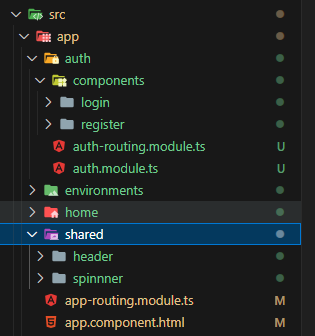
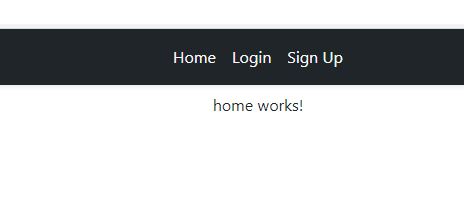
## **Basic set up**

* First project structure



* Output result



* Install packages

npm install @ngrx/store

npm install @ngrx/store-devtools --save

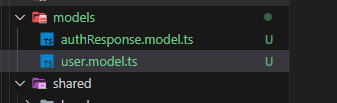
npm install @ngrx/effects

* You can clone all of this from github

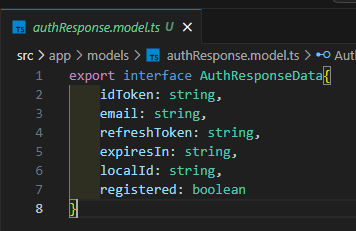
<https://github.com/lukeangular/NGRX_AUTH>

## **Create Models**

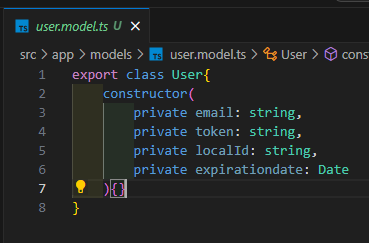
* Create models folder



* Auth response model is for mapping data

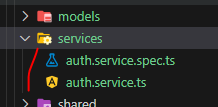


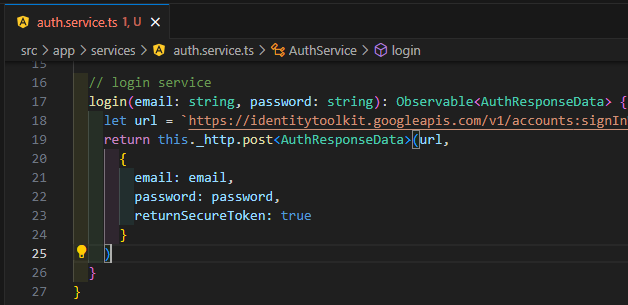
* User model if for formatting data



## **Create services**

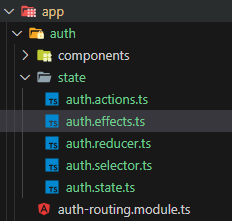
* Create models folder and generate auth files





