RAPPORT PROJET JEE

CREATION D'UN SYSTEME DE GESTION DES EMPRUNTS DANS UNE BIBLIOTHEQUE

Réalisé par :
Yao Yann Vianney BOHOUSSOU

Sous la direction de :
M. Mahmoud Elhamlaoui

Année universitaire
2015-2016

Table des figures

Table des matières

Introduction

Dans le cadre du cours de développement avec JEE, nous avons été emmenés à réaliser des projets afin d'évaluer notre maitrise des outils auxquels nous avons été initiés. Dans cette mesure, j'ai décidé de réaliser un système de gestion des emprunts de livres dans une bibliothèque. Ce rapport présente donc les détail sur le projet, l'analyse et la conception et enfin la réalisation.

Chapitre 1

Spécifications

1.1 Spécifications fonctionnelles

Le système a 2 utilisateurs principaux :
- les employés
- l'administrateur

1.1.1 Les employés

Un employé doit pouvoir :
- se connecter au système
- ajouter, modifier et supprimer des clients
- afficher la liste des livres
- afficher la liste des clients
- enregistrer des emprunts de clients sur place

Un employé peut aussi être un client.

1.1.2 Administrateur

l'administrateur doit pouvoir :
- faire tout ce que peut faire un employé
- ajouter, modifier et supprimer des employés
- ajouter, modifier des livres
- afficher la liste des employés

Etant donné qu'un livre peut avoir été emprunté plusieurs fois, la suppression d'un livre revient à mettre son nombre en stock à 0.

1.2 Spécifications techniques

Le projet sera réalisé avec les outils suivants :
- Jsf
- Spring
- Hibernate
- Apache shiro
- Maven

Le système doit utiliser des cookies http pour gérer les connexion des utilisateurs ainsi que des bean oriemtés message pour envoyer des alertes aux clients lorsqu'il y a de nouveaux livres ou lorsque leur délai d'emprunt est dépassé.

