

Atelier 03 Angular :

Modification et Suppression des produits

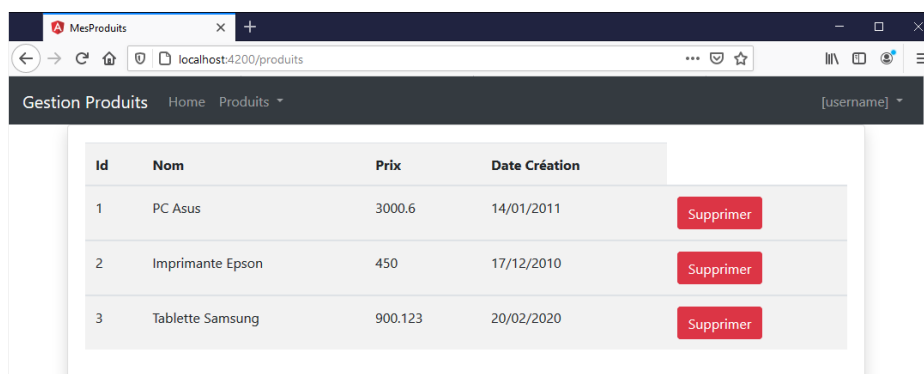
Objectifs :

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. Trier les produits à la suite d'une modification,
6. 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 :

```
8. supprimerProduit(p: Produit)  
9. {  
10.     //console.log(p);  
11.     let conf = confirm("Etes-vous sûr ?");  
12.     if (conf)  
13.         this.produitService.supprimerProduit(p);  
14. }
```

Ajouter un formulaire pour modifier un produit

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

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

16. Créer le Web component updateProduit :

ng g c update-produit --skipTests=true --inlineStyle=true

17. Ajouter la route au fichier app-routing.module.ts :

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

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

18. Tester le lien « Modifier »

19. 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*

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

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

@Component({
  selector: 'app-update-produit',
  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);
  }
}

```

21. 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">Modifier</button>
    </div>
    </form>
</div>

```

22. Ajouter la méthode updateProduit à la classe ProduitService :

```

updateProduit(p:Produit)
{
    // console.log(p);
    this.supprimerProduit(p);
    this.ajouterProduit(p);
}

```

23. 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

24. 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']);
  }

```

Trier les produits à la suite d'une modification

25. Ajouter le tri des produits à la classe ProduitService :

```

trierProduits(){
    this.produits = this.produits.sort((n1,n2) => {
        if (n1.idProduit! > n2.idProduit!) {
            return 1;
        }
        if (n1.idProduit! < n2.idProduit!) {
            return -1;
        }
        return 0;
    });
}

updateProduit(p:Produit)
{
    // console.log(p);
    this.supprimerProduit(p);
    this.ajouterProduit(p);
    this.trierProduits();
}

```

Cacher le champ Id Produit ou le rendre Read Only:

26. 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>

```