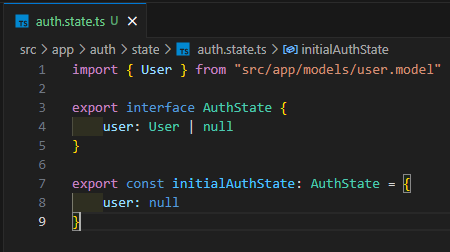
## **Auth state**

* Create auth state and files inside it



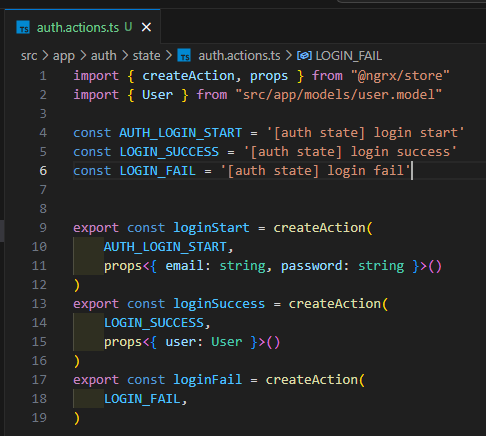
### State

* At first go to state file and create state



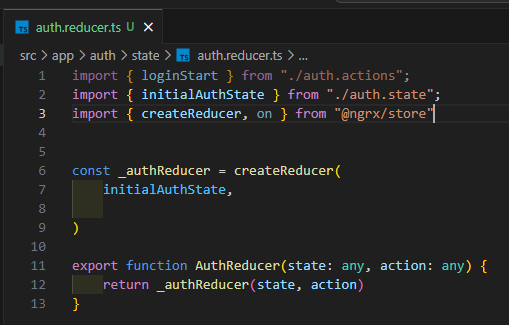
### Action

* Go to action and create actions



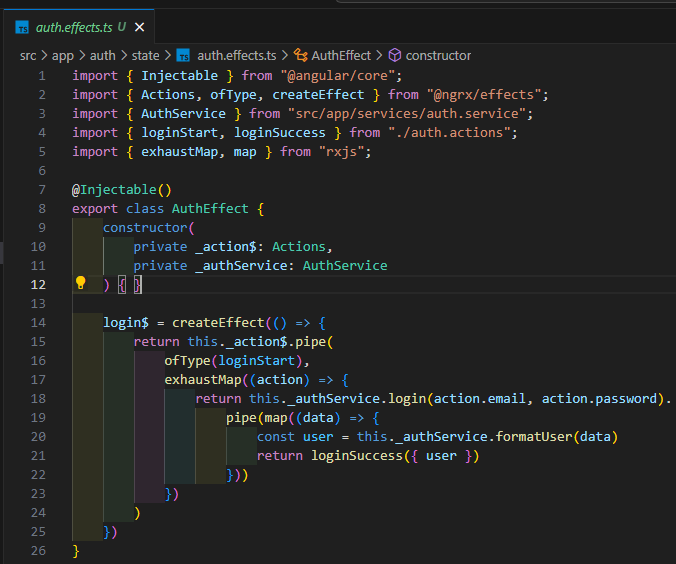
### Reducer

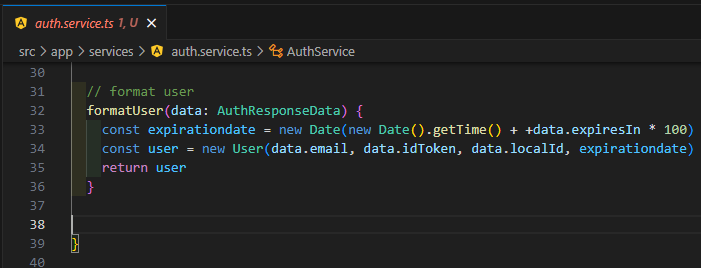
* Open reducer file and write code



### Effects

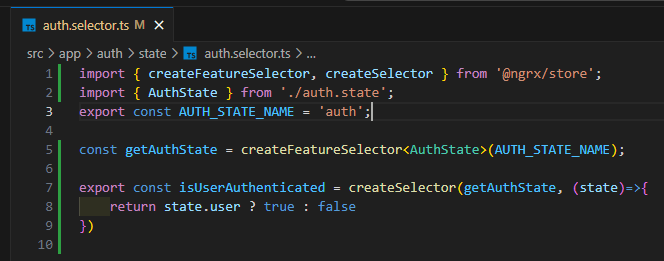
* Open effects file and





### Selector

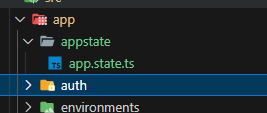
* Go to auth selector

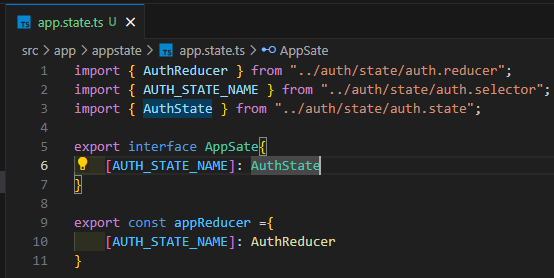


## **App state and register state**

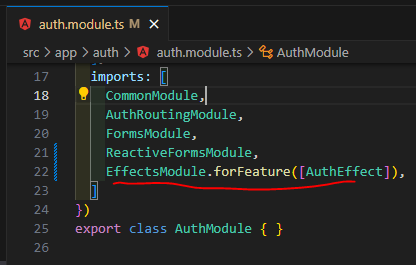
* Auth state should be available among all the components, so we need to make it global

