Formation Angular

TP N°12: Les Guards

SOMMAIRE

1. Pré-requis :
   - Réaliser les TPs 1-11 en premier.

2. Objectifs
   - ✓ Sécuriser une route avec le guard ***canActivate***.
   - ✓ Sécuriser une route avec ***canDeactivate***.
   - ✓ Sécuriser une route avec ***canActivateChild***.
   - ✓ Exécuter un ***Resolver*** avant le chargement d'un composant.

3. Développement de la couche front

a. Importer le projet TP12

- Créer un nouveau dossier (TP12) et copier dans ce dernier le contenu du dossier TP11.
- Lancer les commandes suivantes :
  - **cd tp12**
  - **npm install**

b. Création du guard canActivate

A- Premier exemple : Développement d'un Guard pour vérifier l'authentification des utilisateurs

Vous remarquez que même si l'utilisateur n'est pas connecté, l'application permet l'accès au composant Emp-list pour consulter les employés. Pour permettre l'accès uniquement aux utilisateurs connectés, nous allons créer le guard ***canActivate***.

- Lancer la commande suivante : **ng g guard guards/chech-auth**. CLI vous invite à choisir le type de guard que vous voulez créer :

```
? Which interfaces would you like to implement? (Press <space> to select, <a> to toggle all, <i> to invert sel
ection, and <enter> to proceed)
>(*) CanActivate
 ( ) CanActivateChild
 ( ) CanDeactivate
 ( ) CanLoad
```

- Choisir CanActivate et cliquer sur Entrer. La classe ChechAuth est bien créée.
- Modifier la classe ChechAuthGuard comme suit :

```
import { Injectable } from '@angular/core';
import { MatSnackBar } from '@angular/material/snack-bar';
import { ActivatedRouteSnapshot, CanActivate, Router, RouterStateSnapshot, UrlTree }
from '@angular/router';
import { Observable } from 'rxjs';
import { AuthService } from '../services/auth.service';

@Injectable({
  providedIn: 'root'
})
export class ChechAuthGuard implements CanActivate {
```

```
  canActivate(
    route: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean |
UrlTree> | boolean | UrlTree {
      if (!this.authService.isUserLoggedIn()) {
        this.snackBar.open('You are not allowed to view this page. You are redirected
to login Page','OK');
        this.router.navigate([{ outlets: { primary: 'login', contenu: null } }]);
        return false;
    }
    return true;
  }

  constructor(private router:Router,private authService:AuthService,private
snackBar:MatSnackBar) {

  }
}
```

- Modifier le service AuthService comme suit :

```
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { TokenStorageService } from './token-storage.service';
const AUTH_API = 'http://localhost:9090/auth/signin';

const httpOptions = {
  headers: new HttpHeaders({ 'Content-Type': 'application/json' })
};

@Injectable({
  providedIn: 'root'
})
export class AuthService {

  constructor(private http: HttpClient,private tokenStorage : TokenStorageService) {
}

  login(username: string, password: string): Observable<any> {
    return this.http.post(AUTH_API , { username, password }, httpOptions);
  }

  isUserLoggedIn() :boolean {
    console.log("========!!this.tokenStorage.getToken()=======",!!this.tokenStorage.g
etToken());
    return !!this.tokenStorage.getToken();
```

```
  }
}
```

- Modifier la classe AppModule comme suit :

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { HttpClientModule } from '@angular/common/http';
import { AuthComponent } from './auth/auth.component';
import { NavbarComponent } from './navbar/navbar.component';
import { authInterceptorProviders } from './interceptors/auth.interceptor';
import { WelcomeComponent } from './welcome/welcome.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { EmpDetailComponent } from './emp/emp-detail/emp-detail.component';
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import {MatPaginatorModule} from '@angular/material/paginator';
import {MatProgressSpinnerModule} from '@angular/material/progress-spinner';
import {MatCardModule} from '@angular/material/card';
import {MatTableModule} from '@angular/material/table';
import {MatButtonModule} from '@angular/material/button';
import {MatSnackBarModule} from '@angular/material/snack-bar';

const materialComponents=[
  MatPaginatorModule,
  MatProgressSpinnerModule,
  MatCardModule,
  MatTableModule,
  MatButtonModule,
  MatSnackBarModule
]


@NgModule({
  declarations: [
    AppComponent,
    AuthComponent,
    NavbarComponent,
    WelcomeComponent,
    EmpListComponent,
    EmpDetailComponent,
    EmpCreateComponent
  ],
```

```
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    HttpClientModule,
    ReactiveFormsModule,
    BrowserAnimationsModule,
    materialComponents
  ],
  providers: [authInterceptorProviders],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

