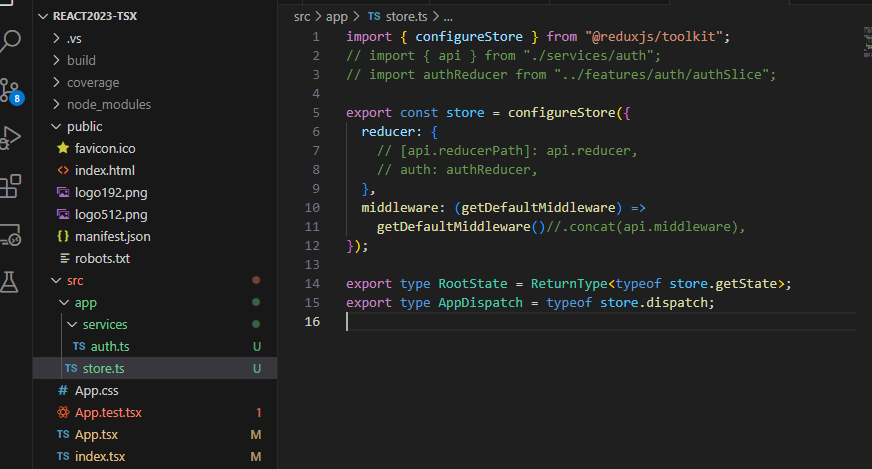
# Part 0 : Required libs

* React-router-dom >> npm i react-router-dom
* React-redux >> npm i react-redux
* Redux toolkit >> npm i @reduxjs/toolkit
* React msw (mock service worker) >> npm i msw (\*need more configuration such as npx msw init public/ --save)

# Part I : Create Store and Service

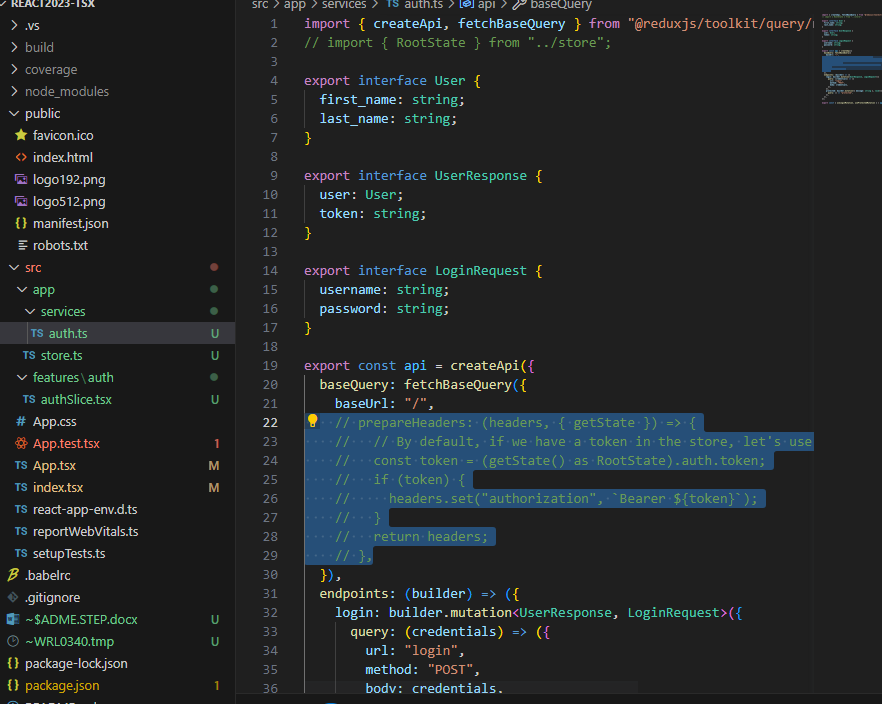
1. Initial boilerplate "default" store without configuration