Chapitre 2

conception

2.1 Diagramme de cas d'utilisation

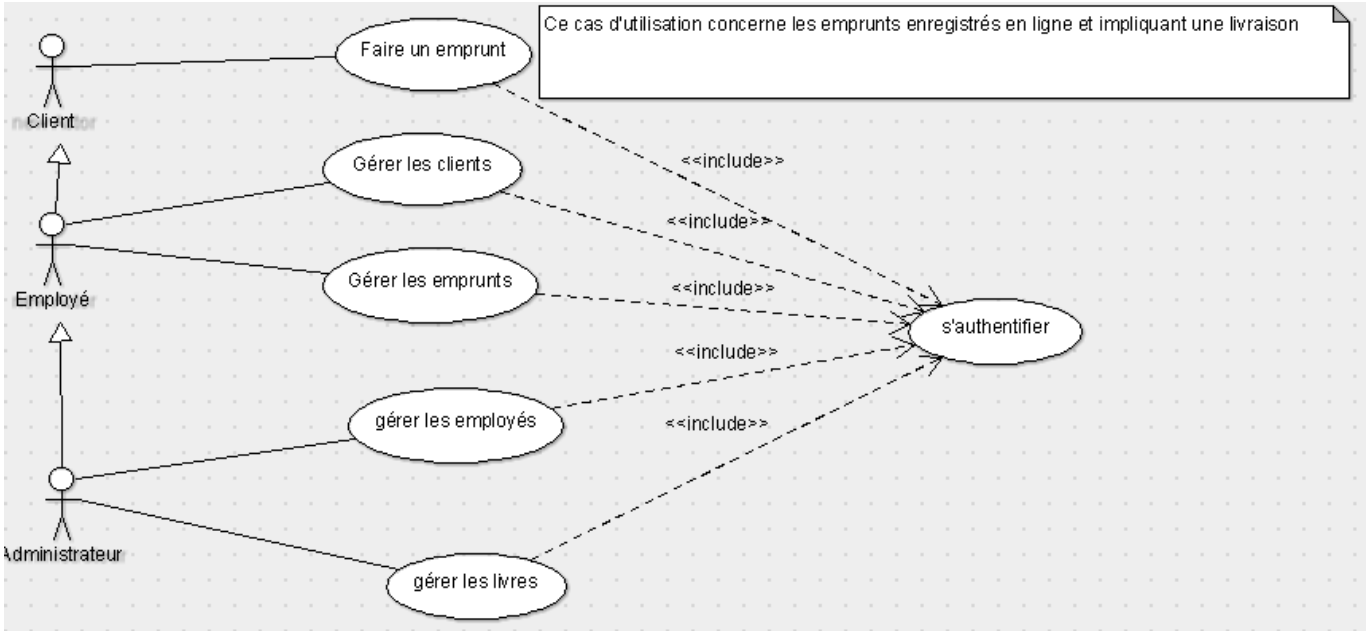

FIGURE 2.1 – Diagramme de cas d'utilisation

2.2 Diagramme de classes

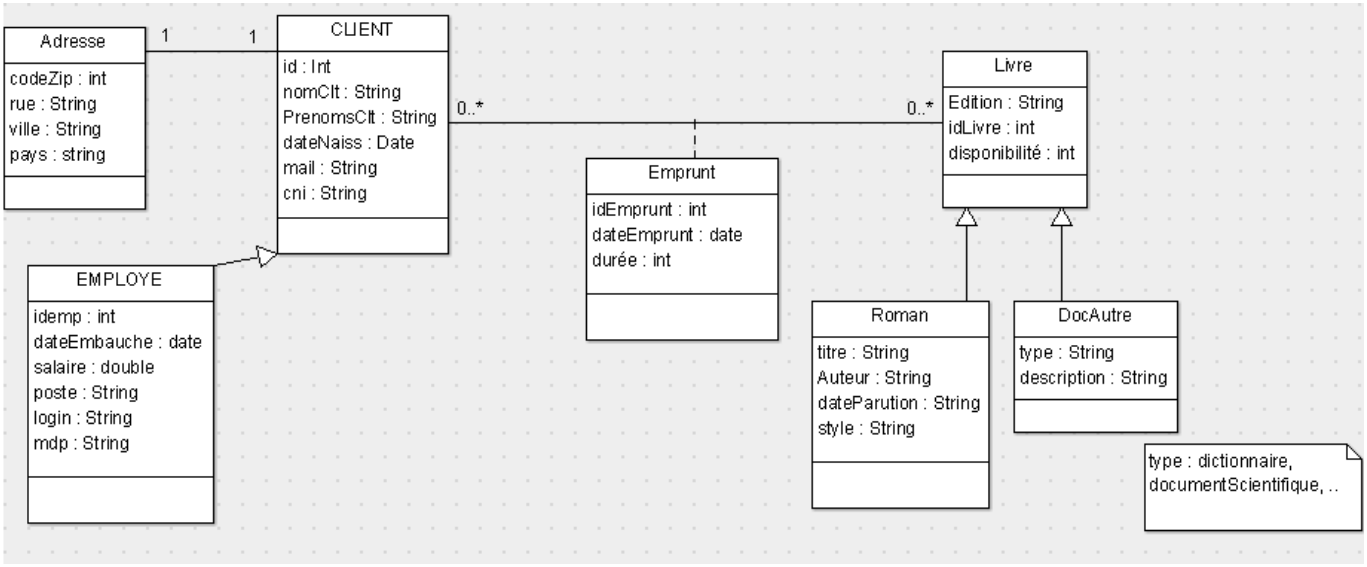

FIGURE 2.2 – Diagramme de classes

Chapitre 3

Réalisation

3.1 Travail réalisées

L'ensemble des spécifications fonctionnelles mentionnées au début de ce rapport ont été remplies. L'apect traitement avec les managed beans et les ejb, les vues avec les JSF et la persistence en utilisant le framework hibernate oont été réalisés. Cependant, les spécifications techniques n'ont pas toutes été remplies. En effet, l'aspect sécurité en utilisant apache shiro et l'aspect envoi de message en utilisant les beans orientés message n'ont pas été réalisés. Aussi le framework Spring n'a pas été utilisé.

3.2 Outils utilisés

3.2.1 Eclipse

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Son objectif est de produire et fournir des outils pour la réalisation de logiciels, englobant les activités de programmation (notamment environnement de développement intégré et frameworks) mais aussi d'AGL recouvrant modélisation, conception, test, gestion de configuration, reporting... Son EDI, partie intégrante du projet, vise notamment à supporter tout langage de programmation à l'instar de Microsoft Visual Studio.

Bien qu'Eclipse ait d'abord été conçu uniquement pour produire des environnements de développement, les utilisateurs et contributeurs se sont rapidement mis à réutiliser ses briques logicielles pour des applications clientes classiques. Cela a conduit à une extension du périmètre initial d'Eclipse à toute production de logiciel : c'est l'apparition du framework Eclipse RCP en 2004.

Figurant parmi les grandes réussites de l'Open source, Eclipse est devenu un standard du marché des logiciels de développement, intégré par de grands éditeurs logiciels et sociétés de services. Les logiciels commerciaux Lotus Notes 8, IBM Lotus Symphony ou WebSphere Studio Application Developer sont notamment basés sur Eclipse.

FIGURE 3.1 – logo Eclipse

3.2.2 Wildfly

WildFly, anciennement JBoss Application Server ou JBoss, est un serveur d'applications Java EE Libre écrit en Java, publié sous licence GNU LGPL. étant écrit en Java, WildFly peut être utilisé sur tout système d'exploitation fournissant une machine virtuelle Java (JVM). Le nom JBoss est aujourd'hui utilisé pour JBoss EAP, produit dérivé WildFly et faisant l'objet d'un support commercial. WildFly implémente entièrement l'ensemble des services Java EE. Il embarque :

- Tomcat : serveur web Tomcat pour exécuter les parties servlets et JSP des applications déployées sur le serveur ;
- JBoss Portal (en) : framework de portail ;
- JBoss Seam (en) : framework web ;
- Hibernate : framework de persistance ;
- jBPM : moteur de workflow ;
- Drools (ou JBoss Rules) : système de gestion de règles métier.

FIGURE 3.2 – logo Wildfly

3.2.3 Maven

Apache Maven est un outil pour la gestion et l'automatisation de production des projets logiciels Java en général et Java EE en particulier. L'objectif recherché est comparable au système Make sous Unix : produire un logiciel à partir de ses sources, en optimisant les tâches réalisées à cette fin et en garantissant le bon ordre de fabrication.

Il est semblable à l'outil Ant, mais fournit des moyens de configuration plus simples, eux aussi basés sur le format XML. Maven est géré par l'organisation Apache Software Foundation. Précédemment Maven était une branche de l'organisation Jakarta Project.

Maven utilise un paradigme connu sous le nom de Project Object Model (POM) afin de décrire un projet logiciel, ses dépendances avec des modules externes et l'ordre à suivre pour sa production. Il est livré avec un grand nombre de tâches pré-définies, comme la compilation de code Java ou encore sa modularisation.

Un élément clé et relativement spécifique de Maven est son aptitude à fonctionner en réseau. Une des motivations historiques de cet outil est de fournir un moyen de synchroniser des projets indépendants : publication standardisée d'information, distribution automatique de modules jar. Ainsi en version de base, Maven peut dynamiquement télécharger du matériel sur des dépôts logiciels connus. Il propose ainsi la synchronisation transparente de modules nécessaires.

Maven1 et Maven2 ont été développés en parallèle mais les versions futures seront basées sur la structure de la deuxième version.

FIGURE 3.3 – logo maven

3.2.4 Enterprise java bean

Enterprise JavaBeans (EJB) est une architecture de composants logiciels côté serveur pour la plateforme de développement Java EE.

Cette architecture propose un cadre pour créer des composants distribués (c'est-à-dire déployés sur des serveurs distants) écrit en langage de programmation Java hébergés au sein d'un serveur applicatif permettant de représenter des données (EJB dit entité), de proposer des services avec ou sans conservation d'état entre les appels (EJB dit session), ou encore d'accomplir des tâches de manière asynchrone (EJB dit message). Tous les EJB peuvent évoluer dans un contexte transactionnel.

De la version 1.0 à la version 2.1, un EJB était accompagné d'un ou plusieurs fichiers de déploiement écrit en XML qui permettait au serveur applicatif de déployer correctement l'objet au sein d'un conteneur. C'était notamment dans ces fichiers de déploiement que le développeur avait la possibilité de préciser le cadre transactionnel dans lequel l'objet allait s'exécuter. Depuis la version 3.0, le modèle EJB utilise le principe d'annotation java (meta-données) pour spécifier toute la configuration et les propriétés transactionnelles de l'objet. Le fichier de code source de l'EJB se suffit à lui-même.

