Projet MediaWeb

Le projet Mediatheque dont le thème a été abordé en AppServJava – période B - a donné satisfaction et il vous est désormais demandé de réaliser des éléments d'une application JavaEE sur le même thème. La **médiathèque** propose des **documents** (livres, DVDs, CDs) et l'application web sera utilisée principalement par les abonnés pour emprunter/rendre des documents et les bibliothécaires pour ajouter de nouveaux documents. Documents et utilisateurs de l'application seront maintenus sur une BD relationnelle.

Déploiement - mise en œuvre dans la médiathèque

Le serveur JavaEE support applicatif de votre application web sera installé au sein du réseau local de la médiathèque (intranet) et les postes de la médiathèque seuls y auront accès via une adresse IP locale - si ça n'était pas le cas, un abonné pourrait emprunter ou retourner un document depuis n'importe quel poste équipé d'un navigateur web, par exemple chez lui, ce qui serait absurde.
Un jeu d'essai minimum sera installé dans la BD.

Abonnés/bibliothécaires : les utilisateurs

L'accès aux données de la médiathèque suppose une authentification (login, mot de passe). A partir de là, un utilisateur ***abonné*** pourra emprunter ou rendre un document, un utilisateur ***bibliothécaire*** pourra ajouter un document. Ces 2 catégories seront les ***utilisateurs*** de l'application web. L'ajout/modification/suppression d'utilisateurs par votre application n'est pas à envisager.

Découplage et (non-)dépendance : services – mediatek2022 -persistance

L'architecture de votre application permet de mettre en œuvre un découplage strict entre trois modules :

- les services : le package ***services*** contient toutes les classes servant aux échanges client/serveur http, donc toutes les servlets ;
- la médiathèque : le package ***mediatek2022*** contient le domaine (Mediatheque, Document, Utilisateur) ainsi qu'une interface PersistantMediatheque qui permet de ne pas être couplé au modèle de persistance (technique d'inversion de dépendance) ;
- persistance : le package ***persistance***, comme son nom l'indique, représente les éléments de persistance des données ; on y trouvera donc le code JDBC et la classe « point d'entrée » de ce package, MediathequeData, implémentant l'interface mediatek2022.PersistantMediatheque. On trouvera également ici les classes implémentant mediatek2022.Document.

Le package ***mediatek2022*** vous est fourni compilé et opérationnel sous forme d'un fichier *mediathek.jar*. Ce code compilé est donc <u>stable</u> et respecte bien le non-couplage du package ***mediatek2022*** aux 2 autres.

Si A→B signifient que A est couplé à B

services → mediatek2022 ← persistance

Vous écrirez les parties ***services*** et ***persistance***, sachant que *persistance.MediathequeData* doit s'auto-déclarer à *mediatek2022.Mediatheque* par **injection de dépendance** – ce qu'elle fait dans son bloc static. Un embryon de MediathequeData qui effectue cette injection vous est fourni.

A rendre

Le projet est à réaliser par binômes ou seul – au sein du bigroupe uniquement, aucun trinôme ne sera accepté ni binôme inter-bigroupes. Une soutenance/recette sera effectuée lors de la dernière séance de tp et un fichier zippé sera rendu par dépôt dans le puits en fin de semaine. Vous pouvez développer les services sous forme de servlets ou de JSP (dans ce cas, pas de package ***services****).*

Dans ce projet, le cahier des charges implémenté dans le package mediatek2022 et la technique utilisée pour garantir le découplage sont contractuels, donc :

Il est strictement interdit de faire la moindre modification à l'intérieur de *mediatek2022*.
Aucun couplage ne doit apparaître dans votre code entre *services* et *persistance*

Un non respect de l'une de ces deux règles vaudra zéro au projet.