TP (doc) : IPI - JVA320 Servlets, Spring MVC et Thymeleaf

1. Rendu des pages web développées en phase de design (Natural Templating)

Afficher cette page sans rendu Thymeleaf. Il suffit d'aller à http://localhost:8080/home.html. Si vous lisez ces lignes, c'est que vous y êtes arrivés !

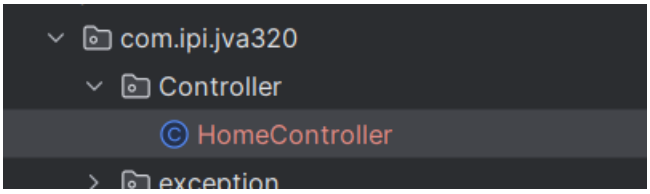
Installation du projet et premier run

2. Page d'accueil

La racine de l'application doit rediriger sur le présent template Thymeleaf. Développer le bon HomeController pour cela..

Se rendre dans les pages p.27-28 du cours (pdf) :

@Controller + @GetMapping(...)




```java
package com.ipi.jva320.Controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.GetMapping;

@Controller  no usages
public class HomeController {

    @GetMapping(value = "/")  no usages
    public String home(final ModelMap model) {
        return "home";
    }
}
```

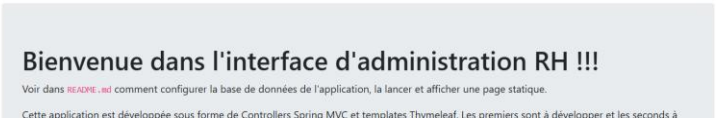
3. Message en page d'accueil

Rendre configurable le message de bienvenue sur la présente page (Bienvenue dans l'interface d'administration RH !) dans le fichier de messages de l'application (main/resources/messages(en/fr(_FR)).properties).

Se rendre sur la p.35 du pdf

```java
import org.springframework.web.bind.annotation.GetMapping;

@Controller  no usages
public class HomeController {

    @GetMapping(value = "/")  no usages
    public String home(final ModelMap model) {
        model.put("messageHome","Bienvenue dans l'interface d'administration RH !!!");
        return "home";
    }
}
```

```html
<div class="container mt-2">
    <div class="jumbotron">
        <!-- <h1>Bienvenue dans l'interface d'administration RH !
        <h1 th:text="${messageHome}"></h1>
        <p>Voir dans <code>README.md</code> comment configurer la
        <p>Cette application est développée sous forme de Controll
        <p/>
```



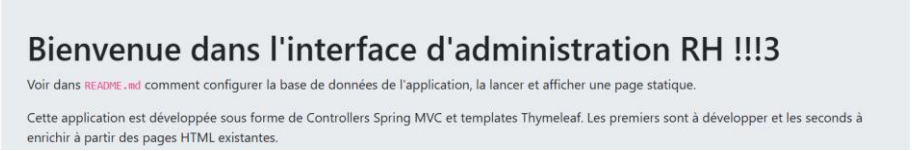
4. Afficher le nombre de salariés

Dans le titre au-dessus, "26" doit être remplacé par le nombre de salariés réellement en base. Développer ou enrichir la bonne méthode mappée de Controller en utilisant méthode countSalaries() de SalarieAideADomicileService pour cela (à injecter dans le Controller en l'annotannt @Autowired).

Se rendre sur la p.36

```java
@Autowired  1 usage
SalarieAideADomicileService salarieService;
```

```java
@GetMapping(value = "/")  no usages
public String home(final ModelMap model) {
    model.put("messageHome","Bienvenue dans l'interface d'administration RH !!!");
    model.put("nbSalarie", salarieService.countSalaries());
    return "home";
}
```



5. Afficher la page des détails d'un salarié

Développer ou enrichir le bon Controller pour rendre le template du détail d'un salarié "detail_Salarie.html" en allant à http://localhost:8080/salaries/1.

