

Atelier 03 Spring Boot :

Spring MVC Coté Serveur :

Thymeleaf et Bootstrap

Dans cet Atelier on va apprendre, d'abord, comment intégrer **Bootstrap** à nos pages JSP développées dans l'atelier précédent, puis comment utiliser le moteur de template **Thymeleaf** pour développer la couche présentation (View) de notre application Web MVC.

Objectifs :

1. Ajouter la dépendance *Bootstrap* au fichier *pom.xml*,
2. Ajouter *Bootstrap* à la vue *createProduit.jsp*,
3. Ajouter *Bootstrap* à la vue *listeProduit.jsp*,
4. Ajouter *Bootstrap* à la vue *editerProduit.jsp*.
5. Ajouter la dépendance Thymeleaf au fichier *pom.xml*,
6. Ma première page Thymeleaf,
7. Utiliser la pagination,
8. Ajouter un lien permettant de supprimer des produits,
9. Ajouter un menu Navbar.

Ajouter la dépendance Bootstrap au fichier pom.xml

1. Editer le fichier *pom.xml* de votre application et ajouter la dépendance suivante :

```
<dependency>
    <groupId>org.webjars</groupId>
    <artifactId>bootstrap</artifactId>
    <version>4.3.1</version>
</dependency>
```

2. Enregistrer le fichier *pom.xml* pour que Maven télécharge les fichiers jar nécessaires dans votre local repository (C:\Users\NADHEM\.m2) où NADHEM est le nom de l'utilisateur Windows.

Ajouter Bootstrap à la vue createProduit.jsp

3. Modifier le fichier *createProduit.jsp* comme suit :

```
<%@ page language="java" contentType="text/html; charset=windows-1256"
    pageEncoding="windows-1256"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
```

```

<link rel="stylesheet" type="text/css"
href="webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
<:url value="/css/main.css" var="jstlCss" />
<link href="${jstlCss}" rel="stylesheet" />
<script type="text/javascript"
src="webjars/bootstrap/4.3.1/js/bootstrap.min.js"></script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Ajouter Produit</title>
</head>
<body>
<div class="container mt-5" >
<div class="card-body">
<form 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="date" class="form-control"/>
  </div>

  <div>
    <button type="submit" class="btn btn-primary">Ajouter</button>
  </div>
</form>
</div>
${msg}
<br/>
<br/>
<a href="ListeProduits">Liste Produits</a>
</div>
</body>
</html>

```

The screenshot shows a web browser window with the title 'Ajouter Produit'. The address bar displays 'localhost:8080/produits/showCreate'. The form contains three input fields: 'Nom Produit', 'Prix Produit', and 'date création'. Below these fields is a blue button labeled 'Ajouter'. At the bottom left of the form, there is a link labeled 'Liste Produits'.

Ajouter Bootstrap à la vue listeProduit.jsp

4. Modifier le fichier *listeProduit.jsp* comme suit :

```
<%@ page language="java" contentType="text/html; charset=windows-1256"
    pageEncoding="windows-1256"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
    <link rel="stylesheet" type="text/css"
href="webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
    <c:url value="/css/main.css" var="jstlCss" />
    <link href="${jstlCss}" rel="stylesheet" />
    <script type="text/javascript"
src="webjars/bootstrap/4.3.1/js/bootstrap.min.js"></script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Liste Produits</title>
</head>
<body>
<div class="container mt-5">
<div class="card">
    <div class="card-header">
        Liste des Produits
    </div>
    <div class="card-body">

        <table class="table table-striped">
            <tr>
<th>ID</th><th>Nom Produit</th><th>Prix</th><th>Date
Création</th><th>Suppression</th><th>Edition</th>
            </tr>
            <c:forEach items="${produits}" var="p">
                <tr>
                    <td>${p.idProduit }</td>
                    <td>${p.nomProduit }</td>
                    <td>${p.prixProduit }</td>
                    <td><fmt:formatDate pattern="dd/MM/yyyy"
value="${p.dateCreation}" /></td>
                    <td><a onclick="return confirm('Etes-vous sûr ?')"
href="supprimerProduit?id=${p.idProduit }">Supprimer</a></td>
                    <td><a href="modifierProduit?id=${p.idProduit }">Edit</a></td>
                </tr>
            </c:forEach>
        </table>
    </div>
</div>
</body>
</html>
```

