Redux-toolkit in Just Four (4) Step

① **Step ①:-** Install Redux Toolkit & React-Redux

```
npm install @reduxjs/toolkit react-redux
```

② **Create a Redux Store:-**

src/redux/store/index.js

```
import {configureStore} from '@reduxjs/toolkit'

expert const store = configureStore({
    reducer: {},
});
```

② **Step ②:-** Wrap main.jsx or index.jsx with store

main.jsx:-

```
import {store} from './redux/store'
import {Provider} from 'react-redux'

ReactDOM.createRat(document.getElementById("rat")).render(

<Provider store={store}>
    <App/>
</Provider>
```

Date: / / Page no:

③ Step ③ :- Create a Redux state slice

slice means function [Normal function]

⇒ redux/counterSlice/index.jsx :-

```
import { createSlice } from "@reduxjs/toolkit"

export const counterSlice = createSlice({
  name: "Counter",
  initialState: 10,
  reducers: {
    increaseby1: (state) => state+1,
    decreaseby1: (state) => state-1,
    increaseby10: (state) => state+10,
    decreaseby10: (state) => state-10,
  },
});

export const { increaseby1, decreaseby1, increaseby10,
               decreaseby10 } = counterSlice.actions;

export default counterSlice.reducer,
```

③ Step ① again :- Import the counterSlice in store

```
import {configureStore} from "@reduxjs/toolkit"
import counter from "../counterSlice";

export const store = configureStore ({
  reducer : {
    counter : counter,
      },
  });
```

④ Step ④ :- Use Redux state & Actions in React Component

```
import { useSelector, useDispatch} from 'react-redux'
import { increaseby1, decreaseby1, increaseby10, decreaseby10 }
                                    from "./redux/counterslice"

const App = () => {
  const counter = useSelector( (state) => state.counter);
  const dispatch = useDispatch();
return(
    <>
  <h1> {counter} </h1>

<button onClick={() => dispatch(increaseby1())}> Increase by 1 </button>
<button onclick={() => dispatch(decreaseby1())}> decreaseby1 </button>
<button onClick={() => dispatch(increaseby10())}> Increaseby10 </button>
<button onClick={() => dispatch(decreaseby10())}> decreaseby10 </button>
    </> )}
export default App
```

Redux - Toolkit

Date / / Page no:

(A) Terminology :-

(i) **Redux Store :-** It is a central container that holds the entire state of application

Easy Explanation :-

Assume ki app ek Shopkeeper (dukandaar) ho, aur stockroom hai, jha saare products ko organized aur list track rkhte ho. jisse pure shop ko easily manage kiya ja ske.

(ii) **Reducers :-** Reducers are functions in Redux, responsible for specifying how the application's state changes in response to actions. They take the current state & an action as input & return a new state.

Easy Explanation :-

Imagine shop me different departments hai, like clothing, electronics, or groceries. Each department has staff members who know how to handle updates their product category.

(iii) **Slice :-**

A slice refers to piece of the application's state & the corresponding reducer logic that handles updates to that piece.

It combines a reducer function & the initial state for a specific part of the application state.

Easy Explanation :- Picture of different section in shop for shoes, gadgets or snacks. Each section is like a slice, and there's someone in charge of making sure products are well-stocked & organized.

(iv) useSelector :- To Select with slice want to select

(v) useDispatch :- To select with function/reducers want to use/ dispatch