Create app state folder for one global state

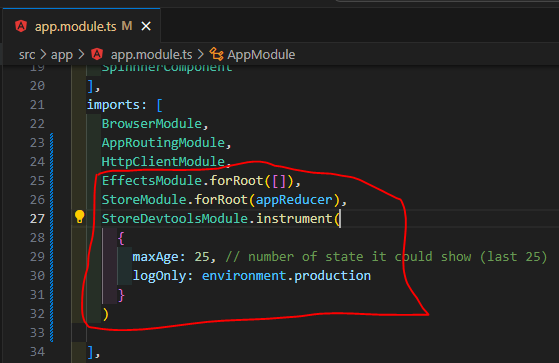




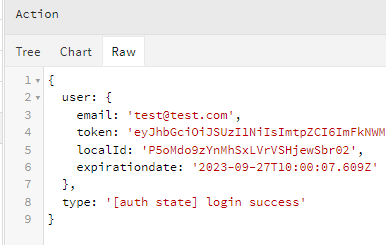
* Now add feature effect of auth, on auth module



* Go to app module (main module)

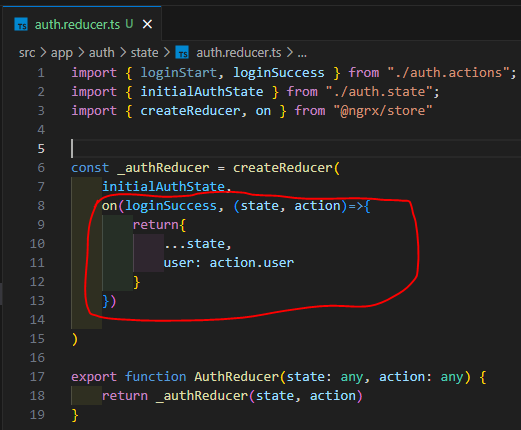


* If use provide valid login and username, we get this



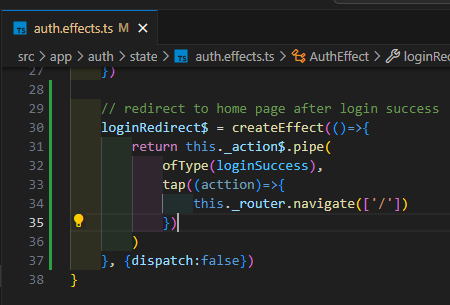
## **Save token in store**

* Go to reducer and update reducer

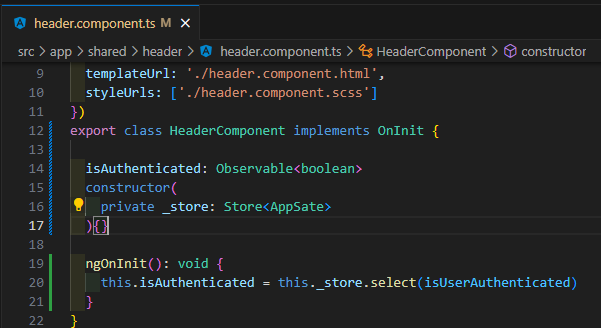


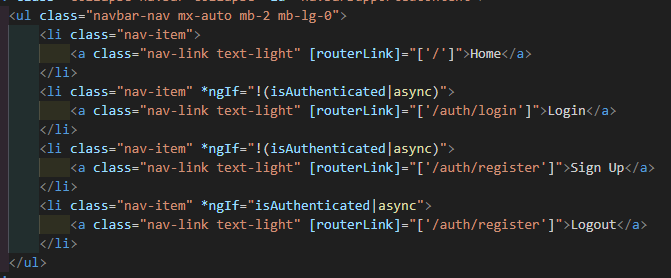
## **Redirect to home page when login**

* For this create redirect effect with dispatch false
* Go to effect file of auth



