

Atelier 06 : Spring MVC Coté Serveur :

Formulaires et Validation

Cet Atelier concerne la création des formulaires pour ajouter et modifier des produits ainsi que la validation des données.

Objectifs :

1. Créer un formulaire pour ajouter des produits,
2. La validation des données,
3. Créer un formulaire pour modifier les produits,
4. Utiliser le même formulaire pour la création et la modification.

Créer un formulaire pour ajouter des produits

1. Créer, dans le dossier `src/main/resources/templates` le fichier `createProduit.html` :

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
      layout:decorator="template">
<link rel="stylesheet" type="text/css"
      href="/webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
<head>
<meta charset="utf-8">
<title>Ajout des Produits</title>
</head>
<body>
<div layout:fragment="Mycontent">
<div class="container mt-5">
<div class="card">
<div class="card-header"> Ajout des Produits </div>
<div class="card-body">
<form th:action="@{saveProduit}" method="post">
<div class="form-group">
<label class="control-label">Nom Produit :</label>
<input type="text" name="nomProduit" class="form-control" />
</div>
<div class="form-group">
<label class="control-label">Prix Produit :</label>
<input type="text" name="prixProduit" class="form-control" />
</div>
<div class="form-group">
<label class="control-label">date création :</label>
<input type="date" name="dateCreation" class="form-control" />
</div>
<div>
<button type="submit" class="btn btn-primary">Valider</button>
</div>
</form>
</div>
</div>
</div>
</body>
</html>
```

2. Modifier la méthode `saveProduit` de la classe `ProduitController` comme suit :

```
@RequestMapping("/saveProduit")
public String saveProduit(@ModelAttribute("produit") Produit produit)
{
    produitService.saveProduit(produit);
    return "createProduit";
}
```

3. Ajouter les annotations à l'attribut `dateCreation` de l'entité `Produit` :

```
@Temporal(TemporalType.DATE)
@DateTimeFormat(pattern = "yyyy-MM-dd")
private Date dateCreation;
```

La validation des données

4. Ajouter la dépendance `spring-boot-starter-validation`

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-validation</artifactId>
</dependency>
```

5. Editer l'entité `Produit` et ajouter les contraintes de validation suivantes :

```
import javax.validation.constraints.Max;
import javax.validation.constraints.Min;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
```

...

```
@NotNull
@Size (min = 4,max = 15)
private String nomProduit;
```

```
@Min(value = 10)
@Max(value = 10000)
private Double prixProduit;
```

```
@Temporal(TemporalType.DATE)
@DateTimeFormat(pattern = "yyyy-MM-dd")
@PastOrPresent
private Date dateCreation;
```

6. Modifier les méthodes `showCreate` et `saveProduit` de la classe `ProduitController` :

```

@RequestMapping("/showCreate")
public String showCreate(ModelMap modelMap)
{
    modelMap.addAttribute("produit", new Produit());
    return "createProduit";
}

@RequestMapping("/saveProduit")
public String saveProduit(@Valid Produit produit,
                          BindingResult bindingResult)
{
    if (bindingResult.hasErrors()) return "createProduit";

    produitService.saveProduit(produit);
    return "createProduit";
}

```

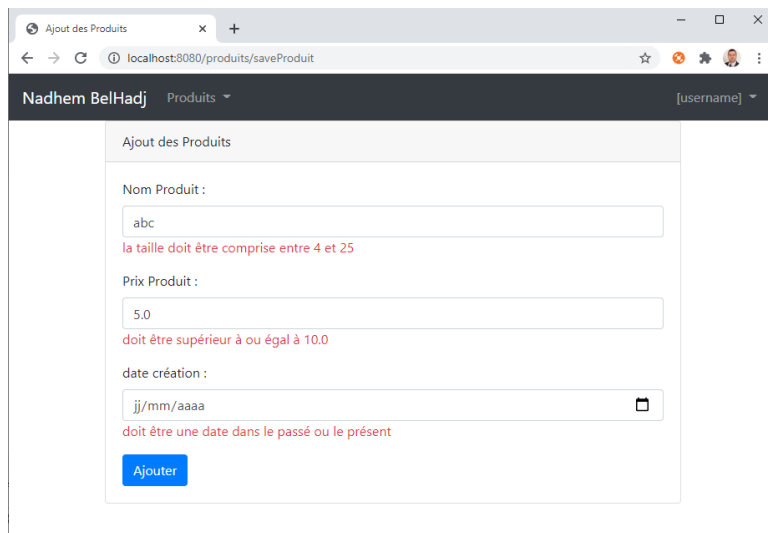
7. Modifier le fichier createProduit.html :

```

<form th:action="@{saveProduit}" method="post">
    <div class="form-group">
        <label class="control-label">Nom Produit :</label>
        <input type="text" name="nomProduit" class="form-control"
th:value="${produit.nomProduit}" />
        <span th:errors=${produit.nomProduit} class="text-danger"> </span>
    </div>
    <div class="form-group">
        <label class="control-label">Prix Produit :</label>
        <input type="text" name="prixProduit" class="form-control"
th:value="${produit.prixProduit}" />
        <span th:errors=${produit.prixProduit} class="text-danger"> </span>
    </div>
    <div class="form-group">
        <label class="control-label">date création :</label>
        <input type="date" name="dateCreation" class="form-control"
th:value="${produit.dateCreation}" />
        <span th:errors=${produit.dateCreation} class="text-danger"> </span>
    </div>
    <div>
        <button type="submit" class="btn btn-primary">Ajouter</button>
    </div>
</form>

