Atelier Angular :
L'authentification et les droits d'accès

Objectifs :

1. Créer une page de login,
2. Création du Service Auth,
3. Contextualisation du menu,
4. Création d'un guard,
5. Utilisation de *LocalStorage*.

Créer une page de login

1. Créer le composant Web login :

   ng g c login --skip-tests -s

2. Ajouter le component login au fichier app.routes.ts

```
    {path:  'login', component: LoginComponent}
```

3. Créer dans le dossier model, la classe User :

```
export class User{
    username!:string ;
    password !: string ;
    roles!:string[];
}
```

4. Ajouter l'attribut user à la classe LoginComponent :

```
 user = new User();
```

5. Ajouter la méthode *onLoggedin*() à la classe LoginComponent :

```
 onLoggedin()
  {
    console.log(this.user);
  }
```

6. Editer le fichier login.component.html :

```
<div class="container mt-5">
    <div class="row justify-content-md-center">
        <div class="col-md-4">
           <form >
                   <div class="form-group">
                        <label >Nom Utilisateur :</label>
                        <input type="text" name="username"
                        class="form-control"
                        [(ngModel)]="user.username" >
                    </div>
                    <div class="form-group">
                        <label >Mot de passe :</label>
                        <input type="password"    name="password"
                         [(ngModel)]="user.password"   class="form-control">
                    </div>
                   <div  class="row justify-content-md-center">
                        <button type="button" (click)="onLoggedin()"
                        class="btn btn-lg btn-primary mt-3">Connexion</button>
                  </div>
            </form>
        </div>
    </div>
</div>
```

7. Tester votre travail : http://localhost:4200/login

Création du Service Auth

8. Accéder au dossier services, et y créer le service produit :

   cd .\src\app\services\

   ng g service auth

9. Modifier le fichier auth.service.ts comme suit :

```
users: User[] = [{"username":"admin","password":"123","roles":['ADMIN']},
                 {"username":"nadhem","password":"123","roles":['USER']} ];

public loggedUser!:string;
public isloggedIn: Boolean = false;
```

```
  public roles!:string[];


  constructor(private router: Router) { }



logout() {
  this.isloggedIn= false;
  this.loggedUser = undefined!;
  this.roles = undefined!;
  localStorage.removeItem('loggedUser');
  localStorage.setItem('isloggedIn',String(this.isloggedIn));
  this.router.navigate(['/login']);
}

  SignIn(user :User):Boolean{
    let validUser: Boolean = false;
    this.users.forEach((curUser) => {
      if(user.username== curUser.username && user.password==curUser.password) {
        validUser = true;
        this.loggedUser = curUser.username;
        this.isloggedIn = true;
        this.roles = curUser.roles;
        localStorage.setItem('loggedUser',this.loggedUser);
        localStorage.setItem('isloggedIn',String(this.isloggedIn));
      }
    });

     return validUser;
  }

  isAdmin():Boolean{
    if (!this.roles) //this.roles== undefiened
       return false;
    return  (this.roles.indexOf('ADMIN') >-1);
  }
```

10. Modifier la méthode *onLoggedin*() de la classe LoginComponent :

```
constructor(private authService : AuthService,
            private  router: Router) { }
```

```
onLoggedin(){

 console.log(this.user);
 let isValidUser: Boolean = this.authService.SignIn(this.user);

 if (isValidUser)
    this.router.navigate(['/']);
 else
    alert('Login ou mot de passe incorrecte!');
 }
```

11. Tester votre travail : http://localhost:4200/login
12. Ajouter ces lignes au fichier login.component.html

```
<div class="row justify-content-md-center">
       <div class="col-md-4">
 @if(erreur==1)   {
            <div  class="alert alert-danger" >
                <strong>login ou mot de passe erronés..</strong>
            </div>
          }
            <form >
```

13. Ajouter l'attribut erreur à la classe LoginComponent :

```
   erreur=0;
```

14. Modifier la méthode onLoggedin() comme suit :

```
else
     //alert('Login ou mot de passe incorrecte!');
     this.erreur = 1;
```

Contextualisation du menu

Afficher l'utilisateur connecté,

15. Modifier la classe AppComponent :

```
constructor (public authService: AuthService) {}

onLogout(){
   this.authService.logout();
 }
```

16. Remplacer dans le fichier app.component.html username par {{authService.loggedUser}} :

```
 <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button"
data-bs-toggle="dropdown" aria-expanded="false">
            [{{authService.loggedUser}}]
          </a>
```

Seulement les utilisateurs qui ont le rôle ADMIN peuvent ajouter des produits.