ID	Nom Produit	Prix	Date Création	Suppression	Edition
1	PC Asus	2546.0	25/07/2020	Supprimer	Edit
2	imprimante Epson	500.0	09/10/2011	Supprimer	Edit
4	PS 4	1200.0	27/07/2020	Supprimer	Edit

Ajouter Bootstrap à la vue `editProduit.jsp`

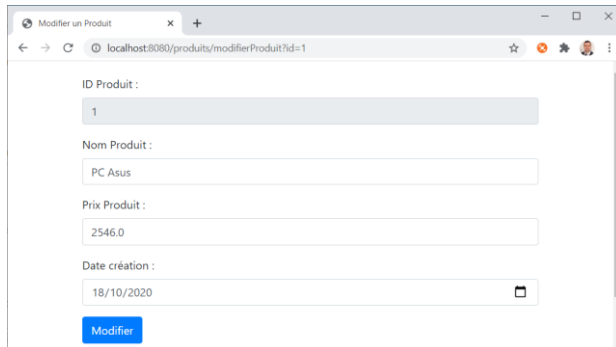
5. Modifier le fichier `editProduit.jsp` comme suit :

```
<%@ page language="java" contentType="text/html; charset=windows-1256"
    pageEncoding="windows-1256"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
    <link rel="stylesheet" type="text/css"
href="webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
    <c:url value="/css/main.css" var="jstlCss" />
    <link href="${jstlCss}" rel="stylesheet" />
    <script type="text/javascript"
src="webjars/bootstrap/4.3.1/js/bootstrap.min.js"></script>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
<title>Modifier un Produit</title>
</head>
<body>
<div class="container mt-5">
<div class="card-body">
<form action="updateProduit" method="post">
    <div class="form-group">
        <label class="control-label">ID Produit :</label>
        <input type="text" name="idProduit" value="${produit.idProduit}"
readonly class="form-control"/>
    </div>
    <div class="form-group">
        <label class="control-label">Nom Produit :</label>
        <input type="text" name="nomProduit" value="${produit.nomProduit}"
class="form-control"/>
    </div>
    <div class="form-group">
        <label class="control-label">Prix Produit :</label>
        <input type="text" name="prixProduit" value="${produit.prixProduit}"
class="form-control"/>
    </div>
    <div class="form-group">
        <label class="control-label"> Date création :</label>
        <fmt:formatDate pattern="yyyy-MM-dd" value="${produit.dateCreation}"
var="formatDate" />
        <input type="date" name="date" value="${formatDate}" class="form-
control"/>
    </div>
    </div>
</div>
</body>
</html>
```

```

    </div>
    <div>
        <button type="submit" class="btn btn-primary">Modifier</button>
    </div>
</form>
</div>
<br/>
<br/>
<a href="ListeProduits">Liste Produits</a>
</div>
</body>
</html>

```



Ajouter la dépendance Thymeleaf au fichier pom.xml

6. Editer le fichier pom.xml de votre application et ajouter la dépendance suivante :

```

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

```

Ma première page Thymeleaf

7. Désormais on va plus utiliser les JSP pour cela, éditer le fichier application.properties et commenter les deux lignes suivantes :

```

#spring.mvc.view.prefix=/WEB-INF/view/
#spring.mvc.view.suffix=.jsp

```

8. Créer, dans le dossier src/main/resource/template le fichier listeProduits.html :

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<link rel="stylesheet" type="text/css"
href="webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
<head>
<meta charset="utf-8">
<title>Liste des Produits</title>
</head>
<body>
<div class="container mt-5">
<div class="card">
    <div class="card-header">
        Liste des Produits
    </div>