C'est le serveur applicatif qui est chargé de la création, la destruction, la passivation ou l'activation de ses composants en fonction des besoins. Le client via un appel RMI (ou une de ses dérivées) va rechercher un EJB par son nom logique JNDI et appeler une ou plusieurs méthodes de cet objet.

3.2.5 Hibernate

Hibernate est un framework open source gérant la persistance des objets en base de données relationnelle.

Hibernate est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un développement client lourd, que dans un environnement web léger de type Apache Tomcat ou dans un environnement Java EE complet : WebSphere, JBoss Application Server et Oracle WebLogic Server.

Hibernate apporte une solution aux problèmes d'adaptation entre le paradigme objet et les SGBD en remplaçant les accès à la base de données par des appels à des méthodes objet de haut niveau.

FIGURE 3.4 – logo hibernate

3.3 Structure du projet

3.3.1 Côté Serveur

Le projet comprend 3 packages :
- metier : Contient les principales classes représentant la logique métier décrite dans le diagramme de classe réalisé durant la phase de conception.
- beans : Contient les beans de sessions qui implémentent les opérations métier telles que l'ajout, la suppression ...
- Dao : Contient les classes qui sechargent d'utiliser le framework hibernate pour accéder à la base de données.

3.3.2 Côté Client

Il s'agit ici d'une application web réalisée selon le modèle MVC et appelant les méthodes des ejb de la partie serveur. Il est composé principalement des pages JSF pour l'administrateur et l'employé et des managed beans qui appellent les éthodes de la partie serveur.

3.3.3 La base de données

La base de données est composée de 4 tables :
- adresse : Qui contient les adresses des clients et employés
- client : Qui contient toute la hierarchie d'héritage client employé
- Livre : Qui contient toute la hierarchie d'héritage client Livre, Roman, Docautre
- Emprunt : Qui contient les différents emprunts réalisés.

Le script de création de la base de données est fourni en annexe.

3.4 Problèmes rencontrés

L'éssentiel des problèmes rencontrés étaient des problèmes de configuration de l'aspect persistence du projet. En effet, Il manquait un datasource pour que les dao puissent aoir accès à la base de données. Il a donc fallu ajouter manuellement un datasource et un driver à wildfly afin de pouvoir accéder à la base

de données . La méthode utilisée pour ajouter le driver et le datasource est fournie sur le lien suivant http ://giordanomaestro.blogspot.com/2015/02/install-jdbc-driver-on-wildfly.html.

Conclusion

Pour conclure, nous pouvons dire que ce projet a été très bénéfique dans la mesure où il nous a éffectivement permis de mieux maitriser le développement JEE via un cas pratique, mais aussi de découvrir de nouvelles technologies telles que apache shiro.

Annexe A

Script de création de la BD

```
    create table Adresse (
id integer ,
codezip varchar(50),
rue varchar(50),
ville varchar(50),
pays varchar(50),
constraint pk_key_adr PRIMARY KEY (id)
);

    create table Client
(
id integer ,
nomClt varchar(50),
prenomsClt varchar(100),
dateNaissance date,
mail varchar(50),
cni varchar(50),
idAdr int,
dateEmbauche date,
salaire FLOAT,
poste varchar(50),
login varchar(50),
mdp varchar(50),
typeClt varchar(50) not null,
constraint pk_key_Client PRIMARY KEY(id),
constraint fk_key_Client_Adress FOREIGN KEY (idAdr) REFERENCES Adresse(id)
);

    create table Livre ( id integer ,
edition varchar(50) ,
nombrestock integer ,
titre varchar(50) ,
auteur varchar(100) ,
dateparution date ,
style varchar(50),
typedoc varchar(50) ,
description varchar(50) ,
typeLivre varchar(50),
constraint pk_key_livre PRIMARY KEY (id)
);
```

```
    create table Emprunt
(
id integer,
dateemprunt date,
duree integer,
idClient integer,
idLivre integer,
constraint pk_key_emprunt PRIMARY KEY (id),
constraint fk_key_emprunt_client Foreign key (idClient) references Client(id),
constraint fk_key_emprunt_livre Foreign key (idLivre) references Livre(id)
);
```

Bibliographie

[W1] *https ://fr.wikipedia.org/wiki/Apache_Maven.*

[W2] *https ://fr.wikipedia.org/wiki/Enterprise_JavaBeans.*

[W3] *http ://www.tutorialspoint.com/ejb/.*

[W4] *http ://www.tutorialspoint.com/hibernate/.*