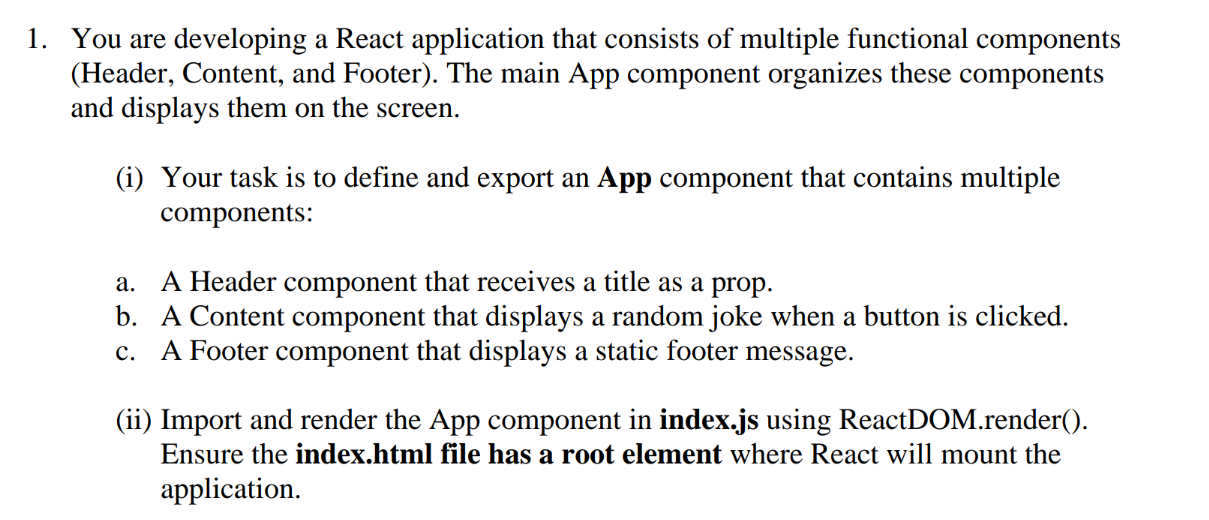
WEB PROGRAMMING

EXERCISE-15

M MENMANGAI

23BCE1022



I) import React from "react";

const Header=()=>{

    return(

        <header>

            <h1>This is where you click to get the code</h1>

        </header>

    );

};

export default Header;

II) import React, { useState } from "react";

const jokes = [

  "Why don’t skeletons fight each other? They don’t have the guts!",

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

];

const Content = () => {

  const [joke, setJoke] = useState("Click the button to see a joke!");

  const showRandomJoke = () => {

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

    setJoke(jokes[randomIndex]);

  };

  return (

    <div>

      <p>{joke}</p>

      <button onClick={showRandomJoke}>Get a Joke</button>

    </div>

  );

};

export default Content;

III) import React from "react";

const Footer = () => {

  return (

    <footer>

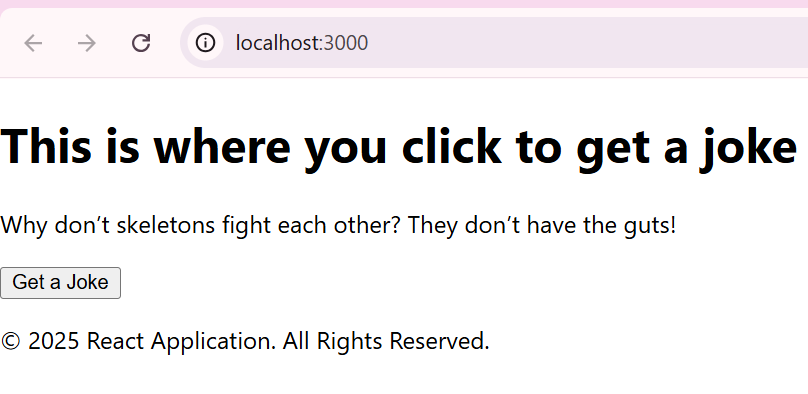
      <p>© 2025 React Application. All Rights Reserved.</p>

    </footer>

  );

};

export default Footer;



