Projet JAVAEE : Mediatek

Rapport de projet

Bounezou Elyes

Groupe 205

Ye Alexandre

Groupe 205

DUT 2ème Année

Table des matières

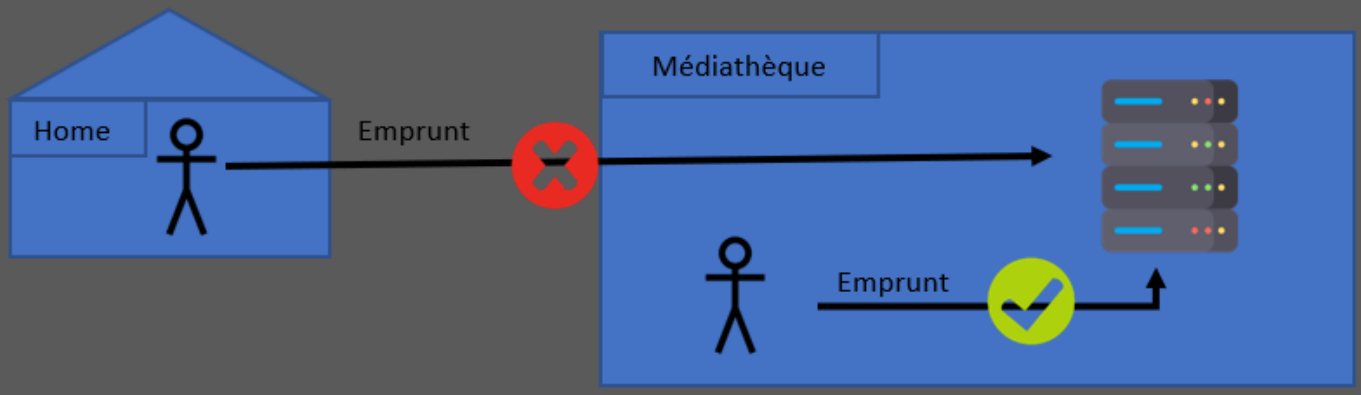
Introduction

L'objectif de ce projet était de réaliser une médiathèque en ligne, dynamique avec des documents enregistrés dans une base de données et des utilisateurs, ayant un rôle dans la médiathèque. Tout le backend de ce projet était à réaliser en Java afin d'apprendre à diriger et réaliser une application JavaEE.

Il y a différents utilisateurs dans le projet : les abonnés et les bibliothécaires :

- Les abonnés peuvent se connecter à la médiathèque, s'ils ont déjà un compte au sein de la médiathèque et peuvent alors consulter tous les documents disponibles (c'est-à-dire les documents empruntables) de la médiathèque ainsi que sa liste de documents personnels, c'est-à-dire les documents qu'il a déjà empruntés. Parmi les documents disponibles il peut choisir d'emprunter n'importe quel document et possède des infos sur le nom de l'auteur, le titre du document, le type de document (CD,DVD,LIVRE). Les abonnés peuvent également se déconnecter et cela les redirige vers la page de connexion.
- Les bibliothécaires peuvent ajouter des documents et n'ont pas accès à l'espace Abonné et leur espace se résume à entrer le type de document, l'auteur et le titre et cela ajoute le document dans la base de données pour qu'il puisse être emprunté par les abonnés de la médiathèque.

Le projet est hébergé sur serveur, en l'occurrence Apache-Tomcat, installé au sein de la médiathèque : donc seuls les postes de la médiathèque y auront accès via une adresse IP locale et donc un utilisateur ne pourra pas interagir avec la médiathèque en dehors de celle-ci.



