

Industrial Training Daily Diary

MERN Stack Training

Company: Sensation Software Solutions

Student Name: Gurvinder Singh

Training Duration: 6 Months

Day: 18

Day 18 – Redux Deep Dive: Actions, Reducers, Store & React-Redux Integration

Objective of the Day

The objective of Day 18 was to gain **hands-on understanding of Redux**, including **actions, reducers, store creation**, and **integrating Redux with React** using **React-Redux** library.

Recap of Redux Basics

Before diving deeper, the trainer briefly revised:

- Redux purpose and importance
- Single source of truth
- Unidirectional data flow

This helped set the stage for advanced concepts.

Actions in Redux

The trainer explained that **actions** are plain JavaScript objects that describe events in the application.

Key points:

- Must have a `type` property
- Can have optional payload
- Actions are dispatched to reducers

We created sample actions for incrementing and decrementing a counter.

Reducers in Redux

Reducers are pure functions that specify how the state changes in response to actions.

Key characteristics:

- Accept current state and action as parameters
- Return new state without mutating the old state
- Should be pure (no side-effects)

Example: Counter reducer updating state based on increment or decrement action.

Store in Redux

The **store** holds the global state of the application.

Trainer explained:

- Creating a Redux store using `createStore`
 - Using `store.getState()` to view current state
 - Subscribing to state changes using `store.subscribe()`
 - Dispatching actions to update the state
-

React-Redux Integration

We learned how to connect Redux with React using the **React-Redux library**:

- `Provider` component wraps the root App component
- `connect` function or `useSelector` & `useDispatch` hooks to access state and dispatch actions

This enables React components to read data from Redux store and update state efficiently.

Practical Work Done

During lab session:

- Created a Redux store
- Defined actions and reducers for a counter app
- Connected Redux with React using `Provider` and hooks

- Implemented increment/decrement functionality
- Observed real-time UI updates based on Redux state

Hands-on practice reinforced concepts.

Weekly Task – Week 4 Extended

As part of weekly task:

- Build a small React-Redux application
 - Implement multiple actions and reducers
 - Demonstrate state updates in components
 - Push code and diary updates to GitHub Pages
-

Importance in MERN Stack Development

Redux is essential for managing complex state in large MERN stack applications. Integrating Redux with React ensures consistency and predictability of application data.

Conclusion of Day 18

Day 18 gave me practical knowledge of Redux internals and how to integrate it with React. I am now able to manage global state effectively in React applications.

Outcome of the Day:

- Understanding of actions, reducers, and store
- Ability to integrate Redux with React
- Hands-on implementation of state updates
- Completion of Week 4 exten