

# PROJET SERVEUR D'APPLICATIONS JAVA



*Elaboré par :*

*Ismail ELAMOURI (Master 2 MIAGE – SITN)*

*Paul VONG (Master 2 MIAGE – SITN)*

*Shahram MAHDAVI (Master 2 MIAGE – IF)*

*Yusuf YILDIZ (Master 2 MIAGE – ID)*



*Année Universitaire : 2019-2020*

# TABLE DES MATIERES

<b>INTRODUCTION GENERALE</b>	<b>7</b>
<b>CHAPITRE 1 CONTEXTE DU PROJET</b>	<b>9</b>
1.1 METHODOLOGIE DE TRAVAIL	10
1.2 SPECIFICATIONS DES BESOINS	11
1.2.1 SPECIFICATIONS DES BESOINS FONCTIONNELS	11
1.2.1.1 Besoins d'un visiteur	11
1.2.1.2 Besoin d'un membre de site web	11
1.2.1.3 Besoins de l'administrateur du site web	12
1.2.2 SPECIFICATION DES BESOINS NON-FONCTIONNELS	12
1.2.2.1 Les exigences	12
1.2.2.2 Les contraintes	13
<b>CHAPITRE 2 SPRINT 0 : ARCHITECTURE LOGICIELLE ET FRAMEWORK DE DEVELOPPEMENT</b>	<b>14</b>
2.1 OBJECTIFS	15
2.2 CHOIX DE L'ARCHITECTURE DU SYSTEME : ARCHITECTURE MVC	15
2.3 PRESENTATION DES OUTILS DE DEVELOPPEMENT	15
2.4 CONFIGURATION ET MISE EN PLACE DE SYSTEME	17
2.4.1 LES MODULES DE L'APPLICATION	17
2.4.2 PRINCIPE DE FONCTIONNEMENT DE L'APPLICATION	18
2.5 DIAGRAMME DE DEPLOIEMENT	19
CONCLUSION	19
<b>CHAPITRE 3 SPRINT 1 : AUTHENTIFICATION ET GESTION DES COMPTES</b>	<b>20</b>
3.1 OBJECTIFS	21
3.2 ANALYSE	21
3.2.1 DIAGRAMME DE CAS D'UTILISATION ET SCENARIOS	21
3.2.1.1 Diagramme de cas d'utilisation	21
3.2.1.2 Les scénarios	22
3.2.2 DIAGRAMME DE CLASSE	23
3.3 CONCEPTION	26
3.3.1 DIAGRAMMES DE SEQUENCES	26

3.3.1.1	Diagramme de séquence relatif à la création de compte	26
3.3.1.1	Diagramme de séquence relatif à la gestion de compte	27
3.3.1.2	Diagramme de séquence relatif à l'authentification	27
3.3.1.3	Diagramme de séquence relatif à la déconnexion	28
3.3.2	DIAGRAMMES D'ACTIVITES	29
3.3.2.1	Diagramme d'activités relatif à l'authentification	29
3.3.2.2	Diagramme d'activités relatif à la gestion de compte	30
3.3.3	DESCRIPTION DE L'IHM	30
3.3.3.1	Page d'accueil du site web	30
3.3.3.2	Page d'inscription	31
3.3.3.3	Page d'authentification	31
3.3.3.4	Page d'accueil du membre / administrateur	32
3.3.3.5	Profile d'un membre / administrateur	33
<b>3.4</b>	<b>EVALUATION</b>	<b>33</b>
	<b>CONCLUSION</b>	<b>33</b>

---

## **CHAPITRE 4    SPRINT 2 : GESTION DES OFFRES ET DES DEMANDES DE SERVICES** **34**

<b>4.1</b>	<b>OBJECTIFS</b>	<b>35</b>
<b>4.2</b>	<b>ANALYSE</b>	<b>35</b>
4.2.1	DIAGRAMME DE CAS D'UTILISATION ET SCENARIOS	35
4.2.1.1	Diagramme de cas d'utilisation	35
4.2.1.2	Les scénarios	36
4.2.2	DIAGRAMME DE CLASSE	38
<b>4.3</b>	<b>CONCEPTION</b>	<b>40</b>
4.3.1	DIAGRAMMES DE SEQUENCES	41
4.3.1.1	Diagramme de séquence relatif à la création d'une offre / demande de service	41
4.3.1.2	Diagramme de séquence relatif à la visualisation des offres ou demandes de services	41
4.3.1.3	Diagramme de séquence relatif à la proposition ou l'acceptations du service	42
		43
4.3.2	DIAGRAMMES D'ACTIVITES	43
4.3.2.1	Diagramme d'activité relatif à la création des services	43
4.3.2.2	Diagramme d'activité relatif à la gestion des services (Propositions / acceptations)	44
4.3.3	DESCRIPTION DE L'IHM	45
4.3.3.1	Page de création d'une demande ou d'une offre de service	45
4.3.3.2	Page de la visualisation des services	46
4.3.3.3	Page d'affichage d'un service sélectionné	47
<b>4.4</b>	<b>EVALUATION</b>	<b>48</b>
	<b>CONCLUSION</b>	<b>48</b>

---

## **CHAPITRE 5    SPRINT 3 : GESTION DES NOTIFICATIONS ET DES SERVICES PAR L'ADMINISTRATEUR** **49**

<b>5.1</b>	<b>OBJECTIFS</b>	<b>50</b>
------------	------------------	-----------

<b>5.2</b>	<b>ANALYSE</b>	<b>50</b>
5.2.1	DIAGRAMME DE CAS D'UTILISATION	50
5.2.1.1	Les Scénarios	51
5.2.2	DIAGRAMME DE CLASSE	52
<b>5.3</b>	<b>CONCEPTION</b>	<b>54</b>
5.3.1	DIAGRAMMES DE SEQUENCES	54
5.3.1.1	Diagramme de séquence relatif à la gestion des services d'un utilisateur	54
5.3.1.2	Diagramme de séquence relatif à la validation des services	55
5.3.2	DIAGRAMME D'ACTIVITE RELATIF A LA VISUALISATION DES NOTIFICATIONS ET GESTION DES SERVICES	56
5.3.3	DESCRIPTION DE L'IHM	56
5.3.3.1	Page de visualisation des services par utilisateur	58
<b>5.4</b>	<b>EVALUATION</b>	<b>59</b>
<b>CONCLUSION</b>		<b>59</b>
 <b>CONCLUSION GENERALE</b>		 <b>60</b>
 <b>ANNEXES</b>		 <b>61</b>
 <b>BIBLIOGRAPHIE</b>		 <b>65</b>
 <b>NETOGRAPHIE</b>		 <b>66</b>

## TABLE DES FIGURES

Figure 1: Cycle de vie de SCRUM [N2] .....	10
Figure 2: Modèle MVC .....	15
Figure 3: Les modules de l'application .....	17
Figure 4: schéma du principe de fonctionnement de système .....	18
Figure 5: Diagramme de déploiement .....	19
Figure 6: Diagramme des cas d'utilisation relatif au package "Authentication & gestion de compte" .....	21
Figure 7: Diagramme de classe visant la classe "Utilisateur" .....	24
Figure 8: Diagramme de séquence relatif à la création de compte .....	26
Figure 9: Diagramme de séquence relatif à la gestion de compte.....	27
Figure 10: Diagramme de séquence relatif à l'authentification .....	28
Figure 11: Diagramme de séquence relatif à la déconnexion .....	28
Figure 12: Diagramme d'activités relatif à l'authentification.....	29
Figure 13: Diagramme d'activités relatif à la gestion de compte .....	30
Figure 14: Page d'accueil du site web .....	30
Figure 15: Page d'inscription .....	31
Figure 16: Page d'authentification .....	32
Figure 17: Page d'accueil du membre / administrateur .....	32
Figure 18: Profile d'un membre / administrateur.....	33
Figure 19: Diagramme de cas d'utilisation relatif au package « Gestion des services" .....	35
Figure 20: Diagramme de classe visant les classes "Service" et "NatureDeService" .....	38
Figure 21: Diagramme de séquence relatif à la création d'une offre / demande de service.....	41
Figure 22: Diagramme de séquence relatif à la visualisation des offres ou demandes de services .....	42
Figure 23: Diagramme de séquence relatif à la proposition ou l'acceptations du service .....	43
Figure 24 : Diagramme d'activité relatif à la création des services .....	44
Figure 25: Diagramme d'activité relatif à la gestion des services (Propositions / acceptations) .....	45
Figure 26: Page de création d'une demande ou d'une offre de service.....	46
Figure 27: Page de la visualisation des services .....	46
Figure 28: Page d'acceptation d'une offre de service .....	47
Figure 29: Page de proposition d'une demande de service .....	47
Figure 30: Page de saisi du message personnel .....	48
Figure 31: Page de message affichant une erreur.....	48
Figure 32 : Diagramme des cas d'utilisation relatif au package "Gestion des notifications et seervices par administrateurs » .....	50
Figure 33 : Diagramme de classe visant la classe "Notification" .....	52
Figure 34: Diagramme de séquence relatif à la gestion des services d'un utilisateur.....	54
Figure 35 : Diagramme de séquence relatif à la validation des services .....	55
Figure 36 : Diagramme d'activité relatif à la visualisation des notifications et le gestion des services .....	56

Figure 37: IHM de notification envoyé à l'administration.....	56
Figure 38: IHM de détails de la notification.....	57
Figure 39: IHM de notification d'acceptation d'un service par l'administrateur .....	57
Figure 40: IHM de détails de la notification envoyée par l'administrateur .....	58
Figure 41: Page de visualisation des services par utilisateur .....	58
Figure 42: IHM pour modifier ou supprimer un service .....	59
Figure 43: Architecture d' EclipseLink .....	63

## TABLE DES TABLEAUX

Tableau 1: Scénario "Création de compte" .....	22
Tableau 2: Scénario « Gestion de compte » .....	22
Tableau 3: Scénario « Authentification » .....	23
Tableau 4: Scénario « Déconnexion » .....	23
Tableau 5: Attributs de la classe "Utilisateur" .....	25
Tableau 6: Méthodes de la classe "Utilisateur" .....	25
Tableau 7: Scénario « Création de l'offre / Création de la demande » .....	36
Tableau 8: Scénario « Visualisation des offres et des demandes de services » .....	37
Tableau 9: Scénario « Proposition d'un service / Acceptation d'une offre » .....	37
Tableau 10: Attributs de la classe "NatureDeService" .....	38
Tableau 11: Méthodes de la classe "NatureDeService" .....	39
Tableau 12: Attributs de la classe "Service" .....	39
Tableau 13: Méthodes de la classe "Service" .....	40
Tableau 14: Scénario « Gestion des services d'un utilisateur » .....	51
Tableau 15: Scénario « Visualisations des notifications » .....	51
Tableau 16: Scénario "Validation des offres et des demandes de service" .....	52
Tableau 17: Diagramme de classe visant la classe "Notification" .....	52
Tableau 18 : Attributs de la classe "Notification" .....	53
Tableau 19 : Méthodes de la classe « Notification » .....	53
Tableau 20: Elément de l'architecture d'EclipseLink.....	64

## Introduction générale

---

Ce projet a pour objectif la création d'un site web pour une association permettant la mise en relation d'internautes, dans le but de se rendre des services mutuellement.

La conception du site web a demandé la mise en place d'une dizaine de processus permettant aux différents acteurs d'interagir entre eux et le bon séquençage de l'éventail des actions possibles.

Il nous a fallu tout au long de notre projet, développer dans un environnement Java EE à l'aide de la méthode de développement agile SCRUM.

On détaillera dans la suite du projet notre méthodologie de travail mise en place, ainsi que l'ensemble de spécifications qui nous ont été demandées. Par la suite on rentrera plus en détail dans les différents sprints de notre projet

Enfin on finalisera le rapport par une conclusion de notre projet.



## Chapitre 1 *Contexte du projet*

---

---

## 1.1 Méthodologie de travail

Avec les progrès en technologie de l'information et les investissements dans les infrastructures, beaucoup de méthodes de gestion de projet ont vu le jour. Ces méthodes jouent un rôle primordial dans la réussite ou l'échec d'un projet, leur choix représente donc une décision importante pour les entreprises. Dans le cadre de notre projet, nous avons adopté la méthode SCRUM comme étant une méthode agile [A1].

- **Méthode « SCRUM »**

SCRUM est une méthodologie agile qui permet de livrer un logiciel de qualité plus rapidement. SCRUM permet au client d'un logiciel développé de contrôler la qualité du travail tandis que l'équipe contrôle la quantité de travail effectué. Cette méthodologie permet de s'adapter rapidement aux changements d'un client puisque régulièrement (à chaque fin d'itération) l'équipe et le client réévaluent les spécifications du logiciel. Ainsi, le client reçoit les spécifications demandées plus fréquemment (parfois jusqu'à un livrable par semaine) ce qui ne fait qu'accroître sa satisfaction. La réévaluation constante des spécifications à développer permet de s'adapter aux priorités du client. Cela permet d'optimiser les ressources de développement en fonction des besoins réels. [B3].

- **Equipe « SCRUM »**

- **Product Owner** : Dimcea DAN
- **Scrum Master** : Ismail EL AMOURI
- **Développeurs** : Ismail EL AMOURI, Shahram MAHDAVI, Paul VONG et Yusuf YILDIZ

- **Cycle de vie « SCRUM »**

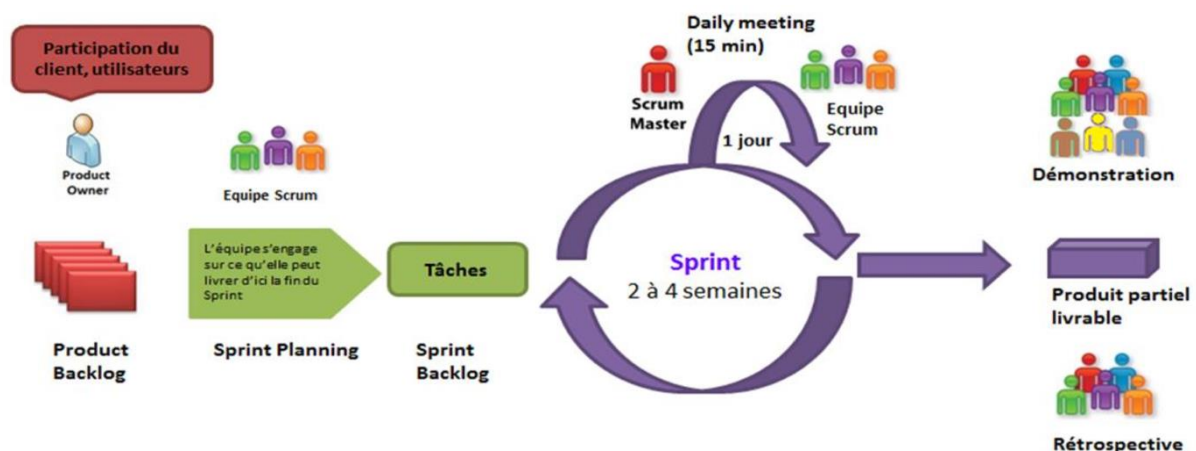


Figure 1: Cycle de vie de SCRUM [N2]

La figure 1 donne un aperçu de la méthodologie SCRUM dont nous traiterons dans les prochains chapitres.

