Documentation technique

Contents

Fraisseix César
Rio Gwenaël
De Carvalho Alexandre

M1 MIAGe
Web Services

Technologies utilisées :

Outil de Développement :

Intellij Idea 2017.3.4 : IDE (Integrated Development Environment) Java développé par JetBrains.

Apache Tomcat : Le projet a été testé sous Tomcat 7.0 & 9.0. Celui-ci permet d'utiliser un serveur http pour tester et utiliser notre application Web.

Librairies utilisées :

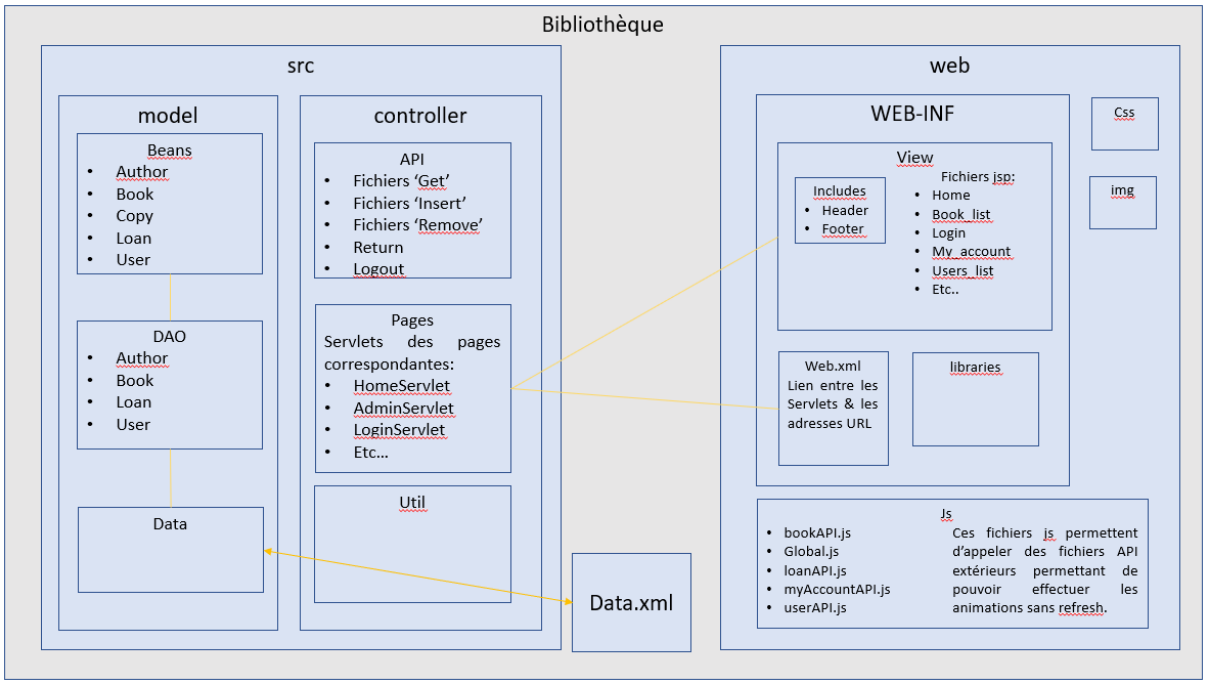
Les librairies utilisées sont stockées dans le répertoire web/WEB-INF/librairies. Les principales librairies utilisées sont :

Javax.json : Nous permet de convertir nos beans en String JSON pour communiquer entre le serveur et le client.

Jstl : Nous permet d'utiliser de manipuler les fichiers jsp plus simpement : tags de structures, requêtes, etc...

XStream : Nous permet de convertir nos beans en XML pour communiquer avec notre fichier data.xml

Architecture globale de l'application



Configurations

Pour se connecter : les identifiants/mdp par défaut sont admin/admin. Ils donnent un accès à un compte de type Administrateur à partir duquel il sera possible de remplir et de gérer la bibliothèque.

Tous les comptes administrateurs sont également des clients et peuvent donc utiliser la bibliothèque comme tel.