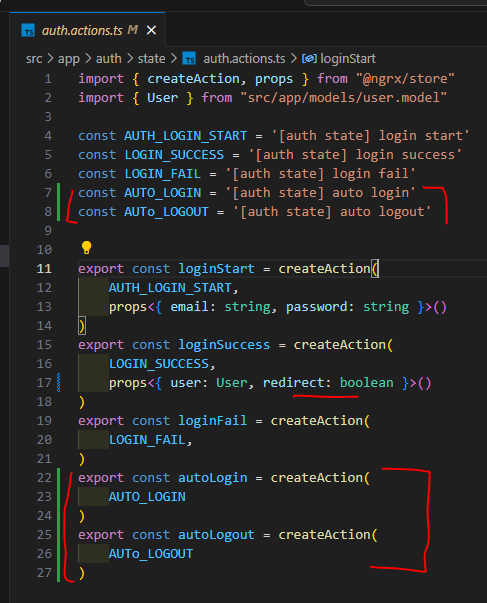
* Now go to header



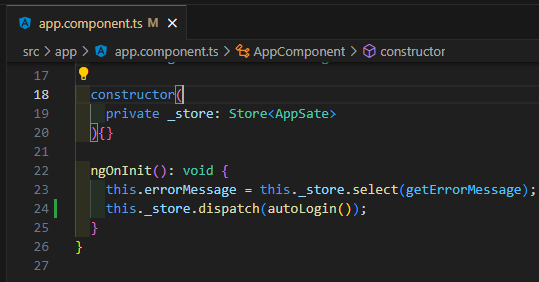


## **Save token in local storage and auto login**

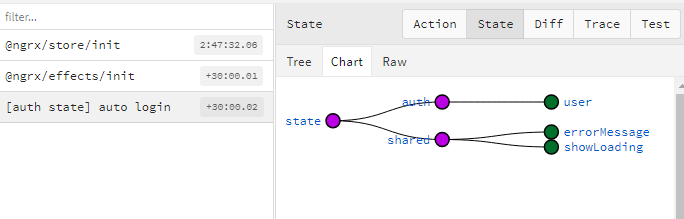
* At first update our action



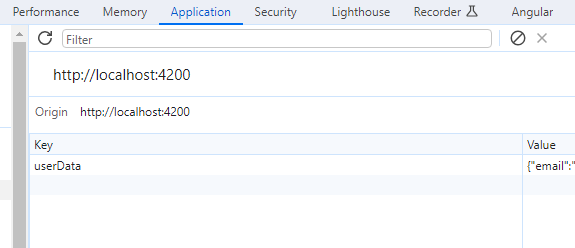
* Go to component and dispatch action on



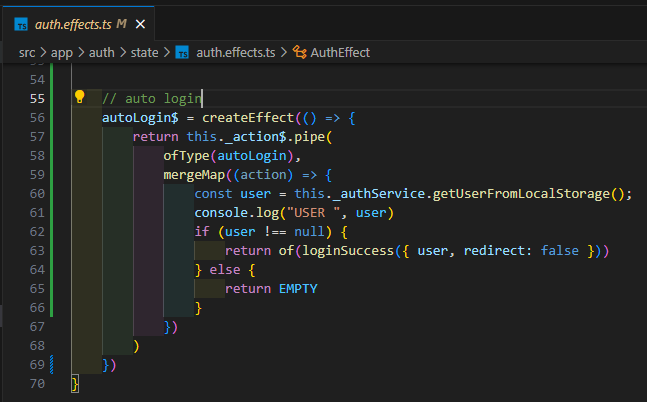
If we see in redux auto login action is firing

