "10px",

              cursor: "pointer",

              backgroundColor: activeComponent === key ? "#007bff" : "#f0f0f0",

              color: activeComponent === key ? "#fff" : "#000",

              border: "none",

              borderRadius: "5px",

            }}

          >

            {key}

          </button>

        ))}

      </div>

      <div

        style={{

          padding: "20px",

          border: "1px solid #ccc",

          borderRadius: "10px",

          backgroundColor: "#fafafa",

        }}

      >

        {componentMap[activeComponent]}

      </div>

    </div>

  );

}

export default App;

Child.jsx

import PropTypes from 'prop-types';

function Child({ message }) {

  return <p>Message from Parent: {message}</p>;

}

Child.propTypes = {

  message: PropTypes.string.isRequired,

};

export default Child;

Content

import { useState } from 'react';

const jokes = [

  "Why don't scientists trust atoms? Because they make up everything!",

  "Why did the scarecrow win an award? Because he was outstanding in his field!",

  "Why don’t programmers like nature? It has too many bugs."

];

function Content() {

  const [joke, setJoke] = useState("");

  const handleClick = () => {

    const random = jokes[Math.floor(Math.random() \* jokes.length)];

    setJoke(random);

  };

  return (

    <div>

      <button onClick={handleClick}>Get Joke</button>

      <p>{joke}</p>

    </div>

  );

}

export default Content;

Counter

import { useReducer } from 'react';

const reducer = (state, action) => {

  switch (action.type) {

    case 'increment': return state + 1;

    case 'decrement': return state - 1;

    default: return state;

  }

};

function CounterReducer() {

  const [count, dispatch] = useReducer(reducer, 0);

  return (

    <div>

      <h3>Count: {count}</h3>

      <button onClick={() => dispatch({ type: 'increment' })}>Increase</button>

      <button onClick={() => dispatch({ type: 'decrement' })}>Decrease</button>

    </div>

  );

}

export default CounterReducer;

State

import { useState } from 'react';

function CounterState() {

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

  return (

    <div>

      <h3>Count: {count}</h3>

      <button onClick={() => setCount(count + 1)}>Increase</button>

      <button onClick={() => setCount(count - 1)}>Decrease</button>

    </div>

  );

}

export default CounterState;

Focus

import { useRef } from 'react';

function FocusInput() {

  const inputRef = useRef();

  const handleClick = () => {

    inputRef.current.focus();

  };

  return (

    <div>

      <input ref={inputRef} type="text" placeholder="Type here..." />

      <button onClick={handleClick}>Focus Input</button>

    </div>

  );

}

export default FocusInput;

Footer

function Footer() {

    return <footer>© 2025 React Lab</footer>;

  }

  export default Footer;

Form

import { useRef, useState } from 'react';

function FormUseRef() {

  const nameRef = useRef();

  const emailRef = useRef();

  const [submitted, setSubmitted] = useState(null);

  const handleSubmit = (e) => {

    e.preventDefault();

    setSubmitted({

      name: nameRef.current.value,

      email: emailRef.current.value,

    });

  };

  return (

    <form onSubmit={handleSubmit}>

      <input ref={nameRef} placeholder="Name" />

      <input ref={emailRef} placeholder="Email" />

      <button type="submit">Submit</button>

      {submitted && (

        <div>

          <p>Name: {submitted.name}</p>

          <p>Email: {submitted.email}</p>

        </div>

      )}

    </form>

  );

}

export default FormUseRef;

Formuse state

import { useState } from 'react';

function FormUseState() {

  const [form, setForm] = useState({ name: '', email: '' });

  const handleChange = (e) => {

    setForm({ ...form, [e.target.name]: e.target.value });

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    console.log(form);

  };

  return (

    <form onSubmit={handleSubmit}>

      <input name="name" value={form.name} onChange={handleChange} placeholder="Name" />

      <input name="email" value={form.email} onChange={handleChange} placeholder="Email" />

      <button type="submit">Submit</button>

      <p>Name: {form.name}</p>

      <p>Email: {form.email}</p>

    </form>

  );

}

export default FormUseState;

Header

function Header({ title }) {

    return <h1>{title}</h1>;

  }

  export default Header;