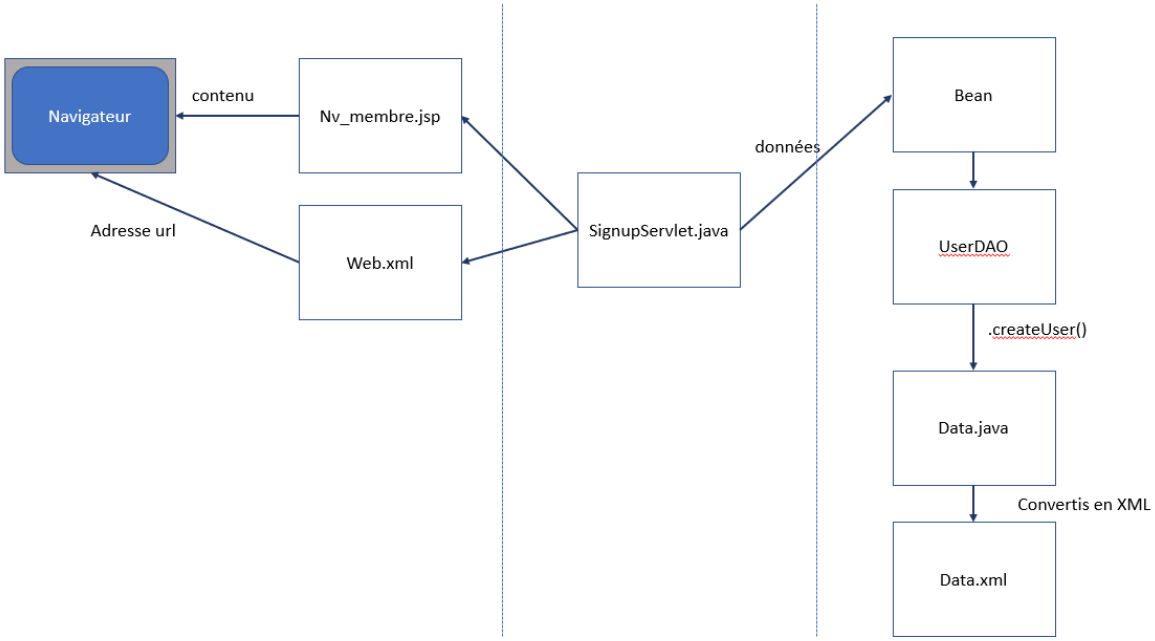
Le transfert de données du navigateur jusqu'à la base de données :

Voici deux exemples qui permettent de comprendre les deux approches utilisées pour concevoir le workflow de l'application.
Le premier ne fait pas la différence entre l'envoi des données et l'envoi des pages, tout est fait dans la même servlet. Pour récupérer d'autres données, il faut donc effectuer une nouvelle requête synchrone.

Le second tire parti de l'approche par API et des requêtes asynchrone d'ajax. Nous permettons d'abord à l'utilisateur de charger la page et les scripts en interrogeant une première servlet puis en fonction de ses actions nous allons interroger d'autres servlets pour récupérer des données spécifique et modifier la page en fonction.

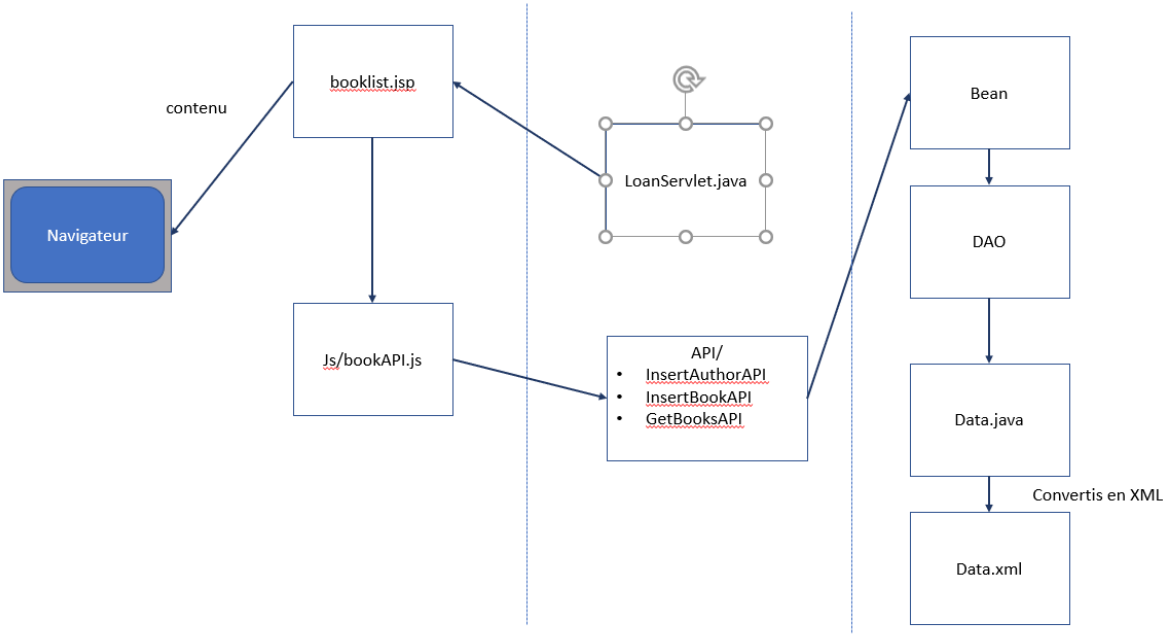
Nous utilisons principalement cette dernière méthode, plus moderne et flexible du point de vue de l'expérience utilisateur.