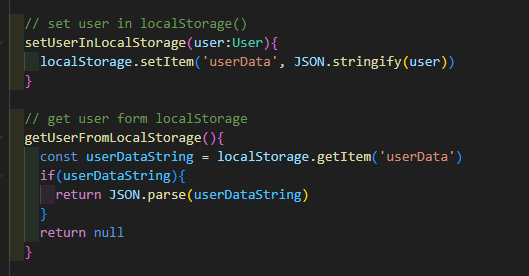


* If we login we are also able to store user data on local storage

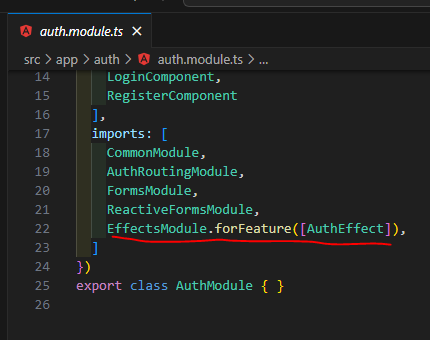


* No go to auth effect

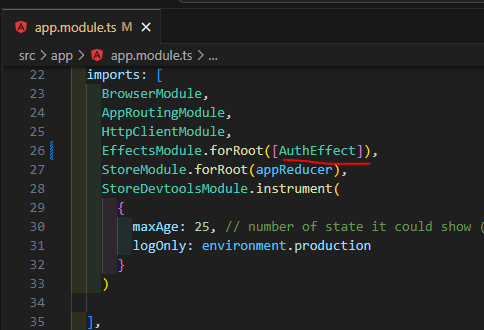




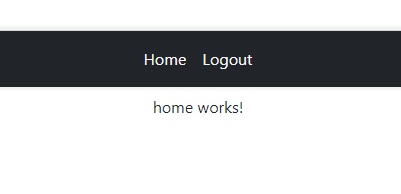
* Now go to auth module



Remove this and put into main module



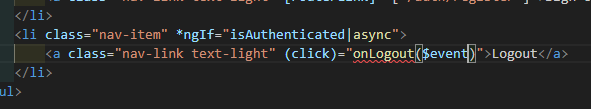
* Now reload page, it should do auto login now

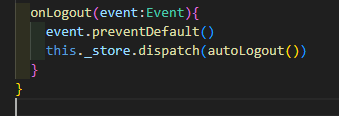




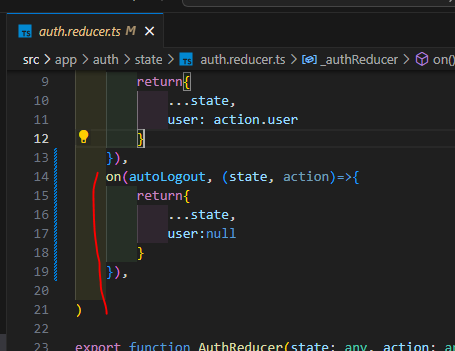
## **When click at logout**

* Go to header

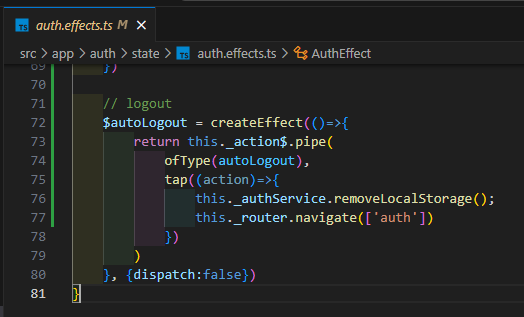


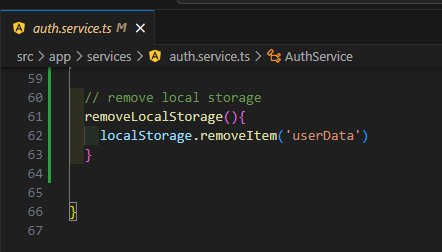


* On reducer (we already wrote this code, so no need to worry)



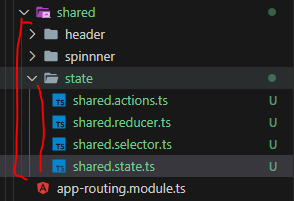
* On effect





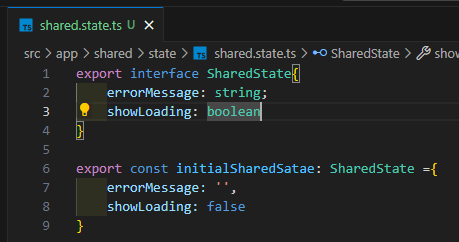
## **What if we get error? (Shared state)**

* To handle to error, we will create sharable state this will be also for the global spinner
* Create state for sharable state



