React Hooks, Lifecycle & Routing - Coding Assessment

1. Counter with Timer (Hooks + Lifecycle)

Create a CounterWithTimer component using functional React with Hooks that:

- Increments a counter every second using setInterval
- Has Start, Stop, and Reset buttons
- Uses useEffect and useRef
- Performs cleanup using the return function inside useEffect when the component unmounts or the timer is stopped

This tests understanding of useEffect as replacement for componentDidMount and componentWillUnmount.

2. React Router Setup (Navigation)

Implement React Router v6 in your app with the following pages:

- / -> Home page (include CounterWithTimer)
- /about -> Static about message
- /contact -> Simple contact form with name and email

Include a Navbar using <Link> for navigation between routes.

3. Lifecycle Handling with Navigation (Cleanup on Route Change)

Ensure that when navigating away from the Home route, the interval gets cleared automatically.

- Use useEffect with cleanup logic inside CounterWithTimer
- Show an alert like 'Timer stopped due to navigation' inside the cleanup

4. Protected Route with Redirect

Simulate a Protected Route using:

const isAuthenticated = false;

- When visiting /dashboard, show 'Welcome to Dashboard' only if isAuthenticated === true
- If not, redirect to /login

React Hooks, Lifecycle & Routing - Coding Assessment

Use React Router and conditionally render or use <Navigate /> from v6.

5. Create Custom Hook - useDocumentTitle

Create a custom hook named useDocumentTitle(title)

Usage: useDocumentTitle('Home Page');

- It should set the document.title dynamically whenever the component renders
- Use useEffect inside the custom hook

6. useEffect Dependency Practice

Create a component called SearchTracker that:

- Has a text input field for the user to type a search term
- Stores the input using useState
- Logs 'Search term changed: [value]' to the console every time the input value changes using useEffect

Bonus: Log only when the search term length > 3

7. Dynamic Routing with Parameters

Set up a route like /user/:id using React Router.

- Create a UserProfile component that extracts the user ID from the URL using useParams()
- Display the ID on the page like: User ID: 17
- Add a few Link buttons on the Home page like 'View User 1', 'View User 2', etc.

8. useReducer - Basic Todo List

Build a basic Todo List using the useReducer hook:

- Initial state: an empty list of todos
- User can add a todo using a text input
- Reducer should handle ADD_TODO and DELETE_TODO
- Display the list of todos with a delete button beside each