- Modifier la classe AppRoutingModule comme suit :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AuthComponent } from './auth/auth.component';
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { ChechAuthGuard } from './guards/chech-auth.guard';
import { NavbarComponent } from './navbar/navbar.component';
import { WelcomeComponent } from './welcome/welcome.component';


const routes: Routes = [
  { path: '', redirectTo: 'login', pathMatch: 'full' },
  { path: 'login', component: AuthComponent },
  { path: 'navbar', component: NavbarComponent },
  { path: 'employees', component: EmpListComponent,  outlet:
'contenu',canActivate:[ChechAuthGuard] },
  { path: 'create', component: EmpCreateComponent, outlet: 'contenu'},
  { path: 'welcome', component: WelcomeComponent,  outlet: 'contenu' },
  { path: 'logout', component: AuthComponent}

];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

B- <u>Deuxième exemple : Développement d'un Guard pour vérifier si l'utilisateur est un Admin</u>

- Lancer la commande suivante : **ng g guard guards/check-admin**
- Choisir CanActivate et cliquer sur Entrer.

- Modifier la classe CheckAdminGuard comme suit :

```
import { Injectable } from '@angular/core';
import { MatSnackBar } from '@angular/material/snack-bar';
import { ActivatedRouteSnapshot, CanActivate, Router, RouterStateSnapshot, UrlTree }
from '@angular/router';
import { Observable } from 'rxjs';
import { TokenStorageService } from '../services/token-storage.service';

@Injectable({
  providedIn: 'root'
})
export class CheckAdminGuard implements CanActivate {
  canActivate(
    route: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean |
UrlTree> | boolean | UrlTree {
    if (!this.tokenService.hasRole('ADMIN')) {
      this.snackBar.open('You are not allowed to get this function','OK');
      return false;
    } else
      return true;
  }

  constructor(private router:Router,private tokenService:TokenStorageService,private
snackBar:MatSnackBar) {

  }
 }
```

- Modifier la classe AppRoutingModule comme suit :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AuthComponent } from './auth/auth.component';
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { ChechAuthGuard } from './guards/chech-auth.guard';
import { CheckAdminGuard } from './guards/check-admin.guard';
import { NavbarComponent } from './navbar/navbar.component';
import { WelcomeComponent } from './welcome/welcome.component';


const routes: Routes = [
  { path: '', redirectTo: 'login', pathMatch: 'full' },
  { path: 'login', component: AuthComponent },
  { path: 'navbar', component: NavbarComponent },
  { path: 'employees', component: EmpListComponent,  outlet:
```

```
'contenu',canActivate:[ChechAuthGuard] },
  { path: 'create', component: EmpCreateComponent, outlet:
'contenu',canActivate:[CheckAdminGuard]},
  { path: 'welcome', component: WelcomeComponent,  outlet: 'contenu' },
  { path: 'logout', component: AuthComponent}

];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

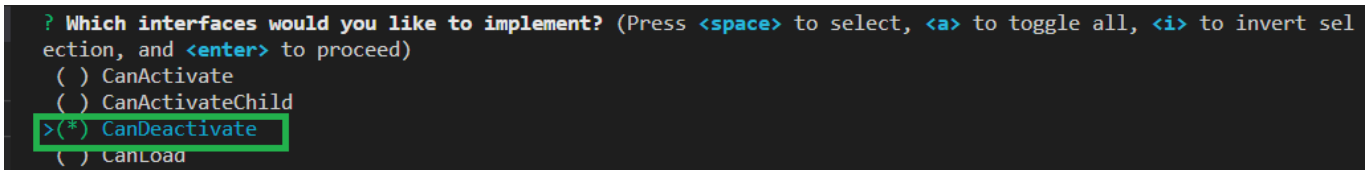
- Modifier la page emp-list.component.html comme suit :

```
<div align="center" class="mat-display-1">
    <mat-card>
        <mat-card-header>
            <mat-card-title>
                Liste des employées :
            </mat-card-title>

        </mat-card-header>

    </mat-card>
    <div align="right"> <button [routerLink]="['/',{ outlets: { primary:
['navbar'],contenu: ['create'] } }]" mat-raised-button color="primary">Nouvel
employé</button></div>
    <mat-paginator [pageSizeOptions]="[5, 10, 15]" showFirstLastButtons
(page)="pageChanged($event)">
    </mat-paginator>
    <div *ngIf="loading;else table">
        <mat-spinner class="center"></mat-spinner>
    </div>

    <ng-template #table>
        <table mat-table [dataSource]="dataSource">
            <ng-container matColumnDef="id">
                <th mat-header-cell *matHeaderCellDef> Id </th>
                <td mat-cell *matCellDef="let element"> {{element.id}} </td>

            </ng-container>

            <ng-container matColumnDef="name">
                <th mat-header-cell *matHeaderCellDef> Nom </th>
```

```
                <td mat-cell *matCellDef="let element"> {{element.name}} </td>
            </ng-container>

            <ng-container matColumnDef="salary">
                <th mat-header-cell *matHeaderCellDef> Salaire </th>
                <td mat-cell *matCellDef="let element"> {{element.salary}} </td>
            </ng-container>

            <ng-container matColumnDef="fonction">
                <th mat-header-cell *matHeaderCellDef> Fonction </th>
                <td mat-cell *matCellDef="let element"> {{element.fonction}} </td>
            </ng-container>


            <tr mat-header-row *matHeaderRowDef="['id', 'name',
'salary','fonction']"></tr>
            <tr mat-row *matRowDef="let row; columns: ['id', 'name',
'salary','fonction'];"></tr>
        </table>
    </ng-template>
</div>
```

