# Module - 13

**React – Applying Redux**

**Q – 1 :- What is Redux ?**

**Ans :- Redux is an open-source JavaScript library used to manage application state. React uses Redux for building the user interface.**

**React Redux is the official React binding for Redux. It allows React components to read data from a Redux Store, and dispatch Actions to the Store to update data. Redux helps apps to scale by providing a sensible way to manage state through a unidirectional data flow model. React Redux is conceptually simple. It subscribes to the Redux store, checks to see if the data which your component wants have changed, and re-renders your component.**

**Q – 2 :- What is Redux Thunk used for ?**

**Ans :- Thunk allow us to write additional redux-related logic seprate from a UI layer. This logic can include side effcts, such a async requests or generating values, as well as logic that requires dispatching multiple actions or access to the redux store state.**

**It’s common to have logic directly in components, such as making an async request In a click handler or a {useEffect} hook and then processing the results. However it’s pften necessary to move as much that logic as possible outside the ui layer. This may be done to improve testability of the logic, to keep the ui layer as thin and “presentational” as possible, or to improve code reuse and sharing.**

**Q – 3 :- What is Pure Component? When to use Pure Component over Component ?**

**Ans:- A PureComponent is a type of component in React that optimizes rendering performance by only re-rendering when its props or state change. PureComponents are also called "stateless components" or "dumb components".**

**React components are either PureComponents or Components. PureComponents are a special type of Component that implements a shouldComponentUpdate() method that prevents the component from re-rendering if its props and state have not changed.**

**PureComponents are typically used when you want to optimize performance and prevent unnecessary re-renders. They are especially useful for components that are frequently rendered, such as list items or table rows.**

**Q – 3 :- What is the second argument that can optionally be passed tosetState and what is its purpose ?**

**Ans :- The second argument that can optionally be passed to setState is a callback function which gets called immediately after the setState is completed and the components get re-rendered.**

**The purpose of the second argument to setState() is to perform actions immediately after a state change. This ensures that the data that is rendered will use the newest state.**