Compte rendu de la réalisation des missions A, B et C pour la production attendue

Au sein de la mise en œuvre de la production attendue, les missions A, B, et C ont été abordées successivement pour atteindre les objectifs spécifiques du projet.

1. **Mission B : Intégration du nombre total de types d'armures dans le tableau d'administration**

   Pour débuter, la mission B visait à afficher le nombre total de types d'armures dans le tableau d'administration. L'implémentation de cette fonctionnalité a été initiée en établissant le code nécessaire. Le résultat de cette phase du développement peut être observé ci-dessous :

```
        </div>
    </td>
</tr>
<!-- Afficher le nombre total de types d'armures -->
<a>Nombre total de types d'armures : <span th:text="${#lists.size(typeArmures)}"></span></a>
</tbody>
able>


replace="mesFragments :: fragFooter"></footer>
```

Mission A : Choix d'une armure parmi la liste pour le joueur

La mission A consistait à permettre au joueur de choisir une armure parmi la liste complète. Pour ce faire, des modifications ont été apportées au contrôleur de Personnage, intégrant les éléments liés aux armures. La progression s'est ensuite dirigée vers la partie du template Personnage. Le résultat obtenu en HTML est présenté ci-dessous :

```
/** DAO pour l'accès aux données des armures. */
fun create(model: Model): String {
    val nouvellePersonnage = Personnage( id: null, nom: "", attaque: 1, defense: 1, endurance: 1, vitesse: 1, armeEquipee: null, armureEquipee: null)
    // Récupère les valeurs d'Armure
    val lesArmures = armureDao.findAll()

    model.addAttribute( attributeName: "nouvellePersonnage", nouvellePersonnage)
    // Ajoute les valeurs de Armure au nouveau personnage
    model.addAttribute( attributeName: "armures",lesArmures)

    return "joueur/personnage/create"
}
```

```
@GetMapping("/joueur/personnage/{id}/edit")
fun edit(@PathVariable id: Long, model: Model): String {
    val personnage = this.personnageDao.findById(id).orElseThrow()
    // Récupère les valeurs d'Armure
    val lesArmures = armureDao.findAll()

    model.addAttribute( attributeName: "personnage", personnage)
    // Ajoute les valeurs de Armure au nouveau personnage
    model.addAttribute( attributeName: "armures",lesArmures)
    return "joueur/personnage/edit"
}
```

```
personnageModifier.nom = personnage.nom
personnageModifier.attaque = personnage.attaque
personnageModifier.defense = personnage.defense
personnageModifier.endurance = personnage.endurance
personnageModifier.vitesse = personnage.vitesse
personnageModifier.armureEquipee = personnage.armureEquipee
personnageModifier.utilisateur = utilisateur
val savedPersonnage = this.personnageDao.save(personnageModifier)
redirectAttributes.addFlashAttribute( attributeName
return "redirect:/joueur/personnage"
```

```
val personnageModi
springaventure
```

```
    <!-- Select pour l'armure équipée -->
    <label for="armures" class="form-label">Armure</label>
    <select class="form-select" id="armures" th:field="${nouvellePersonnage.armureEquipee.id}">
        <option th:each="uneArmure : ${armures}" th:value="${uneArmure.id}" th:text="${uneArmure.nom}"></option>
    </select>

</div>
```

```
<!-- Ajoutez d'autres champs du formulaire si nécessaire -->
<div class="mb-3">
    <!-- Select pour l'armure équipée -->
    <label for="armures" class="form-label">Armure</label>
    <select class="form-select" id="armures" th:field="${personnage.armureEquipee.id}">
        <option th:each="uneArmure : ${armures}" th:value="${uneArmure.id}" th:text="${uneArmure.nom}"></option>
    </select>

</div>
```

Mission C : Affichage détaillé d'une potion sur une nouvelle page

Enfin, la mission C demandait la création d'une nouvelle page pour afficher en détail les informations d'une potion. Pour répondre à cette exigence, le fichier `show.html` de la potion a été créé. Voici le code correspondant :

```
<!DOCTYPE html>
<html lang="fr">
<head th:replace="mesFragments :: fragHead">
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<div class="min-vh-100">
    <nav th:replace="mesFragments :: fragNav"></nav>
    <main class="container">
        <h1>Détails de la potion</h1>
        <div>
            <p><strong>Nom : </strong> <span th:text="${potion.nom}"></span></p>
            <p><strong>Description : </strong> <span th:text="${potion.description}"></span></p>
            <p><strong>Description : </strong> <span th:text="${potion.soin}"></span></p>
        </div>
        <div>
            <a th:href="@{'/admin/potion/'+${potion.id}+'/edit'}" class="btn btn-warning">
                Modifier
            </a>
            <a th:href="@{'/admin/potion'}" class="btn btn-primary">
                Retour à la liste
            </a>
        </div>
    </main>

</div>
<footer th:replace="mesFragments :: fragFooter"></footer>
</body>
</html>
```