Gestion d'un annuaire

1 Introduction

Le but de cette page est de présenter le projet qui va servir de base à votre évaluation dans cette unité d'enseignement.

2 Cahier des charges

L'objectif du mini projet est simple : gérer à l'aide de la technologie JEE un annuaire de personnes. Plus précisément,

Lot 1 : présentation.

- Une personne est représentée par un ensemble d'informations : identifiant, nom, prénom, adresse électronique, site WEB, date de naissance et mot de passe.
- Chaque personne est placée dans un groupe. Un groupe est composé de quelques dizaines de personnes (par exemple les étudiants du M1 IDL 2019/2020). Un groupe a donc un nom et un identifiant.
- L'application doit présenter une liste de groupes, une liste de personnes de chaque groupe et une vue détaillée de chaque personne (sauf adresse électronique et date de naissance). Une fonction de recherche doit être offerte.
- L'application doit être fonctionnelle si nous avons plusieurs milliers de personnes et plusieurs centaines de groupes (**il faut le montrer**).

Lot 2 (si le lot 1 est terminé) : authentification et modification.

- L'application doit permettre à chaque personne de modifier sa propre description.
- Les personnes présentes dans l'annuaire peuvent avoir accès à toutes les informations (y compris les adresses électroniques et les dates de naissance).

Lot 3 (si le lot 2 est terminé) :

- Il faut prévoir un mécanisme de récupération du mot de passe.

3 Étape 1 : Stockage (DAO/Spring/JPA)

Objectifs :

- Conception et création de la base de données.
- Conception et création des *JavaBeans* permettants de représenter les données de l'annuaire.
- Utilisation de la technologie Spring/JPA pour gérer la persistance des *JavaBeans*.

Architecture : Vous devez mettre en place une couche de service d'accès aux données (souvent appelée DAO pour *Data Access Object*), Cette couche est constituée d'une interface (**indépendante du choix de JPA**) et d'une implantation (liée à JPA). Cette interface **pourrait** ressembler à ceci (**ce n'est qu'une très vague proposition**) :

```java
public interface IDirectoryDao {

   // récupérer les groupes
   Collection<Group> findAllGroups();

   // lire une personne
   Person findPerson(long id);

   // lire un groupe et ses personnes
   Group findGroup(long id);

   // modification ou ajout d'une nouvelle personne
   void savePerson(Person p);

   // modification ou ajout d'un nouveau groupe
   void saveGroup(Group g);

   ...
   }
```

Contrainte à respecter :

- Vous devez **absolument** prévoir une classe de test unitaire *Junit* pour valider les méthodes offertes par votre implantation.
- Vos composants (DAO, DataSource, tests JUnit) doivent **absolument** être connectés et paramétrés par une couche Spring-boot.
- Vous devez **absolument** prévoir une phase de préparation de la base de données pour insérer des données cohérentes (prévoir un service de peuplement pour générer les milliers de personnes).

4 Étape 2 : Metier (Spring)

Cette étape consiste à mettre en place une couche métier basée sur Spring. Cette couche devra prendre en charge l'accès et la modification des données métier. Elle assure

- l'authentification des personnes,
- la modification des données personnelles,
- l'accès aux données.

L'interface de cette couche métier **pourrait** ressembler à ceci (**ce n'est qu'une proposition**) :

```java
public interface IDirectoryManager {

    // rendre un utilisateur anonyme
    void newUser(User user);

    // chercher une personne
    Person findPerson(User user, long personId);

    // chercher un groupe
    Group findGroup(User user, long groupId);

    // identifier un utilisateur
    boolean login(User user, long personId, String password);

    // oublier l'utilisateur
    void logout(User user);

    // enregistrer une personne
    void savePerson(User user, Person p);
}
```

5 Étape 3 : Présentation (JSP/SpringMVC)

Cette étape consiste à mettre en place une application WEB basée sur la technologie **Spring MVC/JSP** qui respecte les contraintes suivantes :

- Votre application WEB sera construite autour du framework Spring MVC.
- Les réponses seront construites par des pages JSP. Dans la mesure du possible utilisez les balises de contrôle de la JSTL (boucles, tests et affichage).
- Faites en sorte que votre application produise des pages HTML 5 valides accompagnées d'un framework CSS.
- **Votre application WEB doit vérifier toutes les requêtes utilisateur avant de les exécuter.**

6 Échéances

Ce projet est à rendre, par équipe de **deux personnes**, pour le **12 avril 2023 au soir**.

Documents attendus : une archive ZIP (de la forme `nom1-nom2.zip`) qui regroupe

- un cahier des charges (fichier `CdC.pdf` une ou deux pages),
- un rapport (fichier `rapport.pdf` de moins de 10 pages),
- le projet de l'application WEB (répertoire `projet`) au format maven,
- votre application packagée sur la forme d'un fichier WAR basé sur **SpringBoot** (fichier `annuaire.war`),

Points importants :

- **Point 1 :** Votre application doit utiliser une base de données **embarquée et stockée en mémoire** (voir exemple sur le TP Spring Boot) afin qu'elle soit facile à tester.
- **Point 2 :** Dès la deuxième page, le rapport doit donner clairement des informations d'authentification afin que je puisse utiliser votre application.
- **Point 3 :** La forme du rendu doit **ABSOLUMENT être respectée**. Les travaux sont à rendre sur la

plateforme AMETICE [1] (le lien est valide).

- **Point 4 :** Choisissez un créneau en modifiant en ligne le fichier du planning [2]. Ne laissez pas de créneau vide avant le votre et ne modifiez pas les créneaux déjà choisis.

1. ref:ametice
2. https://amubox.univ-amu.fr/s/3Ta9DoqszpzmgqE