c. Création du guard canDeactivate

- Lancer la commande suivante : ng g guard guards/check-save.
- Choisir le Guard canDeactivate comme montre l'écran suivant :



- Modifier la classe CheckSaveGuard comme suit :

```
import { Injectable } from '@angular/core';
import { ActivatedRouteSnapshot, CanDeactivate, RouterStateSnapshot, UrlTree } from
'@angular/router';
import { Observable } from 'rxjs';
import { EmpCreateComponent } from '../emp/emp-create/emp-create.component';

@Injectable({
  providedIn: 'root'
})
export class CheckSaveGuard implements CanDeactivate<unknown> {
  canDeactivate(
    component: unknown,
    currentRoute: ActivatedRouteSnapshot,
    currentState: RouterStateSnapshot,
    nextState?: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean |
```

```
UrlTree> | boolean | UrlTree {
    if (!(component as EmpCreateComponent).isSaved)
      return window.confirm('Attention, vous risquez de perdre les modifications!.
Voulez vous continuer ?');
    else
      return true;
  }

}
```

- Modifier la classe AppRoutingModule comme suit :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AuthComponent } from './auth/auth.component';
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { ChechAuthGuard } from './guards/chech-auth.guard';
import { CheckAdminGuard } from './guards/check-admin.guard';
import { CheckSaveGuard } from './guards/check-save.guard';
import { NavbarComponent } from './navbar/navbar.component';
import { WelcomeComponent } from './welcome/welcome.component';


const routes: Routes = [
  { path: '', redirectTo: 'login', pathMatch: 'full' },
  { path: 'login', component: AuthComponent },
  { path: 'navbar', component: NavbarComponent },
  { path: 'employees', component: EmpListComponent,  outlet:
'contenu',canActivate:[ChechAuthGuard] },
  { path: 'create', component: EmpCreateComponent, outlet:
'contenu',canActivate:[CheckAdminGuard],canDeactivate:[CheckSaveGuard] },
  { path: 'welcome', component: WelcomeComponent,  outlet: 'contenu' },
  { path: 'logout', component: AuthComponent}

];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

- Modifier la page emp-create.component.html comme suit :

```
<div *ngIf="submitted">
```

```
    <h3>Employé crée avec succés!</h3>
</div>
<div style="width: 500px; margin: auto;">
    <div class="submit-form">
        <div *ngIf="!submitted">

            <div class="form-group">
                <label for="name">Name</label>
                <input type="text" class="form-control" id="name" required
[(ngModel)]="employee.name" name="name"
                    ngModel />
            </div>

            <div class="form-group">
                <label for="salary">Salaire</label>
                <input type="text" class="form-control" id="salary" required
[(ngModel)]="employee.salary" name="salary"
                    ngModel />
            </div>

            <div class="form-group">
                <label for="fonction">Fonction</label>
                <input type="text" class="form-control" id="fonction" required
[(ngModel)]="employee.fonction"
                    name="fonction" />
            </div>
            <button (click)="createEmployee()" class="btn btn-
success">Create</button>
            <button (click)="save()" class="btn btn-success">Sauvegarder</button>
        </div>
    </div>
</div>
```

- Modifier la classe emp-create.component.ts comme suit :

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { Emp } from 'src/app/model/emp';
import { EmpService } from 'src/app/services/emp.service';

@Component({
  selector: 'app-emp-create',
  templateUrl: './emp-create.component.html',
  styleUrls: ['./emp-create.component.css']
})
export class EmpCreateComponent implements OnInit {
```