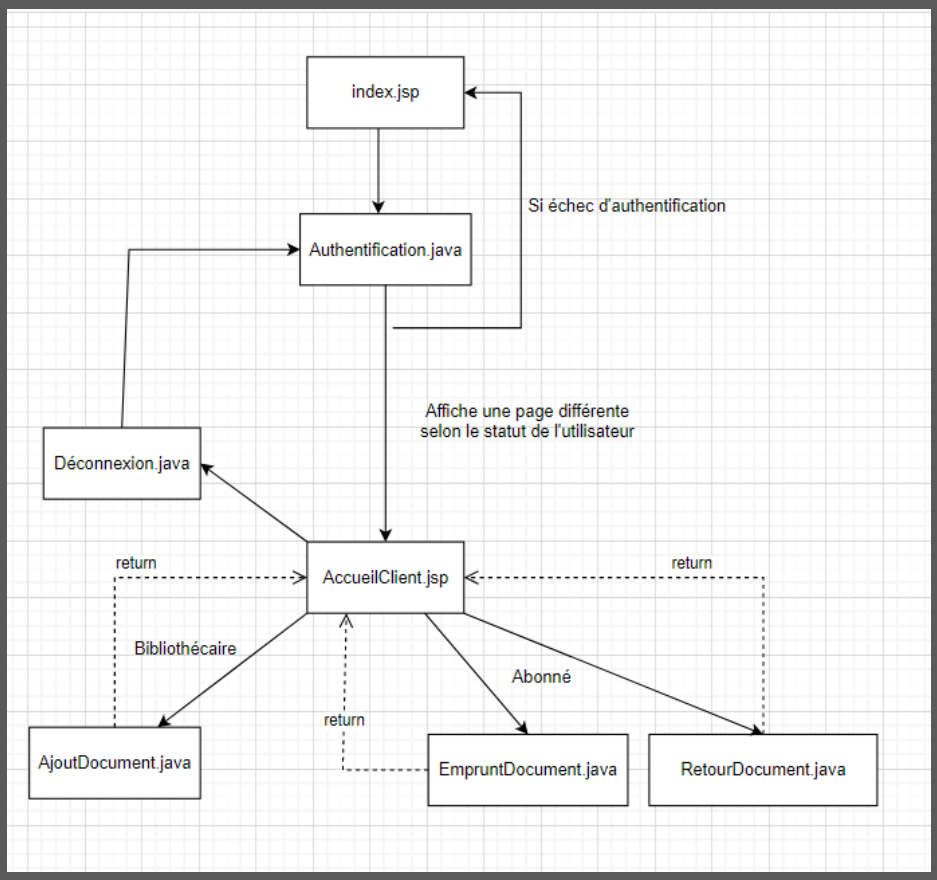
Nous avons décidé de réaliser ce projet sur IntelliJ à l'aide de Maven, un outil qui permet de déployer des applications Java. En ce qui concerne la base de données, c'est une BDD MySql déployée sur PhpMyAdmin directement intégrée au sein du projet (on retrouve la BDD sur l'IDE grâce à l'outil « Database »).

Au sein du projet, tout est déclaré dans différents packages, qui ont chacun leur propre fonction. Le package « persistance » (là où on retrouve principalement le code SQL) n'interagit pas avec le package « services ». Les deux sont reliés au package principal de l'application « mediatek2022 » qui contient la classe principale « Médiathèque » qui fonctionne comme un singleton pour être sur que la médiathèque n'est déclarée qu'une unique fois et qu'il n'y ait bien qu'une instance de la médiathèque.

Le package « mediatek2022 » a été ajouté en JAR au sein du projet dans les librairies externes.

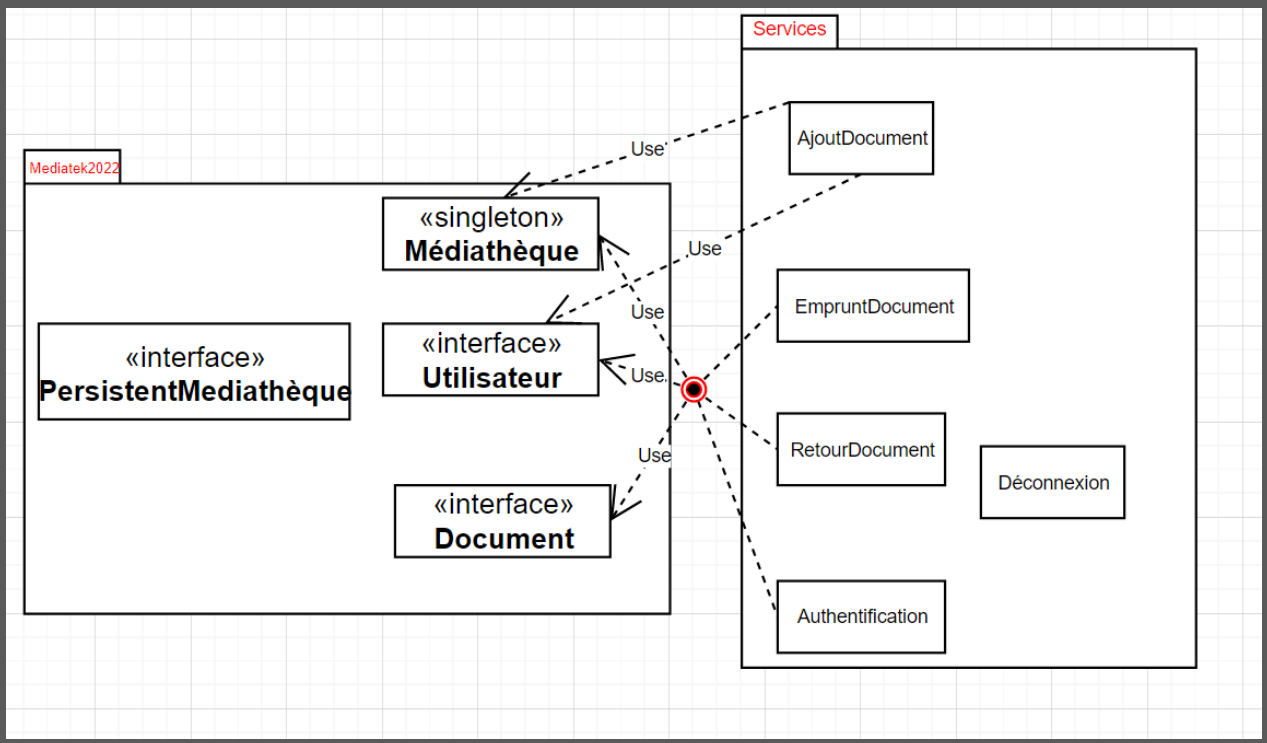
Structuration de code, découplage et injection de dépendance