Index css

:root {

  font-family: system-ui, Avenir, Helvetica, Arial, sans-serif;

  line-height: 1.5;

  font-weight: 400;

  color-scheme: light dark;

  color: rgba(255, 255, 255, 0.87);

  background-color: #242424;

  font-synthesis: none;

  text-rendering: optimizeLegibility;

  -webkit-font-smoothing: antialiased;

  -moz-osx-font-smoothing: grayscale;

}

a {

  font-weight: 500;

  color: #646cff;

  text-decoration: inherit;

}

a:hover {

  color: #535bf2;

}

body {

  margin: 0;

  display: flex;

  place-items: center;

  min-width: 320px;

  min-height: 100vh;

}

h1 {

  font-size: 3.2em;

  line-height: 1.1;

}

button {

  border-radius: 8px;

  border: 1px solid transparent;

  padding: 0.6em 1.2em;

  font-size: 1em;

  font-weight: 500;

  font-family: inherit;

  background-color: #1a1a1a;

  cursor: pointer;

  transition: border-color 0.25s;

}

button:hover {

  border-color: #646cff;

}

button:focus,

button:focus-visible {

  outline: 4px auto -webkit-focus-ring-color;

}

@media (prefers-color-scheme: light) {

  :root {

    color: #213547;

    background-color: #ffffff;

  }

  a:hover {

    color: #747bff;

  }

  button {

    background-color: #f9f9f9;

  }

}

body {

  background-color: white;

  color: black;

}

Joke fetch

import { useEffect, useState } from 'react';

function JokeFetcher() {

  const [joke, setJoke] = useState('');

  const fetchJoke = async () => {

    const res = await fetch('https://official-joke-api.appspot.com/random\_joke');

    const data = await res.json();

    setJoke(`${data.setup} - ${data.punchline}`);

  };

  useEffect(() => {

    fetchJoke();

  }, []);

  return (

    <div>

      <h3>Random Joke:</h3>

      <p>{joke}</p>

      <button onClick={fetchJoke}>Get New Joke</button>

    </div>

  );

}

export default JokeFetcher;

Lifecycle demo

import React, { Component } from 'react';

class LifecycleDemo extends Component {

  constructor(props) {

    super(props);

    this.state = { count: 0 };

    console.log('Constructor');

  }

  componentDidMount() {

    console.log('Component Did Mount');

  }

  componentDidUpdate() {

    console.log('Component Did Update');

  }

  componentWillUnmount() {

    console.log('Component Will Unmount');

  }

  render() {

    return (

      <div>

        <p>Count: {this.state.count}</p>

        <button onClick={() => this.setState({ count: this.state.count + 1 })}>

          Update Count

        </button>

      </div>

    );

  }

}

export default LifecycleDemo;

main

import React from 'react';

import ReactDOM from 'react-dom/client';

import App from './App';

import { ThemeProvider } from './components/ThemeContext';

import './styles.css'; // if you’re using external CSS

ReactDOM.createRoot(document.getElementById('root')).render(

  <React.StrictMode>

    <ThemeProvider>

      <App />

    </ThemeProvider>

  </React.StrictMode>

);

Parent

import Child from './Child';

function Parent() {

  return <Child message="Hello from Parent!" />;

}

export default Parent;

import './styles.css';

function StyledButton() {

  return <button className="external-btn">Externally Styled Button</button>;

}

export default StyledButton;

import { createContext, useState } from 'react';

export const ThemeContext = createContext();

export function ThemeProvider({ children }) {

  const [theme, setTheme] = useState('light');

  const toggleTheme = () => {

    setTheme(prev => (prev === 'light' ? 'dark' : 'light'));

  };

  const value = { theme, toggleTheme };

  return (

    <ThemeContext.Provider value={value}>

      <div style={{ background: theme === 'light' ? '#fff' : '#333', color: theme === 'light' ? '#000' : '#fff', minHeight: '100vh' }}>

        {children}

      </div>

    </ThemeContext.Provider>

  );

}

import { useContext } from 'react';

import { ThemeContext } from './ThemeContext';

function ThemeToggler() {

  const { theme, toggleTheme } = useContext(ThemeContext);

  return (

    <div>

      <p>Current Theme: {theme}</p>

      <button onClick={toggleTheme}>Toggle Theme</button>

    </div>

  );

}

export default ThemeToggler;

Output:

