Documentation Technique

"Candidature à un projet informatique"

Informations sur le document	
Nom du projet	**WebApp**
Groupe	LEGRAND Steve, LEVEQUE Matthieu, MAQUINGHEN Tanguy, RANAIVOSOA François
Type de document	docx
Version	1.0.0
Statut du document	Terminé

Suivi des modifications			
Version	**Date**	**Auteur**	**Description**
0.1.0	26/05/2018	RANAIVOSOA	Création du document
0.2.0	29/05/2018	LEGRAND, LEVEQUE, MAQUINGHEN	Rédaction du document

1.0.0	01/06/2018		Relecture

Table des matières

Objectif du document

Ce document est à destinations des utilisateurs qui auront pour rôle de maintenir à jour, sauvegarder et exploiter la solution. Il s'adresse aussi aux développeurs qui auront à charge de faire évoluer l'application. Il décrit l'ensemble des composants à installer et la configuration de l'environnement.

Prérequis logiciel

- Serveur TomCat v9
- Java JDK8 (minimum)

Installation

Le code source de l'application est disponible sur github à l'adresse suivante : https://github.com/RocalL/webApp. Il convient de l'importer dans un IDE approprié tel que Eclipse.

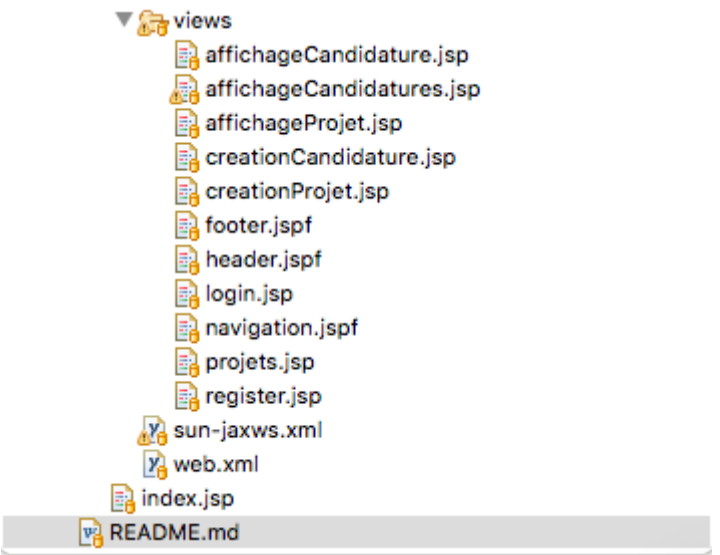
Sur la machine ou le serveur sur lequel l'application est installée, il est nécessaire de modifier le chemin d'accès au répertoire contenant la base de données. Pour ce faire, il faut modifier le fichier 'web.xml' qui se trouve au chemin suivant de l'arborescence du projet : webApp/WebContent/WEB-INF/web.xml .

La valeur à changer est celle contenue dans la balise <param-value>. Il est nécessaire de mettre le chemin absolu du fichier en entier depuis le disque.

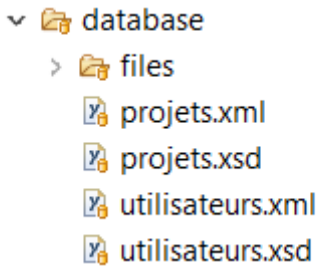
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://java.sun.com/xml/ns/javaee"
      xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
      version="3.0">
      <display-name>Candidatures</display-name>
      <welcome-file-list>
            <welcome-file>index.jsp</welcome-file>
      </welcome-file-list>
      <context-param>
            <param-name>localDirectoryPath</param-name>
            <param-
value>/Users/Tanguy/Documents/M1/S8/webService/webApp/WebContent/WEB-
INF/database</param-value>
      </context-param>
</web-app>
```

A l’attention de l’exploitant, deux identifiants de tests ont étaient pré-créés :
admin@root.fr root avec les permissions d’un administrateur
user@root.fr root avec les permissions d’un utilisateur lambda

Arborescence des fichiers



Structure de la base de donnée



La base de données est constituée de 4 fichiers + un répertoire files.

Le répertoire ‘files’ contient la liste des images uploadés par les utilisateurs lors de la création d’un projet.

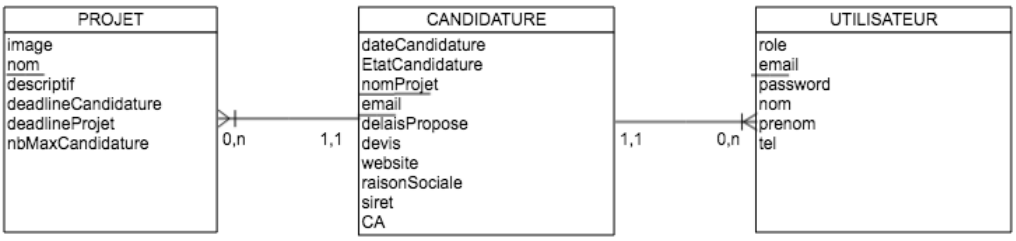
Le fichier ‘projets.xml’ contient les informations détaillées de tous les projets. Notamment, à chaque projet est associé une liste de candidature qui lui est propre.

Le fichier projets.xsd définit la structure et le type de contenu de projets.xml.

Le fichier utilisateurs.xml contient les données de la liste des utilisateurs.

Le fichier utilisateur.xsd définit la structure et le type de contenu de utilisateurs.xml

Voici un MERISE correspondant à l'organisation de nos données :



Structure de la liste des projets

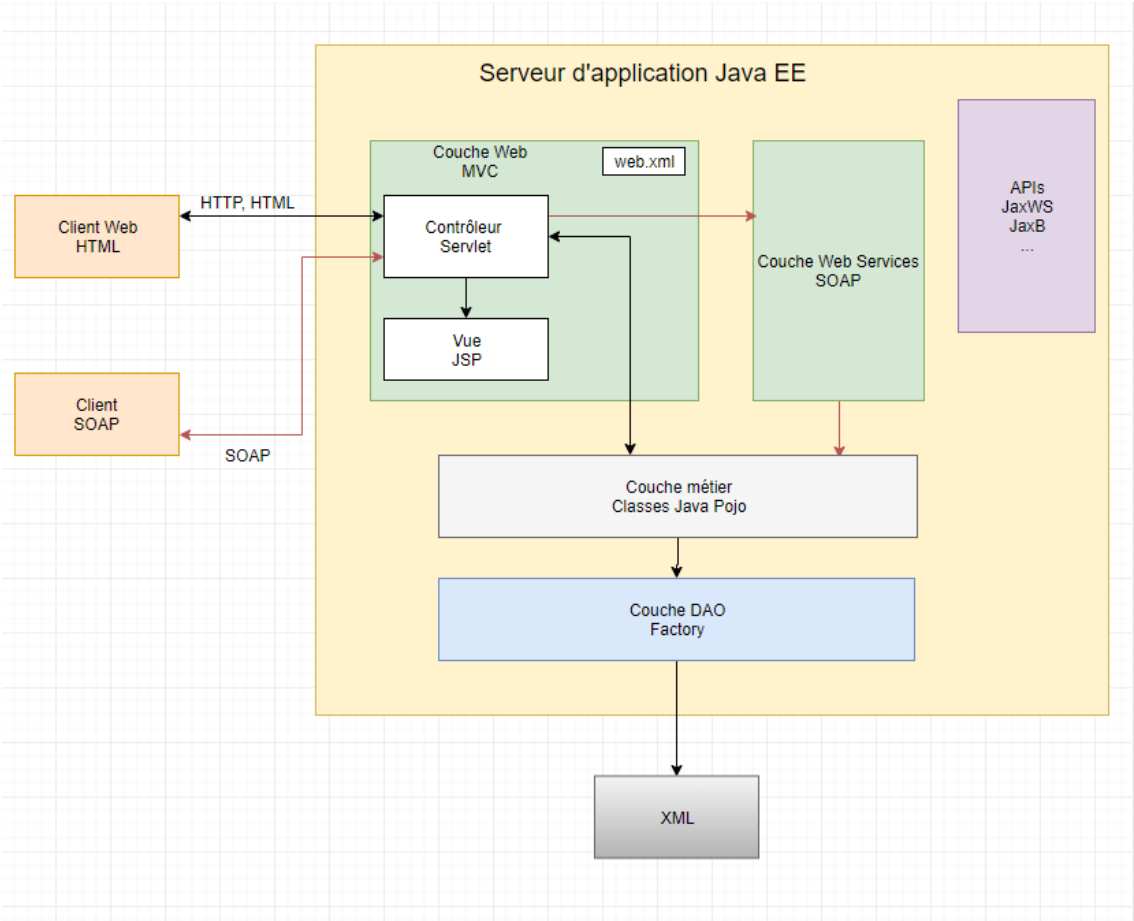
```
projets
      projet (nb 0,n)
            image
            nom
            descriptif
            deadLineCandidature
```