2)inline css button styling

import React from "react";

const StyledButtonInline = () => {

  const buttonStyle = {

    backgroundColor: "blue",

    color: "white",

    padding: "10px 20px",

    fontSize: "16px",

    border: "none",

    cursor: "pointer",

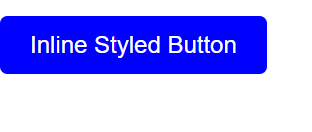
    borderRadius: "5px"

  };

  return <button style={buttonStyle}>Inline Styled Button</button>;

};

export default StyledButtonInline;



3)internal CSS styling

import React from "react";

const StyledButtonInternal = () => {

  return (

    <>

      <style>

        {`

          .internal-button {

            background-color: green;

            color: white;

            padding: 10px 20px;

            font-size: 16px;

            border: none;

            cursor: pointer;

            border-radius: 5px;

          }

        `}

      </style>

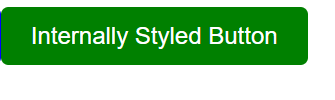
      <button className="internal-button">Internally Styled Button</button>

    </>

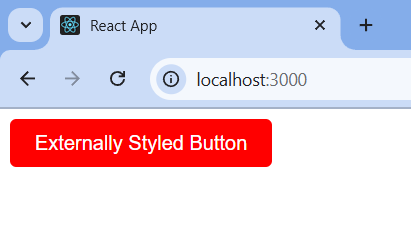
  );

};

export default StyledButtonInternal;



4)External CSS styling



Combining all the three in app.js

import React from "react";

import StyledButtonInline from "./components/StyledButtonInline";

import StyledButtonInternal from "./components/StyledButtonInternal";

import StyledButtonExternal from "./components/StyledButtonExternal";

const App = () => {

  return (

    <div>

      <h1>React CSS Styling Examples</h1>

      <StyledButtonInline />

      <br /><br />

      <StyledButtonInternal />

      <br /><br />

      <StyledButtonExternal />

    </div>

  );

};

export default App;

5) import React, { Component } from "react";

class LifecycleDemo extends Component {

  constructor(props) {

    super(props);

    this.state = { count: 0 };

    console.log("Constructor: Component is being initialized.");

  }

  componentDidMount() {

    console.log("componentDidMount: Component mounted.");

  }

  componentDidUpdate() {

    console.log("componentDidUpdate: Component updated.");

  }

  componentWillUnmount() {

    console.log("componentWillUnmount: Component is being removed.");

  }

  increment = () => {

    this.setState({ count: this.state.count + 1 });

  };

  render() {

    return (

      <div>

        <h2>Lifecycle Demo</h2>

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

        <button onClick={this.increment}>Increase</button>

        <button onClick={this.props.unmount}>Unmount Component</button>

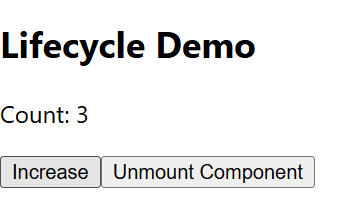
      </div>

    );

  }

}

export default LifecycleDemo;



6) import React, { useState } from "react";

const Counter = () => {

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

return (

<div>

<h2>Counter (useState)</h2>

<p>Count: {count}</p>

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

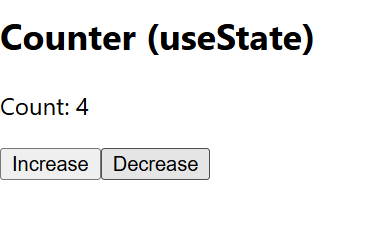
<button onClick={() => setCount(count - 1)}>Decrease</button>

</div>

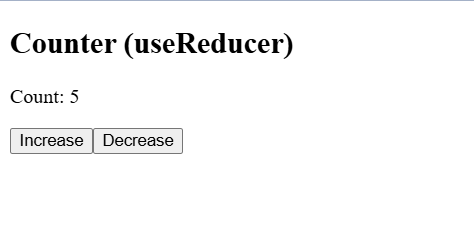
);

};

export default Counter;



Using useReducer():



7) import React, { useState, useEffect } from "react";

const JokeFetcher = () => {

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

  const fetchJoke = async () => {

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

    const data = await response.json();