Voici notre diagramme qui présente comment sont indexées les pages JSP et comment elles interagissent avec les servlets correspondantes.

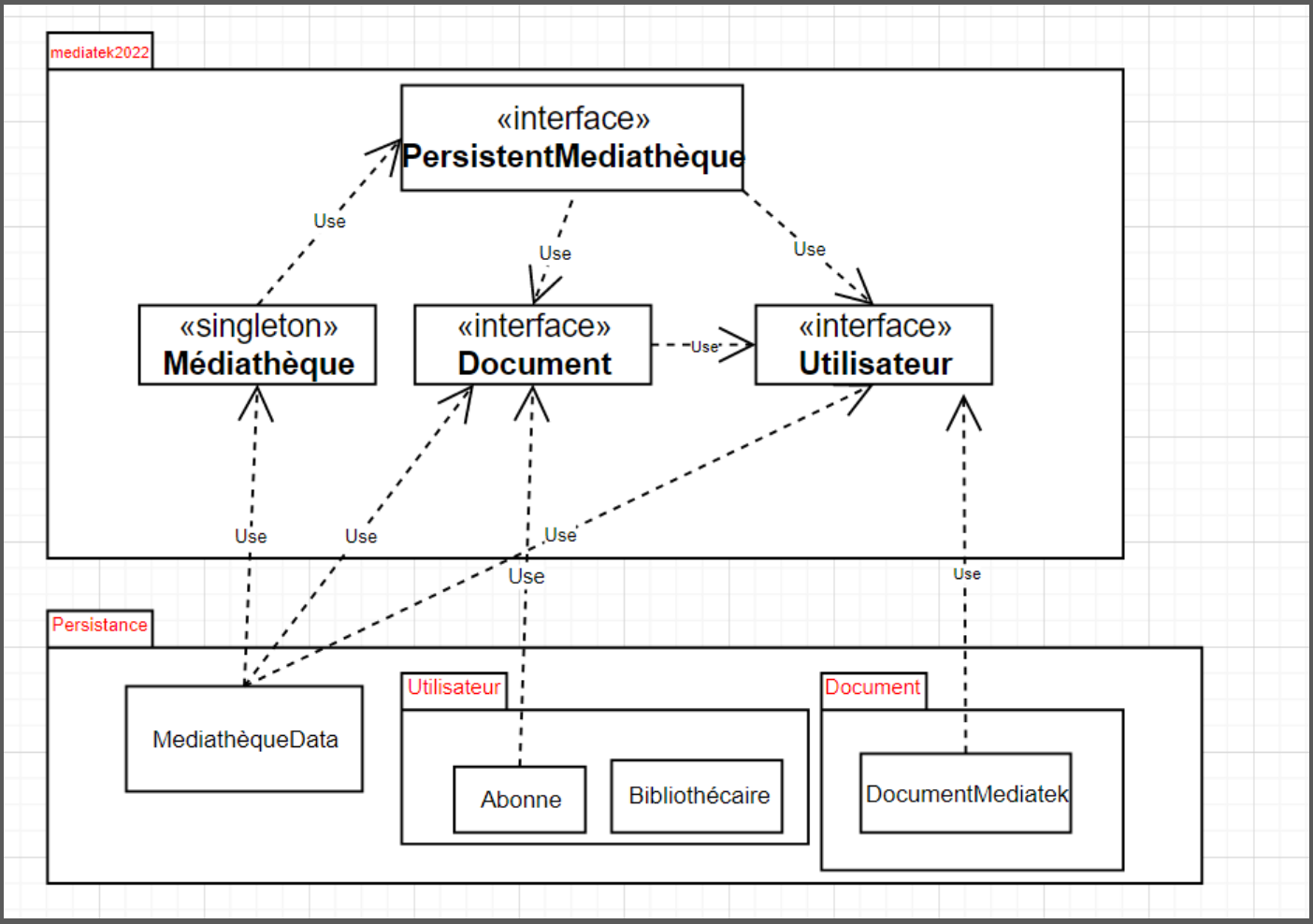


Ensuite par rapport au package « services » celui ne peut, selon la consigne, pas avoir de relations avec le package persistance et ne peut avoir de relations qu'avec le package « mediatek2022 ».