: src\app\store.ts



1. Initial api services (without authorize module right now)

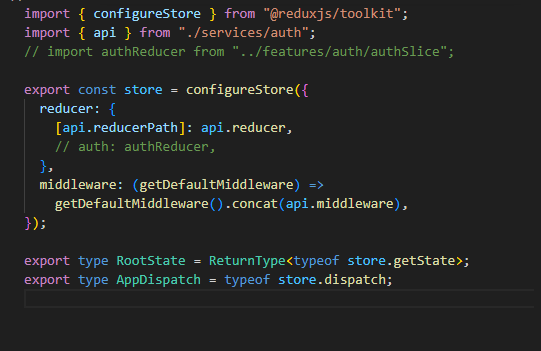
: src\app\services\auth.ts



1. config store

: src\app\store.ts

* config api reducerPath
* config middleware api



1. create authen Slice

: src\features\auth\authSlice.tsx

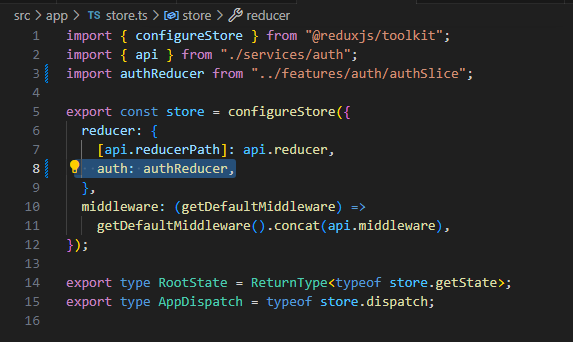
* config reducer -> setCredential, export action and reducer



1. config store

: src\app\store.ts

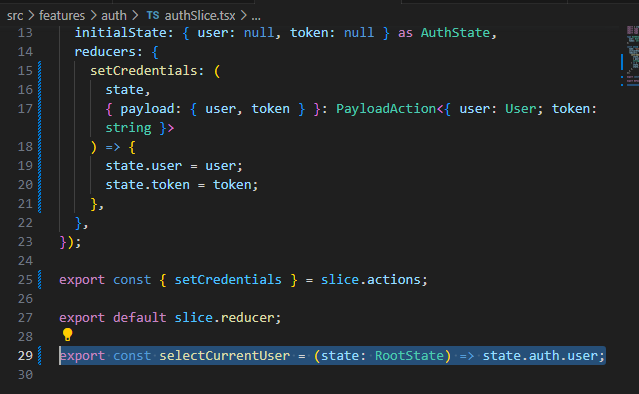
* import authSlice and add authReducer to store



1. config “auth Slice”

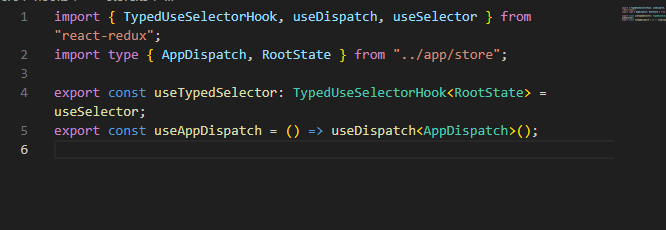
: src\features\auth\authSlice.tsx

* export authen user state “selectCurrentUser”



1. create hooks for store

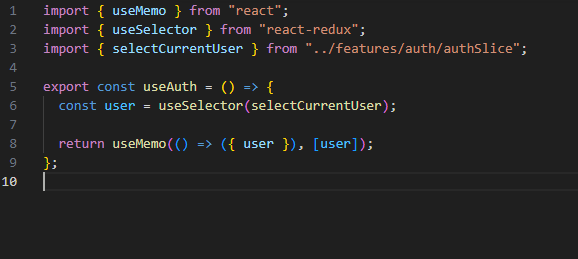
: src\hooks\store.ts



1. create custom hook “useAuth”

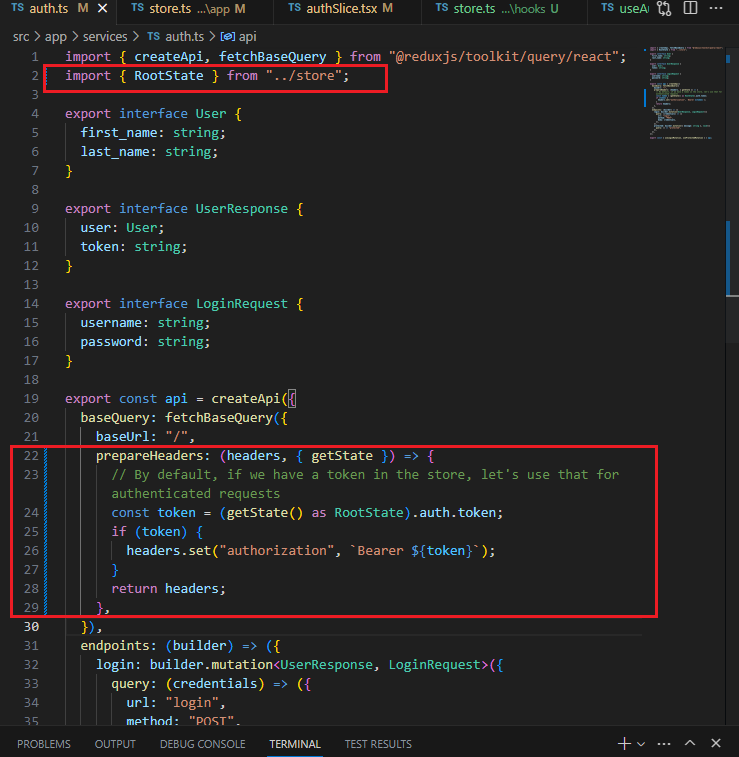
:src\hooks\useAuth.ts

\* return useMemo(() => ({user}), [user]) => re-render when **user** state change