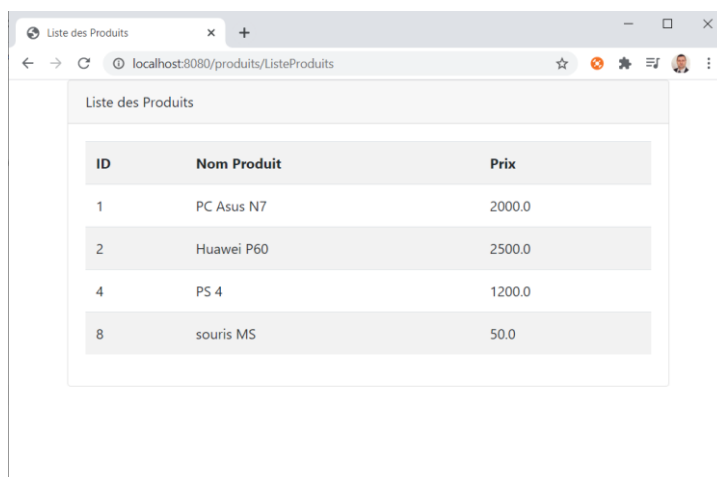
```

```

</div>
<div class="card-body">

    <table class="table table-striped">
        <tr>
            <th>ID</th><th>Nom Produit</th><th>Prix</th>
        </tr>
        <tr th:each="p:${produits}">
            <td th:text="${p.idProduit}"></td>
            <td th:text="${p.nomProduit}"></td>
            <td th:text="${p.prixProduit}"></td>
        </tr>
    </table>
</div>
</div>
</div>
</body>
</html>

```



ID	Nom Produit	Prix
1	PC Asus N7	2000.0
2	Huawei P60	2500.0
4	PS 4	1200.0
8	souris MS	50.0

Utiliser la pagination

9. Ajouter à l'interface ProduitService :

```
Page<Produit> getAllProduitsParPage(int page, int size);
```

10. Ajouter son implémentation à la classe ProduitServiceImpl :

```

@Override
public Page<Produit> getAllProduitsParPage(int page, int size) {
    return produitRepository.findAll(PageRequest.of(page, size));
}

```

11. Tester la méthode getAllProduitsParPage :

```

@Test
public void testFindByNomProduitContains()
{
    Page<Produit> prods = produitService.getAllProduitsParPage(0,2);
    System.out.println(prods.getSize());
    System.out.println(prods.getTotalElements());
}

```

```

        System.out.println(prods.getTotalPages());

        prods.getContent().forEach(p -> {System.out.println(p.toString());
                                          });

        /*ou bien
        for (Produit p : prods)
        {
            System.out.println(p);
        } */
    }
}

```

12. Au niveau de la classe *ProduitController*, modifier la méthode *listeProduits* pour qu'elle prenne en charge la pagination :

```

@RequestMapping("/ListeProduits")
public String listeProduits(ModelMap modelMap,
    @RequestParam (name="page",defaultValue = "0") int page,
    @RequestParam (name="size", defaultValue = "2") int size)
{
    Page<Produit> prods = produitService.getAllProduitsParPage(page, size);
    modelMap.addAttribute("produits", prods);
    modelMap.addAttribute("pages", new int[prods.getTotalPages()]);
    modelMap.addAttribute("currentPage", page);
    return "listeProduits";
}

```

13. Modifier la page *listeProduits.html* :

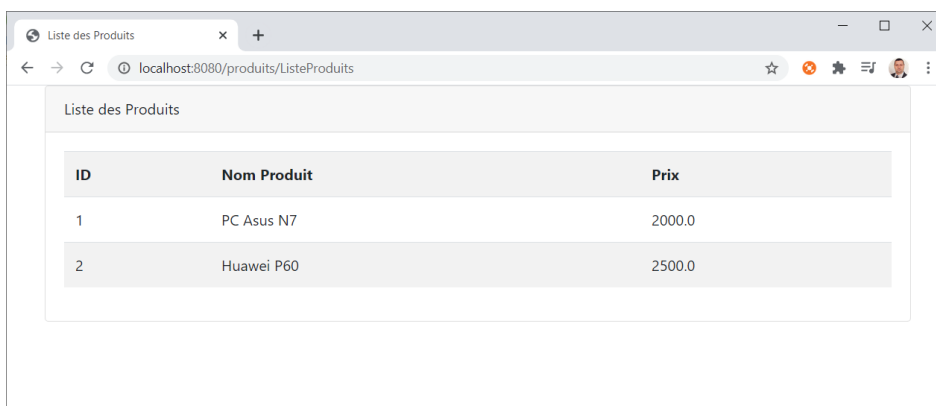
```