Se rendre sur la p.44 du pdf

```
@Autowired  3 usages
SalarieAideADomicileService salarieService;
SalarieAideADomicile s = new SalarieAideADomicile();  1 usage

@GetMapping(value = "/salaries/{id}")  no usages
public String getSalarie(final ModelMap model,
                         @PathVariable Long id) {
    SalarieAideADomicile salarie = salarieService.getSalarie(id);
    model.put("salarie", salarie);
    model.put("nbSalarie", salarieService.countSalaries());
    return "detail_Salarie";
}
```

6. Afficher le détail d'un salarié de test

Dans le Controller, passer à la vue un salarié de test créé à la volée, par exemple par :

SalarieAideADomicile aide = new SalarieAideADomicile("Jeannette Dupontelle", LocalDate.of(2021, 7, 1), LocalDate.now(), 0, 0, 10, 1, 0);

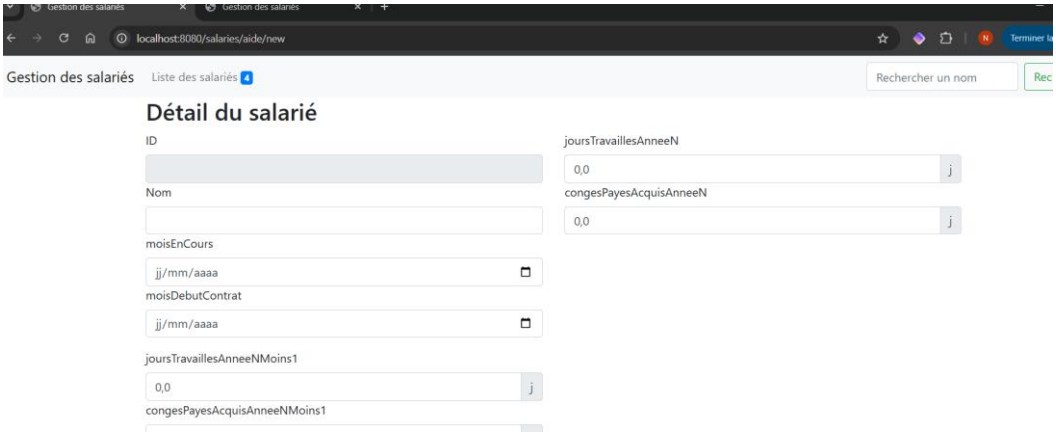
et en afficher les données lors du rendu dynamique, en enrichissant la page detail_Salarie.html des bons attributs Thymeleaf (utiliser th:text...). Il est aussi possible de réaliser à la place directement l'étape "Afficher le détail des salariés créés".

7. Création d'un salarié aide à domicile

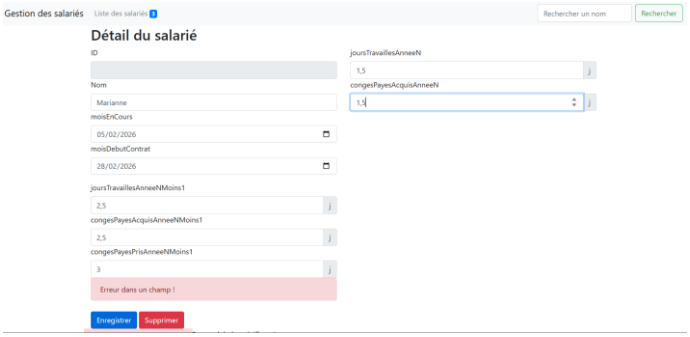
En allant à http://localhost:8080/salaries/aide/new ou via le bouton *Nouveau salarié* présent dans la liste des salariés, on accède au formulaire de création d'un salarié. Développer ou enrichir le bon Controller pour traiter l'appel POST /salaries/save avec les données du formulaire qui sont envoyées de manière à récupérer ces dernières côté serveur et en créer un nouveau salarié en utilisant la bonne méthode de SalarieAideADomicileService (donc à injecter dans le controller à l'aide de l'annotation @Autowired).

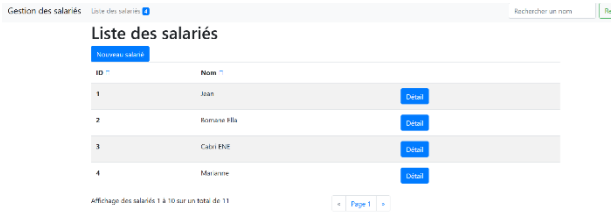
Se rendre sur la p.43

```java
@GetMapping(value = "/salaries/aide/new")  no usages
public String addSalarie(final ModelMap model) {
    model.put("salarie", new SalarieAideADomicile());
    model.put("nbSalarie", salarieService.countSalaries());
    return "detail_Salarie";
}

@PostMapping(value = "/salaries/save")  no usages
public String saveSalarie(SalarieAideADomicile salarie) throws SalarieException {
    salarieService.creerSalarieAideADomicile(salarie);
    return "redirect:/salaries/" + salarie.getId();
}
```



(Fonctionnalité new après les autres tp :)





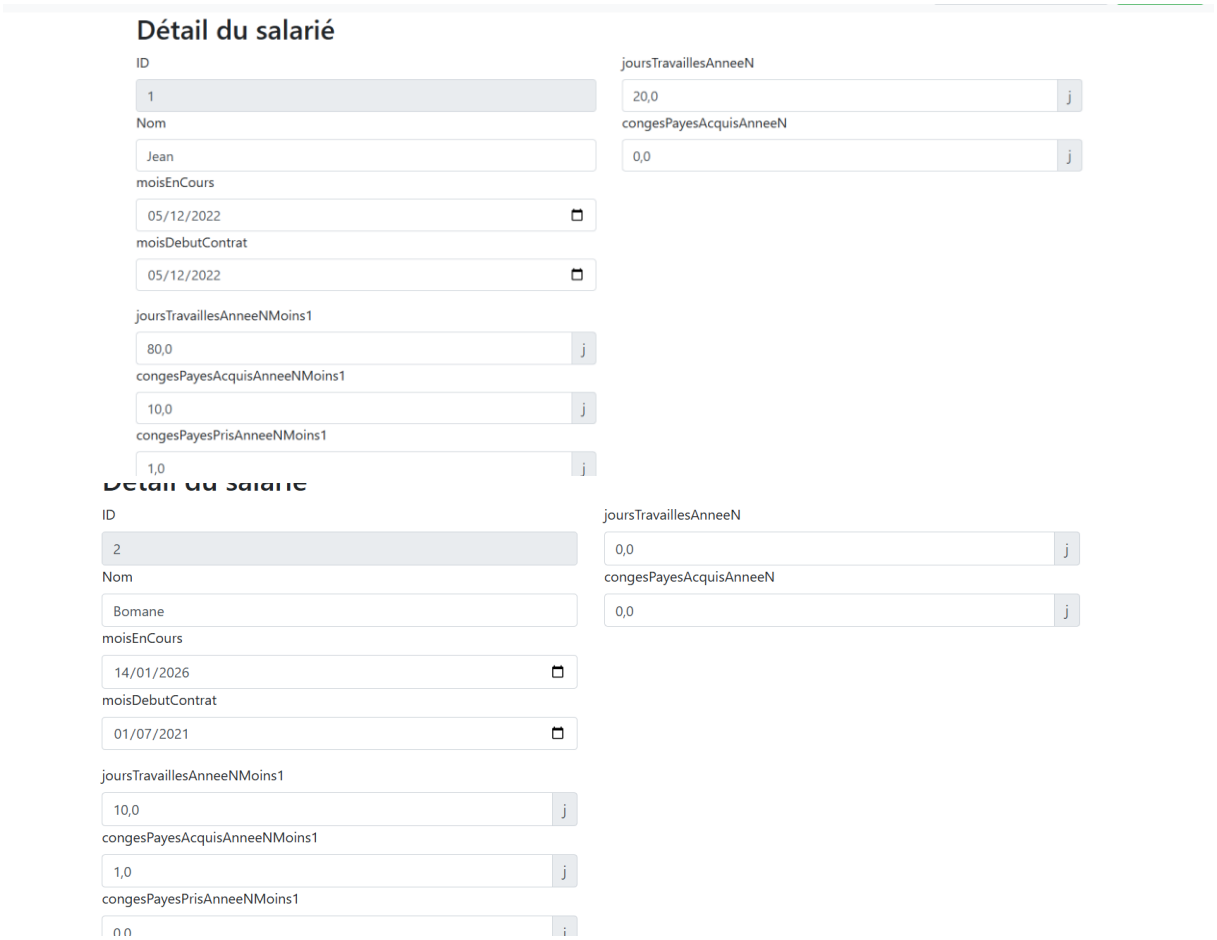
8. Afficher le détail des salariés créés

Modifier la (méthode de) Controller précédemment créée pour qu'elle ne retourne plus le salarié de test créé à la volée (ou uniquement s'il n'y en a pas ou sur id 0) mais les données réelles stockées dans la base de données, en utilisant la bonne méthode de SalarieAideADomicileService.

