Redux DevTools is a good extention

### **Create a React Redux App**

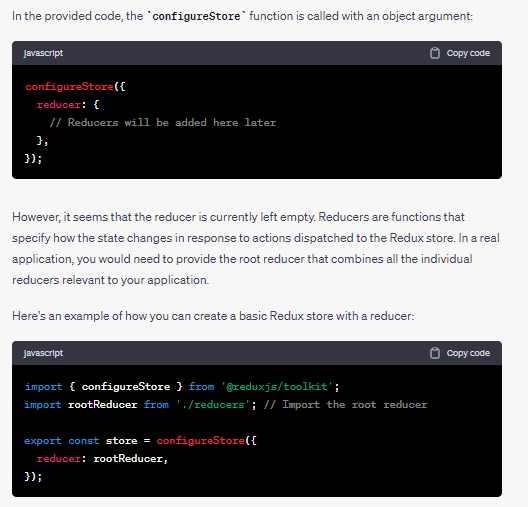
npx create-next-app --example with-redux my-app

Isky bad hum apna store banaty hain.It is recommended convetion ka hum src folder ma ek app ka nam sa folder banaien aur usky andr store.js ka nam sa file aur usky andr store ka code likhain.otherwise hum kuch aur apni mrzi mutabik bhi structure kr skty.



**configureStore** is a function imported from the **@reduxjs/toolkit** library. It is used to create a Redux store, which is a central data store in a Redux-based application. The Redux store holds the complete state tree of your application and allows you to manage the state and perform state updates in a predictable way.

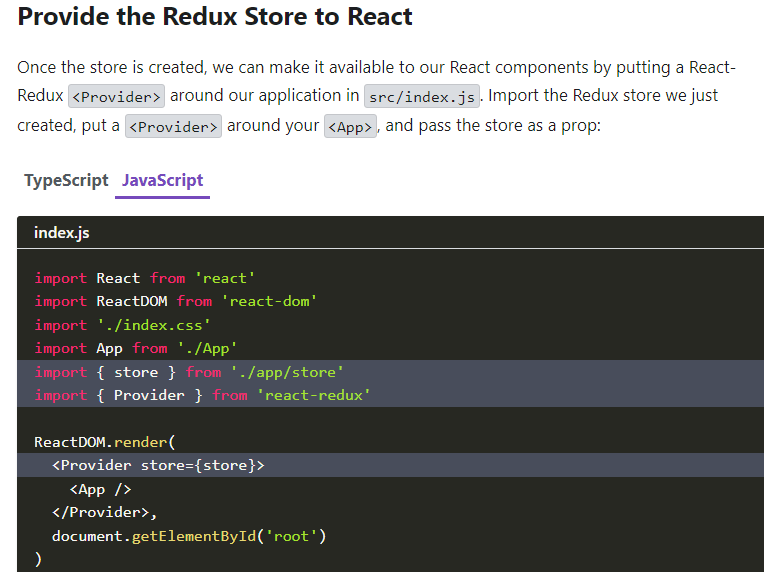
The purpose of using **configureStore** is to set up and configure the Redux store with certain default behaviors and middleware provided by Redux Toolkit. Redux Toolkit is an official package from the Redux team that simplifies and streamlines the process of working with Redux. It incorporates commonly used middleware and simplifies the Redux store setup, making it easier to manage and maintain your application's state.



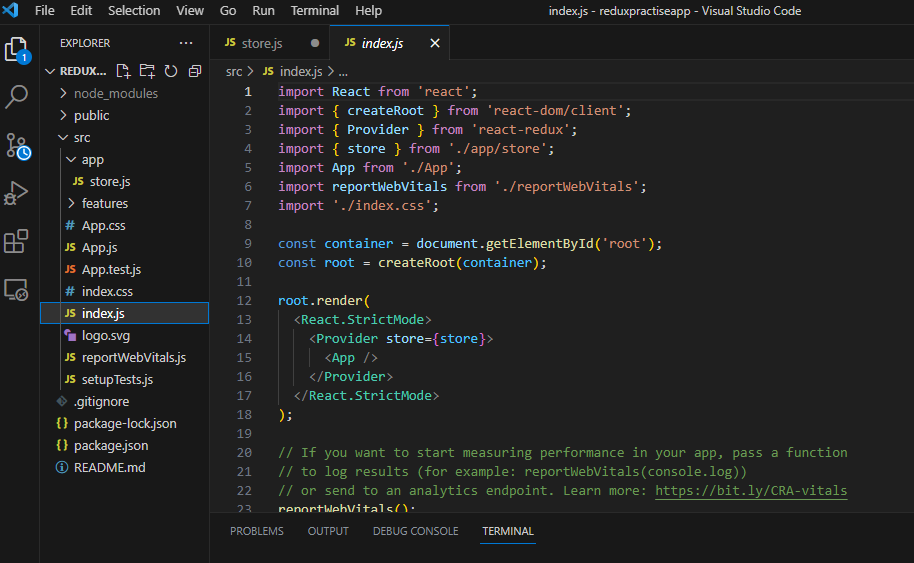
The configureStore() function is a higher-order function from the Redux Toolkit library. It is used to create a Redux store with a single function call. The configureStore() function takes a number of options as input, including the reducers, middleware, and enhancers that will be used by the store.

