Industrial Training Daily Diary

MERN Stack Training

Company: Sensation Software Solutions
Student Name: Gurvinder Singh
Training Duration: 6 Months
Day: 13

Day 13 – React Component Lifecycle & useEffect Hook

Objective of the Day

The main objective of Day 13 was to understand how React components behave during different phases of their life and how **side effects** are handled in functional components using the **useEffect hook**. This knowledge is very important for real-world React applications where data fetching, subscriptions, and DOM updates are required.

React Component Lifecycle Overview

The trainer explained that every React component goes through a **lifecycle** from creation to removal. Understanding the lifecycle helps developers write efficient and optimized applications.

The lifecycle mainly includes:

- Mounting phase
- Updating phase
- Unmounting phase

In modern React, lifecycle behavior is mostly handled using **hooks** instead of class-based lifecycle methods.

Mounting Phase

The mounting phase occurs when a component is created and inserted into the DOM for the first time. During this phase, initial state is set and JSX is rendered on the screen.

We discussed how initial API calls and setup tasks are commonly performed during this phase using hooks.

Updating Phase

The updating phase occurs whenever props or state values change. React automatically re-renders the component to reflect updated data.

The trainer explained how unnecessary re-renders can impact performance and how hooks help control updates efficiently.

Unmounting Phase

The unmounting phase occurs when a component is removed from the DOM. This phase is important for cleaning up resources such as timers, event listeners, or subscriptions to avoid memory leaks.

Introduction to useEffect Hook

The **useEffect hook** was introduced as a replacement for multiple lifecycle methods in class-based components. It allows us to perform side effects in functional components.

Key uses of useEffect:

- Fetching data from APIs
- Updating the DOM
- Setting timers
- Subscribing and unsubscribing to events

---

useEffect Dependency Array

The trainer explained how the **dependency array** controls when useEffect is executed:

- Without dependency array → runs after every render
- Empty dependency array → runs only once (on mount)
- With dependencies → runs when specific values change

This concept helped us understand performance optimization.

Cleanup Function in useEffect

We learned about cleanup functions used inside useEffect to handle component unmounting. Cleanup functions help remove listeners, clear intervals, and cancel API calls.

This prevents memory leaks and improves application stability.

Practical Work Done

During the practical session, we:

- Created components using useEffect
- Simulated API calls using dummy data
- Observed component behavior on state changes
- Implemented cleanup functions

Hands-on practice made the lifecycle concept very clear.

Importance in MERN Stack Development

The trainer explained that lifecycle management and useEffect are essential for building scalable React applications that interact with backend services and handle asynchronous operations efficiently.

Conclusion of Day 13

Day 13 helped me understand how React components behave during their lifecycle and how useEffect manages side effects. This knowledge is crucial for building real-world MERN stack applications.

Outcome of the Day:

- Clear understanding of component lifecycle
- Practical knowledge of useEffect hook
- Ability to handle side effects and cleanup
- Improved React development confidence