Atelier 03 Angular :

Modification et Suppression des produits

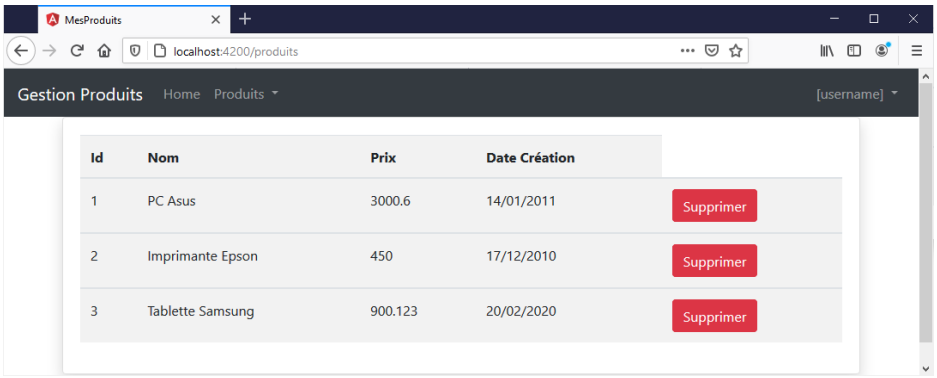
Objectifs :

1. Ajouter un lien pour supprimer un produit,
2. Ajouter un formulaire pour modifier un produit,
3. Transmettre des paramètres avec *ActivatedRoute,*
4. Naviguer entre les pages avec *Router,*
5. Cacher le champ Id Produit ou le rendre Read Only.

Ajouter un lien pour supprimer un produit

1. Editer le fichier produit.component.html et ajouter la ligne suivante :

```
<td><a class="btn btn-danger" (click)="supprimerProduit(produit)">Supprimer</a></td>
```



2. Editer le fichier produit.component.ts et ajouter la méthode *supprimerProduit* :

```
supprimerProduit(p: Produit)
  {
    console.log(p);
  }
```

3. Tester le lien « Supprimer »

4. Ajouter la méthode *supprimerProduit* au service *produit.service.ts* qui permet de supprimer un produit passé en argument du tableau produits (déclaré dans la classe *ProduitService*) :

```
supprimerProduit( prod: Produit){
    //supprimer le produit prod du tableau produits
     const index = this.produits.indexOf(prod, 0);
     if (index > -1) {
       this.produits.splice(index, 1);
     }
     //ou Bien
     /*  this.produits.forEach((cur, index) => {
        if(prod.idProduit === cur.idProduit) {
             this.produits.splice(index, 1);
          }
     }); */
   }
```

5. Editer le fichier produit.component.ts et modifier la méthode *supprimerProduit* :

```
supprimerProduit(p: Produit)
   {
      //console.log(p);
      this.produitService.supprimerProduit(p);
   }
```

6. Tester le lien « Supprimer »
7. Modifier la méthode *supprimerProduit* pour demander une confirmation :

```
supprimerProduit(p: Produit)
   {
      //console.log(p);
let conf = confirm("Etes-vous sûr ?");
if (conf)
        this.produitService.supprimerProduit(p);
   }
```

Ajouter un formulaire pour modifier un produit

8. Editer le fichier produit.component.html et ajouter la ligne suivante :

```
<td><a class="btn btn-success" [routerLink]="['/updateProduit',produit.idProduit]">Modifier</a></td>
```

9. Importer le modue RouterLink au niveau du fichier produit.component.ts

```
import { RouterLink } from '@angular/router';
…
 imports: [CommonModule,RouterLink],
```

10. Créer le Web component updateProduit (sans créer le fichier .spec ni le fichier .css):
    ng g c update-produit --skip-tests -s

11. Ajouter la route au fichier app-routes.ts :

```
import { UpdateProduitComponent } from './update-produit/update-produit.component';

const routes: Routes = [
  …
  {path: "updateProduit/:id",  component: UpdateProduitComponent}
```

12. Tester le lien « Modifier »
13. Ajouter la méthode consulterProduit à la classe ProduitService (cette méthode accepte comme paramètre un id d'un produit et retourne le produit en le cherchant dans le tableau produits) :

```
export class ProduitService {
 …
  produit! : Produit;
```

```
consulterProduit(id:number): Produit{
     this.produit =  this.produits.find(p => p.idProduit == id)!;
       return this.produit;
    }
```

