**LAB 4**

### 1. ****Need & Benefits of Component Lifecycle****

React components go through several phases from creation to destruction. Managing these phases is essential because:

**Control Side Effects:** Run code at specific times (e.g., fetch data when the component mounts).

**Manage Resources:** Initialize or clean up timers, subscriptions, or event listeners.

**Optimize Performance:** Decide when to update the UI and avoid unnecessary renders.

**Handle State & Props Changes:** React to data changes properly.

**Enhance User Experience:** Ensure smooth UI transitions and loading states.

**Benefits:**

| **Benefit** | **Explanation** |
| --- | --- |
| Predictable Component Behavior | Lifecycle hooks provide clear, structured points to execute code. |
| Better Resource Management | Helps allocate and release resources efficiently (e.g., API calls, timers). |
| Cleaner Code & Separation | Keeps setup, update, and teardown logic organized and separate. |
| Performance Optimization | Avoid unnecessary rendering with methods like shouldComponentUpdate. |

### 2. ****Various Lifecycle Hook Methods****

React class components have lifecycle methods grouped into phases:

| **Phase** | **Method** | **Purpose** |
| --- | --- | --- |
| **Mounting** | constructor() | Initialize state and bind methods. |
|  | static getDerivedStateFromProps() | Sync state with props before render (rarely used). |
|  | render() | Returns JSX to render UI. |
|  | componentDidMount() | Runs after component is inserted in the DOM; ideal for API calls. |
| **Updating** | static getDerivedStateFromProps() | Runs before rendering on prop/state changes. |
|  | shouldComponentUpdate() | Determines if component should re-render (performance optimization). |
|  | render() | Re-renders UI with updated state/props. |
|  | getSnapshotBeforeUpdate() | Capture some info (e.g., scroll position) before DOM changes. |
|  | componentDidUpdate() | Runs after updates are flushed to DOM. |
| **Unmounting** | componentWillUnmount() | Cleanup before component is removed (e.g., clear timers). |
| **Error Handling** | componentDidCatch() | Catches errors in child components. |

### 3. ****Sequence of Steps in Rendering a Component****

Here’s the typical lifecycle flow for mounting and updating:

#### Mounting Phase:

constructor() — Initialize component.

static getDerivedStateFromProps() — Update state based on props.

render() — Create React elements.

React updates the **real DOM** with rendered elements.

componentDidMount() — Component is fully rendered; start side effects here.

#### Updating Phase (when props or state change):

static getDerivedStateFromProps() — Sync state with props.

shouldComponentUpdate() — Decide if update is necessary.

render() — Re-render UI.

getSnapshotBeforeUpdate() — Capture info before DOM updates.

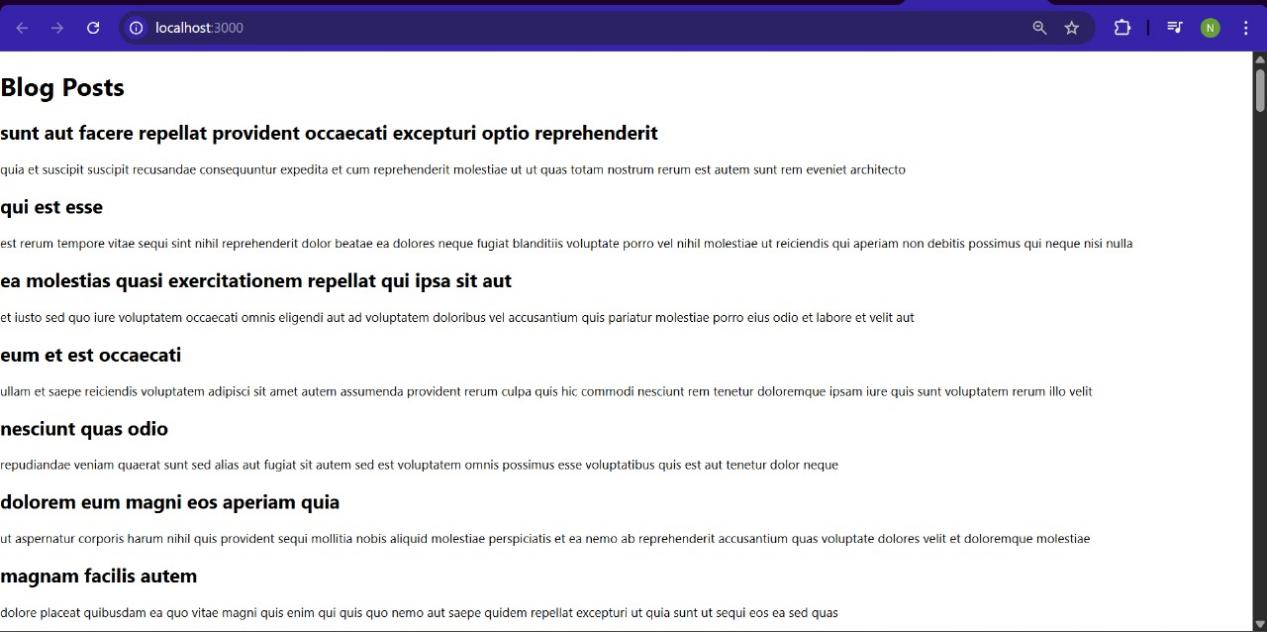
React updates the real DOM.

componentDidUpdate() — Post-update operations.

#### Unmounting Phase:

componentWillUnmount() — Cleanup operations before removal.

**Output of Lab 4:**

****