## 1.2 Spécifications des besoins

L'analyse du sujet nous a permis de montrer les diverses fonctionnalités que l'application doit satisfaire.

Le site WEHO aura un système de hiérarchie comportant trois grades : le visiteur, le membre du site et l'administrateur. Les visiteurs peuvent accéder uniquement à la page d'accueil du site et visualiser les offres de service. Les membres du site ont tous les droits, excepté les droits d'administrer le site. À la fin, l'administrateur a les mêmes droits que les membres, plus le droit d'administration (par exemple un administrateur a le droit de valider une offre ou une demande).

### 1.2.1 *Spécifications des besoins fonctionnels*

#### 1.2.1.1 Besoins d'un visiteur

- Informations générales : accès aux offres et demandes publiées par les membres de site.
- Recherche : permet au visiteur de rechercher et visualiser les offres et les demandes publiées et valides.
- S'inscrire à notre site

#### 1.2.1.2 Besoin d'un membre de site web

- Authentification : permet à un membre de s'authentifier auprès du site.
- Déconnexion : permet à un membre de site de se déconnecter du site.
- Création d'une offre : permet à un membre de créer une offre de service.
- Création d'une demande : permet à un membre de créer une demande de service.
- Visualisation des offres : permet à un membre connecté de visualiser toutes les offres et les demandes de service qui sont valide.
- Recherche : permet à un membre de rechercher des offres de service en les filtrant à l'aide des critères de recherche.
- Proposition de service : permet à un membre de proposer un service à un autre membre qui a mis une demande de service sur le site.

- Acceptation d'une offre de service : permet à un membre d'accepter un service proposé par un autre membre.
- Gestion des services : permet à un membre de gérer les offres et les demandes de services qu'il a créés sur le site.
- Visualisation des notifications : permet à un membre visualiser la liste de toutes ses notifications récents et anciennes, triées par date d'envoi.

### 1.2.1.3 Besoins de l'administrateur du site web

- Validation des offres et des demandes de service : permet à un administrateur de valider ou d'annuler les nouvelles offres et demandes de service dans l'état A\_VALIDER.
- Authentification : permet à un administrateur de s'authentifier auprès du site.
- Déconnexion : permet à un administrateur de site de se déconnecter du site.
- Création d'une offre : permet à un administrateur de créer une offre de service.
- Création d'une demande : permet à un administrateur de créer une demande de service.
- Visualisation des offres : permet à un administrateur connecté de visualiser toutes les offres et les demandes de service qui sont valides.
- Recherche : permet à un administrateur de rechercher des offres de service en les filtrant à l'aide des critères de recherche.
- Proposition de service : permet à un administrateur de proposer un service à un autre utilisateur qui a mis une demande de service sur le site.
- Acceptation d'une offre de service : permet à un administrateur d'accepter un service proposé par un autre utilisateur.
- Gestion des services : permet à un administrateur de gérer les offres et les demandes de services qu'il a créé sur le site.
- Visualisation des notifications : permet à un administrateur d'afficher la liste de toutes ses notifications récents et anciennes, triées par date d'envoi.

### 1.2.2 Spécification des besoins non-fonctionnels

#### 1.2.2.1 Les exigences

- Le système doit intégrer des aides contextuelles dans certaines interfaces.
- La procédure de saisie des informations doit être simple et rapide.

- Le serveur doit pouvoir gérer les différents processus de saisie, de vérification et de validation des données.
- Le système doit permettre la saisie décentralisée et simultanée des données.
- Les temps de réaction doivent être de l'ordre de la seconde pour garder l'impression d'interactivité forte.
- Le système doit s'exécuter et fonctionner entièrement sans avoir recours à d'autres applications.
- Le système doit accéder de façon transparente au système de gestion de bases de données.
- Les tables de données en consultation doivent automatiquement être mises à jour si les données changent.

### 1.2.2.2 Les contraintes

Différentes contraintes de sécurité devront être respectées lors de la réalisation du site :

- Attaques d'une menace extérieure :

Le site utilisant une base de données, il y a de fortes chances qu'un pirate informatique tente d'y rentrer par différents types d'attaque telles que des injections SQL ou l'utilisation de scripts malveillants.

- Mots de passes cryptés :

Afin de renforcer la protection des données personnelles des utilisateurs, tous les mots de passe sont stockés dans la base de données après avoir été cryptés

## Chapitre 2 *Sprint 0 : Architecture logicielle et Framework de développement*

---

---

## 2.1 Objectifs

Ce sprint est consacré à comprendre l'architecture logicielle et la mise en œuvre de cette dernière à travers les Framework de développement.

## 2.2 Choix de l'architecture du système : Architecture MVC

L'ensemble des outils et Framework de développement mène à adopter l'architecture logicielle MVC [A2]. En effet, c'est une méthode d'organisation et de structuration des applications logicielles. Ce design est basé sur la décomposition de l'application en trois éléments indispensables : modèle, vue et contrôleur.

Ceci est bien explicite à travers la figure 2 :

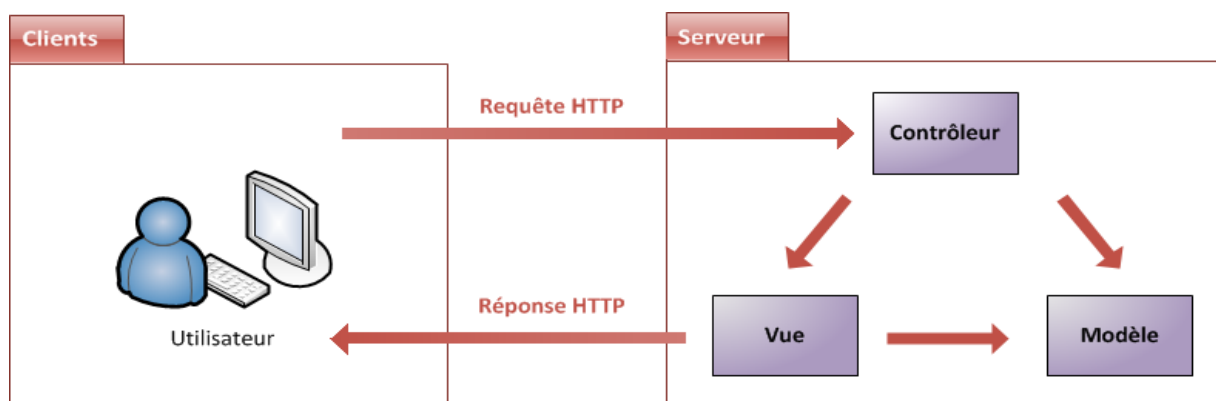


Figure 2: Modèle MVC

## 2.3 Présentation des outils de développement

Tout au long de la phase de développement, nous nous sommes servis de l'environnement logiciel suivant :

- **Entreprise Architecture (EA)** est un outil complet d'analyse et de conception UML pour UML, SysML, BPMN et de nombreuses autres technologies. Couvrir le développement de logiciels, de la collecte des exigences aux étapes d'analyse, des modèles de conception, des tests et de la maintenance. EA déploie toute votre équipe, y compris les analystes, les testeurs, les gestionnaires de projet, le personnel de contrôle de la qualité et l'équipe **de** déploiement, pour une fraction du coût de certains produits concurrents.
- **JavaServer Pages Standard Tag Library (JSTL)** est un composant de la plate-forme JEE de développement.mapping objet/relationnel.
- **NetBeans** est un environnement de développement intégré (EDI), placé en open source par Sun JPA Modeler

- **EclipseLink** est un Framework open source de mapping objet-relationnel pour les développeurs Java. Il fournit une plateforme puissante et flexible permettant de stocker des objets Java dans une base de données relationnelle et/ou de les convertir en documents XML [A3].
- **GlassFish** est le nom du serveur d'applications Open Source Java EE 5 et désormais Java EE 7 avec la version 4.1 qui sert de socle au produit OracleGlassFish Server (anciennement Sun Java System Application Server de Sun Microsystems)
- **Bootstrap** est un Framework CSS proposé par Twitter sous licence Apache. Bootstrap dépasse les Framework CSS classiques et propose carrément des éléments graphiques complets avec une garantie maximale de compatibilité entre les divers navigateurs [N3].
- **JQuery** est une bibliothèque JavaScript rapide, de taille petite et riche en fonctionnalités. Elle s'intègre facilement dans les documents HTML. Elle offre une manipulation simple, une gestion des événements facile, des animations et des outils ajaxifiés[N4].
- **HTML (HyperText Markup Language)** : Langage de structuration des pages web.
- **CSS (Cascading Style Sheets)** : Langage de présentation des pages web.
- **JS (Java Script)** : est un langage de programmation de scripts principalement employé dans les pages web interactives.
- **Visual-paradigme** : un outil de modélisation des diagrammes UML.

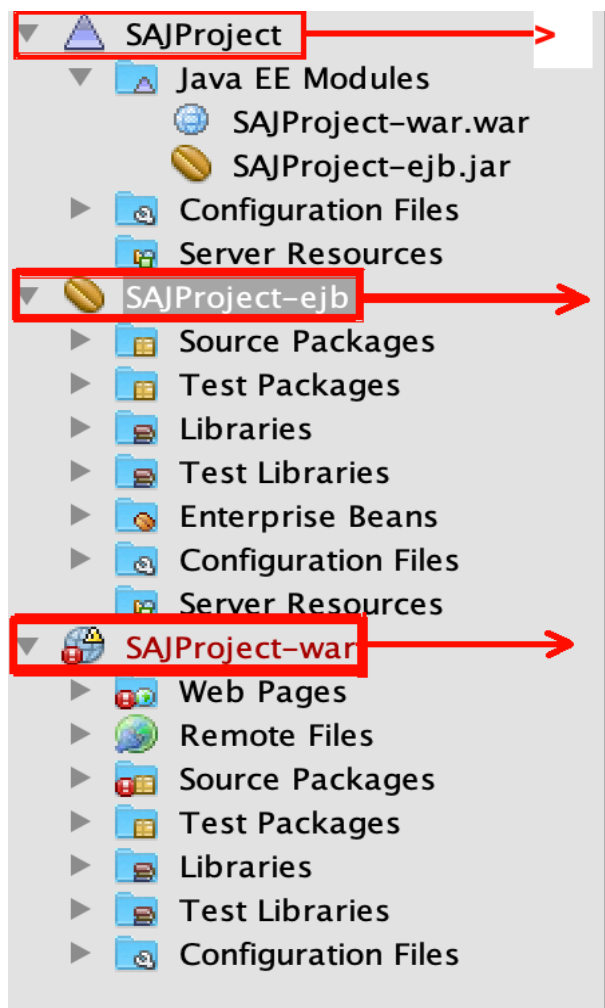


## 2.4 Configuration et mise en place de système

Dans cette partie, on va énumérer les différents outils utilisés pour l'étude et la mise en place de notre application, éventuellement on va donner une description générale sur les principaux composants nécessaires ainsi que la liaison entre eux.

### 2.4.1 Les modules de l'application

Au moment de la création d'une Enterprise Application, l'éditeur « Netbeans » crée automatiquement 3 modules comme le décrit la figure suivante :



#### **EAR (pour Enterprise Application Archive) :**

Contient que des informations de configuration et packaging concernant l'application

**Entreprise JavaBeans (EJB) :** est une architecture de composants logiciels côté serveur pour la plateforme de développement Java JEE.

Ce module encapsule tous les opérations et les

**Application web :** l'objectif de ce module est de communiquer avec les EJB pour envoyer ou réceptionner des traitements.

Figure 3: Les modules de l'application

### 2.4.2 Principe de fonctionnement de l'application

Le schéma suivant montre le principe de fonctionnement de notre application en mettant en valeur les relations entre les différents composants.

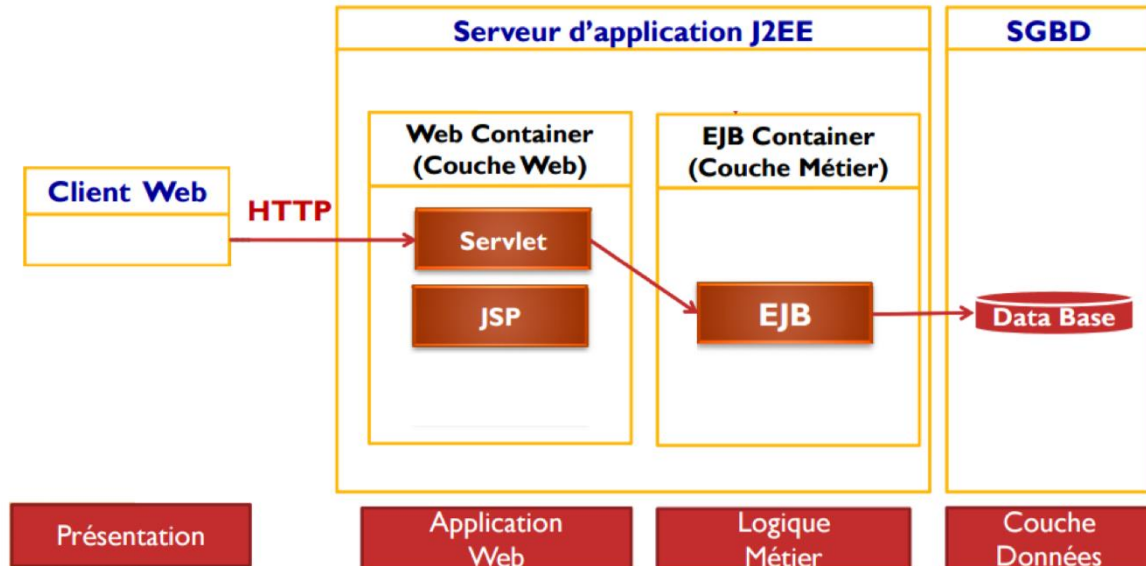


Figure 4: schéma du principe de fonctionnement de système

- **La couche de présentation** dialogue avec l'utilisateur via une interface web. Elle a pour rôle de fournir des données provenant du client à la couche métier ou de présenter au client des données fournies par la couche métier.
- **La couche métier** est la couche qui applique les règles dites métier, c'est-à-dire la logique spécifique de l'application, détachée de la provenance des données, ni de la postérité des résultats qu'elle produit.
- **La couche persistance** est la couche qui fournit à la couche métier des données préenregistrées (fichiers, bases de données...) et qui enregistre certains des résultats fournis par la couche métier.

## 2.5 Diagramme de déploiement

Le diagramme de déploiement permet de représenter l'architecture physique supportant l'exploitation du système. Cette architecture comprend des nœuds. Les nœuds peuvent être interconnectés pour former un réseau d'éléments physiques correspondant aux supports physiques (serveurs, routeurs...) ainsi que la répartition des artefacts logiciels (bibliothèques, exécutables...) sur ces nœuds. C'est un véritable réseau constitué de nœuds et de connexions entre ces nœuds qui modélise cette architecture. La figure 5 montre le diagramme de déploiement de notre application.

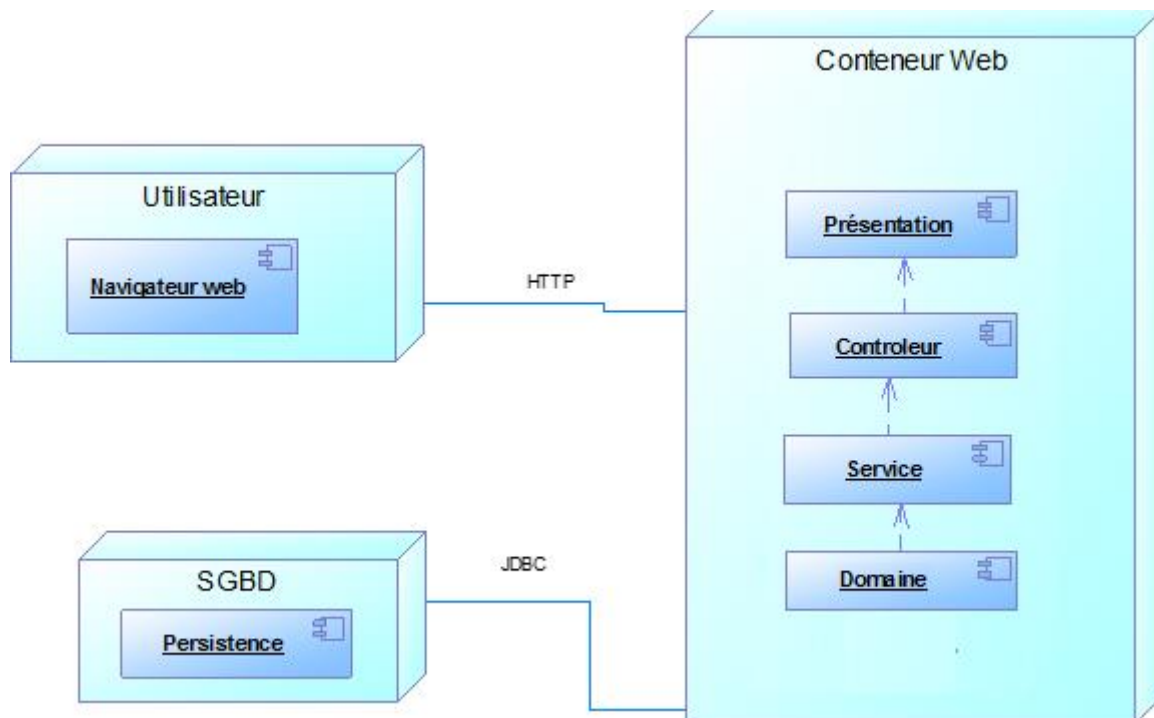


Figure 5: Diagramme de déploiement

## CONCLUSION

Ce sprint est consacré au choix de notre architecture du système, la présentation de l'architecture logicielle et l'architecture physique de notre application ainsi que l'environnement logiciel dans lequel le projet a été élaboré. Dans ce qui suit on va présenter la phase de réalisation des sprints qui suivent tout en commençant par le sprint 1 qui décrit l'authentification et la gestion des comptes.

## Chapitre 3 *Sprint 1 : Authentification et gestion des comptes*

---

---

### 3.1 Objectifs

Dans notre application l'authentification consiste à assurer la sécurité. Le but de ce sprint est de gérer la gestion des utilisateurs pour qu'ils puissent accéder à l'application.

### 3.2 Analyse

Nous avons présenté les besoins spécifiés, nous allons maintenant présenter les diagrammes des cas d'utilisation des différents processus (Création de compte, gestion de compte, Login, logout) ainsi que le diagramme de classe.

#### 3.2.1 Diagramme de cas d'utilisation et scénarios

##### 3.2.1.1 Diagramme de cas d'utilisation

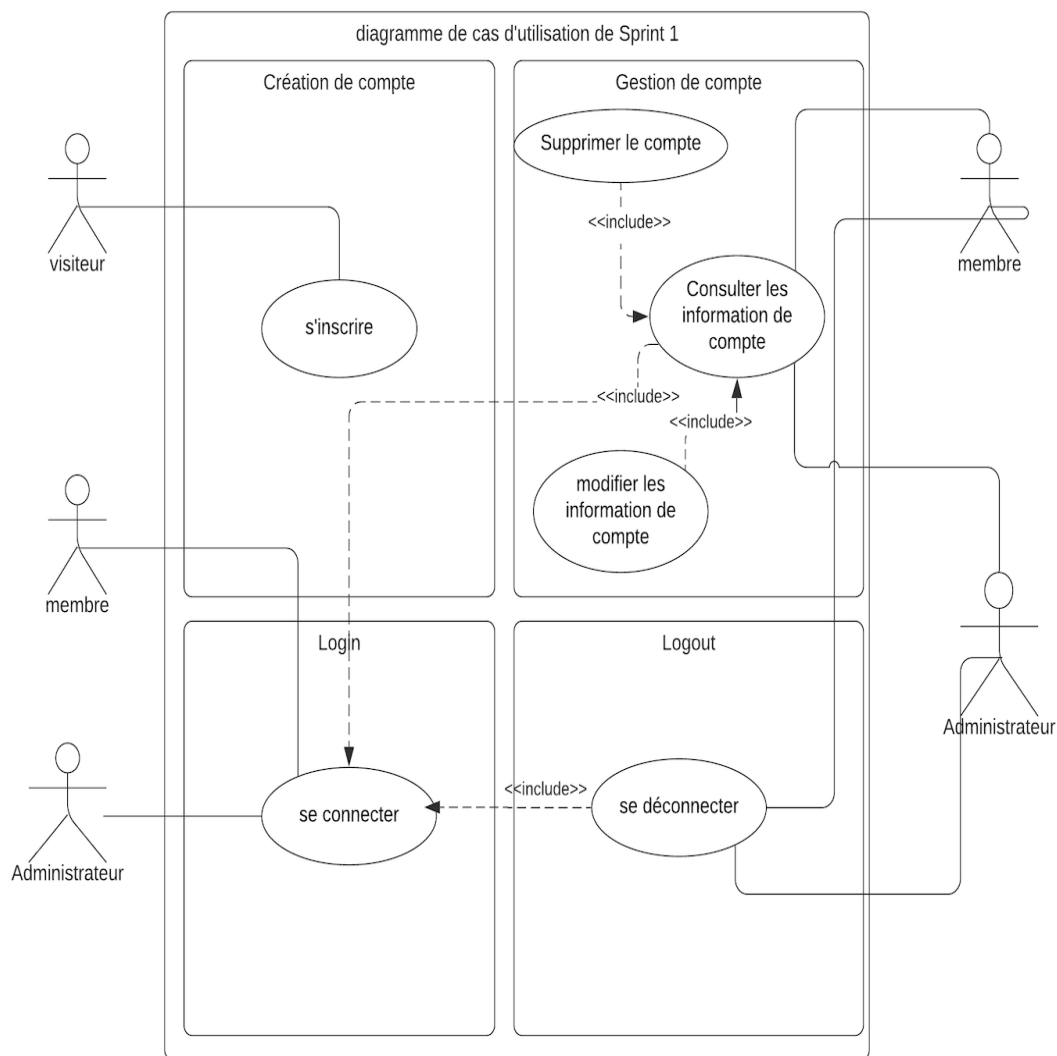


Figure 6: Diagramme des cas d'utilisation relatif au package "Authentification & gestion de compte"

### 3.2.1.2 Les scénarios

#### 3.2.1.2.1 Scénario « Création de compte »

<b>Cas d'utilisation</b>	<b>Création de compte</b>
<b>Acteur</b>	Visiteur
<b>Pré condition</b>	Le visiteur ne possède pas de compte
<b>Post condition</b>	Le compte est créé avec succès
<b>Scénario nominal</b>	<ol style="list-style-type: none"><li>1. Le visiteur doit se rendre sur la page de création de compte à partir de lien dans la page d'accueil de site</li><li>2. Il remplit le formulaire de création de compte</li><li>3. Il valide la saisie.</li></ol>
<b>Exception</b>	E1 : Saisie incorrecte : l'information saisie n'est pas au bon format. E1' : E-mail existe déjà : E-mail saisi correspond à un autre utilisateur  ⇒ Afficher un message d'erreur

Tableau 1: Scénario "Création de compte"

#### 3.2.1.2.2 Scénario « Gestion de compte »

<b>Cas d'utilisation</b>	<b>Gestion de compte</b>
<b>Acteurs</b>	Membre, Administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit posséder un compte et il doit être connecté sur le site.
<b>Post condition</b>	Le compte est consulté/modifié/supprimé avec tous les services de l'utilisateur
<b>Scénario nominal</b>	<ol style="list-style-type: none"><li>1. L'acteur se connecte avec son email et son mot de passe.</li><li>2. Il consulte tous ses informations</li><li>3. Il peut aussi<ol style="list-style-type: none"><li>a. Supprimer son compte après une confirmation de sa part</li><li>b. Modifier ses informations modifiable (Numéro de téléphone, mot de passe, adresse et sa description personnelle) de compte.</li><li>c. Valider les modifications après une confirmation de sa part</li></ol></li></ol>
<b>Exception</b>	E2 : Saisie incorrecte : l'information saisie n'est pas au bon format.  ⇒ Afficher un message d'erreur

Tableau 2: Scénario « Gestion de compte »

### 3.2.1.2.3 Scénario « Authentification »

<b>Cas d'utilisation</b>	<b>Authentification</b>
<b>Acteurs</b>	Membre et administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit posséder un compte.
<b>Post condition</b>	L'utilisateur est connecté en tant que membre ou en tant qu'administrateur du site.
<b>Scénario nominal</b>	<ol style="list-style-type: none"><li>1. L'acteur doit se rendre sur la page de connexion à partir de lien dans la page d'accueil de site</li><li>2. Il saisit son email et son mot de passe.</li><li>3. Il peut consulter ses informations de compte</li></ol>
<b>Exception</b>	E3 : Identifiants de connexions incorrects : E-mail et/ou mot de passe ne sont pas corrects. E3' : Saisie incorrecte : l'information saisie n'est pas au bon format  ⇒ Afficher un message d'erreur

**Tableau 3: Scénario « Authentification »**

### 3.2.1.2.4 Scénario « Déconnexion »

<b>Cas d'utilisation</b>	<b>Déconnexion</b>
<b>Acteurs</b>	Membre et administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit être connecté à son compte.
<b>Post condition</b>	L'acteur est déconnecté du site et ses actions en cours non finalisées sont supprimées.
<b>Scénario nominal</b>	<ol style="list-style-type: none"><li>1. L'acteur clique sur le bouton « se déconnecter » dans la barre de navigation</li><li>2. Il doit confirmer son choix de déconnexion.</li></ol>

**Tableau 4: Scénario « Déconnexion »**

## 3.2.2 Diagramme de classe

Un élément central de conception d'un logiciel est le diagramme de classes, c'est un modèle représentant la structure du système à concevoir. Le diagramme de classes est un schéma utilisé pour présenter les classes et les interfaces d'un système ainsi que les différentes relations. Ce diagramme appartient à la partie statique d'UML car il fait abstraction temporels et dynamiques.

Dans cette partie, nous allons vous présenter l'ensemble des classes de notre diagramme de classe. Sachant que nous avons adopté la méthodologie agile « SCRUM », nous allons donc nous focaliser sur la classe appartenant à notre sprint courant.

Soit la classe « Utilisateur » entourée avec un rectangle rouge dans la figue ci-dessous.

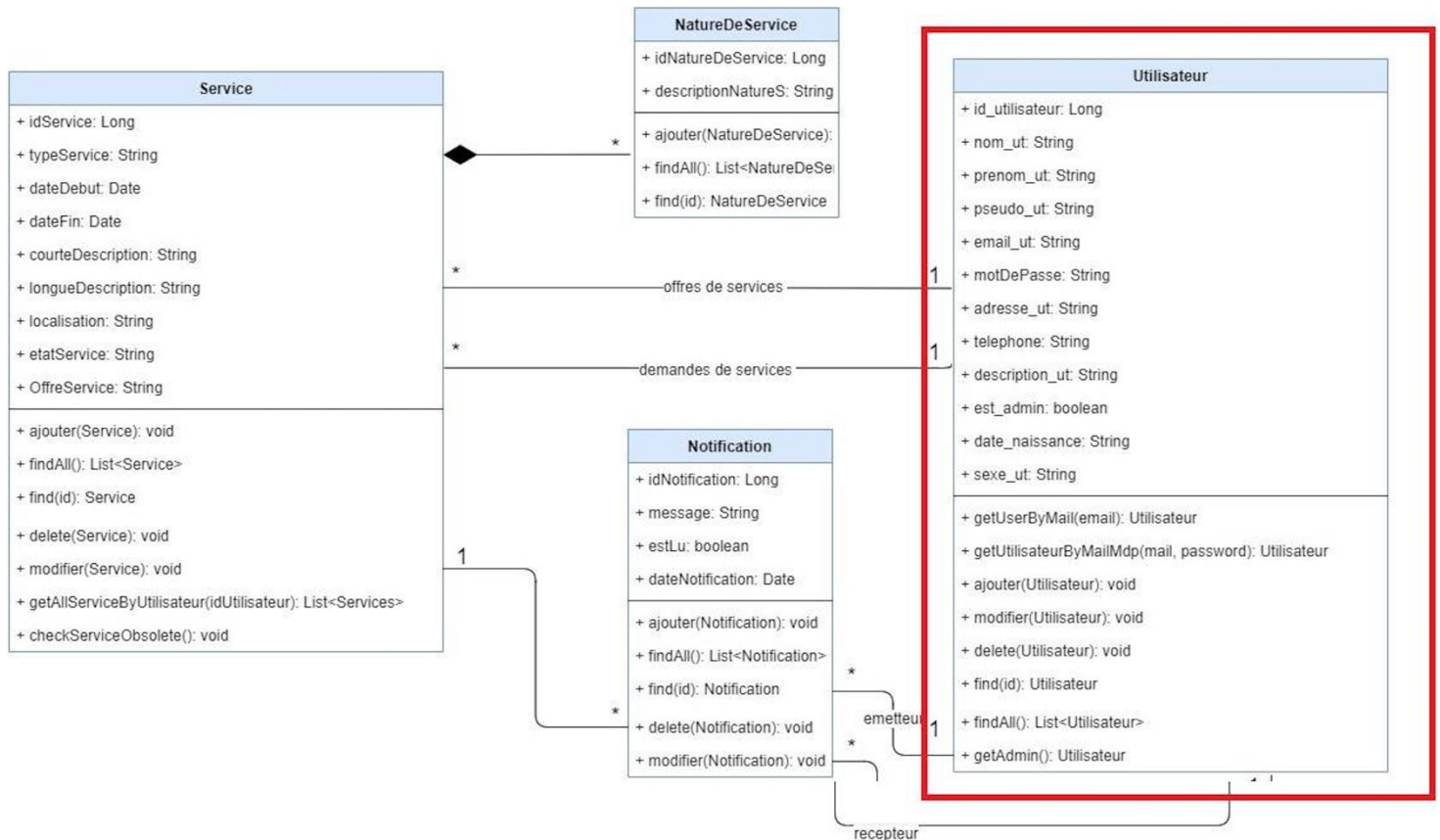


Figure 7: Diagramme de classe visant la classe "Utilisateur"

Nous allons donc décrire notre classe focalisée dans le diagramme de classe avec deux tableaux. Le premier tableau (tableau 5) présente l'ensemble des attributs utilisés dans notre classe. Nous allons définir le type et la description de chaque attribut.