Que se passe-t-il en allant à http://localhost:8080/salaries/1000, est-ce normal ? Il y a une erreur car le salarié avec l'identifiant 1000 n'existe pas.

Se rendre sur la p.44

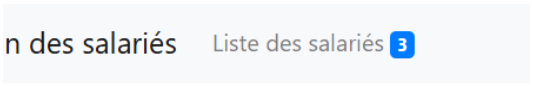
```
<h2>Detail du salarie</h2>

<form id="saveForm" action="http://localhost:8080/salaries/aide/10" method="post" class="row"
      th:action="@{/salaries/{id}(id=${salarie.id})}">

    <div class="col-lg-6">
        <div class="form-group">
            <label class="form-control-label" for="id">ID</label>
            <input type="text" th:value="${salarie.id}" class="form-control" name="id" id="id" disabled="true">

            <label class="form-control-label" for="nom">Nom</label>
            <input type="text" th:value="${salarie.nom}" class="form-control" name="nom" id="nom">

            <label class="form-control-label" for="moisEnCours">moisEnCours</label>
            <input type="date" th:value="${salarie.moisEnCours}" class="form-control" name="moisEnCours"
```



9. Afficher la liste des salariés

Développer ou enrichir le bon Controller pour rendre le template de liste des salariés "list.html" en allant à http://localhost:8080/salaries. Puis lui faire passer à la vue la liste des salariés stockées en base de données en utilisant la bonne méthode de SalarieAideADomicileService, et enfin dans "list.html" faire afficher les données de tous les salariés ainsi passés lors du rendu dynamique, en l'enrichissant des bons attributs Thymeleaf (utiliser aussi th:each...). Faire en sorte que les salariés d'exemple en dur dans la page web soient toujours affichés en statique mais disparaissent en rendu dynamique.

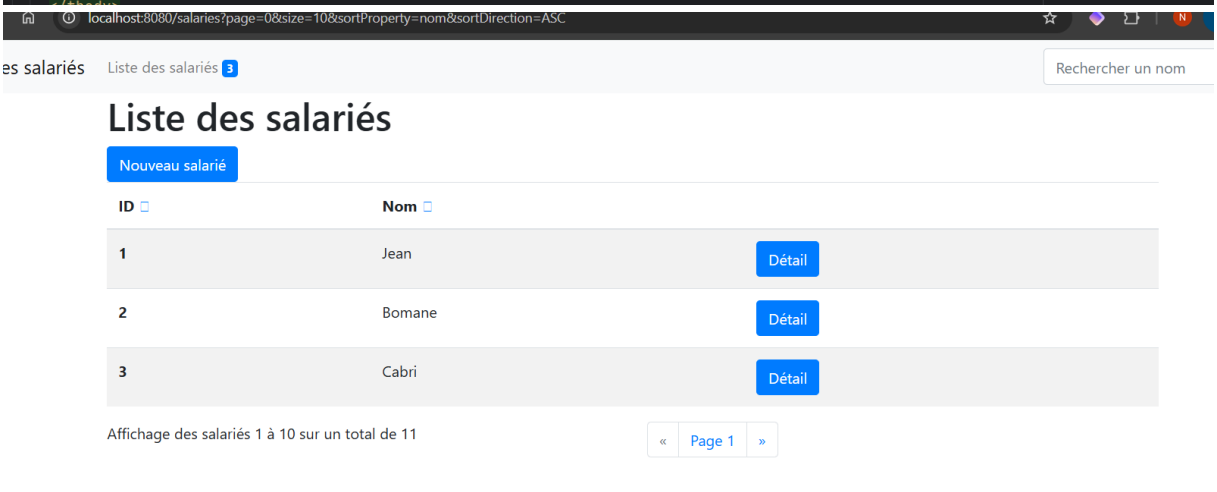
```
lariés <span class="badge badge-primary" th:text="${nbSalarie}">26</span></a></li>
```



