Clase 7 :

RUTEO: cualquiera que ingresaba a nuestra página podía acceder

GUARDS:

Para restringir la navegación existen los GUARDS que nos permiten tomar decisiones en la navegación haber quién puede o no ingresar o no de diferentes maneras y distintas RUTAS

Los guard que existen son:

-canActivate lo que hace es antes de entrar ejecutar

-canActivaChild lo mismo pero cuando tenga un child

-canLoad lo hace antes para cargar

-canDeactivate primero carga y despues hacemos algo

-resolve

Cada uno se va a ejecutar cuando el ruteo y en una determinada ruta :

{path:"",component: HomeComponent ,canActivate:[ authService, AdminService]}

Al canActivate le voy a pasar un array al cual dentro puedo poner "servicios" ejemplo uno que decida si voy a poder ingresar o no , como es un array puedo pasarle varios servicios y se van a ejecutar uno a uno .

Estos servicios deben devolver siempre un BOOLEANO , cuando hagamos el return de FALSE hacemos un router.navigate para dirigirlo a otro lado

ng g guard auth/auth

si no funciona hacemos:

ng g service auth/auth

\*la única diferencia de si lo hacemos con servicio es que vamos a tener que implementar una interfa

Creamos un servicio llamado userService, este va a ser el encargado de mantener el "nombre de usuario" ejemplo vamos a tener un login y una vez que esta okey este service va a tener el objeto user con todos los datos del usuario , es decir datos públicos (nombre, email,type,image,etc)

Creamos un login también y en el guard lo que haremos es ir a este servicio y preguntarle si existe el objeto user

Tenemos una api en el que vamos a enviar un objeto cliente el cual debe tener user y pass por post

Para hacerlo por POST , desde el postman vamos a 192.168.2.85:3003/clientes

luego vamos al body-> raw -> json y ahí escribimos el objeto , ejemplo :

{ "cliente": { "user":"pepe","pass":"123"}}

Para GET hacemos solo :

192.168.2.85:3003/clientes

Luego para el LOGIN tenemos que hacer por POST:

192.168.2.85:3003/login

El cual me va a devolver si ese objeto está creado o no

Creamos metodo setToken() ya que cuando recargue mi pagina el servicio se va a reiniciar entonces debemos guardar el token en el locastorage , entonces cuando me logueo ese token lo guardo alli , arg['token ']

Cuando tenga que volver a traer el usuario voy a tener que traer todo el token y transformarlo para que valide si esta autenticado o no :

Para trabajar con token existen angular2/jwt el cual vamos a instalar y luego a utilizar para poder obtener los datos de ese token , esto debemos instalarlo con :

npm install @auth0/angular-jwt