De même pour le deuxième tableau (tableau 6) pour la définition des différentes méthodes.



Attributs		
code	Type	Description
codeUtilisateur	Long	Le code de l'utilisateur.
email	Chaine de caractères	L'adresse mail de l'utilisateur.
adresse	Chaine de caractères	L'adresse de l'utilisateur.
nom	Chaine de caractères	Le nom de l'utilisateur.
prenom	Chaine de caractères	Le prénom de l'utilisateur.
dateNaissance	Date	La date de naissance de l'utilisateur.
login	Chaine de caractères	L'identifiant de l'utilisateur.
mdp	Chaine de caractères	Le mot de passe de l'utilisateur.
photo	Chaine de caractères	La photo de l'utilisateur.
sexe	Chaine de caractère	Le sexe de l'utilisateur
tel	Long	Le numéro de téléphone de l'utilisateur.
estAdmin	Booléen	Un booléen qui détermine le type d'utilisateur : - True : Admin / - False : membre
description	Chaine de caractère	La description de l'utilisateur, détail supplémentaire

Tableau 5: Attributs de la classe "Utilisateur"

Méthodes		
Nom	Type	Description
getUserByMail(email)	Utilisateur	Afficher l'utilisateur dont le mail correspond au paramètre
getUtilisateurByMailMdp(mail, password)	Utilisateur	Afficher l'utilisateur dont le mail et le mot de passe correspond aux paramètres
getAdmin()	Utilisateur	Afficher l'utilisateur qui possède le statut d'administrateur
+ ajouter(Utilisateur)	void	Ajouter un utilisateur
+ modifier(Utilisateur)	void	Modifier les informations de l'utilisateur sélectionné
+ delete(Utilisateur)	void	Supprimer l'utilisateur sélectionné
+ findAll()	List<Utilisateur>	Afficher la liste de tous les utilisateurs
+ find(id)	Utilisateur	Chercher un utilisateur avec son identifiant

Tableau 6: Méthodes de la classe "Utilisateur"

### 3.3 Conception

La phase de conception est la phase initiale de création et de mise en œuvre de notre projet. En fait, elle représente une étape importante de réflexion dans le cycle de développement logiciel. Dans ce sprint nous allons présenter la conception à travers le diagramme de séquence et description des rôles de chaque type et la description des interfaces homme machine (IHM).

#### 3.3.1 Diagrammes de séquences

Les diagrammes de séquences nous permettent de mieux comprendre la communication Utilisateur-Système. Pour cela, nous présenterons dans cette partie les diagrammes de séquences les plus importants.

##### 3.3.1.1 Diagramme de séquence relatif à la création de compte

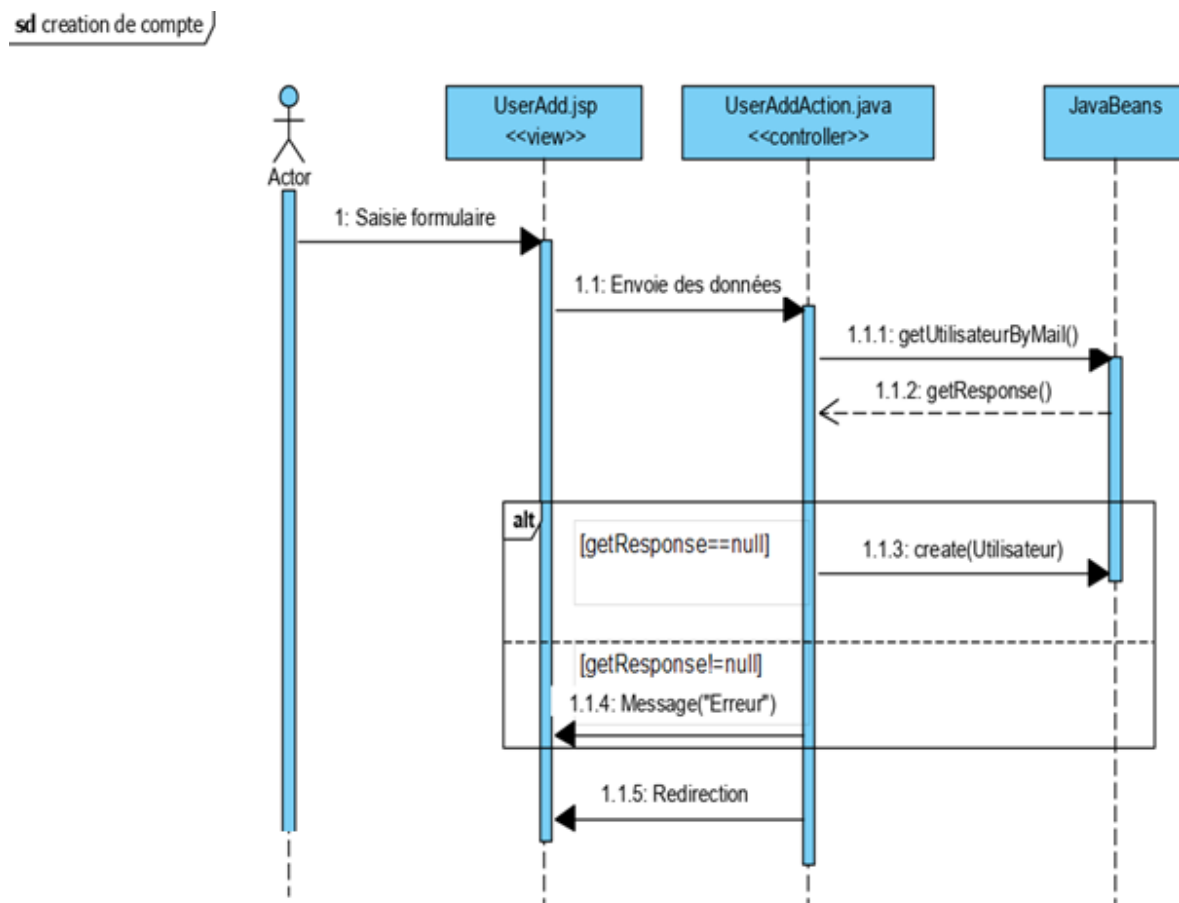


Figure 8: Diagramme de séquence relatif à la création de compte

L'utilisateur se connecte à la page « *UserAdd.jsp* », puis saisit ses informations personnelles pour la création d'un nouveau compte et valide le formulaire.

Les informations sont envoyées vers le Controller « *UserAddAction.java* » qui va vérifier s'il existe déjà un utilisateur avec le même e-mail dans la base de données. Si la réponse est négative, alors un nouveau compte sera créé, sinon un message d'erreur sera renvoyé à l'utilisateur.

### 3.3.1.1 Diagramme de séquence relatif à la gestion de compte

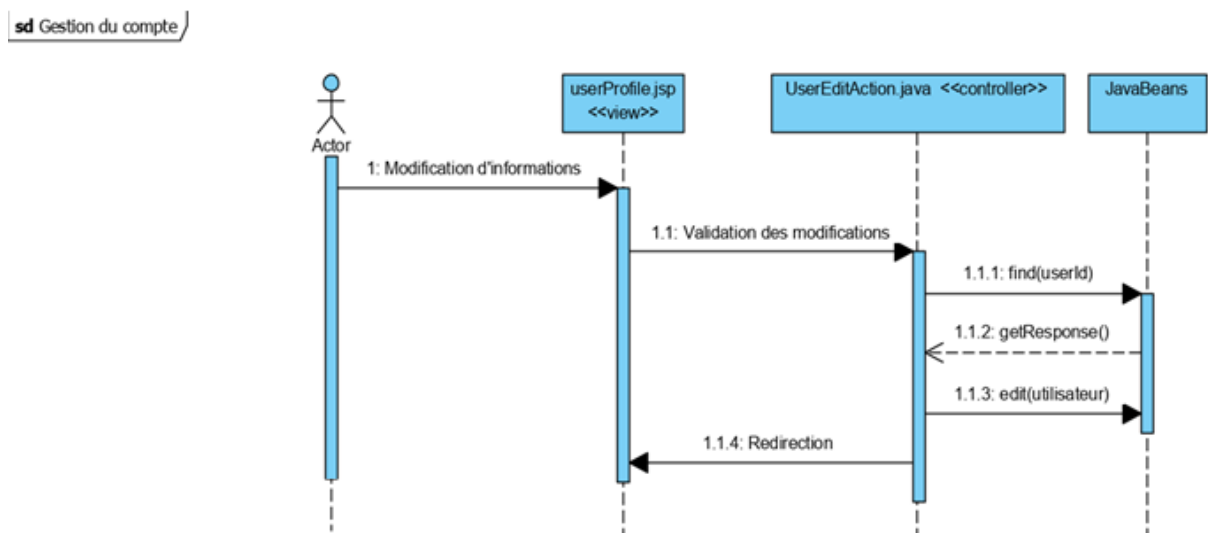


Figure 9: Diagramme de séquence relatif à la gestion de compte

L'utilisateur se connecte à la page « *userProfile.jsp* » pour effectuer les modifications de ses informations personnelles, puis valide ses modifications. Le Controller « *UserEditAction.java* » récupère l'objet de la base de données correspondant à l'utilisateur puis modifie ses données.

### 3.3.1.2 Diagramme de séquence relatif à l'authentification

L'utilisateur se connecte à la page « *login.jsp* », saisit ses identifiants puis valide. Le Controller « *ConnexionAction.java* » va récupérer l'objet correspondant au mail et mot de passe entrés dans le formulaire, de la base de données. Si la réponse renvoyée est « *nulle* », alors l'objet n'existe pas et l'utilisateur est redirigé vers « *login.jsp* » et un message d'erreur sera affiché sinon l'utilisateur est redirigé vers « *accueil.jsp* »

sd Login /

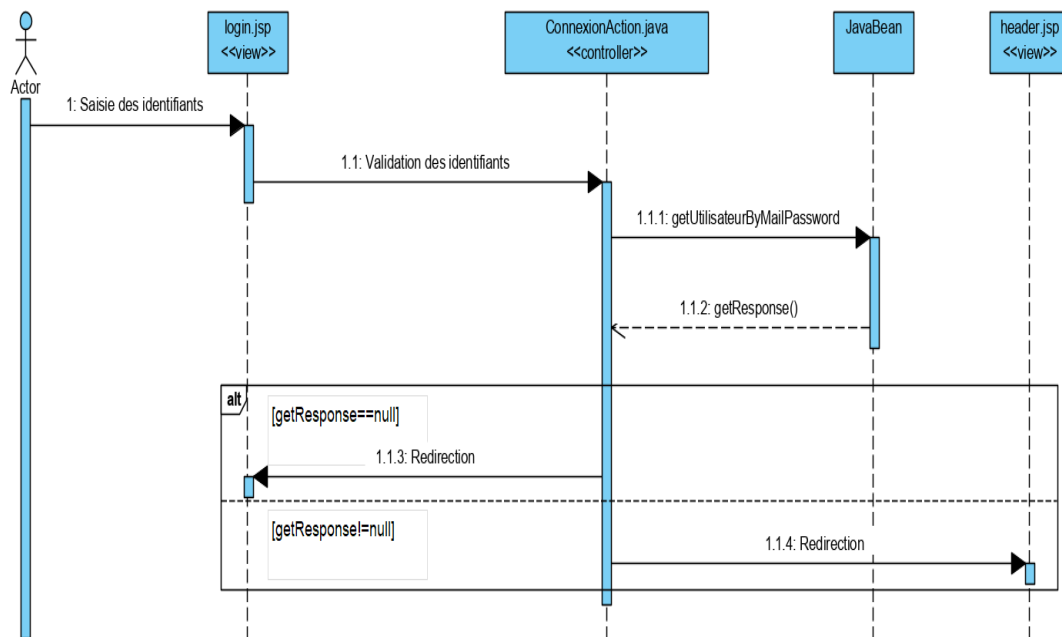


Figure 10: Diagramme de séquence relatif à l'authentification

### 3.3.1.3 Diagramme de séquence relatif à la déconnexion

sd logout /

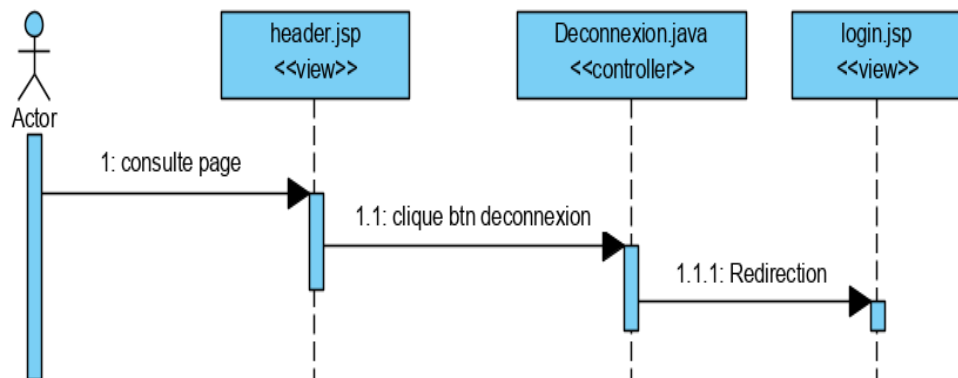


Figure 11: Diagramme de séquence relatif à la déconnexion

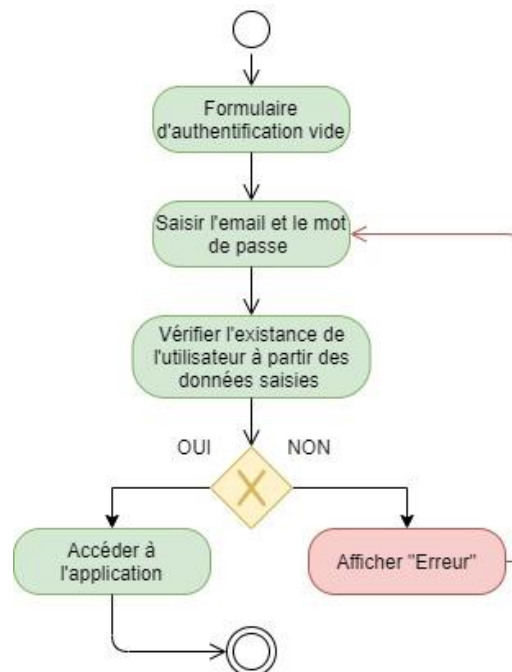
L'utilisateur clique sur le bouton de déconnexion dans la barre de navigation. Le Controller « *Deconnexion.java* » va le rediriger vers la page « *login.jsp* ».