**Next Step**

Next humain ek provider provide krna ho ga apny root component ko



Humny provider ma store bhi dia ha taky pata tu chly ka ye app ye wala store access krny wala ha ye access kr skta ha.Ab isky mtlb ye howa ka <App/> aur isky andr ka tamam components child waly bhi access kr skty hain manipulate kr skty hain is store ko aur isky andr mojod states ko because hum redux use hi state management ka liay karty hain.



**Next Step**

### Create a Redux State Slice

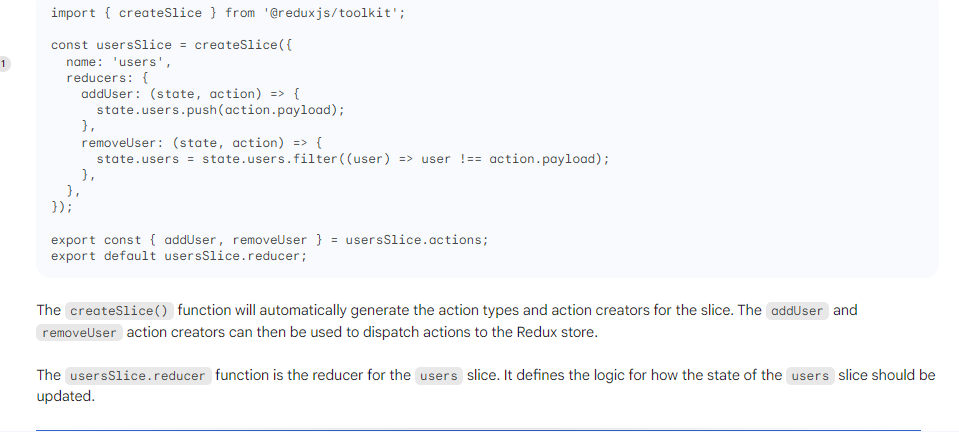
A Redux state slice is a way of organizing your Redux state into smaller, more manageable pieces. This can make your code easier to read, understand, and maintain.

To create a Redux state slice, you use the createSlice() function from the Redux Toolkit library. The createSlice() function takes an object as input, which defines the reducers, action types, and action creators for the slice.

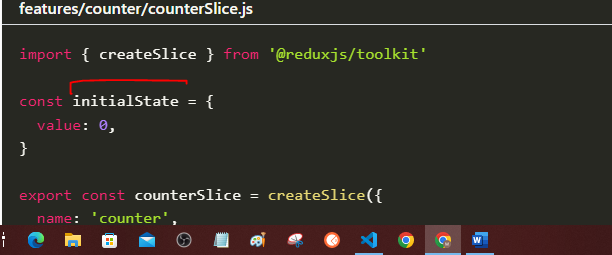
Redux state slices are a powerful way to organize your Redux state. They can help you to create more efficient and maintainable Redux stores.

Here are some of the benefits of using Redux state slices:

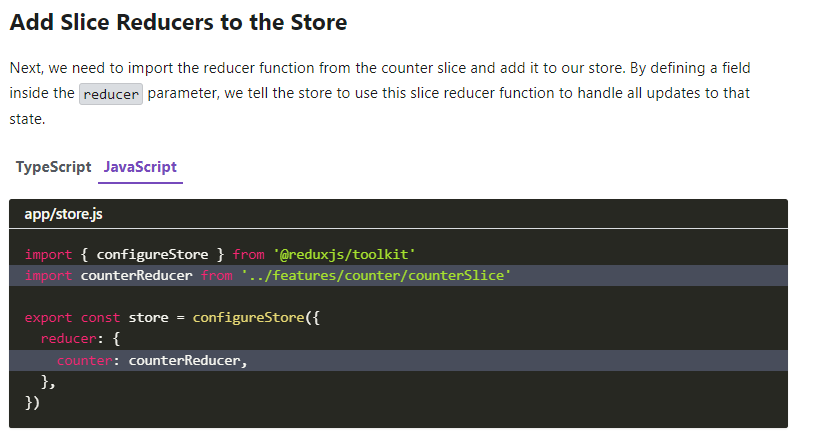
* They make your code easier to read and understand.
* They help you to manage complex state.
* They make it easier to test your Redux code.