Donc on va cacher les menus "Ajouter" aux utilisateurs qui ne sont pas des admin.

17. Modifier le fichier app.component.html :

```
   @if(authService.isAdmin()) {
       <li><a class="dropdown-item"  routerLink="/add-produit">Ajouter</a></li>
             }
```

Seulement les utilisateurs qui ont le rôle ADMIN peuvent modifier et supprimer des produits

18. Modifier la classe ProduitsComponent :

```
  constructor(private produitService : ProduitService,
              public authService: AuthService) {  }
```

19. Modifier le fichier produit.component.html :

```
  @if(authService.isAdmin()) {
  <td><a class="btn btn-danger"  (click)="supprimerProduit(produit)">Supprimer</a></td>
  <td><a class="btn btn-success" [routerLink]="['/updateProduit',produit.idProduit]">Modifier</a></td>
                }
```

Cacher ou Montrer les commandes « Login » et « Logout » selon qu'on est connecté ou pas connecté

20. Modifier le fichier app.component.html :

```
@if(!authService.isloggedIn){
          <li><a class="dropdown-item"routerLink="/login">Login</a></li>
        }
        @else {
          <li><a class="dropdown-item"   (click) = "onLogout()">Logout</a></li>
        }
```

Création d'un guard

Connectez-vous en tant qu'un utilisateur non admin puis essayez d'accéder au formulaire d'ajout de produit :

http://localhost:4200/add-produit

Est-ce-que vous pouvez le faire ?

21. Créer un guard avec la commande :

    ng g guard produit

    Choisir l'option par défaut (CanActivate)

```
import { CanActivateFn, Router } from '@angular/router';
import { AuthService } from './auth.service';
import { inject } from '@angular/core';

export const produitGuard: CanActivateFn = (route, state) => {
    const authService = inject(AuthService);
    const router = inject(Router);

    if (authService.isAdmin())
      return true;
    else {
       router.navigate(['app-forbidden']);
      return false;
    }
};
```

22. Créer un web component *forbidden* pour afficher le message « Vous n'êtes pas autorisé… »

Si vous êtes dans le dossier service tapez cd.. puis

ng g c forbidden --skip-tests -s

23. Modifier le fichier app.routes.ts :

```
 {path:  'app-forbidden', component: ForbiddenComponent},
```

24. Modifier le fichier forbiden.component.html comme suit :

```
<div class="alert alert-danger" >
     <strong> Vous n'êtes pas autorisé…</strong>
 </div>
```

25. Modifier le fichier app.routes.ts :

```
{path: "add-produit", component : AddProduitComponent, canActivate:[produitGuard]},
```

26. Tester : Connectez-vous en tant qu'un utilisateur non admin puis essayez d'accéder au formulaire d'ajout de produit :

    http://localhost:4200/add-produit

Utilisation de *LocalStorage*

Au démarrage de l'application on teste Localstorage pour voir si l'utilisateur n'est pas déjà connecté sinon on le redirige vers login.

27. Modifier la classe AppComponent :

```
ngOnInit () {
  let isloggedin: string;
  let loggedUser:string;
  isloggedin = localStorage.getItem('isloggedIn') !;
  loggedUser = localStorage.getItem('loggedUser') !;
  if (isloggedin!="true" || !loggedUser)
      this.router.navigate(['/login']);
  else
   this.authService.setLoggedUserFromLocalStorage(loggedUser);
}
```

28. Modifier le fichier auth.service.ts comme suit :

```
setLoggedUserFromLocalStorage(login : string) {
    this.loggedUser = login;
    this.isloggedIn = true;
    this.getUserRoles(login);
  }

  getUserRoles(username :string){
    this.users.forEach((curUser) => {
```

```
    if( curUser.username == username ) {
        this.roles = curUser.roles;
    }
  });
}
```