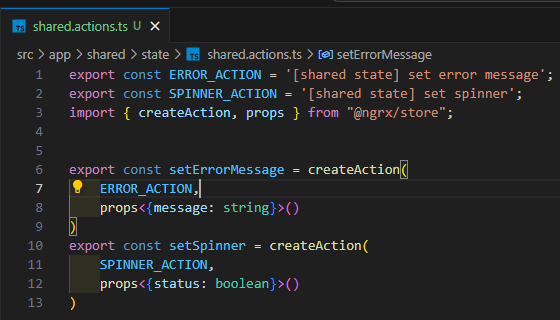
### State

* At first go to state file and create state



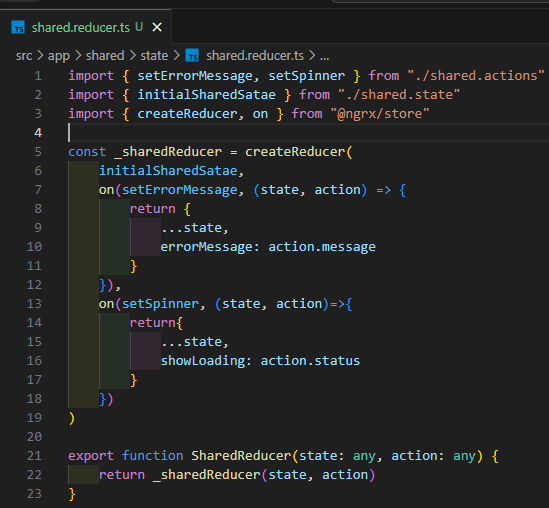
### Action

* Shared actions



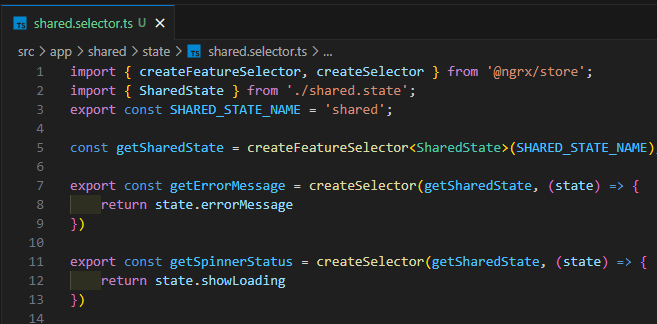
### Reducer

* Open reducer file and



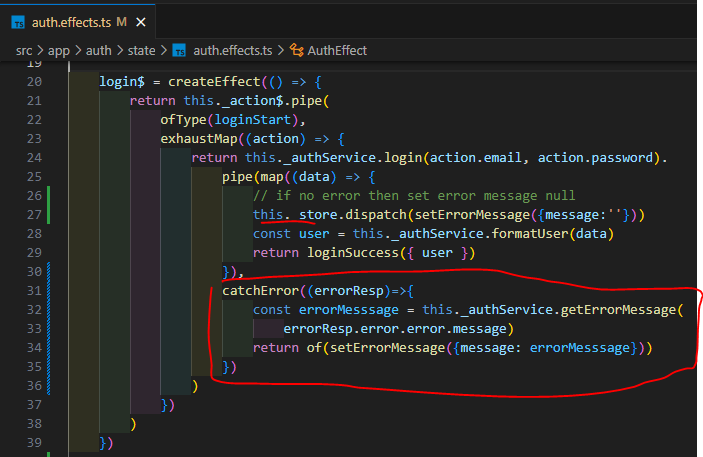
### Selector

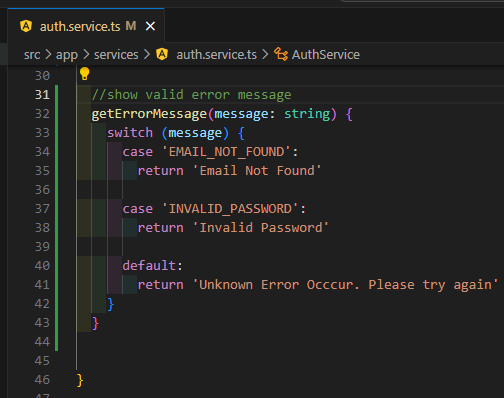
* Go to shared selector and



### Effect

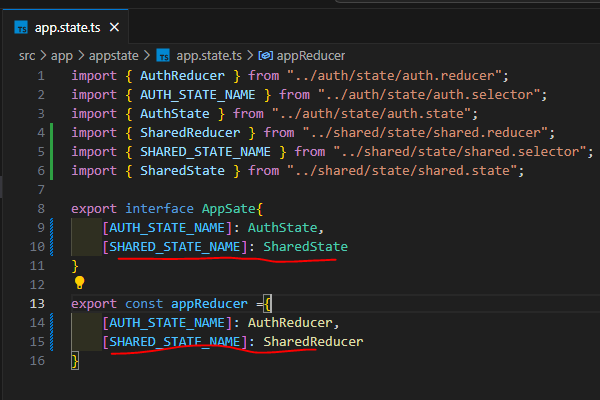