```
  employee = new Emp(0, '', 0, '');
  submitted = false;
  message:string='';
  isSaved:boolean=false;

  constructor(private empService: EmpService, private route: ActivatedRoute, private
router: Router) { }

  ngOnInit(): void { }

  createEmployee(): void {
    this.empService.create(this.employee)
      .subscribe(
        response => {
          this.submitted = true;
          this.router.navigate([{outlets: {primary: 'navbar' ,contenu: 'employees'}}]);
        },
        error => {
          this.message=error.message;
          console.log(error);
        });
  }

  save():void {
    this.isSaved=true;
  }
}
```

d. Création d'un Resolver

- Lancer la commande suivante : **ng g class model/product**
- Modifier la classe Product comme suit :

```
export class Product {
    constructor(public productID: number, public name: string, public price:
number) {
    }
}
```

- Lancer la commande suivante : **ng g s services/product**
- Modifier la classe ProudctService comme suit :

```
import { Injectable } from '@angular/core';
import { Product } from '../model/product';

import { of, Observable, throwError} from 'rxjs';
import { delay, map } from 'rxjs/operators';

@Injectable({
```

```
  providedIn: 'root'
})
export class ProductService {

  products: Product[];

    public constructor() {
        this.products=[
            new Product(1,'Memory Card',500),
            new Product(2,'Pen Drive',750),
            new Product(3,'Power Bank',100),
            new Product(4,'Computer',100),
            new Product(5,'Laptop',100),
            new Product(6,'Printer',100),
        ]
    }

    //Return Products List with a delay
    public getProducts(): Observable<Product[]> {
        return of(this.products).pipe(delay(1500)) ;
    }


    public getProduct(id:any): Observable<Product|undefined> {
        var product= this.products.find(i => i.productID==id)
        return of(product).pipe(delay(1500)) ;
    }
}
```

- Lancer la commande suivante : **ng g resolver services/product-list-resolver**
- Modifier la classe ProductListResolverService comme suit :

```
import { Injectable } from '@angular/core';
import { ActivatedRouteSnapshot, Resolve, RouterStateSnapshot } from '@angular/router';
import { Observable } from 'rxjs';
import { Product } from '../model/product';
import { ProductService } from './product.service';

@Injectable({
  providedIn: 'root'
})
export class ProductListResolverService implements Resolve<Product[]>{

  constructor(private productService:ProductService ) {
  }

  resolve(route: ActivatedRouteSnapshot,
          state: RouterStateSnapshot): Observable<Product[]> {
```

```
      console.log("ProductListResover is called");
      return this.productService.getProducts();
  }

}
```

- Créer les composants : proudct1 et product2 et modifier les deux classes comme suit :

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';
import { Product } from '../model/product';
import { ProductService } from '../services/product.service';

@Component({
  selector: 'app-product1',
  templateUrl: './product1.component.html',
  styleUrls: ['./product1.component.css']
})
export class Product1Component implements OnInit {

  public products?:Product[];

  constructor(private route: ActivatedRoute,private productService:ProductService){
  }

  ngOnInit() {
      this.productService.getProducts().subscribe(data => {
        this.products=data;
     });
  }

}
```

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';
import { Product } from '../model/product';
import { ProductService } from '../services/product.service';

@Component({
  selector: 'app-product2',
  templateUrl: './product2.component.html',
  styleUrls: ['./product2.component.css']
})
export class Product2Component implements OnInit {
```

```
  public products?:Product[];

  constructor(private route: ActivatedRoute,private productService:ProductService){
  }

  ngOnInit() {
     this.products=this.route.snapshot.data['products'];
  }

}
```

- Modifier la page product1.component.html comme suit :

```
<h1> Without Resolve</h1>
<div class='table-responsive'>
    <table class='table'>
        <thead>
            <tr>
                <th>Name</th>
                <th>Price</th>
            </tr>
        </thead>
        <tbody>
            <tr *ngFor="let product of products;">
                <td><a [routerLink]="['/',{ outlets: { primary: ['navbar'],contenu:
['editproduct',product.productID] } }]">{{product.name}} </a> </td>
                <td>{{product.price}}</td>
            </tr>
        </tbody>
    </table>
</div>
```

- Modifier la page product2.component.html comme suit :

```
<h1> Without Resolve</h1>
<div class='table-responsive'>
    <table class='table'>
        <thead>
            <tr>
                <th>Name</th>
                <th>Price</th>
            </tr>
        </thead>
        <tbody>
            <tr *ngFor="let product of products;">
```