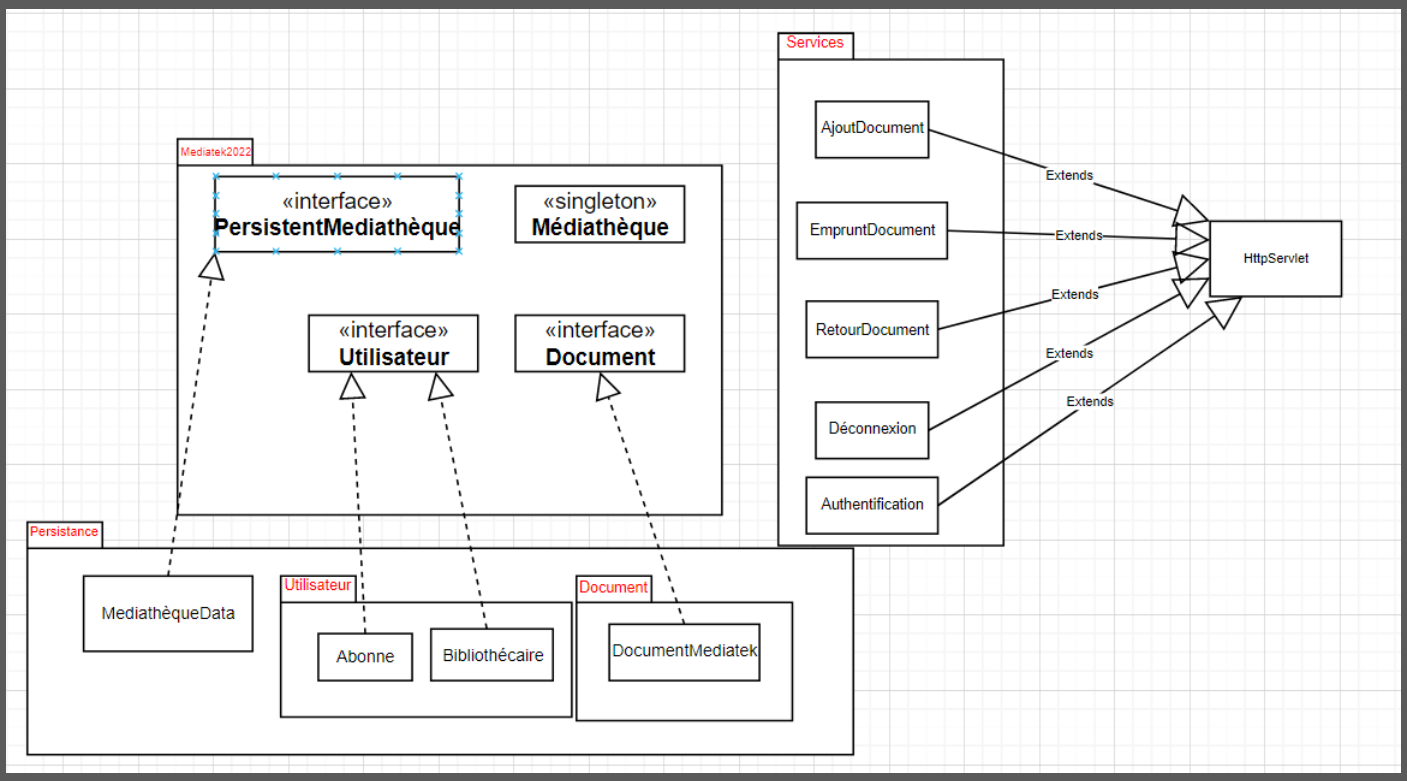
Ainsi voici notre diagramme qui montre les relations entre le package « mediatek2022 » et « Services »



Ensuite voici notre diagramme qui montre les relations entre le package « persistance » et « mediatek2022 » et qui permet également de schématiser les relations au sein du package « mediatek2022 »



Et enfin un aperçu mais qui aurait été illisible avec toutes les flèches, donc nous avons décidé de séparer ca en 3 diagrammes ce qui rend le diagramme plus compréhensible et lisible



Utilisation de Servlet et JSP

Pour cette application, nous avons décidé d'utiliser des servlets afin de pouvoir récupérer les informations et s'occuper du traitement des données de l'utilisateur. Tout ce qui concerne l'affichage sera réalisé en JSP, à l'aide de Framework tels que Bootstrap pour rendre la page plus esthétique et plus visuelle. On récupère toutes les informations de l'utilisateur puis on envoie à l'aide de formulaires (grâce au « action » du formulaire) les données vers les servlets qui vont s'occuper de traiter les informations.