1. go to auth services and prepareHeader for use **authentication with token**

: src\app\services\auth.ts



# Part II : Create Component and use the RTK to call authen api

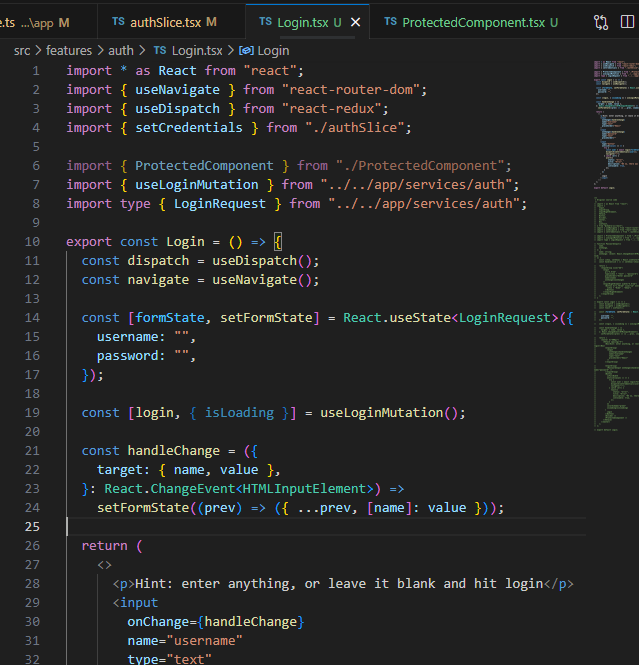
1. Create ProtectedComponent

: src\features\auth\ProtectedComponent.tsx



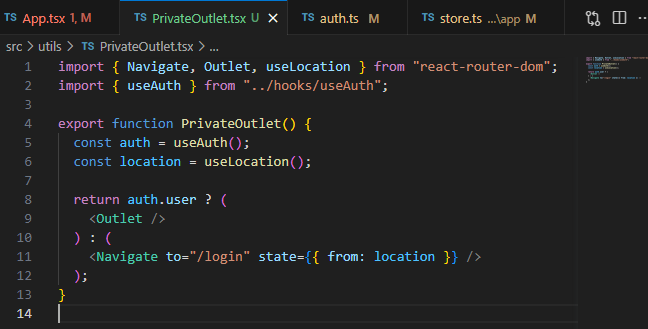
1. Create Login Component

: src\features\auth\Login.tsx



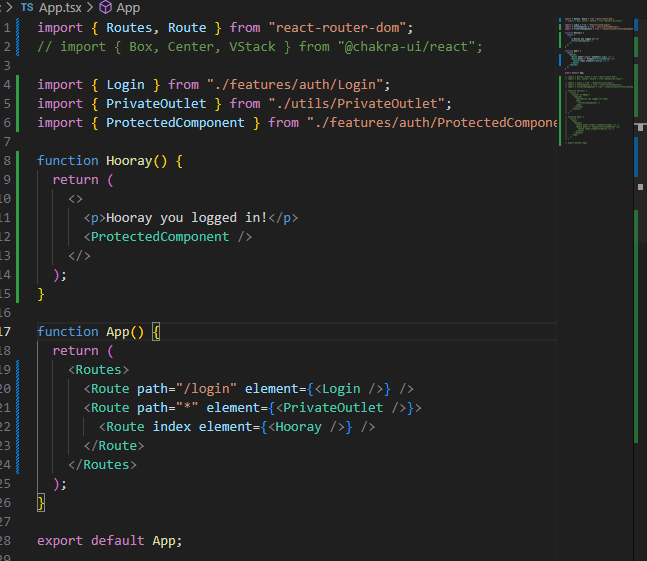
1. Create PrivateOutlet component (that like a landing page for checking authorize status)

: src\utils\PrivateOutlet.tsx



1. Modify App.tsx, integrate all components to application

: src\App.tsx



1. Modify index.tsx, binding store provider and browser route

: src\index.tsx