...
</tr>

<tr th:each="p:${produits.content}">

```

14. Tester



ID	Nom Produit	Prix
1	PC Asus N7	2000.0
2	Huawei P60	2500.0

Ajouter quelques produits à chaque démarrage de l'application

15. Modifier la classe ProduitsApplication comme suit :

```
@SpringBootApplication
public class ProduitsApplication implements CommandLineRunner {

    @Autowired
    ProduitService produitService;

    public static void main(String[] args) {
        SpringApplication.run(ProduitsApplication.class, args);
    }

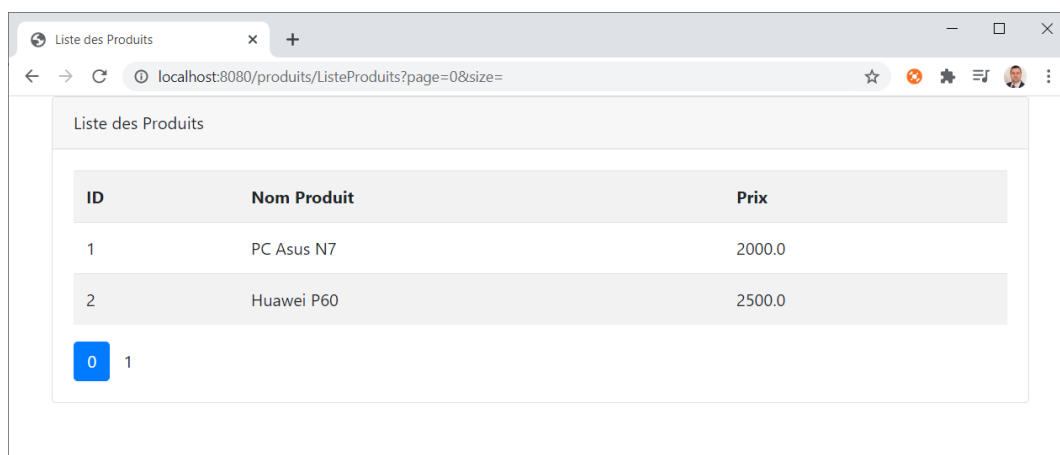
    @Override
    public void run(String... args) throws Exception {
        produitService.saveProduit(new Produit("PC Dell", 2600.0, new Date()));
        produitService.saveProduit(new Produit("PC Asus", 2800.0, new Date()));
        produitService.saveProduit(new Produit("Imprimante Epson", 900.0, new Date()));
    }
}
```

Ajouter la barre de navigation entre les pages

16. Modifier la page *listeProduits.html* comme suit :

```
...
<table class="table table-striped">
...
</table>
<ul class="nav nav-pills">
    <li th:each="page,status:${pages}">
        <a th:class="${status.index==currentPage?'btn btn-primary':'btn' }"
            th:href="@{ ListeProduits(page=${status.index}, size=${size} ) }"
            th:text="${status.index }"></a>
    </li>
</ul>
...
```

17. Tester :



Ajouter un lien permettant de supprimer des produits

18. Editer la page listeProduits.html :

```
...
<td th:text="${p.prixProduit}"></td>
<td><a class="btn btn-danger" th:href="@{supprimerProduit(id=${p.idProduit},
page=${currentPage}, size=${size})}">Supprimer</a></td>
...

```

19. Modifier, au niveau de la classe Produit Controller, la méthode supprimerProduit :

