**Client Function’s Documentation**

**Step-1 :** We wrap our <App /> component inside <BrowserRouter> in the main.js file.  
This allows us to use global navigation and route-based rendering across the entire app

**Step-2 :** Redux Store, It helps manage data (like user info, app settings, etc.) that multiple components need to access or update, Instead of passing props between components, Redux stores all that data in one central place called the store

**Step-3 :** A slice is a small piece of your Redux store that handles a specific area of data. Example: authSlice for login/logout, cartSlice for shopping cart

**Step-4 :** Create our first basic slice called **auth** using createSlice from Redux Toolkit. This function requires a name for the slice, an initialState (typically includes: isAuthenticated, isLoading, and userInfo), and a set of reducers (functions to update the state), Finally, export the actions and the reducer from the slice

**Step-5 :** Creating out store in store.js, our idea is towe combine all the slice to make a global slice thus making a global reducer

**Step-6 :** After setting up Redux and the store, create a pages folder inside the src directory and add your authentication-related pages such as Login.jsx and Register.jsx Then, inside the components folder, create an auth folder. Within this auth folder, create a Layout.jsx file that will serve as the layout for your auth pages and include an <Outlet /> to render nested routes

**Step-7 :** In the Layout.jsx file, the Outlet component renders whatever child route (like AuthLayout) is matched. If no child route is matched, nothing will be rendered