❖ What is Redux?

Redux is a state management library commonly used with javascript applications, particularly those built with frameworks like React. It helps manage and update the application's state in a predictable and organized manner.

Redux follows a unidirectional data flow, which means that the state is read-only, and any changes must be made through actions. Actions are plain javascript objects that describe the type of change or event that occurred in the application. They are dispatched to the Redux store, which then uses reducers to handle these actions and update the state accordingly.

❖ What is Redux Thunk used for?

Redux Thunk is a middleware library for Redux that allows you to write action creators that return functions instead of plain action objects. It provides a way to handle asynchronous actions in Redux applications.

The most common use case for Redux Thunk is handling asynchronous API calls. You can create a thunk that dispatches an initial action to indicate that the API call is in progress, then makes the asynchronous request, and finally dispatches additional actions to update the state based on the API response.

❖ What is Pure Component? When to use Pure Component over Component?

The main difference between a purecomponent and a regular Component is that a purecomponent performs a shallow comparison of props and state by default, whereas a regular Component does not implement shouldcomponentupdate and will re-render whenever its parent component re-renders, regardless of whether the props or state have changed.

- **What is the second argument that can optionally be passed tosetstate and what is its purpose?**

The second argument that can be passed to the setstate function is a callback function. This callback function is invoked after the state update is completed and the component has been re-rendered.

The purpose of the callback function is to perform any actions or operations that need to occur after the state has been updated and the component has re-rendered. It allows you to synchronize any code that relies on the updated state with the component's lifecycle.