Transmettre des paramètres avec *ActivatedRoute*

14. Modifier le fichier update-produit.component.ts comme suit :

```
import { Component, OnInit } from '@angular/core';
import { Produit } from '../model/produit.model';
import { ActivatedRoute } from '@angular/router';
import { ProduitService } from '../services/produit.service';
import { FormsModule } from '@angular/forms';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-update-produit',
  standalone: true,
  imports: [FormsModule, CommonModule],
  templateUrl: './update-produit.component.html',
  styles: ``
})
export class UpdateProduitComponent implements OnInit {
  currentProduit  = new Produit();

  constructor(private activatedRoute: ActivatedRoute,
              private produitService: ProduitService) { }

  ngOnInit() {
  // console.log(this.route.snapshot.params.id);
this.currentProduit =
this.produitService.consulterProduit(this.activatedRoute.snapshot.
params['id']);
   console.log(this.currentProduit);
  }
}
```

15. Modifier le fichier update-produit.component.html comme suit :

```
<div class="container">
     <div>
       <h2>Modifier un Produit :</h2>
     </div>
          <form >
              <div class="col-sm-2 col-md-2 col-lg-2">
                <label >ID  Produit</label>
                <input   type="text"  [(ngModel)]="currentProduit.idProduit"

                    name="idProduit" class="form-control">
               </div>
```

```
              <div class="col-sm-4 col-md-4 col-lg-4" >
                <label >Nom Produit</label>
                <input  type="text"[(ngModel)]= "currentProduit.nomProduit"
                    name="nomProduit" class="form-control">
              </div>

              <div class="col-sm-2 col-md-2 col-lg-2">
                <label >Prix Produit</label>
            <input type="number"   [(ngModel)]= "currentProduit.prixProduit"

                    name="prixProduit" class="form-control">
              </div>

              <div class="col-sm-4 col-md-4 col-lg-4">
                  <label >Date création</label>
     <input  type="date"  [ngModel]="currentProduit.dateCreation | date: 'yyyy-MM-dd'"
     (ngModelChange)=" currentProduit.dateCreation= $event"   name="dateCreation"
      class="form-control">
              </div>

               <div class="mt-2">
<button type="submit"  (click)="updateProduit()" class="btn btn-success">Modifer</button>
               </div>
          </form>
</div>
```

16. Ajouter la méthode updateProduit à la classe ProduitService :

```
17.     updateProduit( prod: Produit){
18.        //chercher le produit prod du tableau produits
19.       const index = this.produits.indexOf(prod, 0);
20.        if (index > -1) {
21.          this.produits.splice(index, 1); //supprimer l'ancien éléments
22.          this.produits.splice(index,0,prod); // insérer le nouvel élément
23.               }
24.             }
```

25. Ajouter la méthode updateProduit à la classe UpdateProduitComponent :

```
  updateProduit()

  { //console.log(this.currentProduit);
    this.produitService.updateProduit(this.currentProduit);
  }
```

Retourner à la page Liste Produits après la modification

26. Modifier la méthode updateProduit de la classe UpdateProduitComponent pour revenir à la page qui liste les produits (produits.component.html) après modification d'un produit :

```
import { ActivatedRoute,Router } from '@angular/router';
 constructor(private activatedRoute: ActivatedRoute,
             private router :Router,
             private produitService: ProduitService)
…

 updateProduit()
  { //console.log(this.currentProduit);
    this.produitService.updateProduit(this.currentProduit);
    this.router.navigate(['produits']);
   }
```

Cacher le champ Id Produit ou le rendre Read Only:

27. Modifier le fichier update-produit.component.html comme suit :

```
<div hidden class="col-sm-2 col-md-2 col-lg-2">
               <label >ID  Produit</label>
               <input   type="text"  [(ngModel)]="currentProduit.idProduit"
                   name="idProduit" class="form-control">
 </div>
```

```
<div  class="col-sm-2 col-md-2 col-lg-2">
               <label >ID  Produit</label>
<input readonly   type="text"  [(ngModel)]="currentProduit.idProduit"
                   name="idProduit" class="form-control">
 </div>
```