1. Props – Practice Tasks

1. Create a UserCard component that takes name and age as props and displays them.

function UserCard({ name, age }) {

return

(

<h2>name is{name} ,age is {age} </h2>

);

}

2. Pass a list of hobbies as a prop to a HobbiesList component and render them as <li> items.

function HobbiesList({ hobbies }) {

return (

<ul>

{hobbies.map((swim, dance) => <li key= {dance}>{swim} </li>)

}

</ul>

);

}

3. Build a Button component that takes label and color as props and styles itself.

function Button({ label, color }) {

return

(

<button style={{ backgroundColor: white, color: "pink" }}>{label}</button>

);

}

4. Create a Profile component that takes a user object prop and displays the username and email.

function Profile({ user })

{

return

(

<p>{user.username} ({user.email})</p>

);

}

5. Pass a function as a prop to a ClickButton component that logs “Button clicked!” when pressed.

function ClickButton({ onClick })

{

return

(

<button onClick={onClick}>Click Me</button>

);

}

6. Build a Greeting component that displays “Good Morning” or “Good Evening” based on a time prop.

function Greeting({ time }) {

return

(

<h2>{time < 12 ? "Good Morning" : "Good Evening"}</h2>

);

}

7. Create a Counter component where the starting value is passed as a prop.

import { useState } from "react";

function Counter({ start })

{

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

return (

<>

<h1>{count}</h1>

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

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

</>

);

}

8. Pass an image URL as a prop to an Avatar component and render it in <img>.

function Avatar({ url })

{

return

(

<img src={url} alt="Avatar" width="100" />

);

}

9. Create a Card component that takes title and children as props and displays them in a styled card.

function Card({ title, children })

{

return (

<div style={{ border: "1px black", padding: "10px" }}>

<h3>{card}</h3>

{children}

</div>

);

}

10. Build a Product component that receives price and discount props and displays the discounted price.

function Product({ price, discount }) {

const final = price - (price \* discount) / 100;

return

(

<p>discounted price is: ${final.toFixed(2)}</p>

);

}

2. Hooks – Practice Tasks

1. Create a Timer component using useState and useEffect to count seconds.

import { useState, useEffect } from ‘react’;

function Timer()

{

const [seconds, setSeconds] = useState(0);

useEffect(() => {

const interval = setInterval(() => setSeconds(s => s + 1), 1000);

return () => clearInterval(interval);

}, []);

return <h1>{seconds}s</h1>;

}

2. Build a MouseTracker component that shows the current mouse position using useState and useEffect.

import { useEffect, useState } from "react";

function MouseTracker() {

const [pos, setPos] = useState({ x: 0, y: 0 });

useEffect(() => {

const onMove = (e) => setPos({ x: e.clientX, y: e.clientY });

window.addEventListener("mousemove", onMove);

return () => window.removeEventListener("mousemove", onMove);

}, []);

return <div>Mouse: {pos.x}, {pos.y}</div>;

}

3. Use useRef to focus an input field when a button is clicked.

import { useRef } from "react";

function FocusInput() {

const inputRef = useRef();

return (

<>

<input ref={inputRef} />

<button onClick={() => inputRef.current.focus()}>Focus</button>

</>

);

}

4. Create a form that uses useState to store and display input values in real time.

import { useState } from ‘react’;

function Form() {

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

return (

<>

<input value={text} onChange={e => setText(e.target.value)} />

<p>{text}</p>

</>

);

}

5. Build a theme switcher using useState and useEffect to store the selected theme in localStorage.

import { useState, useEffect } from ‘react’;

function ThemeSwitcher()

{

const [theme, setTheme] = useState(localStorage.getItem("theme") || "light");

useEffect(() => localStorage.setItem("theme", theme), [theme]);

return (

<button onClick={() => setTheme(theme === "light" ? "dark" : "light")}>

{theme} mode

</button>

);

}

6. Use useReducer to create a counter with increment, decrement, and reset buttons.

import { useReducer } from "react";

function reducer(state, action) {

switch (action.type) {

case "inc": return state + 1;

case "dec": return state - 1;

case "reset": return 0;

default: return state;

}

}

function ReducerCounter() {

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

return (

<div>

<button onClick={() => dispatch({ type: "dec" })}>-</button>

<span style={{ margin: "0 8px" }}>{count}</span>

<button onClick={() => dispatch({ type: "inc" })}>+</button>

<button onClick={() => dispatch({ type: "reset" })}>Reset</button>

</div>

);

}

7. Use useMemo to calculate and display a list of prime numbers up to a given number.

import { useMemo, useState } from "react";

function Primes() {

const [n, setN] = useState(50);

const primes = useMemo(() => {

const res = [];

outer: for (let i = 2; i <= n; i++) {

for (let p of res) { if (p \* p > i) break; if (i % p === 0) continue outer; }

res.push(i);

}

return res;

}, [n]);

return (

<>

<input type="number" value={n} onChange={e => setN(+e.target.value || 0)} />

<div>{primes.join(", ")}</div>

</>

);