### 3.3.2 Diagrammes d'activités

Les diagrammes d'activités sont les représentations proches de l'organigramme. La description d'un cas d'utilisation par un diagramme d'activités correspond à sa traduction algorithmique. Une activité est l'exécution d'une partie de cas d'utilisation. Dans la suite, nous présentons quelques diagrammes d'activités de notre système.

#### 3.3.2.1 Diagramme d'activités relatif à l'authentification

Pour que l'utilisateur puisse accéder au système, il doit s'authentifier en saisissant son login et son mot de passe. Une fois connu, l'utilisateur accède au menu des fonctionnalités qui lui sont offertes. Le processus d'authentification peut être résumé dans le diagramme d'activités suivant :



**Figure 12: Diagramme d'activités relatif à l'authentification**

### 3.3.2.2 Diagramme d'activités relatif à la gestion de compte

La gestion de compte est considérée comme l'une des tâches principales de l'utilisateur. Le processus de gestion de compte peut être résumé dans le diagramme d'activité décrit dans la figure suivante :

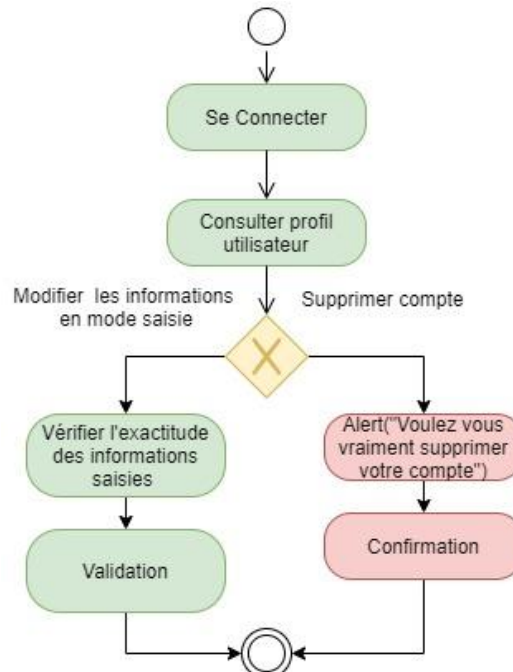


Figure 13: Diagramme d'activités relatif à la gestion de compte

### 3.3.3 Description de l'IHM

#### 3.3.3.1 Page d'accueil du site web

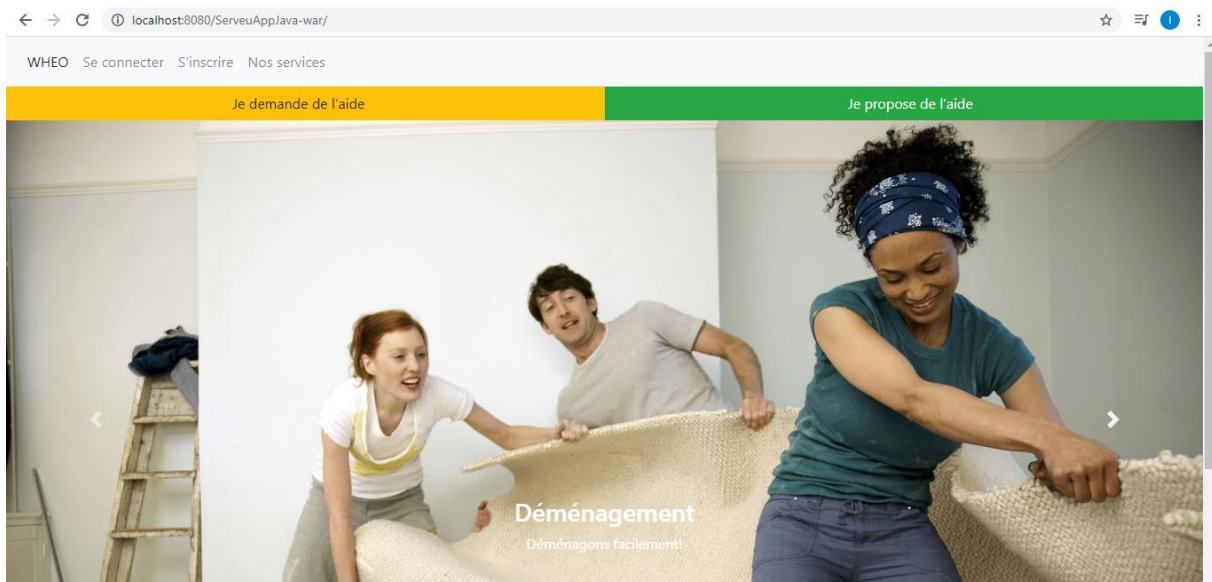


Figure 14: Page d'accueil du site web

Cette interface sera affichée au moment où l'utilisateur saisit l'adresse de notre site dans la barre de l'url du navigateur. A partir de cette page, le visiteur peut consulter les demandes ou

les offres à l'état « VALIDE » avec des accès limités en cliquant sur le bouton « Nos services » dans la barre de navigation. Il peut aussi créer un compte en cliquant sur « S'inscrire » pour avoir le droit de demander ou proposer des services.

### 3.3.3.2 Page d'inscription

The screenshot shows a web browser window with the URL `localhost:8080/ServeurApp/Java-war/UserAddAction`. The page has a navigation bar with the text "WHEO" and links "Se connecter" and "S'inscrire". The main content area is titled "Inscription" and contains a form with the following fields:

- Prénom (text input)
- Nom (text input)
- Pseudonyme (text input with a user icon)
- Date de naissance (date input, showing 03/01/2020)
- Sexe (radio buttons for Feminin and Masculin)
- Téléphone (text input)
- Pays (text input)
- Ville (text input)
- Code Postale (text input)
- E-mail (text input with an @ icon)
- Mot de passe (text input)
- Description personnelle (text area)
- ☐ Accepter les termes et conditions
- 

The footer of the page contains the text "2020 Copyright: WHEO" on the left, and links "Contactez nous" and "We Help Each Other" on the right.

**Figure 15: Page d'inscription**

A travers cette interface le visiteur peut créer un compte. Pour que le compte soit créé avec succès, l'utilisateur doit saisir un numéro de téléphone, une adresse e-mail, une date de naissance et un code postal valides. Les champs nom, prénom, pays et ville doivent contenir que des lettres alphabets. Il doit aussi remplir tous les champs et cocher la case « Accepter les termes et conditions ».

### 3.3.3.3 Page d'authentification

Cette interface sert à restreindre l'accès à l'application et à délimiter ce mode d'accès selon les privilèges accordés à l'utilisateur. L'interface ci-dessous présente notre page d'authentification. Il s'agit de saisir l'identifiant et le mot de passe pour pouvoir accéder aux fonctionnalités accordées selon les droits d'accès

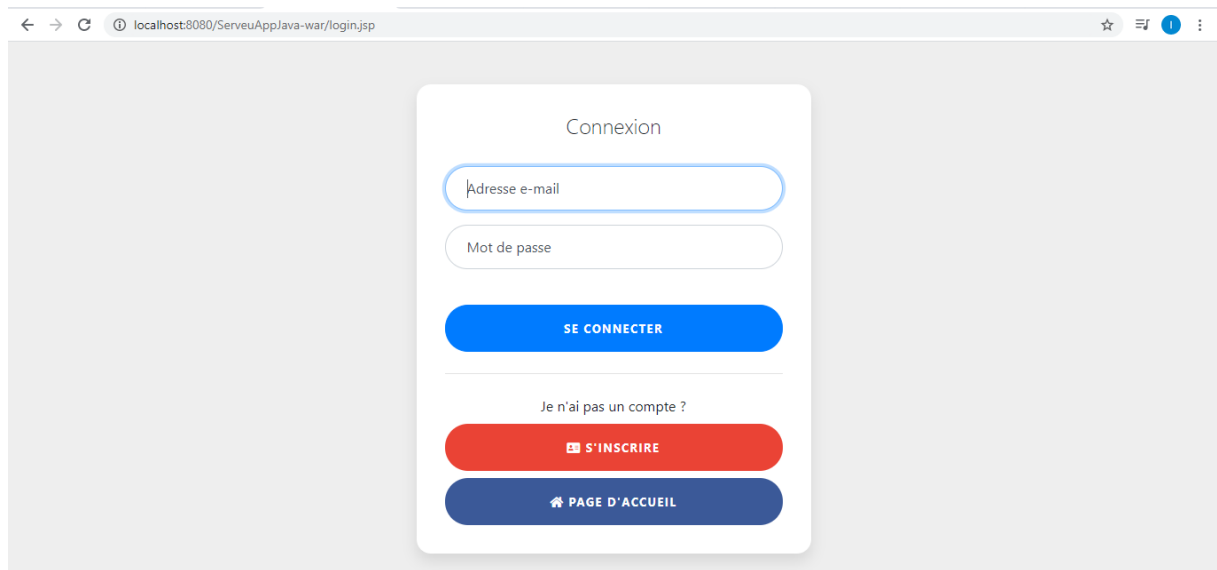


Figure 16: Page d'authentification

### 3.3.3.4 Page d'accueil du membre / administrateur

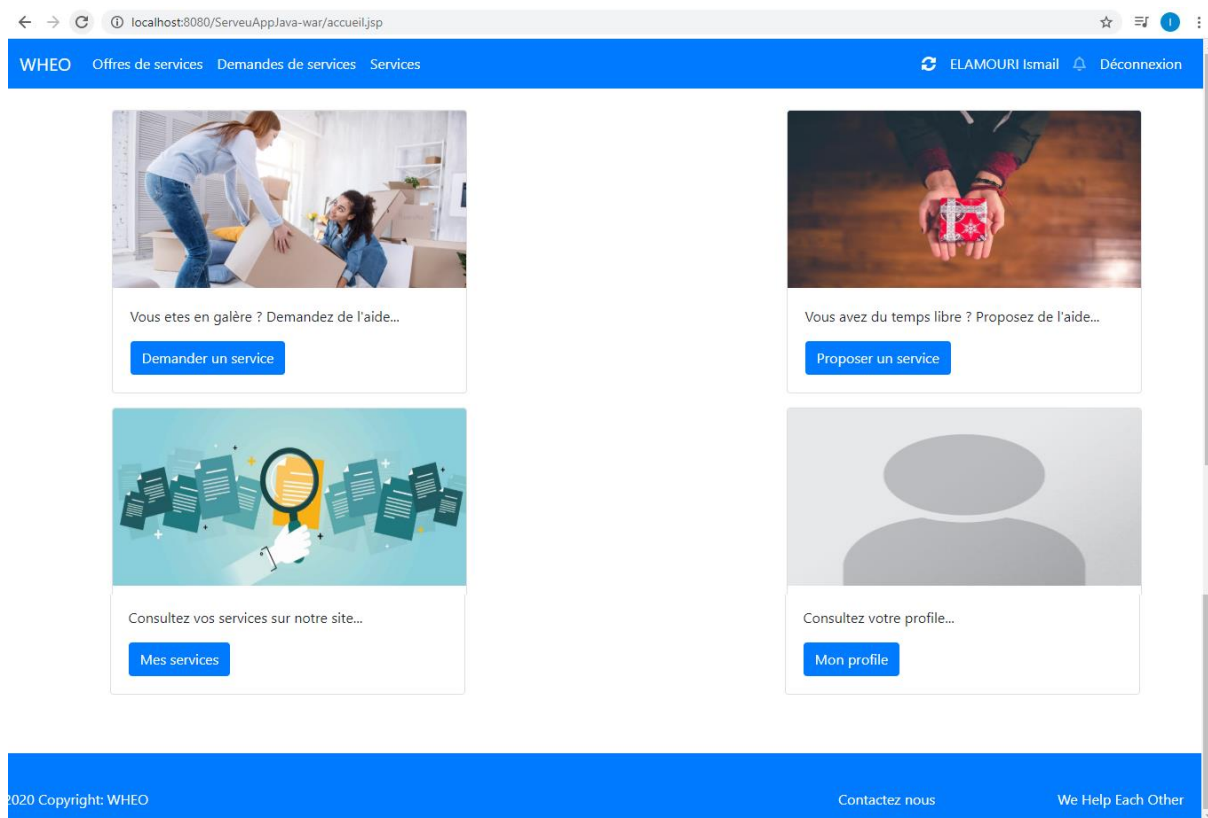


Figure 17: Page d'accueil du membre / administrateur

Une fois que l'utilisateur est connecté, il sera redirigé vers cette interface. À partir de cette interface l'utilisateur peut effectuer plusieurs tâches dont la consultation de son profil.



### 3.3.3.5 Profile d'un membre / administrateur

The screenshot shows a web application interface for a user profile. At the top, a blue navigation bar contains the text 'WHEO Offres de services Demandes de services Services' on the left and 'Paul VONG Déconnexion' on the right. The main content area is titled 'Profile de Paul VONG'. It contains several input fields for personal information: 'Prénom' (Paul), 'Nom' (VONG), 'Pseudonyme' (vpal), 'Date de naissance' (29 févr. 2020), 'Téléphone' (0122334455), 'Pays' (France), 'Ville' (Paris), 'Code Postale' (75011), 'E-mail' (ismail.elamouri@dauphine.eu), and 'Mot de passe'. There is also a 'Description personnelle' text area containing 'Etudiant en M2 Miage à l'université Paris Dauphine'. At the bottom of the form are two buttons: 'Enregistrer' (blue) and 'Supprimer votre compte' (red). The footer of the page is a blue bar with '2020 Copyright: WHEO' on the left, 'Contactez nous' in the center, and 'We Help Each Other' on the right.