```

8. Tester la validation des données



Ajout des Produits

Nom Produit :

abc

la taille doit être comprise entre 4 et 25

Prix Produit :

5.0

doit être supérieur à ou égal à 10.0

date création :

jj/mm/aaaa

doit être une date dans le passé ou le présent

Ajouter

Créer un formulaire pour modifier les produits

9. Modifier la page listeProduit.html pour ajouter le lien « Editer » :

```
<td><a class="btn btn-success"
th:href="@{modifierProduit(id=${p.idProduit})}">Editer</a></td>
```

10. Créer la page editProduit.html :

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"
      layout:decorator="template">
<link rel="stylesheet" type="text/css"
href="/webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
<head>
<meta charset="utf-8">
<title>Ajout des Produits</title>
</head>
<body>
<div layout:fragment="Mycontent">
<div class="container mt-5">
<div class="card">
<div class="card-header">
Modifier des Produits
</div>
<div class="card-body">
<form th:action="@{saveProduit}" method="post">
<div class="form-group">
<label class="control-label" hidden>ID Produit :</label>
<input type="hidden" name="idProduit" class="form-control"
th:value="${produit.idProduit}" />
</div>
<div class="form-group">
<label class="control-label">Nom Produit :</label>
<input type="text" name="nomProduit" class="form-control"
th:value="${produit.nomProduit}" />
<span th:errors= ${produit.nomProduit} class="text-danger">
</span>
</div>
</div>
</div>
</div>
```

```

        </div>
        <div class="form-group">
            <label class="control-label">Prix Produit :</label>
            <input type="text" name="prixProduit" class="form-control"
th:value="${produit.prixProduit}" />
            <span th:errors=${produit.prixProduit} class="text-danger">
</span>
        </div>
        <div class="form-group">
            <label class="control-label">date création :</label>
            <input type="date" name="dateCreation" class="form-control"
th:value="${produit.dateCreation}" />
            <span th:errors=${produit.dateCreation} class="text-danger">
</span>
        </div>
        <div>
            <button type="submit" class="btn btn-primary">Valider</button>
        </div>
    </form>
</div>
</div>
</body>
</html>

```

Utiliser le même formulaire pour la création et la modification

11. Renommer le fichier createProduit.html en formProduit.html
12. Au niveau de la classe ProduitController, modifier les méthodes showCreate et saveProduit :

```

@RequestMapping("/showCreate")
public String showCreate(ModelMap modelMap)
{
    modelMap.addAttribute("produit", new Produit());
    return "formProduit";
}

@RequestMapping("/saveProduit")
public String saveProduit(@Valid Produit produit,
                          BindingResult bindingResult)
{
    if (bindingResult.hasErrors()) return "formProduit";

    produitService.saveProduit(produit);
    return "formProduit";
}

```

13. Tester l'ajout d'un produit,

14. Au niveau de la classe ProduitController, modifier `editerProduit` :

```
@RequestMapping("/modifierProduit")
public String editerProduit(@RequestParam("id") Long id, ModelMap modelMap)
{
    Produit p= produitService.getProduit(id);
    modelMap.addAttribute("produit", p);
    modelMap.addAttribute("mode", "edit");
    return "formProduit";
}
```

```
@RequestMapping("/showCreate")
public String showCreate(ModelMap modelMap)
{
    modelMap.addAttribute("produit", new Produit());
    modelMap.addAttribute("mode", "new");
    return "formProduit";
}
```

15. Modifier le fichier `formProduit.html` comme suit :

```
...
<div class="card">
  <div class="card-header" th:if="${mode=='new'}"> Ajout des Produits </div>
  <div class="card-header" th:if="${mode=='edit'}">Modification des Produits
</div>
  <div class="card-body">
    <form th:action="@{saveProduit}" method="post">
      <div class="form-group">
        <label class="control-label" hidden>ID Produit :</label>
        <input type="hidden" name="idProduit" class="form-control"
          th:value="${produit.idProduit}" />
      </div>
    </form>
  </div>
</div>
...
```