```
            deadLineProjet
            nbMaxCandidatures
            candidatures
                  candidature (nb 0,n)
                        dateCandidature
                        etatCandidature
                        utilisateur
                              role
                              email
                              password
                              nom
                              prenom
                              tel
                        structure
                              raisonSocial
                              siret
                              ca
                        repProjet
                              delaisPropose
                              devis
                              website
```

Structure de la liste des utilisateurs

```
utilisateurs
      utilisateur (nb 1,n)
            role
            email
            password
            nom
            prenom
            tel
```

Architecture technique



L'architecture technique utilise :

- *Différentes technologies relatives*:

    - à **Java Platform, Enterprise Edition** (Servlet, JavaServer Pages (JSP), JavaServer Pages Standard Tag Library (JSTL), Java Architecture for XML Binding (JAXB)).

    - **aux services web** (Java API for XML Web Services (JAX-WS).

- **aux technologies Web** : HTML5, CSS3, JavaScript, XML, XSD, les librairies Bootstrap et Materialize.

- **à des librairies utilitaires Java** : mime-util-2.1.3 pour détecter le type mime d'un fichier téléverser. SLF4J comme librairie de logging.

- *Différentes approches* : Modèle-vue-controlleur, Approche orientée Services, protocole SOAP, design pattern Factory

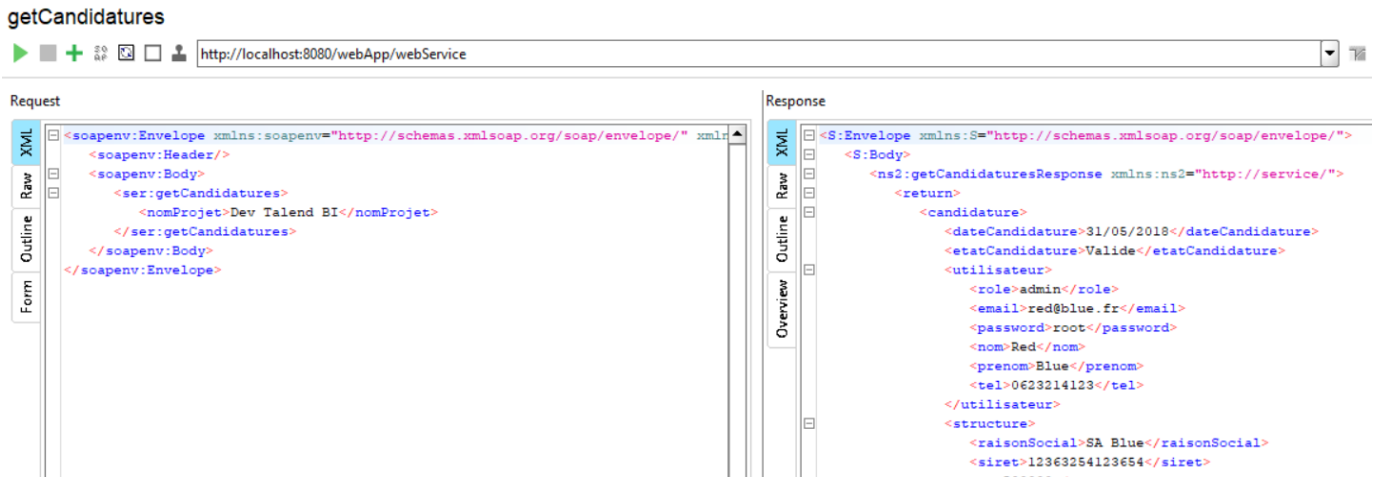
Le Web Service SOAP

Le chemin d'accès du webservice est le suivant : http://localhost:8080/webApp/webService
Son wsl : http://localhost:8080/webApp/webService?wsdl

Les opérations suivantes sont opérationnelles :
addCandidature, getCandidatures, getOneCandidature, getOneProjet, getProjets

Nous n'avons pas développé de client SOAP pour tester notre service web c'est pourquoi nous préconisons l'utilisation de SoapUI.

Exemple de paire requête/réponse soap de l'opération getProjets()



Exemple de paire requête/réponse soap de l'opération getCandidatures() d'un projet dont le nom est passé en paramètre

Cependant, à noter que la création d'une candidature requiert que celui qui candidate ait au préalable enregistré dans la base de données des utilisateurs.
Nous n'avons pas implémenté de mécanisme d'authentification pour les requêtes SOAP qui aurait permis de remplir automatiquement les informations de l'utilisateur qui créé la requête à l'instar de ce qui se passe quand on crée une candidature sur le site web.

Sécurité

Pour sécuriser les mots de passe des utilisateurs, nous faisons appel au PBKDF2 (abréviation de Password-Based Key Derivation Function 2). C' est une fonction de dérivation de clé appartenant à la famille des normes Public Key Cryptographic Standards.

On utilise dans ce projet PBKDF2 pour le hachage de mot de passe, couplé à la fonction SHA-512.

Le PBKDF2 applique la fonction de hachage SHA-512 à un mot de passe ou une phrase secrète avec un sel et répète cette opération plusieurs fois afin de générer une clé, qui peut être ensuite utilisée pour chiffrer le mot de passe original.