L'inscription :



La Servlet SignupServlet utilise le fichier nv_membre.jsp pour afficher le formulaire. Le fichier Web.xml (/web/WEB-INF/web.xml) relie le Servlet à l'adresse url. Les données rentrées dans le formulaire sont traitées dans la Servlet. La Servlet appelle la construction d'un java Bean (User Bean) avant de l'intégrer à la base de données grâce à l'usage d'une méthode de la classe UserDAO.

L'emprunt :



Dans ce cas, les traitements de données ne sont pas directement effectués dans le Servlet mais dans des fichiers API créés à l'occasion. Ceux-ci permettent d'effectuer les opérations sans avoir à réactualiser la page. Ainsi, chaque procédure est reliée à une page API, et chaque page API est reliée grâce au fichier Web.xml à une adresse url. Cela permet d'avoir une application plus fluide et efficace.

La Base de Données

Les données sont sauvegardés dans le fichier data.xml. Il faudra configurer son chemin d'acces via l'option suivante au lancement de la VM :

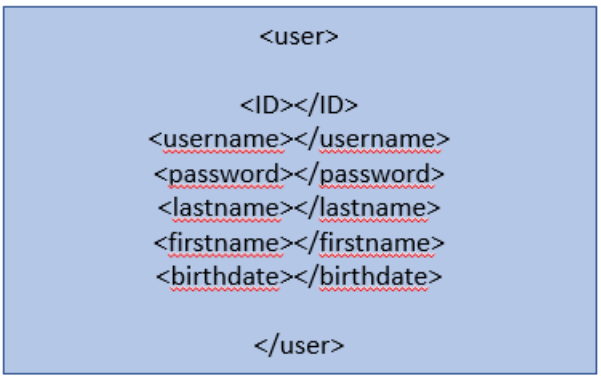
-Duser.dir=<Le chemin ou je souhaite sauvegarder mes données>

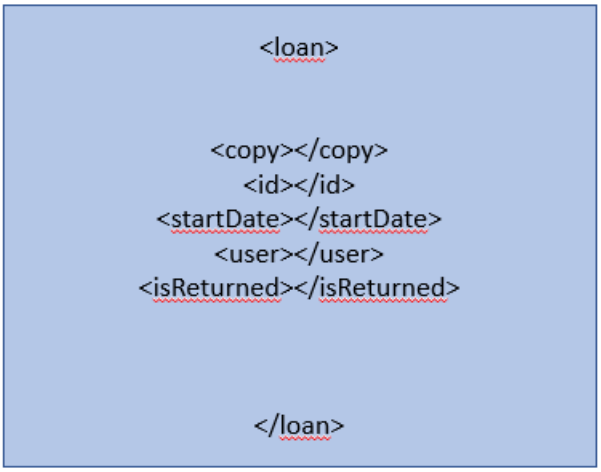
Veillez simplement à choisir un chemin où vous avez les droits en lecture et en écriture.

La base de données est de format XML. Elle est manipulée à travers le fichier Data.java (dossier src/model), ce fichier lui-même étant manipulé par les fichiers DAO.

La structure du fichier est de la forme :


<book>
<ID>
</ID>
<Author>
<ID></ID>
<lastname></lastname>
<firstname></firstname>
<naissance></naissance>
<nationalite></nationalite>
</Author>
<titre>
</titre>
<copies>
<ID></ID>
<bookID></lbookID>
</copies>
</book>


<user>
<ID></ID>
<username></username>
<password></password>
<lastname></lastname>
<firstname></firstname>
<birthdate></birthdate>
</user>


<loan>
<copy></copy>
<id></id>
<startDate></startDate>
<user></user>
<isReturned></isReturned>
</loan>