**Module-18) React - Applying Redux**

State Management (Redux, Redux-Toolkit or Recoil)

1. **What is Redux, and why is it used in React applications? Explain the core concepts of actions, reducers, and the store.**

* Redux is a state management library that helps manage application state in a predictable way.
* It is commonly used in React applications to maintain a centralized state and manage complex state interactions efficiently.
* **Why is Redux used in React applications?**
  + **Centralized State Management** – Redux provides a single source oftruth for the application state, making it easier to manage and debug.
  + **Predictability** – State updates in Redux follow a strict flow, making it easy to track and reproduce changes.
  + **Easier Debugging** – Tools like Redux DevTools allow time-travel debugging and state tracking.
  + **Improved Scalability** – Helps manage state effectively in large-scale applications with multiple components sharing data.
* **Core Concepts of Redux**

***1. Actions***

* *Actions* are plain JavaScript objects that describe what should happen in the application.
* They must have a type property that tells Redux what kind of action is being performed.

1. **Reducers**

* Reducers are pure functions that take the current state and an action and return a new state.
* They specify how the application’s state should change in response to actions.

1. **Store**

* The store is an object that holds the entire application state.
* It allows access to the state, dispatching actions, and subscribing to updates.

**2. How does Recoil simplify state management in React compared to Redux?**

* Recoil is a state management library for React that provides a simpler and more intuitive way to manage shared state. It is designed specifically for React applications, unlike Redux, which is framework-agnostic.
* **How Recoil Works: -**

1️.**Atoms (State Units)**

* An atom represents a piece of state.
* Any component that uses an atom will automatically re-render when its value changes.

2️.**Selectors (Derived State)**

* A selector allows you to compute derived state based on atoms.

3️.**Using Atoms in Components**

* Components can access and modify atoms using hooks like useRecoilState and useRecoilValue.