Une fois les informations traitées dans la servlet, on redirige ensuite ces informatiques vers la page JSP grâce au RequestDispatcher après avoir crée des attributs de session et attributs de requêtes. On stocke l'utilisateur dans un attribut de session et pour ainsi accéder à son nom et son boolean « isBibliothecaire » pour pouvoir différencier l'affichage en fonction de ce boolean.

En effet, en ce qui concerne l'accueil d'un utilisateur, tout se fait dans une même page, puis, en fonction du boolean « isBibliothecaire », on propose un formulaire d'ajout de document si c'est un bibliothécaire ou alors 2 tableaux HTML qui stockent les documents de l'utilisateur et les documents disponibles afin qu'il puisse, soit rendre le document, soit en emprunter.

Nous avons 5 services à implémenter soit 5 servlets qui sont les suivantes :

- Authentification : qui permet de vérifier si un utilisateur qui se connecte est bien présent dans la base de données, sinon on le renvoie vers la page d'accueil avec un message d'erreur pour le prévenir que nous n'avons pas pu l'authentifier.
- EmpruntDocument : qui permet de faire le traitement de l'emprunt d'un document par un utilisateur. On recupère l'ID du document à emprunter, on l'envoie à travers un formulaire avec un input invisible et on exécute la fonction emprunt de la médiathèque en récupérant l'utilisateur courant et on le redirige vers l'accueil utilisateur.
- RetourDocument : qui permet de faire le traitement de retour d'un document. On récupère, comme pour l'emprunt l'id du document à retourner en l'envoyant à travers un formulaire et un input invisible et on exécute la fonction retour de la médiathèque et on le redirige vers l'accueil utilisateur.
- AjoutDocument : qui concerne le bibliothécaire et qui lui permet d'ajouter un Document dans la médiathèque. On fait en sorte qu'il soit obligé de remplir tous les champs du formulaire et on lui impose les 3 types : CD, DVD, Livre. Il n'a plus qu'à entrer l'auteur et le titre du document.
- Deconnexion : qui permet de faire la déconnexion d'un utilisateur. Lorsqu'on appuie sur le bouton déconnexion, cela redirige vers la servlet « Deconnexion » dans laquelle on utilise la fonction invalidate() sur la session. On est ensuite redirigé vers le formulaire de connexion.

Ensuite, on a 2 pages HTML/JSP :

- Index.jsp : qui possède un formulaire de connexion et qui permet à l'utilisateur de s'authentifier.

- AccueilClient.jsp : qui s'occupe de l'affichage en fonction du rôle de l'utilisateur (abonné ou bibliothécaire comme dit plus haut). Il affichera soit les documents disponibles ainsi que les documents de l'abonné (si c'est un abonné) soit les documents qu'il a empruntés ou alors, un formulaire d'ajout de documents, si c'est un bibliothécaire.
  Espace abonné :

MEDIATEK2022 BIENVENUE DANIYAL ESPACE ABONNÉ DECONNEXION

Bienvenue dans l'espace abonné, vous pouvez consulter les documents disponibles et les emprunter ou bien retourner un de vos documents empruntés.

Voici tous les documents proposés dans la médiathèque.