```
                <td><a [routerLink]="['/',{ outlets: { primary: ['navbar'],contenu:
['editproduct',product.productID] } }]">{{product.name}} </a> </td>
                <td>{{product.price}}</td>
            </tr>
        </tbody>
    </table>
</div>
```

- Modifier le fichier app-routing.module.ts comme suit :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AuthComponent } from './auth/auth.component';
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { ChechAuthGuard } from './guards/chech-auth.guard';
import { CheckAdminGuard } from './guards/check-admin.guard';
import { CheckSaveGuard } from './guards/check-save.guard';
import { NavbarComponent } from './navbar/navbar.component';
import { Product1Component } from './product1/product1.component';
import { Product2Component } from './product2/product2.component';
import { ProductListResolverService } from './services/product-list-
resolver.service';
import { WelcomeComponent } from './welcome/welcome.component';


const routes: Routes = [
  { path: '', redirectTo: 'login', pathMatch: 'full' },
  { path: 'login', component: AuthComponent },
  { path: 'navbar', component: NavbarComponent },
  {
    path: 'employees',
    component: EmpListComponent,
    outlet: 'contenu',
    canActivate: [ChechAuthGuard],
  },
  { path: 'create', component: EmpCreateComponent, outlet: 'contenu', canActivate:
[CheckAdminGuard], canDeactivate: [CheckSaveGuard] },
  { path: 'welcome', component: WelcomeComponent, outlet: 'contenu' },
  { path: 'logout', component: AuthComponent },
  { path: 'product1', component: Product1Component, outlet: 'contenu' },
  {
    path: 'product2',
    component: Product2Component,
    resolve: { products: ProductListResolverService }, outlet: 'contenu'
  }
```

```
]
  ;

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

e. **Création du guard canActivatechild**

- Lancer la commande suivante **ng g guard guards/admin** ensuite choisir canActivateChild comme le montre l'écran suivant :

```
PS C:\workspace_angular_forGL\demotp12> ng g  guard guards/admin
? Which interfaces would you like to implement? (Press <space> to
o toggle all, <i> to invert selection, and <enter> to proceed)
 ( ) CanActivate
>(*) CanActivateChild
 ( ) CanDeactivate
 ( ) CanLoad
```

- Modifier la classe AdminGuard comme suit :

```
import { Injectable } from '@angular/core';
import { ActivatedRouteSnapshot, CanActivateChild, RouterStateSnapshot, UrlTree } from
'@angular/router';
import { Observable } from 'rxjs';
import { TokenStorageService } from '../services/token-storage.service';

@Injectable({
  providedIn: 'root'
})
export class AdminGuard implements CanActivateChild {
  canActivateChild(
    childRoute: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<boolean | UrlTree> | Promise<boolean |
UrlTree> | boolean | UrlTree {
    if (!this.tokenStorage.hasRole('ADMIN')) {
      console.log("vous n'êtes pas autorisé à accéder à cette foncytionnalité");
      return false;
    }
    else return true;
  }
```

```
  constructor(private tokenStorage: TokenStorageService) {}}
```

- Créer le composant product-edit en lançant la commande suivante : **ng g c product-edit**
- Modifier le fichier product-edit.component.ts comme suit :

```
import { Component, OnInit } from '@angular/core';
import { FormBuilder, Validators } from '@angular/forms';
import { ActivatedRoute, Router } from '@angular/router';
import { Product } from '../model/product';
import { ProductService } from '../services/product.service';

@Component({
  selector: 'app-product-edit',
  templateUrl: './product-edit.component.html',
  styleUrls: ['./product-edit.component.css']
})
export class ProductEditComponent implements OnInit {
  productForm = this.fb.group({
    productID: ['', [Validators.required]],
    name: ['', [Validators.required]],
    price: ['', [Validators.required]]
  })
  constructor(private fb: FormBuilder, private produitService: ProductService, private
router: Router, private route: ActivatedRoute) {
    console.log('iam here')
  }

  ngOnInit(): void {

    this.produitService.getProduct('2').subscribe(
     response => {
        if (response)
          this.productForm.setValue(
            {
              'productID': response.productID,
              'name': response.name,
              'price': response.price
            });
      },
      err => {
        console.log(err);
      }

    )
```

```
  }

  save() {
    if (this.productForm.status === 'VALID') {
      let dto = new Product(
        this.productForm.get('productID')?.value,
        this.productForm.get('name')?.value,
        this.productForm.get('price')?.value,
      )
      this.produitService.save(dto);
      this.router.navigate([{ outlets: { primary: 'navbar', contenu: 'product' } }]);
    }
  }
}
```

- Modifier le fichier product-edit.component.html comme suit :

```
<div class="container">
    <form [formGroup]="productForm">
        <div class="form-group">
            <label for="Id">Id</label><br>
            <input formControlName="productID" type="text" class="form-control">
        </div>

        <div class="form-group">
            <label for="Intitulé">Name</label><br>
            <input formControlName="name" type="text" class="form-control">
        </div>