**Figure 18: Profile d'un membre / administrateur**

À travers cette interface le membre du site peut modifier ses informations, ainsi que supprimer son compte. L'administrateur peut uniquement modifier ses données. Nous avons jugé que la suppression d'un compte administrateur est impossible car aucune demande ou offre de service peut être acceptée. Pour cela nous avons enlevé le bouton « Supprimer votre compte » pour le compte de l'administrateur. Certaines données comme le nom, le prénom... ne sont pas modifiables. Nous avons géré les erreurs comme dans l'interface d'inscription.

## 3.4 Evaluation

Lors de cette phase nous avons montré que tous les utilisateurs, bien que leurs rôles soient différents, peuvent accéder à la même interface. Nous avons géré les rôles avec le type de l'utilisateur connecté.

## CONCLUSION

Au cours de ce sprint nous avons présenté l'analyse, la conception et la réalisation de la gestion des comptes utilisateurs ainsi que le principe d'authentification. Nous allons à présent aborder la question de la gestion des offres et des demandes de service.

## Chapitre 4 *Sprint 2 : Gestion des offres et des demandes de* *services*

---

---

### 4.1 Objectifs

La gestion des offres et des demandes est un processus important pour réaliser notre site WEHO. Le but de ce sprint est de créer des offres et des demandes de service, les visualiser et, les proposer et accepter.

### 4.2 Analyse

Nous allons maintenant présenter les diagrammes des cas d'utilisation des processus (Création de l'offre, création de la demande, visualisation des offres et des demandes, proposition de service, acceptation d'une offre de service) ainsi que le diagramme de classe.

#### 4.2.1 Diagramme de cas d'utilisation et scénarios

##### 4.2.1.1 Diagramme de cas d'utilisation

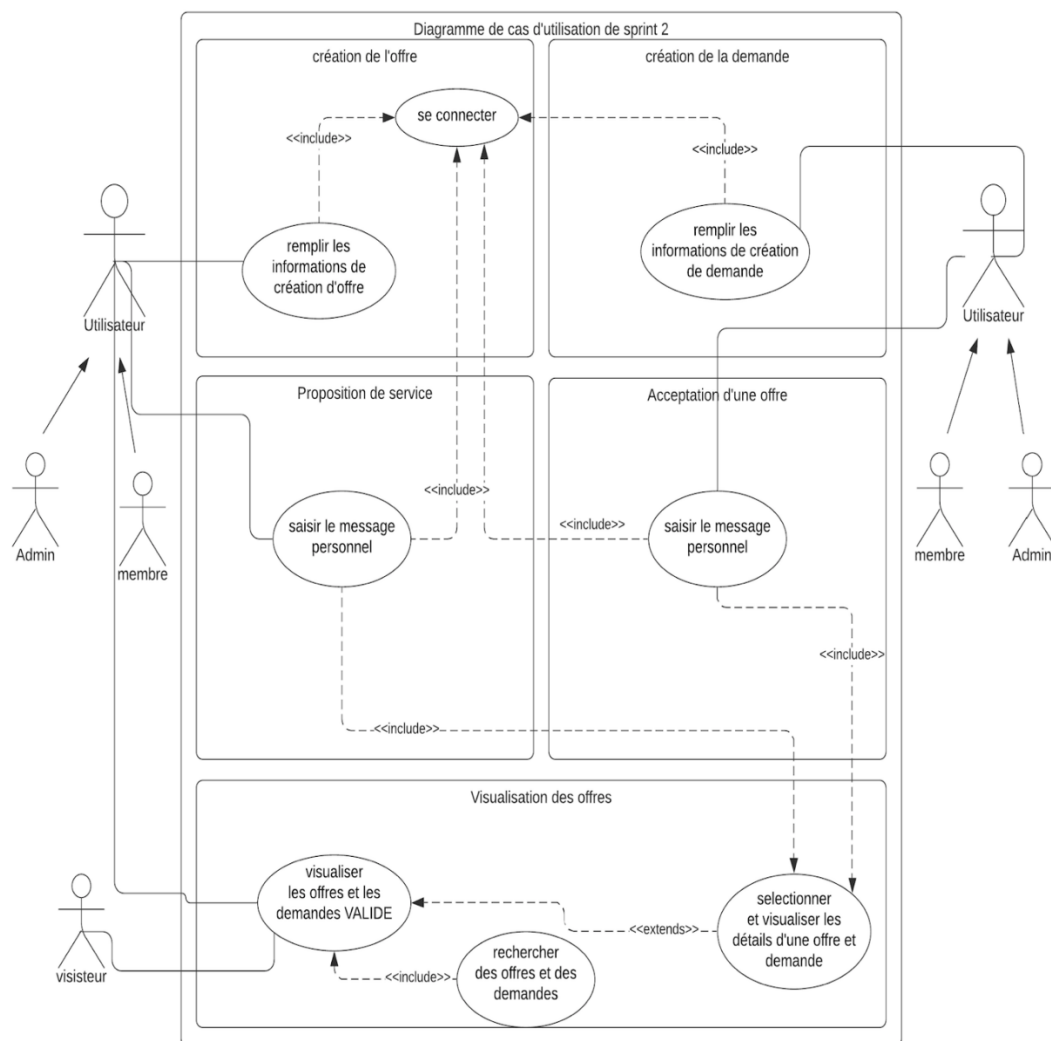


Figure 19: Diagramme de cas d'utilisation relatif au package « Gestion des services »

#### 4.2.1.2 Les scénarios

##### 4.2.1.2.1 Scénario « Création de l'offre / Création de la demande »

Pour ne pas dupliquer le code nous avons utilisé une présentation « JSP » et une servlet pour créer l'offre ou la demande de service. En fonction du bouton sélectionné, le service sera défini comme une offre ou une demande.

<b>Cas d'utilisation</b>	<b>Création de l'offre</b>
<b>Acteurs</b>	Membre, Administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit posséder un compte et doit être connecté sur le site.
<b>Post condition</b>	L'offre a été créé avec succès, le service passe en état A_VALIDER
<b>Scénario nominal</b>	<ol style="list-style-type: none"> <li>1. L'acteur doit se connecter à son compte à partir de lien dans sa page d'accueil</li> <li>2. Il doit remplir tous champs obligatoires sur le formulaire de création de l'offre.</li> <li>3. Il valide la saisie.</li> </ol>
<b>Exception</b>	<p>E1 : Saisie incorrecte : les erreurs sur les données saisies, par exemple, le format de la date ou de la période de validité du service n'est pas correct</p> <p>⇒ Affiche un message d'erreur.</p>

**Tableau 7: Scénario « Création de l'offre / Création de la demande »**

##### 4.2.1.2.2 Scénario « Visualisation des offres et des demandes de services »

Pour cette partie nous avons donné plusieurs choix à l'utilisateur pour visualiser les offres ou / et les demandes. Il peut donc afficher soit toutes les demandes, soit toutes les offres, soit tous les services à l'état « VALIDE » Un service « VALIDE » est un service accepté par l'administrateur du site web. Dans le cas où l'utilisateur est un visiteur il peut juste visualiser tous les services.

<b>Cas d'utilisation</b>	<b>Visualisation des offres et des demandes de services</b>
<b>Acteurs</b>	Visiteur, Membre, Administrateur
<b>Pré condition</b>	Aucune
<b>Post condition</b>	Le service a été affiché sous forme d'un tableau. Le résultat peut être un tableau vide si aucun service n'est trouvé avec des critères de recherche.
<b>Scénario nominal</b>	<ol style="list-style-type: none"> <li>1. L'acteur se rend sur la barre de navigation <ol style="list-style-type: none"> <li>a. Acteur = visiteur : Il clique sur « Nos services »</li> <li>b. Sinon :</li> </ol> </li> </ol>

	<ol style="list-style-type: none"> <li>i. Il clique sur Offre pour afficher les offres</li> <li>ii. Il clique sur Demande pour afficher les demandes</li> <li>iii. Il clique sur Services pour afficher tous les services</li> <li>c. Les services affichés sont tous à l'état VALIDE.</li> </ol> <ol style="list-style-type: none"> <li>2. Il peut rechercher des offres et des demandes en ajoutant des critères de recherche.</li> <li>3. Il peut également sélectionner un service parmi les services affichés et visualiser les détails de service.</li> </ol>
--	---

**Tableau 8: Scénario « Visualisation des offres et des demandes de services »**

#### 4.2.1.2.3 Scénario « Proposition d'un service / Acceptation d'une offre »

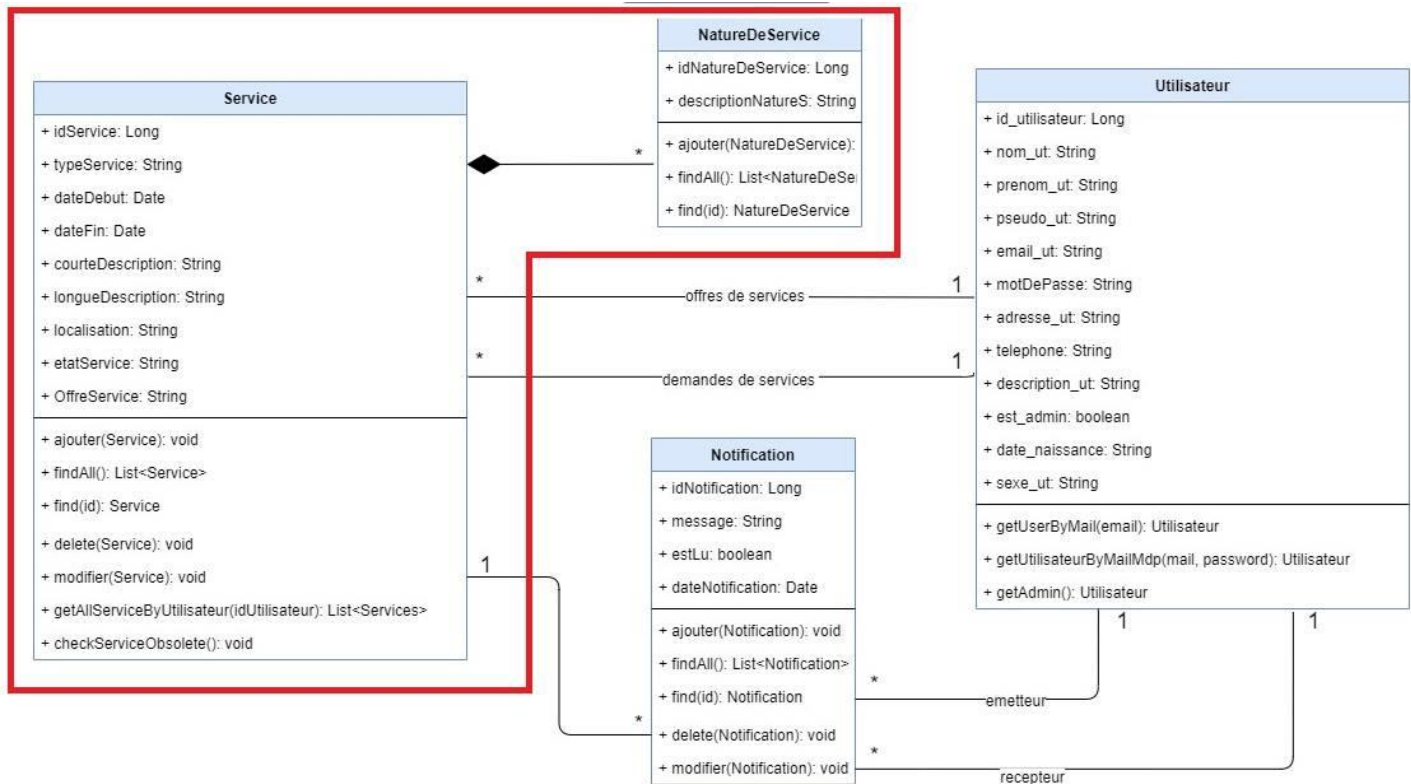
Pour une raison de ne pas dupliquer le code, nous avons géré les deux processus « Proposition de service » et « Acceptation de service » en fonction du type de service sélectionné (offre ou demande). Dans le cas où c'est une offre de service le bouton « Accepter service » sera affiché et vice versa.

<b>Cas d'utilisation</b>	<b>Proposition de service / Acceptation d'un service</b>
<b>Acteurs</b>	Membre, Administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit posséder un compte et doit être connecté sur le site.
<b>Post condition</b>	Le service a été proposé ou accepté avec succès et passe en état EN_COURS.
<b>Scénario nominal</b>	<ol style="list-style-type: none"> <li>1. L'acteur doit se connecter puis se rendre sur la barre de menu</li> <li>2. Il clique sur l'onglet Offre ou / et Demande</li> <li>3. Il visualise les offres ou / et les demandes de service qui se trouvent dans l'état VALIDE.</li> <li>4. Il peut rechercher des offres et des demandes en ajoutant des critères de recherche.</li> <li>5. Il peut également sélectionner un service parmi les services affichés et visualiser les détails de service.</li> <li>6. Il clique sur « Proposer ce service » ou « Accepter ce service »</li> <li>7. L'application demande à l'acteur de saisir un message personnel</li> <li>8. Une notification sera envoyée à l'autre utilisateur avec les coordonnées du créateur de service et vice versa.</li> </ol>
<b>Exception</b>	<p>E1 : gestion de concurrence : - A propose / accepte un service X - ensuite B propose / accepte un service X</p> <p>→ Message d'erreur sera affiché à B</p>

**Tableau 9: Scénario « Proposition d'un service / Acceptation d'une offre »**

### 4.2.2 Diagramme de classe

Dans cette partie nous allons donc nous focaliser sur les classes appartenant à notre sprint courant. Soit les classes « Service » et « NatureDeService » entourées avec un rectangle rouge dans la figure ci-dessous.



**Figure 20: Diagramme de classe visant les classes "Service" et "NatureDeService"**

Nous allons donc décrire nos classes focalisées dans le diagramme de classe avec deux tableaux. Les tableaux 10 et 12 présentent l'ensemble des attributs utilisés dans notre les classes « Service » et « NatureDService ». Nous allons définir le type et la description de chaque attribut.

De même pour les deux autres tableaux 11 et 13 pour la définition des différentes méthodes.