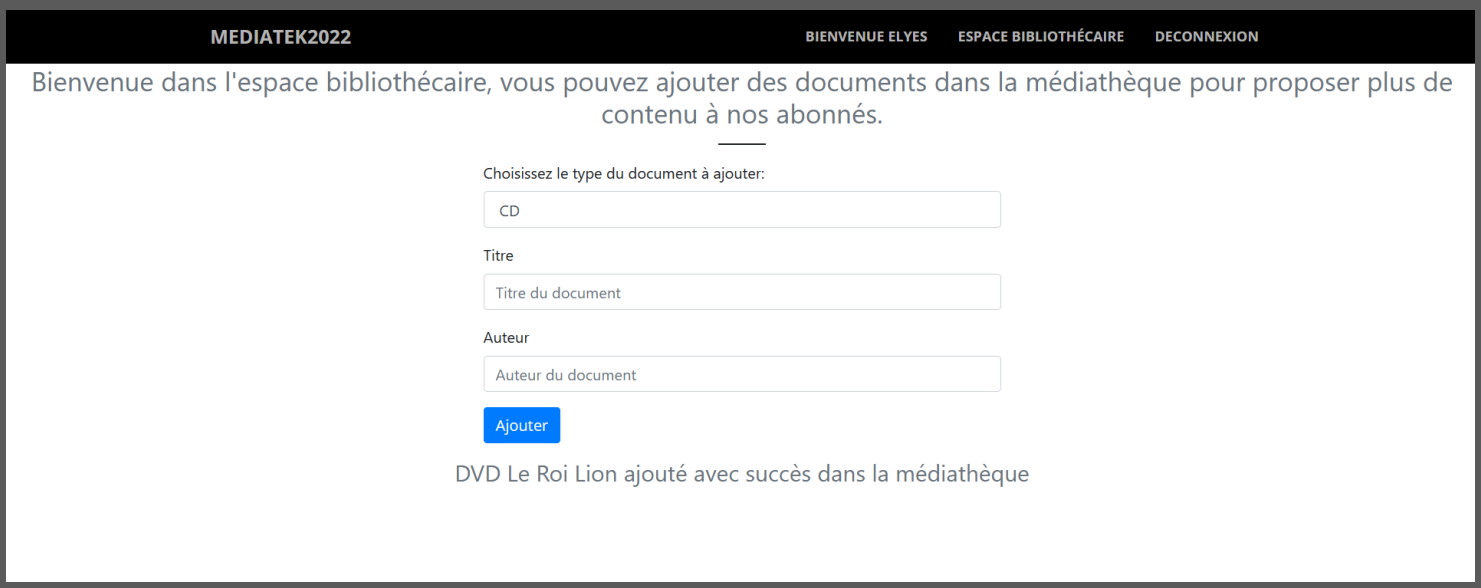
#	Titre	idDocument	Disponible	idEmprunteur	Type Document	Auteur	Emprunter
0	Cendrillon	39	true	0	DVD	Clyde Geronimi	Emprunter
1	Spider-Man : No Way Home	40	true	0	DVD	Jon Watts	Emprunter
2	Les 101 Dalmatiens	41	true	0	DVD	Wolfang Reitherman	Emprunter
3	Au revoir la-haut	42	true	0	Livre	Pierre Lemaitre	Emprunter
4	Les fleurs du Mal	43	true	0	Livre	Charles Baudelaire	Emprunter
5	Peter Pan	46	true	0	DVD	Hamilton Luske	Emprunter

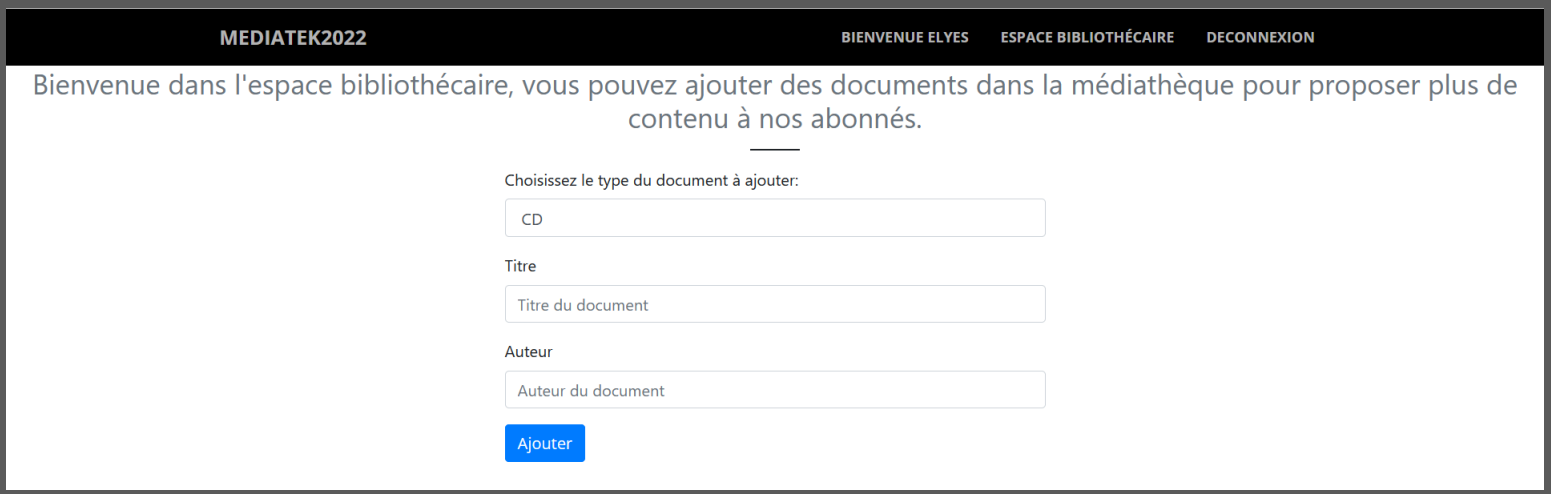
8	Meteora	51	true	0	CD	Linkin Park	Emprunter