}

8. Create a WindowSize component using useEffect to track window width/height.

import { useEffect } from ‘react’;

function WindowSize() {

const [size, setSize] = useState({ w: window.innerWidth, h: window.innerHeight });

useEffect(() => {

const handler = () => setSize({ w: window.innerWidth, h: window.innerHeight });

window.addEventListener("resize", handler);

return () => window.removeEventListener("resize", handler);

}, []);

return <p>{size.w} x {size.h}</p>;

}

9. Build a Stopwatch component that starts, stops, and resets using hooks.

import { useState, useEffect } from ‘react’;

function Stopwatch() {

const [time, setTime] = useState(0);

const [running, setRunning] = useState(false);

useEffect(() => {

if (running) {

const id = setInterval(() => setTime(t => t + 1), 1000);

return () => clearInterval(id);

}

}, [running]);

return (

<>

<h1>{time}s</h1>

<button onClick={() => setRunning(true)}>Start</button>

<button onClick={() => setRunning(false)}>Stop</button>

<button onClick={() => setTime(0)}>Reset</button>

</>

);

}

10. Create a DarkModeToggle component using useState and useEffect to switch background color.

import { useState, useEffect } from ‘react’;

function DarkModeToggle() {

const [dark, setDark] = useState(false);

useEffect(() => {

document.body.style.background = dark ? "#333" : "#fff";

}, [dark]);

return (

<button onClick={() => setDark(!dark)}>

{dark ? "Light Mode" : "Dark Mode"}

</button>

);

}

3. State – Practice Tasks

1. Create a counter using useState with + and – buttons.

import { useState } from ‘react’;

function Counter() {

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

return (

<>

<h1>{count}</h1>

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

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

</>

);

}

2. Make a toggle switch that changes between “ON” and “OFF” state.

import { useState } from ‘react’;

function Toggle() {

const [on, setOn] = useState(false);

return <button onClick={() => setOn(!on)}>{on ? "ON" : "OFF"}</button>;

}

3. Build a form that stores name, email, and age in a single state object.

import { useState } from "react";

function ProfileForm() {

const [data, setData] = useState({ name: "", email: "", age: "" });

const upd = e => setData({ ...data, [e.target.name]: e.target.value });

return (

<>

<input name="name" value={data.name} onChange={e => setForm({ ...form, name: e.target.value })}

placeholder="Name" />

<input name="email" value={data.email} onChange={e => setForm({ ...form, email: e.target.value })}

placeholder="Email" />

<input name="age" value={data.age} onChange={e => setForm({ ...form, age: e.target.value })}

placeholder="Age" />

<pre>{JSON.stringify(data, null, 2)}</pre>

</>

);

}

4. Create a random quote generator where each click changes the displayed quote.

import { useState } from ‘react’;

const quotes = ["Be happy", "Stay strong", "Keep going"];

function Quote() {

const [q, setQ] = useState(quotes[0]);

return <button onClick={() => setQ(quotes[Math.floor(Math.random() \* quotes.length)])}>{q}</button>;

}

5. Store an array of todo items in state and render them in a list.

import { useState } from "react";

function Todos() {

const [todos, setTodos] = useState([]);

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

const add = () => { if (!text.trim()) return; setTodos([...todos, { id: Date.now(), text }]); setText(""); };

return (

<>

<input value={text} onChange={e => setText(e.target.value)} placeholder="Add todo" />

<button onClick={add}>Add</button>

<ul>{todos.map(t => <li key={t.id}>{t.text}</li>)}</ul>

</>

);

}

6. Create a text input that converts text to uppercase in real time.

import { useState } from ‘react’;

function Uppercase() {

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

return <input value={text} onChange={e => setText(e.target.value.toUpperCase())} />;

}

7. Make a “Like” button that increments a count each time it’s clicked.

function LikeButton() {

const [likes, setLikes] = useState(0);

return <button onClick={() => setLikes(likes + 1)}>Like {likes}</button>;

}

8. Build a color picker that changes the page background color.

import { useEffect } from ‘react’;

function ColorPicker() {

const [color, setColor] = useState("#ffffff");

useEffect(() => { document.body.style.background = color; }, [color]);

return <input type="color" value={color} onChange={e => setColor(e.target.value)} />;

}

9. Store a list of images in state and cycle through them with next/previous buttons.

import { useState } from "react";

function ImageCarousel({ images = [] }) {

const [i, setI] = useState(0);

const next = () => setI((i + 1) % images.length);

const prev = () => setI((i - 1 + images.length) % images.length);

if (!images.length) return <p>No images.</p>;

return (

<div>

<img src={images[i]} alt={`img-${i}`} style={{ maxWidth: 300 }} />

<div>

<button onClick={prev}>Prev</button>

<button onClick={next}>Next</button>

</div>

</div>

);

}

10. Implement a “character counter” for a textarea

import { useState } from ‘react’;

function CharCounter() {

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

return (

<>

<textarea value={text} onChange={e => setText(e.target.value)} />

<p>Characters: {text.length}</p>

</>

);

}