* Now go to effect and update effect to handle error



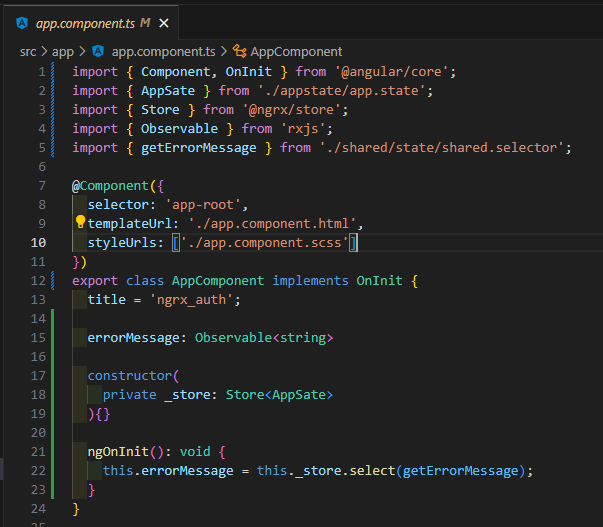


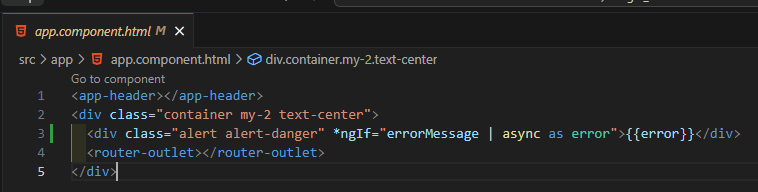
### Add reducer and state to the app sate

* Go app state and



* We want to show error globally so we will add this in app component HTML
* At first go to app component

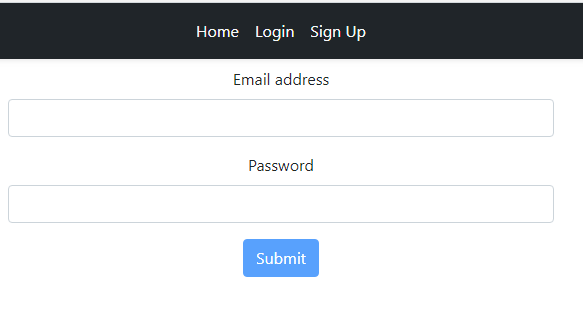




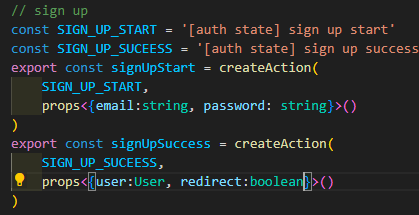
* We can show spinner with too using same logic