Cette génération rajoute du temps de calcul qui complique le cassage du mot de passe, notamment par force brute. Le sel ajouté permet d'éviter l'utilisation de rainbow tables et donc limite les attaques sur plusieurs mots de passe en simultané.

Limites du projet et problèmes rencontrés

La principale difficulté rencontrée au cours du projet et non résolue concerne la “création d’un projet”. En effet, dans un premier temps, celle-ci a fonctionné parfaitement quand nous n’avons pas encore pris en compte le mécanisme d’upload d’image (associée à chaque projet).
Cependant, quand nous avons commencé à implémenter l’upload, nous avons dû changer le format d’envoie de données de notre formulaire de application/x-www-form-urlencoded, l’encodage par défaut, à multipart/form-data.
Cette nouvelle façon de faire a rendu possible l’envoie de fichier (dans notre cas de l’image du projet) mais à rendu impossible la récupération des paramètres tels que ‘deadLineCandidature ‘et autres dates, integer pour qui la méthode request.getParameter() ne fonctionne plus.

Nous avons cherché plusieurs solutions à ce problème, mais faute de temps et de compétence, nous n’avons pas réussi à trouver comment parser la request nouvellement encodé. Il est donc toujours possible de créer un projet, mais ses informations seront partiellement enregistrés. A noter que cela n’affecte pas le processus de candidature.

Cette limite nous a également ralenti sur la gestion de l’upload des images, elle fonctionne mais ne remplit pas toutes les fonctionnalités que nous aurions voulus mettre en place, à savoir par exemple renommer les images par une concaténation du timestamp, du nom du projet et du mail de l’utilisateur. Cela aurait permis de distinguer les images dans le dossier files et surtout d’éviter les doublons concernant les noms d’images uploadées par les utilisateurs.

Évolutions possibles du projet

Les évolutions autour de ce projets sont nombreuses, en voici quelques-unes :

- Pour l’utilisateur :
    - Recevoir un mail de confirmation lors de la création de son compte.
    - Possibilité de modifier une candidature en attente de validation.
    - Possibilité de modifier des informations comme le mail, mot de passe etc…
    - Création d'un tableau de bord permettant de suivre l’état des candidatures.
    - Possibilité d’avoir des notifications sur l’évolution des états d’une candidatures.
    - Possibilité de créer des dossiers de bases pouvant être appliqués pour de nouvelles candidatures.
    - Possibilité de se mettre en liste d’attente sur un projet complet en cas de désistement.
- Relations entre utilisateurs :
    - Permettre la mise en relation entre les différents candidats sur un même projet accepté.
    - Création d’un espace de travail et d'échange (messagerie, dépôt de documents etc…).

    - Outils permettant aux candidats d’indiquer (saisir) un niveau de progression estimé sur le projet en cours.
- Pour les administrateurs :
    - Possibilité de définir un rôle à un utilisateur choisit.
    - Possibilité d’inviter un utilisateur via email.
    - Possibilité de supprimer un utilisateur.
    - Tableaux de bords récapitulant :
        - le niveaux de “remplissage” du nombre de candidatures sur les projets.
        - La progression estimée sur les projets en cours.
        - Divers indicateurs comme taux de réussite, échec etc…
    - Permettre de mettre en avant certains projets.
    - Gérer une liste d’attente des candidatures pour les projets complets pour pouvoir, éventuellement proposer un projet similaire à ses utilisateurs.