        <div class="form-group">
            <label for="Contenu">Prix</label><br>
            <input formControlName="price" type="text" class="form-control">
        </div>
        <pre></pre>
        <button class="btn btn-primary" type="submit" (click)="save()">
submit</button>
    </form>
</div>
```

- Modifier ensuite le fichier app-routing.module.ts comme suit :

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AuthComponent } from './auth/auth.component';
```

```
import { EmpCreateComponent } from './emp/emp-create/emp-create.component';
import { EmpListComponent } from './emp/emp-list/emp-list.component';
import { AdminGuard } from './guards/admin.guard';
import { ChechAuthGuard } from './guards/chech-auth.guard';
import { CheckAdminGuard } from './guards/check-admin.guard';
import { CheckSaveGuard } from './guards/check-save.guard';
import { NavbarComponent } from './navbar/navbar.component';
import { ProductEditComponent } from './product-edit/product-edit.component';
import { Product1Component } from './product1/product1.component';
import { Product2Component } from './product2/product2.component';
import { ProductListResolverService } from './services/product-list-
resolver.service';
import { WelcomeComponent } from './welcome/welcome.component';


const routes: Routes = [
  { path: '', redirectTo: 'login', pathMatch: 'full' },
  { path: 'login', component: AuthComponent },
  { path: 'navbar', component: NavbarComponent },
  {
    path: 'employees',
    component: EmpListComponent,
    outlet: 'contenu',
    canActivate: [ChechAuthGuard],
  },
  { path: 'create', component: EmpCreateComponent, outlet: 'contenu', canActivate:
[CheckAdminGuard], canDeactivate: [CheckSaveGuard] },
  { path: 'welcome', component: WelcomeComponent, outlet: 'contenu' },
  { path: 'logout', component: AuthComponent },
  {
    path: 'product1', component: Product1Component, outlet: 'contenu', canActivate:
[ChechAuthGuard],
    canActivateChild: [AdminGuard],
    children: [
      { path: 'edit/:id', component: ProductEditComponent }
    ]
  },
  {
    path: 'product2',
    component: Product2Component,
    resolve: { products: ProductListResolverService }, outlet: 'contenu'
  },
]

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

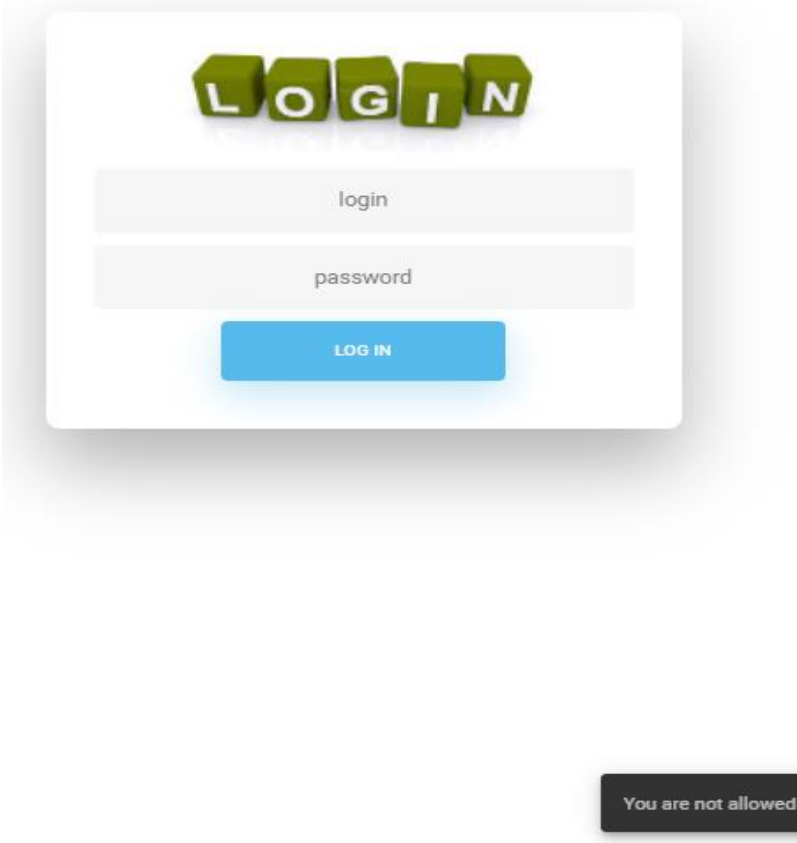
- Modifier le fichier product1.component.html comme suit :

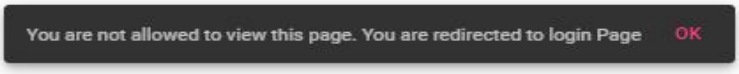
```
<h1> Without Resolve</h1>
<div class='table-responsive'>
    <table class='table'>
        <thead>
            <tr>
                <th>Name</th>
                <th>Price</th>
            </tr>
        </thead>
        <tbody>
            <tr *ngFor="let product of products;">
                <td><a [routerLink]="['..e/',{ outlets: { primary:
['navbar'],contenu: ['product1','edit',product.productID] } }]">{{product.name}} </a>
</td>
                <td>{{product.price}}</td>
            </tr>
        </tbody>
    </table>
</div>
```

4. Test
   - Lancer l'application : **ng serve**.
   - Accéder au lien http://localhost:4200/. La page suivante est affichée :

a. Tester le Guard ChechAuthGuard :

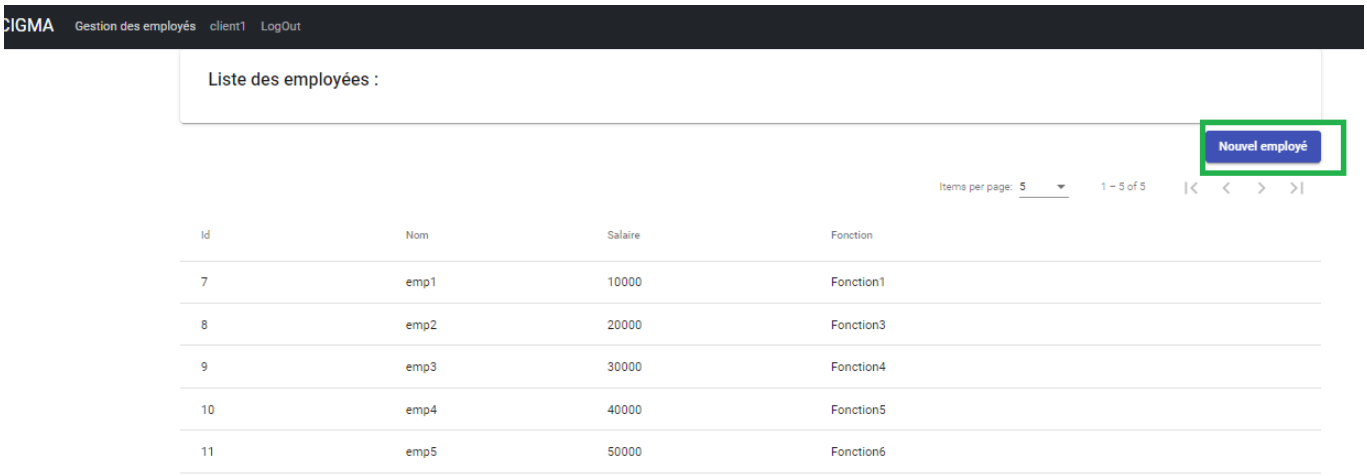
- Sans se connecter, essayer d'accéder au lien http://localhost:4200/navbar(contenu:employees). Le message suivant est affiché et l'application se redirige vers la page d'authentification :