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

  };

  useEffect(() => {

    fetchJoke();

  }, []);

  return (

    <div>

      <h2>Random Joke</h2>

      <p>{joke}</p>

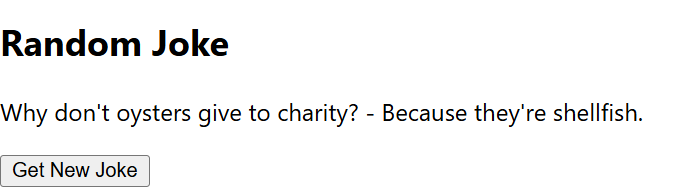
      <button onClick={fetchJoke}>Get New Joke</button>

    </div>

  );

};

export default JokeFetcher;



8) import React, { useRef } from "react";

const RefExample = () => {

  const inputRef = useRef(null);

  const focusInput = () => {

    inputRef.current.focus();

  };

  return (

    <div>

      <h2>useRef Example</h2>

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

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

    </div>

  );

};

export default RefExample;



Focused output:



9)

import React, { useState, createContext, useContext } from "react";

// Creating Theme Context

const ThemeContext = createContext();

const ThemeProvider = ({ children }) => {

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

const toggleTheme = () => {

setTheme((prevTheme) => (prevTheme === "light" ? "dark" : "light"));

};

return (

<ThemeContext.Provider value={{ theme, toggleTheme }}>

<div style={{

background: theme === "light" ? "#fff" : "#333",

color: theme === "light" ? "#000" : "#fff",

padding: "20px",

textAlign: "center"

}}>

{children}

</div>

</ThemeContext.Provider>

);

};

// Theme Toggler Component

const ThemeToggler = () => {

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

return (

<div>

<h2>Current Theme: {theme}</h2>

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

</div>

);

};

// Main App Component

const App = () => {

return (

<ThemeProvider>

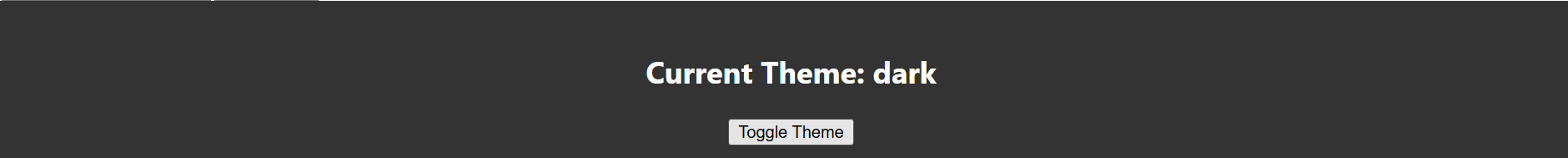
<ThemeToggler />

</ThemeProvider>

);

};

export default App;

10) import React from "react";

const Child = ({ message }) => {

return <h2>Message from Parent: {message}</h2>;

};

const Parent = () => {

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

};

export default Parent;



11) import React from "react";

import PropTypes from "prop-types";

const Child = ({ message }) => {

return <h2>Message from Parent: {message}</h2>;

};

// Prop Validation

Child.propTypes = {

message: PropTypes.string.isRequired,

};

const Parent1 = () => {

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

};

export default Parent1;



12) import React, { useState } from "react";

const Formvalidate = () => {

  const [details, setDetails] = useState({ name: "", email: "" });

  const changeDetails = (e) => {

    setDetails({ ...details, [e.target.name]: e.target.value });

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    alert(`The name is: ${details.name} and the email is: ${details.email}`);

  };

  return (

    <div>

      <form onSubmit={handleSubmit}>

        <label>Enter your name:

          <input

            type="text"

            name="name"

            placeholder="Your name"

            value={details.name}

            onChange={changeDetails}

          />

        </label>

        <br />

        <label>Enter your Email:

          <input

            type="email"

            name="email"

            placeholder="abc.123@gmail.com"

            value={details.email}

            onChange={changeDetails}

          />

        </label>

        <br />

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

      </form>

      <p>Name: {details.name}</p>

      <p>Email: {details.email}</p>

    </div>

  );

};

export default Formvalidate;