```
@RequestMapping("/supprimerProduit")
public String supprimerProduit(@RequestParam("id") Long id,
                                ModelMap modelMap,
                                @RequestParam(name="page", defaultValue = "0") int page,
                                @RequestParam(name="size", defaultValue = "2") int size)
{
    produitService.deleteProduitById(id);
    Page<Produit> prods = produitService.getAllProduitsParPage(page,
size);
    modelMap.addAttribute("produits", prods);
    modelMap.addAttribute("pages", new int[prods.getTotalPages()]);
    modelMap.addAttribute("currentPage", page);
    modelMap.addAttribute("size", size);
    return "listeProduits";
}

```

20. Ajouter la confirmation de suppression

```
<a class="btn btn-danger" onclick="return confirm('Etes-vous sûr ?') " ...

```

Ajouter un menu Navbar

21. Ajouter la dépendance thymeleaf-layout-dialect nécessaire pour les templates thymeleaf :

```
<dependency>
    <groupId>nz.net.ultraq.thymeleaf</groupId>
    <artifactId>thymeleaf-layout-dialect</artifactId>
</dependency>

```

22. Ajouter la dépendance JQuery nécessaire pour faire fonctionner le dropdown list de la navbar bootstrap :

```
<dependency>
    <groupId>org.webjars</groupId>
    <artifactId>jquery</artifactId>
    <version>3.5.1</version>
</dependency>

```

23. Créer la page template.html :

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">

<link rel="stylesheet" type="text/css"
href="webjars/bootstrap/4.3.1/css/bootstrap.min.css" />
<script src="webjars/jquery/3.5.1/jquery.min.js"></script>
<script type="text/javascript"
src="webjars/bootstrap/4.3.1/js/bootstrap.min.js"></script>

<head>
<meta charset="utf-8">
<title>Gestion des Produits</title>
</head>
<body>
<nav class="navbar navbar-expand-sm bg-dark navbar-dark">
  <!-- Brand -->
  <a class="navbar-brand" href="#">Gestion Produits</a>

  <!-- Links -->
  <ul class="navbar-nav">
    <li class="nav-item">
      <a class="nav-link" th:href="@{ListeProduits}" >Home</a>
    </li>

    <!-- Dropdown -->
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">
        Produits
      </a>
      <div class="dropdown-menu">
        <a class="dropdown-item" th:href="@{showCreate}">Ajouter</a>
        <a class="dropdown-item" th:href="@{ListeProduits}">Lister</a>
      </div>
    </li>
  </ul>

  <ul class="navbar-nav ml-auto" >
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">
        [username]
      </a>
      <div class="dropdown-menu">
        <a class="dropdown-item" href="#">login</a>
        <a class="dropdown-item" href="#">logout</a>
        <a class="dropdown-item" href="#">Profile</a>
      </div>
    </li>
  </ul>
</nav>
<section layout:fragment="Mycontent" >
</section>

</body>
</html>
```

24. Modifier la page listeProduit.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>Liste des Produits</title>
</head>
<body>
  <div layout:fragment="Mycontent">
    <!-- Pour la version de spring boot 2.6+ Vous remplacez la ligne ci-dessus
    <div layout:fragment="Mycontent">
      Par la ligne suivante :
        <div th:replace="template"></div>
    -->
  <div class="container">
  <div class="card">
    <div class="card-header">
      Liste des Produits
    </div>
    <div class="card-body">
      <table class="table table-striped">
        <tr>
          <th>ID</th><th>Nom Produit</th><th>Prix</th>
        </tr>
        <tr th:each="p:${produits.content}">
          <td th:text="${p.idProduit}"></td>
          <td th:text="${p.nomProduit}"></td>
          <td th:text="${p.prixProduit}"></td>
          <td><a class="btn btn-danger" onclick="return confirm('Etes-vous
sûr ?')"
              th:href="@{supprimerProduit(id=${p.idProduit},
page=${currentPage},size=${size})}">Supprimer</a></td>
        </tr>
      </table>
      <ul class="nav nav-pills">
        <li th:each="page,status:${pages}">
          <a th:class="${status.index==currentPage?'btn btn-primary':'btn' }"
            th:href="@{ ListeProduits(page=${status.index}, size=${size} ) }"
            th:text="${status.index }"></a>
        </li>
      </ul>
    </div>
  </div>
</div>
</div>
</div>
</body>
</html>
```