## **Register user**

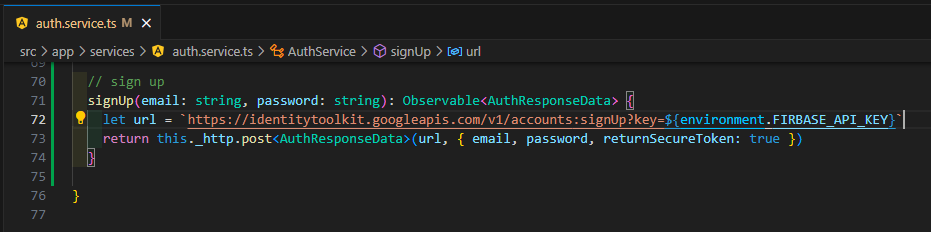
* At create auth form



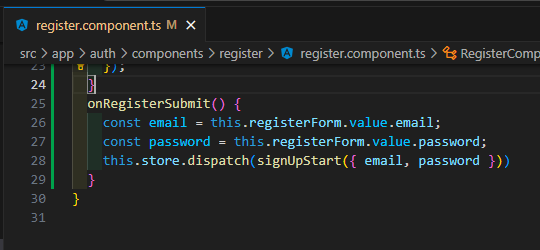
* Then write action code for sign up



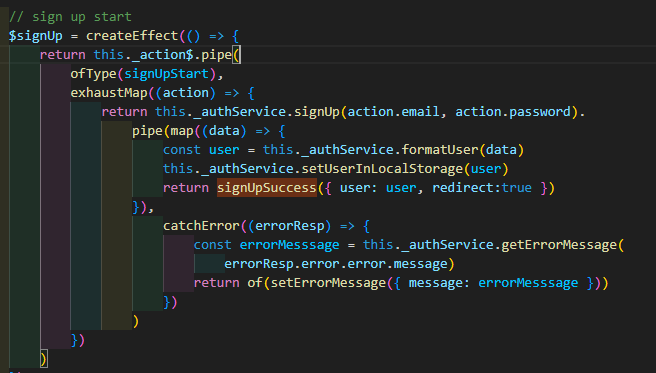
* Go to services



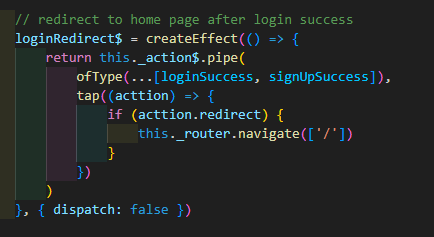
* Go to component



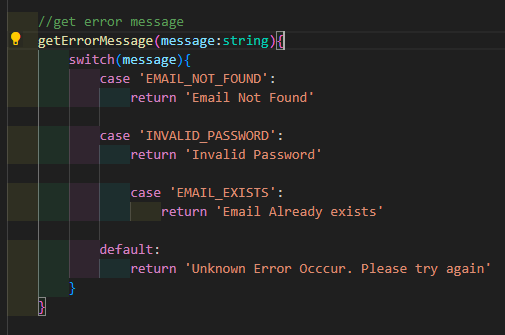
* Go to effect



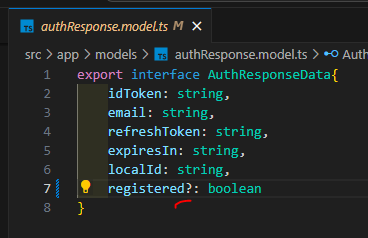
* Update login redirect



* Go to service and update set error message format function

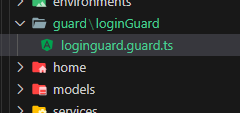


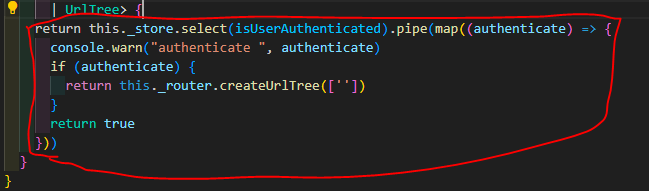
* We have also updated model like this



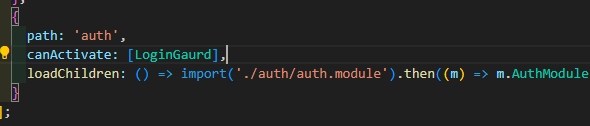
## **Login guard**

* At first generate login guard using angular CLI





* Go to main routing and add this



## **Post**

* At first generate login guard using angular CLI