Détail du salarié

```
@GetMapping(value = "/salaries")  no usages
public String listSalarie(final ModelMap model) {
    model.put("nbSalarie", salarieService.countSalaries());
    model.put("salaries", salarieService.getSalaries());
    return "list";
}
```

```
<tbody>
<tr th:each="salarie : ${salaries}">
  <th th:text="${salarie.id}" scope="row">1</th>
  <td th:text="${salarie.nom}">Jeanne Dupond</td>
  <td><a th:href="@{/salaries/{id}(id=${salarie.id})}" href="/salaries/aide/2" class="btn btn-primary ember-view">Détail</a></td>
</tr>

<!--<tr>
  <th scope="row">2</th>
  <td>Robert Martin</td>
  <td><a href="/salaries/aide/3" class="btn btn-primary ember-view">Détail</a></td>
</tr> robert n'a pas d'id '3' = erreur en cliquant sur détail -->
```



10. Modification d'un salarié

En allant à http://localhost:8080/salaries/1 (page de détail du salarié d'id 1), il est possible de modifier les informations du salarié d'identifiant 1 qui sont persistées en base de donnée lorsqu'on clique sur le bouton *Enregistrer*. L'URL qui est appelée est POST /salaries/1 avec les données du formulaire qui sont envoyées (NB. en API REST ce serait un PUT mais il s'agit de Controllers de pages web).

Développer ou enrichir le bon Controller pour réaliser cela, et côté web réutiliser detail_Salarie.html (comme le code du Controller et donc leurs URLs et les valeurs affichées dans le template HTML diffèrent dans les deux cas création et modification, il faut utiliser des expressions conditionnelles ternaires Thymeleaf pour supporter les deux à la fois). Une alternative permettant de se passer d'expressions conditionnelles ternaires Thymeleaf est de plutôt le copier vers new_Salarie.html.

Se rendre sur les p.44-45-55

```
@PostMapping(value = "/salaries/{id}")  no usages
public String editSalarie(SalarieAideADomicile salarie, @PathVariable Long id) throws SalarieException {
    SalarieAideADomicile compare = salarieService.getSalarie(id);
    if (Objects.equals(salarie.getId(), compare.getId())) {
        salarieService.updateSalarieAideADomicile(salarie);
    }
    return "redirect:/salaries/" + salarie.getId();
}
}
```

```
<form id="saveForm" method="post" class="row" th:action="${salarie.id} == null ? @{/salaries/save} : @{/salaries/{id}(id=${salarie.id})}" th:object="${salarie}" ac

    <div class="col-lg-6">
        <div class="form-group">
            <label class="form-control-label" for="id">ID</label>
            <input type="text"  th:value="*{id} != null ? *{id} : '' " value="10" class="form-control" name="id" id="id" disabled="true">

            <label class="form-control-label" for="nom">Nom</label>
            <input type="text" th:value="${salarie.nom} != null ? ${salarie.nom} : ''" value="Jean Dupont" class="form-control" name="nom" id="nom">

            <label class="form-control-label" for="moisEnCours">moisEnCours</label>
            <input type="date" th:value="${salarie.moisEnCours} != null ? ${salarie.moisEnCours} : ''" value="03/2023" class="form-control" name="moisEnCours"
```

1	Jean	Détail
2	Bomane	Détail
3	Cabri ENE	Détail

Affichage des salariés 1 à 10 sur un total de 11

« Page 1 »

Détail du salarié

ID

2

joursTravaillesAnneeN

0,0

Nom

Bomane Ella

congesPayesAcquisAnneeN

0,0

moisEnCours

04/02/2026

moisDebutContrat

01/07/2021

joursTravaillesAnneeNMoins1

10,0 j

congesPayesAcquisAnneeNMoins1

1,0 j

congesPayesPrisAnneeNMoins1

0,0 j

Erreur dans un champ !

Enregistrer Supprimer

ID	Nom	
1	Jean	Détail
2	Bomane Ella	Détail
3	Cabri ENE	Détail

Affichage des salariés 1 à 10 sur un total de 11

« Page 1 »

Gestion des salariés Liste des salariés 3

Détail du salarié

ID

joursTravaillesAnneeN

0,0

Nom

congesPayesAcquisAnneeN

0,0

moisEnCours

jj/mm/aaaa

moisDebutContrat

jj/mm/aaaa

joursTravaillesAnneeNMoins1

0,0 j

congesPayesAcquisAnneeNMoins1

0,0 j

congesPayesPrisAnneeNMoins1

0,0 j

Erreur dans un champ !