Voici la liste de vos documents

#	Titre	idDocument	Disponible	idEmprunteur	Type Document	Auteur	Emprunter
0	Le Roi Lion	38	false	2	DVD	Rob Minkof	Retourner
1	Madame Bovary	44	false	2	Livre	Gustave Flaubert	Retourner
2	Germinal	45	false	2	Livre	Emile Zola	Retourner
3	Thriller	47	false	2	CD	Michael Jackson	Retourner
4	The Eminem Show	50	false	2	CD	Eminem	Retourner

Espace Bibliothécaire :





On a préféré utiliser JSP et Servlet car d'un point de vue logique, cela a plus sens que de faire de l'affichage et servlet ou alors du traitement en JSP.

Transformation Objet-relationnel

Une base de données MySql a été créée sur PhpMyAdmin afin de pouvoir répondre aux besoins de l'application. On a donc une table utilisateur et une table document :

- La table utilisateur contient : l'id de l'utilisateur (auto-increment), l'age, le boolean bibliothecaire, son login et son password. Cette table contient une contrainte unique sur le login et l'id de l'utilisateur est une clé primaire de la table
- La table document contient : l'id du document (auto-increment), le titre du document, le boolean disponible, le type du document, le nom de l'auteur et l'idUtilisateurEmprunt qui est l'id de l'utilisateur qui a emprunté le document. La clé primaire de cette table est l'id du document, il y a également une contrainte unique sur le triplet (titre du document, type du document, auteur). L'idUtilisateur emprunt est une clé étrangère qui va référence l'id de l'utilisateur dans la table utilisateur.

Ainsi, une fois cette architecture implémentée et cette base de données créée, il fallait alors faire la connexion entre le projet et la base de données. On a donc récupéré le connecteur MySql et on l'a implémenté dans le projet, dans MédiathèqueData afin de pouvoir faire des échanges et des requêtes avec la base de données.

Ensuite pour pouvoir faire les requêtes SQL nécessaires pour l'utilisateur et pour le document, nous avons fait des classes implémentant les interfaces : DocumentMediatek qui implémente Document, Abonne et Bibliothecaire qui implémentent Utilisateur.

Afin d'éviter les relations entre le package « persistance » et « services » on fonctionne seulement avec les interfaces Document et Utilisateur du package principal « mediatek2022 ».

Enfin pour faire ces requêtes SQL on utilise principalement des PreparedStatement qui permet de précompiler le code SQL puis on exécute la requête SQL avec soit executeUpdate() ou executeQuery(). En même qu'on fait le traitement dans la base de données MySql on fait le traitement au sein du projet. Par exemple pour la fonction tousLesDocumentsDisponibles() de la médiathèque, la requête SQL sera la suivante :

`SELECT * FROM document where disponible = true`, on récupère le résultat de la requête avec un ResultSet puis on ajoute dans une ArrayList déjà instanciée un nouveau DocumentMediatek avec les attributs récupérés du ResultSet. Grâce au ResultSet on peut enfin récupérer le résultat de chaque colonne de notre table utilisateur et document (exemple : int id = resultSet.getInt(« idDocument ») puis on construit à l'aide du résultat des colonnes notre objet.

C'est ainsi que dans tout le projet on arrive à passer d'un objet de l'application à une table et aux attributs de la base de données.

Variables sessions

Dans l'application, il y a des cas où nous avons besoin des attributs de requête, donc des attributs qui contiennent des informations propres à une requête (par exemple dans le cas ou rend un document, lorsque l'on passe l'id du document à rendre, celui-ci est stocké dans un attribut de requête, lors de la prochaine requête cet id du document ne sera pas le même), mais nous avons également des variables de session qui stockent les informations durant toute connexion de l'utilisateur.

Dans notre cas, la variable de session est créée lorsque l'utilisateur se connecte à la médiathèque, seulement s'il est reconnu et s'il est stocké dans la base de données. Donc notre variable de session contient l'Utilisateur, c'est-à-dire que l'on stocke une instance de l'objet

Utilisateur dans notre variable de session. On peut ainsi accéder à son nom, son tableau d'objets (dans lequel on sa liste de documents), son boolean isBibliothecaire et cette variable de session est maintenue tout au long de la connexion et des actions de l'utilisateur pour l'emprunt et le retour (si c'est un abonné), l'ajout d'un document (si c'est un bibliothécaire).

```
HttpSession session = request.getSession(true);
Utilisateur u = data.getUser(login,password);
session.setAttribute("profil",u);
```

Enfin cette variable de session est supprimée, mise à null, lorsque l'utilisateur se déconnecte de l'application, grâce à un bouton de déconnexion dans lequel on invalidate() qui permet de supprimer la session courante du registre et donc un nouvel utilisateur pourra se connecter et une nouvelle session lui sera assignée.