Attributs		
Nom	Type	Description
idNatureDeService	Long	L'identifiant de la nature de service
descriptionNatureDeService	String	La description la nature de service

**Tableau 10: Attributs de la classe "NatureDeService"**

Méthodes		
Nom	Type	Description
<b>ajouter(NatureDeService)</b>	void	Ajouter une nature de service
<b>findAll()</b>	List< NatureDeService >	Afficher la liste de toutes les natures de service
<b>find(id)</b>	NatureDeService	Chercher une nature de service avec son identifiant

Tableau 11: Méthodes de la classe "NatureDeService"

Attributs		
Nom	Type	Description
<b>idService</b>	Long	La référence du service
<b>typeService</b>	String	Type de service : <ul style="list-style-type: none"> <li>• Service à offrir</li> <li>• Objet à prêter</li> <li>• Objet à donner</li> </ul>
<b>dateFin</b>	Date	Indique la date finale du service
<b>dateDebut</b>	Date	Indique la date de début du service
<b>courteDescription</b>	String	Courte description du service
<b>longueDescription</b>	String	Description détaillée du service
<b>etatService</b>	String	État du service : <ul style="list-style-type: none"> <li>• A_VALIDER</li> <li>• VALIDE</li> <li>• EN_COURS</li> <li>• FINALISE</li> <li>• OBSOLETE</li> </ul>
<b>localisation</b>	String	Le lieu où le service va se dérouler
<b>OffreService</b>	String	Permet de distinguer entre les offres et les demandes

Tableau 12: Attributs de la classe "Service"

Méthodes		
Nom	Type	Description
<b>getAllServiceByUtilisateur(idUtilisateur):</b>	List<Services>	Afficher la liste des services de l'utilisateur sélectionné
<b>checkServiceObsolete()</b>	void	Rendre un service à l'état « OBSOLETE » lorsque la date de fin est antérieure à la date courante.
<b>ajouter(Service)</b>	void	Ajouter un service
<b>modifier(Service)</b>	void	Modifier les informations du service sélectionné
<b>delete(Service)</b>	void	Supprimer le service sélectionné
<b>findAll()</b>	List< Service >	Afficher la liste de tous les service
<b>find(id)</b>	Service	Chercher un service avec sa référence

Tableau 13: Méthodes de la classe "Service"

### 4.3 Conception

Cette partie a pour objectif de définir la solution conceptuelle de ce sprint. Nous allons présenter en premier lieu une vue dynamique grâce au diagramme de séquences pour les scénarios d'exécution les plus importants et les descriptions D'IHM.



#### 4.3.1 Diagrammes de séquences

##### 4.3.1.1 Diagramme de séquence relatif à la création d'une offre / demande de service

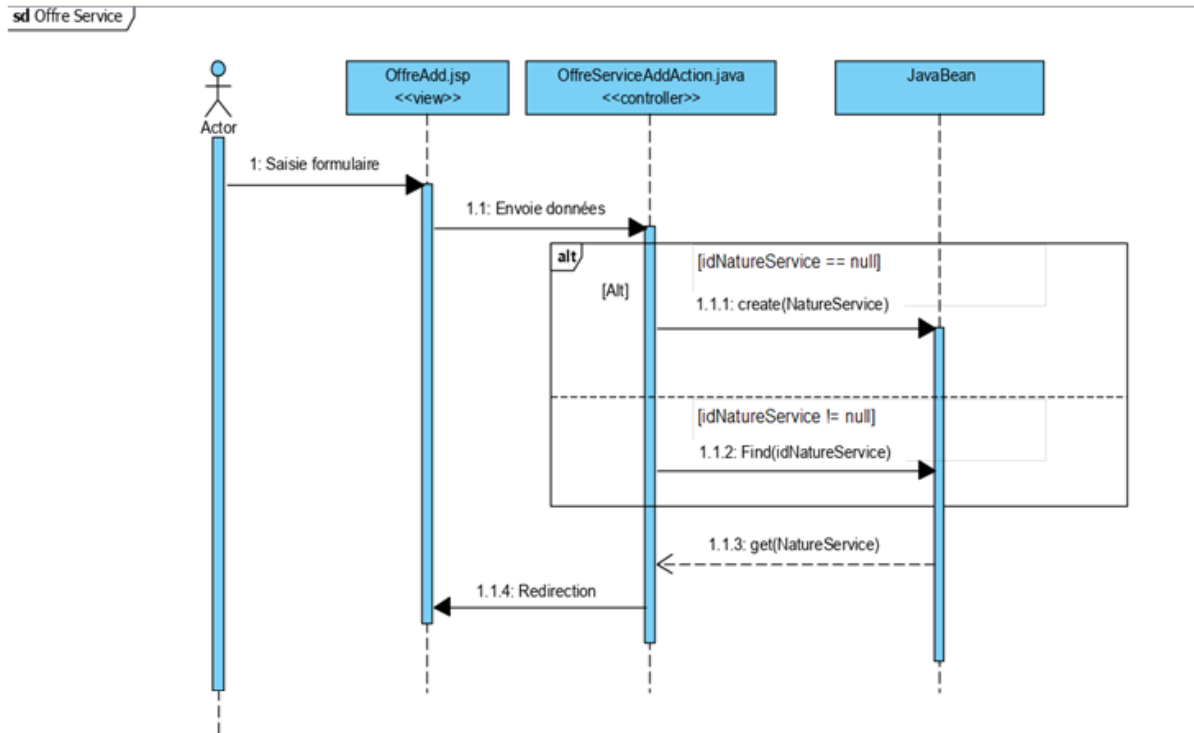


Figure 21: Diagramme de séquence relatif à la création d'une offre / demande de service

L'utilisateur se connecte sur la page « *OffreAdd.jsp* ». Il saisit les informations du service à proposer (ou à demander) puis valide le formulaire. Le Controller « *OffreServiceAddAction.java* » vérifie si l'objet « *NatureService* » du service offert (ou demandé) existe dans la BDD. S'il n'existe pas, alors un nouvel objet « *NatureService* » sera ajouté dans la BDD. Il finit par récupérer l'objet de la BDD puis redirige l'utilisateur vers « *OffreAdd.jsp* ».

##### 4.3.1.2 Diagramme de séquence relatif à la visualisation des offres ou demandes de services

L'utilisateur se connecte au Controller « *AllServicesShow.java* ». Le controller récupère les services de la BDD puis, si les services sont valides, il les affiche sur « *AllServicesShow.jsp* ».

- Si le champ « *rechercher* » est différent de « *null* » les services affichés seront filtrés.
- Si l'utilisateur clique sur l'icône « Œil », le Controller va se charger de récupérer les détails du service et les afficher dans la vue « *OffreShowDetails* »

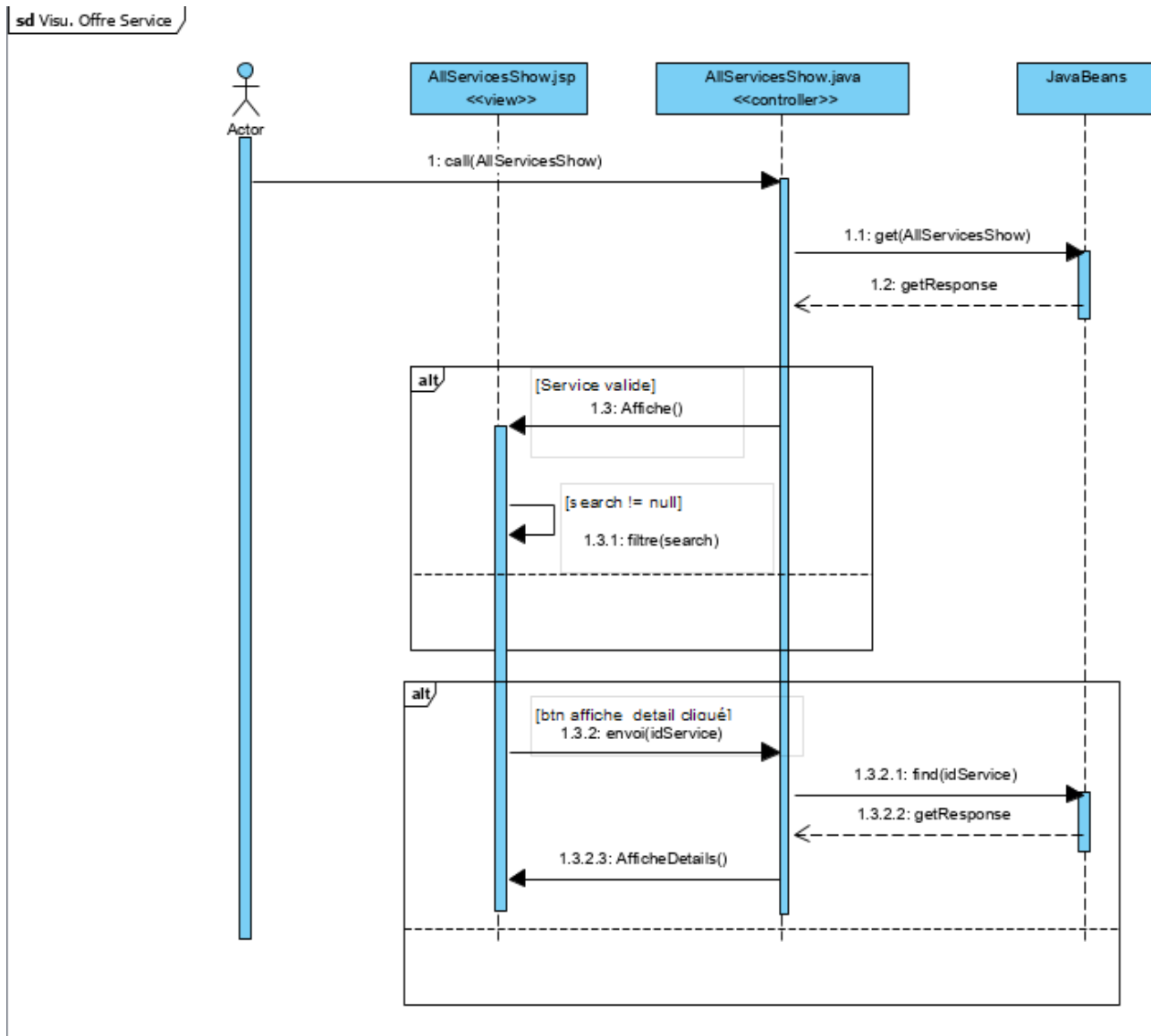


Figure 22: Diagramme de séquence relatif à la visualisation des offres ou demandes de services

### 4.3.1.3 Diagramme de séquence relatif à la proposition ou l'acceptations du service

L'utilisateur accède à la page « *OffreShow.jsp* » puis clique sur le sur l'icône « Œil », puis sur le bouton *accepter* (ou *proposition*). Il accède à la page « *messageProposition.jsp* » puis valide son message. Le Controller des objets « *Utilisateur* » et « *Service* » vérifie si l'état de l'offre est différent de « EN\_COURS ». Si c'est le cas, des notifications seront envoyées aux parties prenantes. Sinon un message d'erreur sera envoyé à l'utilisateur qui a accepté l'offre (ou demandé l'offre). L'utilisateur sera ensuite redirigé vers « *messageProposition.jsp* ».

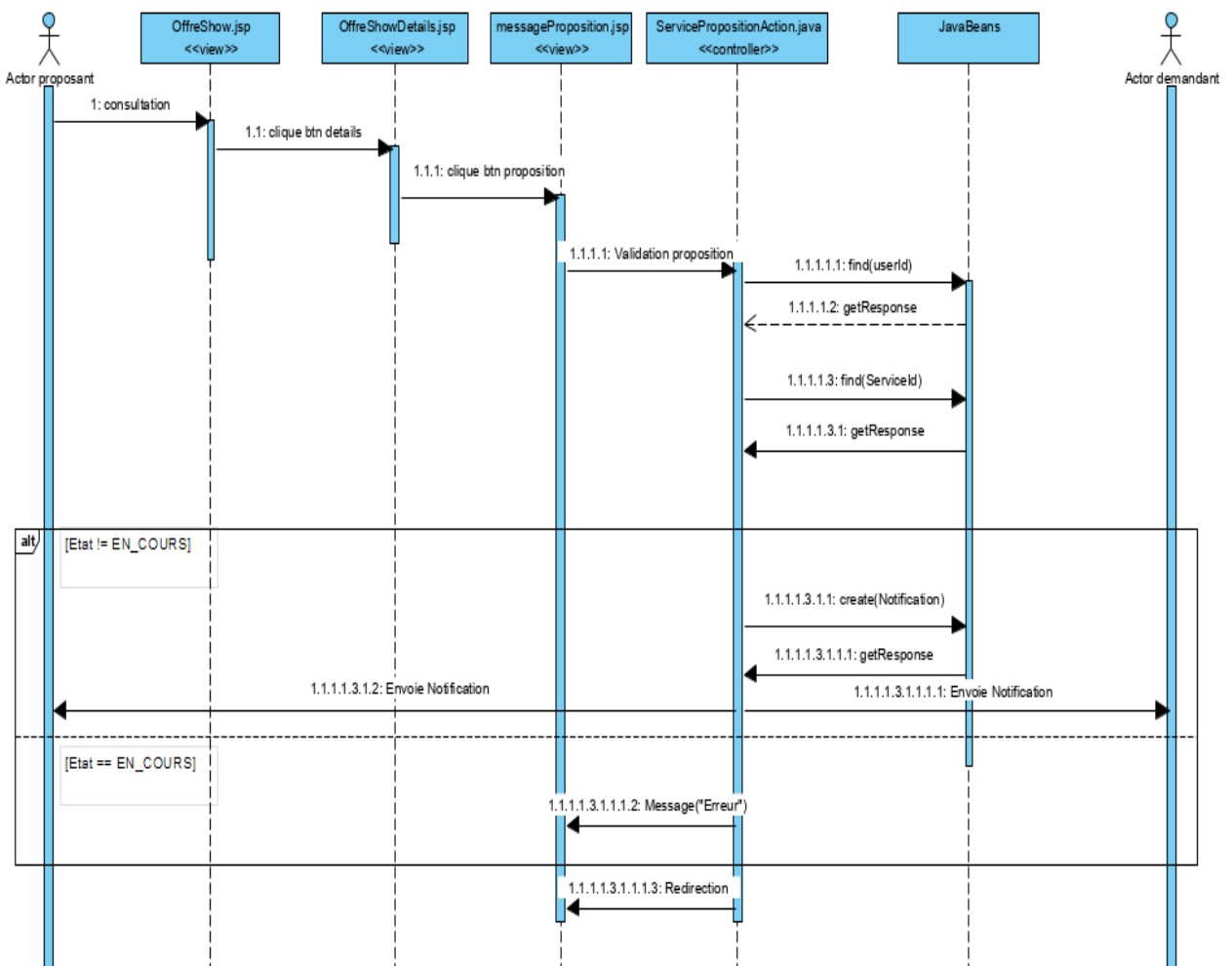
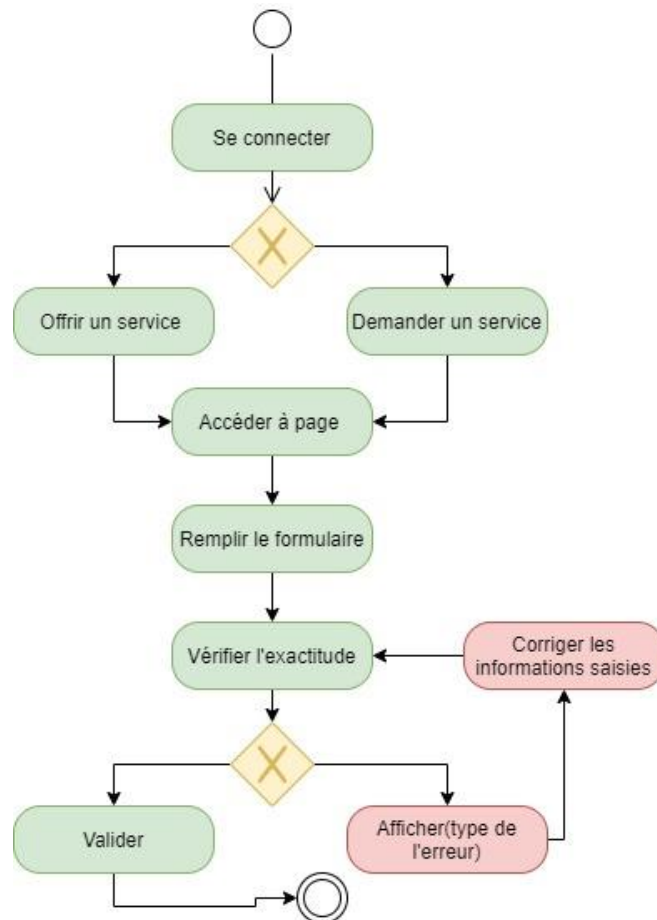