Enregistrer Annuler

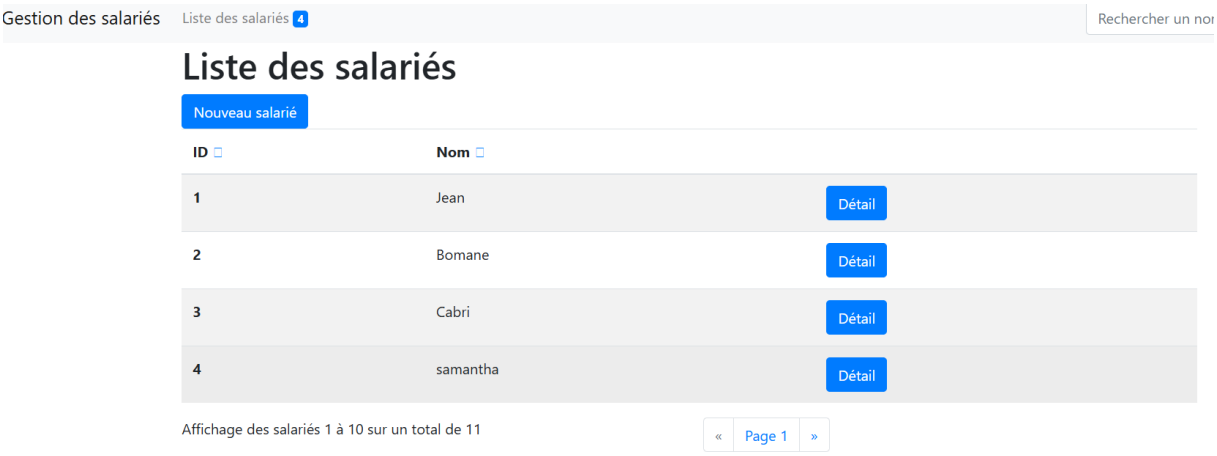
11. TP - Suppression d'un salarié

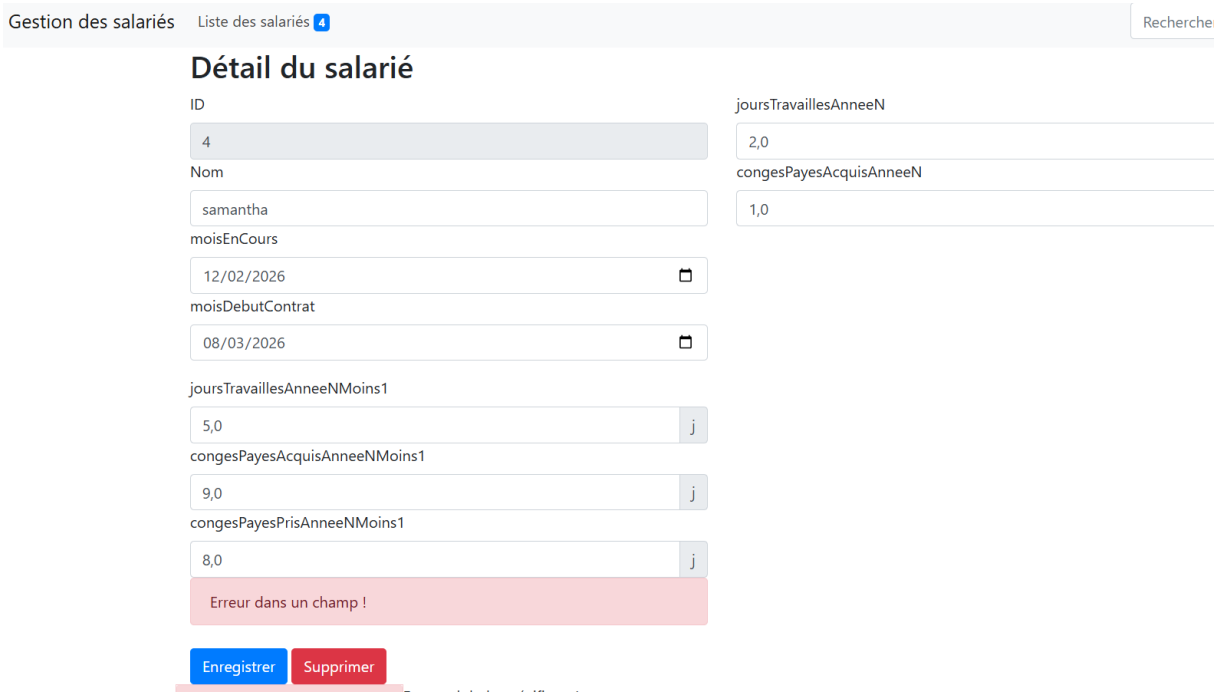
En allant à http://localhost:8080/salaries/1 (page de détail du salarié d'id 1), il est possible de supprimer le salarié affiché lorsqu'on clique sur le bouton *Supprimer*. L'appel qui est effectué est GET /salaries/1/delete (NB. c'est un GET pour simplifier cet exercice ; en API REST ce serait un PUT, mais il s'agit de Controllers de pages web donc selon l'usage ce devrait être un POST).