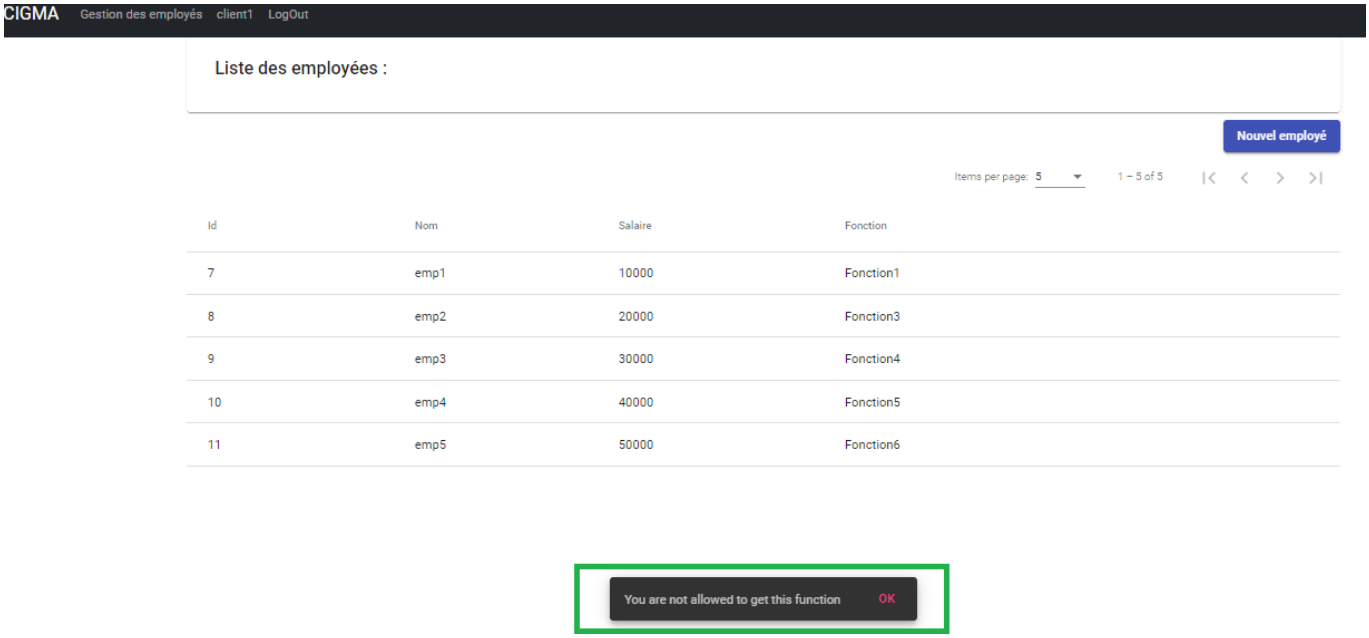


b. Tester le Guard CheckAdminGuard :

- Se connecter avec le compte « client1/client1).
- Cliquer ensuite sur le menu «Gestion des employés ». La page suivante est affichée :

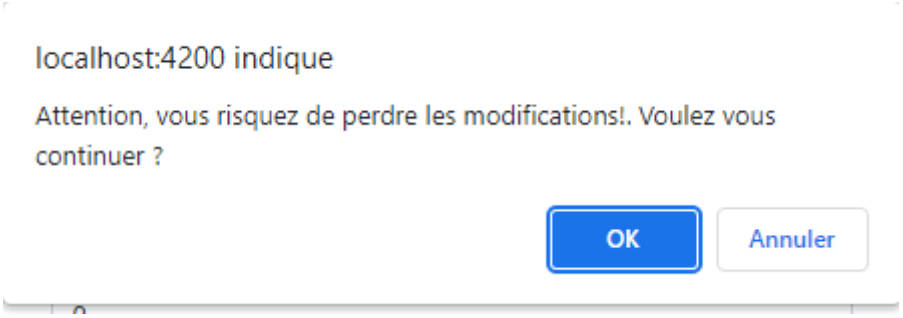


- Cliquer sur le menu « Nouvel employée » et vérifier que l'application affiche le message suivant :



c. Tester le Guard CheckSaveGuard :

- Se connecter avec le compte admin1/admin1.
- Ensuite, cliquer sur le menu « Gestion des employés ».
- Ensuite cliquer sur le bouton « Nouvel employé ».
- Cliquer ensuite sur un autre menu, par exemple : « Gestion des employés », « LogOut » et vérifier que l'application affiche le message suivant :



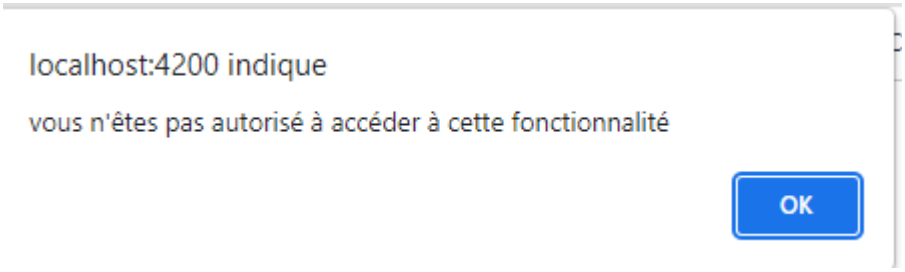
- Si vous cliquez sur « Annuler », vous restez dans le formulaire.
- Si vous cliquez sur « OK », l'application annule les modifications et effectue la redirection vers la page appelée.

d. Tester le Resolver ProductListResolverService :

- Se connecter avec le compte admin1/admin1.
- Ensuite, cliquer sur le menu « Gestion des produits ». Vous constatez que le composant est chargé au début ensuite les données sont affichées.
- Ensuite, cliquer sur le menu « Gestion des produits avec Resolver ». Vous constatez que le composant est chargé après que les données soient disponibles. En effet, Angular exécute les Resolver en denier lieu.

f. Tester le Guard canActivateChild :

- Se connecter avec le compte client1/client1.
- Ensuite, cliquer sur le menu « Gestion des produits ».
- Cliquer ensuite sur un produit pour l'éditer. L'application interdit l'accès et affiche le message suivant :



Fin du TP 12.