Concurrence

Pour ce qui concerne les problèmes de thread-safety et de concurrence des données de l'application, nous avons pensé que les sections critiques se trouvaient au niveau des requêtes SQL que l'on exécute dans l'application donc pour régler ce problème de concurrence, on encercle nos exécutions de requêtes SQL avec un synchronized sur l'attribut connection de la classe Connection déclarée en attribut static.

Notre ressource partagée est bien la connection puisqu'elle concerne à chaque fois un utilisateur.

Exemple :

```
private static Connection connection;
@Override
public Document getDocument(int numDocument) {
    PreparedStatement req = null;
    try {
        synchronized (connection) {
            req = connection.prepareStatement("select * from document where idDocument
= ?");
            req.setInt(1, numDocument);
            ResultSet result = req.executeQuery();
            if (!result.next()) return null;
            return new DocumentMediatek(
                    result.getString("titreDocument"),
                    result.getInt("idDocument"),
                    result.getBoolean("disponible"),
                    result.getString("typeDocument"),
                    result.getString("auteur"),
                    result.getInt("idUtilsateurEmprunt"));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }

    return null;
}
```

Cela permet notamment à ce que 2 utilisateurs ne réservent pas le même document en même temps ou alors de ne pas réserver un document qu'un utilisateur vient tout juste de réserver.

Efficacité des requêtes d'accès à la base de données

On se connecte à la base de données grâce au driver MySql grâce à la fonction Class.forName() et on se connecte avec l'objet Connection à la base de données au port 3306 sur la BD nommée « mediatek2022 » qui contient donc nos 2 tables Utilisateur et Document.

Toutes les requêtes SQL sont effectuées dans le package « persistance » et on récupère par la suite toutes les informations dans la BDD grâce à des PreparedStatement (plus efficace lorsque l'on doit faire des même requêtes plusieurs fois et que le code ne doit pas être exécuté qu'une fois donc en utilisant des preparedStatement, on a des requêtes plus efficace et un temps d'exécution plus court si on a beaucoup de données à traiter) qui comme son nom l'indique, sont des requêtes préparées, soit des requêtes déjà précompilées, utiles si on a besoin d'exécuter le même code plusieurs fois.

On a décidé de travailler sur une BDD MySql car nous avons déjà réalisé des projets avec celle-ci grâce à l'aide de PhpMyAdmin et donc nous connaissions déjà bien la syntaxe des requêtes SQL.

Bilan du Projet

Ce projet a été pour nous une très bonne expérience, c'était pour nous notre première application en JavaEE nous donnant donc un aperçu de ce qu'est la programmation Java dans le monde professionnel. On a pu découvrir le Java sous une autre forme que celle que l'on voit habituellement. Une des grosses surprises a été de découvrir la simplicité de la création du projet avec IntelliJ qui proposait déjà pas mal de fonctionnalités et qui proposait également la mise en place de la base de données.

On a également rencontré certaines difficultés comme par exemple la relation servlet/JSP. On ne savait pas comment créer ce lien entre les 2 mais une fois la logique acquise, il suffisait de faire la même chose partout et de plus il nous suffisait également de mettre en place ce qui a été vu en TP et durant les amphis de JavaEE.

Maven nous a grandement aidé, cela nous a permis de comprendre la compilation du projet et la mise en place des dépendances nécessaires mais encore une fois, notre IDE IntelliJ s'occupait de tout.

On aurait aussi pu également faire en sorte d'éviter les redondances du projet et pourquoi pas créer une classe Session qui s'occupe elle-même de gérer la session d'un utilisateur. Au niveau des JSP on aurait pu faire en sorte de faire un code plus esthétique également et faire un peu plus de design mais nous avons décidé de miser plus sur le back, coté Java. On aurait également plus ajouté plus de fonctionnalités sur la location d'un document, par exemple proposer un âge minimum etc… mais nous avons décidé de ne pas aller trop loin pour ne pas se perdre et de respecter les consignes du projet.

En conclusion, ce projet fut très enrichissant, qui nous a permis de découvrir le Java d'une façon différente, du Java utilisé en entreprise et qui pourra sûrement nous servir pour l'avenir.