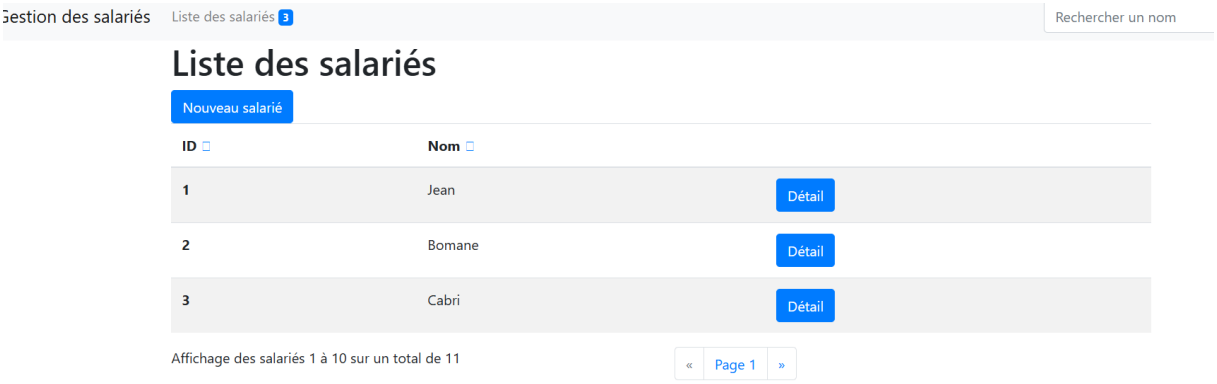
Se rendre sur les p.57

```java
@GetMapping("/salaries/{id}/delete")  no usages
public String deleteSalarie(@PathVariable Long id) throws SalarieException {
    salarieService.deleteSalarieAideADomicile(id);
    return "redirect:/salaries";
}
```

```html
    </div>
</div>
<div class="row mt-2">
    <div class="col-lg-6" th:switch="${salarie.id} != null">
        <input form="saveForm" class="btn btn-primary" type="submit" value="Enregistrer">
        <a href="/salaries/10/delete" th:href="@{/salaries/{id}/delete(id=${salarie.id})}" class="btn btn-danger" th:case="true">Supprimer</a>
        <a href="/salaries" class="btn btn-danger" th:case="false">Annuler</a>
    </div>
</div>
```







12. TP - Recherche par nom

Lorsqu'on recherche le nom *Jeanne Dupond* dans la barre de recherche, son salarié doit être affiché dans les résultats s'il est créé. L'URL auquel on accède est /salariés?nom=Jeanne Dupond. Lorsqu'on recherche un nom inexistant commme *123456*, on obtient une erreur 404. Développer ou enrichir le bon Controller pour cela, en utilisant la bonne méthode de SalarieAideADomicileService.

Se rendre sur les p.54-29

```
<form class="form-inline my-2 my-lg-0" role="search" action="http://localhost:8080/salaries" method="GET"
    <div class="form-group">
        <input name="matricule" class="form-control" placeholder="Rechercher un nom" type="text">
    </div>
    <button type="submit" class="btn btn-outline-success ml-2 my-2 my-sm-0">Rechercher</button>
</form>
```