Creating a slice requires a string name to identify the slice, an initial state value, and one or more reducer functions to define how the state can be updated. Once a slice is created, we can export the generated Redux action creators and the reducer function for the whole slice.

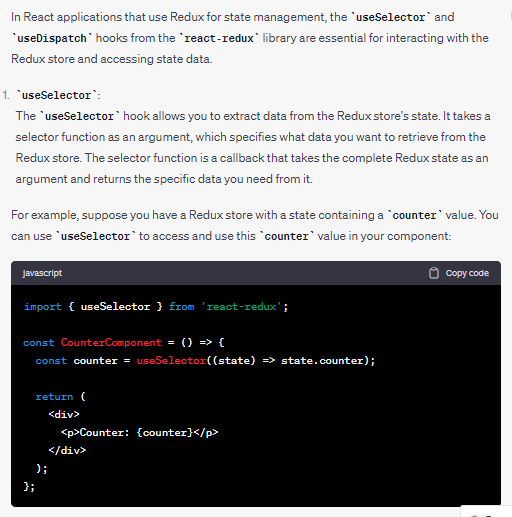






### **Use Redux State and Actions in React Components**

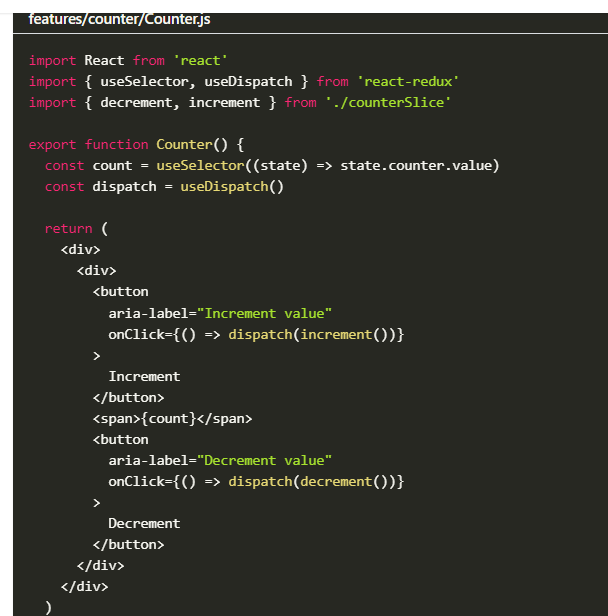
Now we can use the React-Redux hooks to let React components interact with the Redux store. We can read data from the store with useSelector, and dispatch actions using useDispatch.





In this example, when the "Increment" button is clicked, the **handleIncrement** function dispatches the **increment** action to the Redux store, triggering the state update.

Overall, using **useSelector** and **useDispatch** hooks simplifies the process of interacting with the Redux store in functional components. They provide an elegant and efficient way to access and update the state without the need for connect higher-order components and explicit subscription management.

Now, any time you click the "Increment" and "Decrement" buttons:

* The corresponding Redux action will be dispatched to the store
* The counter slice reducer will see the actions and update its state
* The <Counter> component will see the new state value from the store and re-render itself with the new data
* In the line **const dispatch = useDispatch()**, a React hook called **useDispatch** from the **react-redux** library is being used. This hook allows functional components in a React application to access the Redux store's **dispatch** function.
* In Redux, the **dispatch** function is used to dispatch actions to the store. Dispatching an action is the way to communicate with the store and trigger state changes. When an action is dispatched, the store passes the action to the reducers, which then update the state based on the action's type and payload.
* The **useDispatch** hook is used to obtain a reference to this **dispatch** function, so you can use it within your functional component to dispatch actions when needed.
* By using the **useDispatch** hook, you can avoid wrapping your component with the **connect** higher-order component from **react-redux**, as the hook gives you direct access to the **dispatch** function in a more concise and functional way.
* Top of Form