Figure 23: Diagramme de séquence relatif à la proposition ou l'acceptations du service

### 4.3.2 Diagrammes d'activités

#### 4.3.2.1 Diagramme d'activité relatif à la création des services

Dans le cadre de notre projet, l'ajout d'une offre ou d'une demande de service est l'une des tâches les plus importantes. Le processus d'ajout peut être résumé dans le diagramme d'activité décrit dans la figure suivante :



**Figure 24 : Diagramme d'activité relatif à la création des services**

#### 4.3.2.2 Diagramme d'activité relatif à la gestion des services (Propositions / acceptations)

Ce processus permet aux utilisateurs de l'application de s'entraider soit en acceptant une offre ou en proposant une demande. La figure ci-dessous résume ce phénomène.

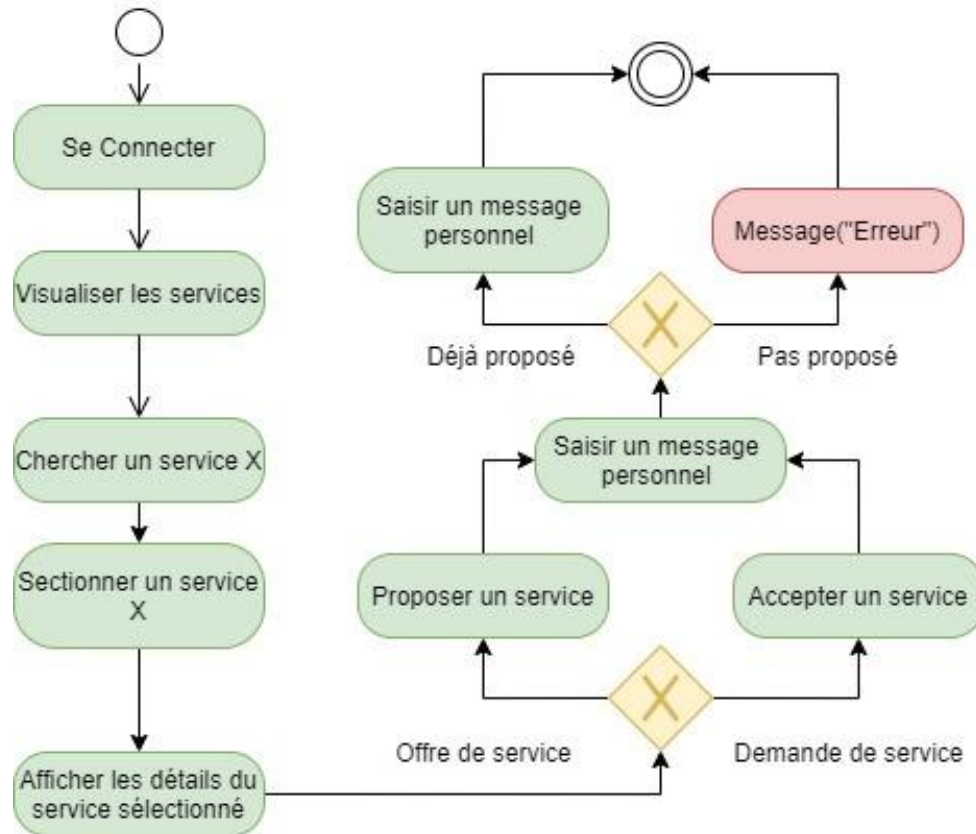


Figure 25: Diagramme d'activité relatif à la gestion des services (Propositions / acceptations)

#### 4.3.3 Description de l'IHM

##### 4.3.3.1 Page de création d'une demande ou d'une offre de service

Cette interface permet à l'utilisateur (membre ou administrateur) de créer une offre ou une demande de service. Elle est accessible via un bouton appelant « offrir un service » ou « demander un service » dans la barre de navigation. L'utilisateur remplit tous les champs. Au moment de la validation le système vérifie l'exactitude des données. Par exemple l'utilisateur ne peut pas choisir une date de fin antérieure à la date de début de service. De même principe que l'inscription pour les champs de l'adresse. Dans le cas où l'utilisateur ne trouve pas la nature de service, il clique sur l'option « Ajouter une nature de service » dans la liste des natures de service. Il saisit donc la nouvelle valeur dans le champ qui sera affiché

The screenshot shows a web application interface for adding a service offer. The header bar is blue with the text 'WHEO Offres de services Demandes de services Services' and user information 'ELAMOURI Ismail Déconnexion'. The form is titled 'Ajouter une offre de service' and contains the following fields:

- Type de service:** A dropdown menu with 'Service à offrir' selected.
- Nature de service:** An empty dropdown menu.
- Date de début:** A date picker.
- Date de fin:** A date picker.
- Description courte du service:** A text input field.
- Description Détaillée:** A large text area.
- Pays:** A text input field.
- Ville:** A text input field.
- Code Postale:** A text input field.

An 'Ajouter' button is located at the bottom left of the form. The footer bar is blue with the text '2020 Copyright: WHEO', 'Contactez nous', and 'We Help Each Other'.

Figure 26: Page de création d'une demande ou d'une offre de service

### 4.3.3.2 Page de la visualisation des services

Cette interface permet de visualiser toute les demandes et les offres à l'état « VALIDE ». Elle est accessible via le bouton « services » dans la barre de navigation.

The screenshot shows a web application interface for viewing all valid service offers and requests. The header bar is blue with the text 'WHEO Offres de services Demandes de services Services' and user information 'ELAMOURI Ismail Déconnexion'. The page title is 'Toutes les offres et les demandes de services'. Below the title is a search bar and a set of icons for filtering and sorting. The main content is a table with the following columns:

ID	Offre / Demande	Type de service	Nature de service	Description du service	Action
112	Offre de service	Service à offrir	Cours	Cours Anglais	
113	Demande de service	Objet à prêter	Jardinage	Pele	

Below the table, it says 'Showing 1 to 2 of 2 rows'. The footer bar is blue with the text '2020 Copyright: WHEO', 'Contactez nous', and 'We Help Each Other'.

Figure 27: Page de la visualisation des services

### 4.3.3.3 Page d'affichage d'un service sélectionné

Via l'interface précédente l'utilisateur appuie sur l'icône « œil », il sera redirigé vers la page des détails.

- Dans le cas où le service sélectionné est une offre de service

WHEO Offres de services Demandes de services Services Paul VONG Déconnexion

Détails de l'offre de service - Réf: 112

Proposé par: elamis  
Type de service: Service à offrir  
Nature de service: Cours  
Période: **Du** 27-03-2020 **Au** 27-03-2020  
Description: Cours Anglais  
Description détaillée: Donner un cours d'anglais  
Localisation: France,Paris,75011

[Accepter un service](#)

2020 Copyright: WHEO [Contactez nous](#) [We Help Each Other](#)

**Figure 28: Page d'acceptation d'une offre de service**

- Dans le cas où le service sélectionné est une demande de service

WHEO Offres de services Demandes de services Services Paul VONG Déconnexion

Détails de la demande de service - Réf: 113

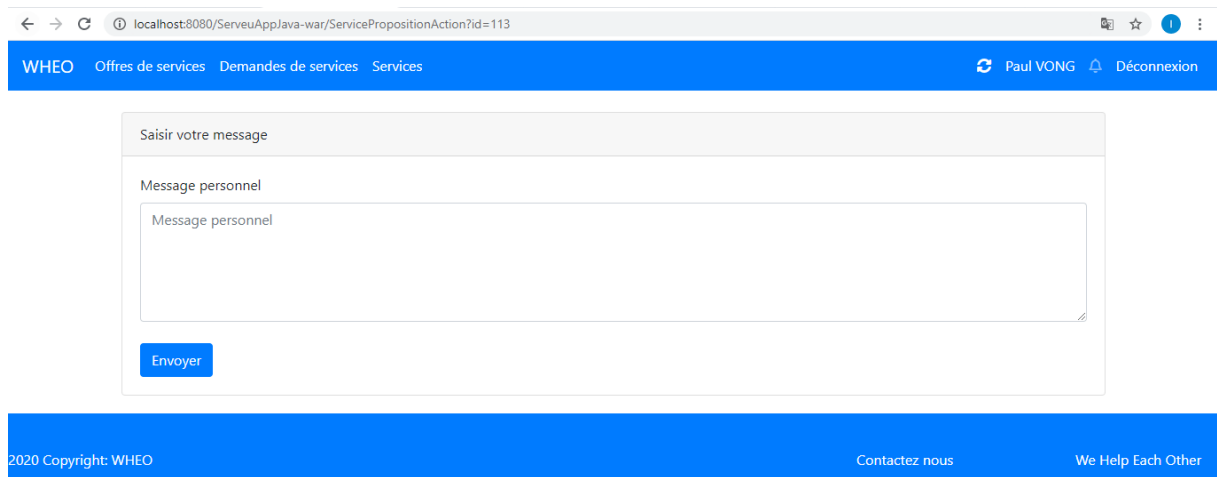
Proposé par: elamis  
Type de service: Objet à prêter  
Nature de service: Jardinage  
Période: **Du** 14-03-2020 **Au** 14-03-2020  
Description: Pele  
Description détaillée: J'ai besoin d'une pele  
Localisation: France,Chelles,77000

[Proposer un service](#)

2020 Copyright: WHEO [Contactez nous](#) [We Help Each Other](#)

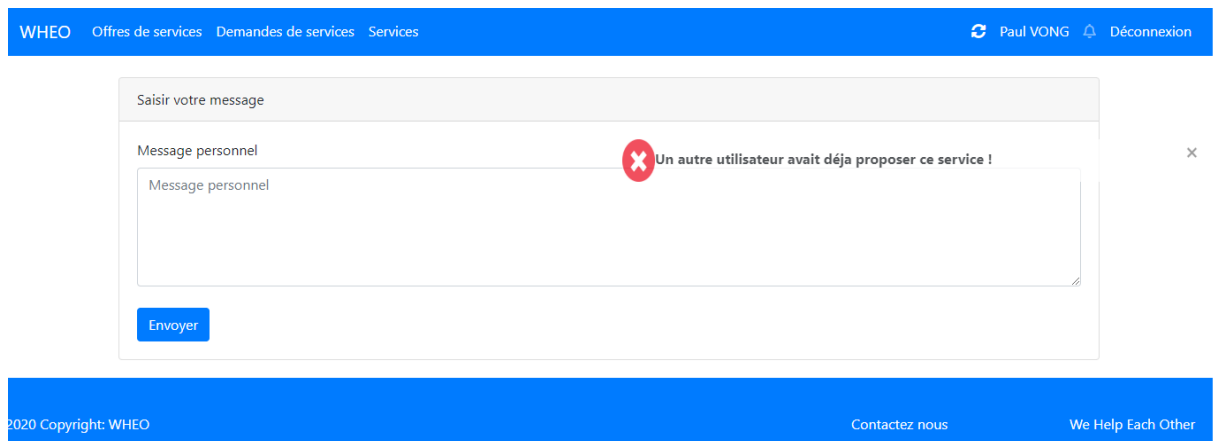
**Figure 29: Page de proposition d'une demande de service**

Lorsque l'utilisateur clique sur l'un des boutons « Proposer » ou « Accepter » il sera redirigé vers la page suivante :



**Figure 30: Page de saisi du message personnel**

Dans le cas où un autre utilisateur a déjà proposé ou accepté le même service une erreur sera affichée comme l'indique la figure ci-dessous.



**Figure 31: Page de message affichant une erreur**

### 4.4 Evaluation

Lors de l'ajout d'un service (offre ou demande), il est important de prendre en compte les champs à remplir : Les champs obligatoires non remplis ou de formats incorrects seront rouge en affichant des messages d'erreur. Le champ date ne doit afficher que des dates postérieures à la date du jour.

## CONCLUSION

Au cours de ce sprint nous avons réalisé les fonctionnalités essentielles du site (création des offres, création des demandes, visualisation des offres de service, proposition de service, acceptation d'une offre de service). Pour ce faire nous sommes passés par l'analyse, la conception et la réalisation. Nous allons aborder à présent la gestion des notifications et la gestion des services par l'administrateur.



## Chapitre 5 *Sprint 3 : Gestion des notifications et des services* par l'administrateur

---

## 5.1 Objectifs

La gestion des notifications est une des fonctionnalités indispensables pour un site de service. Le but de ce sprint est de gérer des services d'un utilisateur, visualiser les notifications et gérer la validation des offres et des demandes de service par un administrateur.

## 5.2 Analyse

Nous allons maintenant présenter les diagrammes des cas d'utilisation des processus 11 à 13 (ainsi que le diagramme de classe. Les processus 10 et 14 ne requièrent pas l'intervention des acteurs humains, donc il n'y a pas d'utilité de les schématiser dans des cas d'utilisations.

### 5.2.1 Diagramme de cas d'utilisation

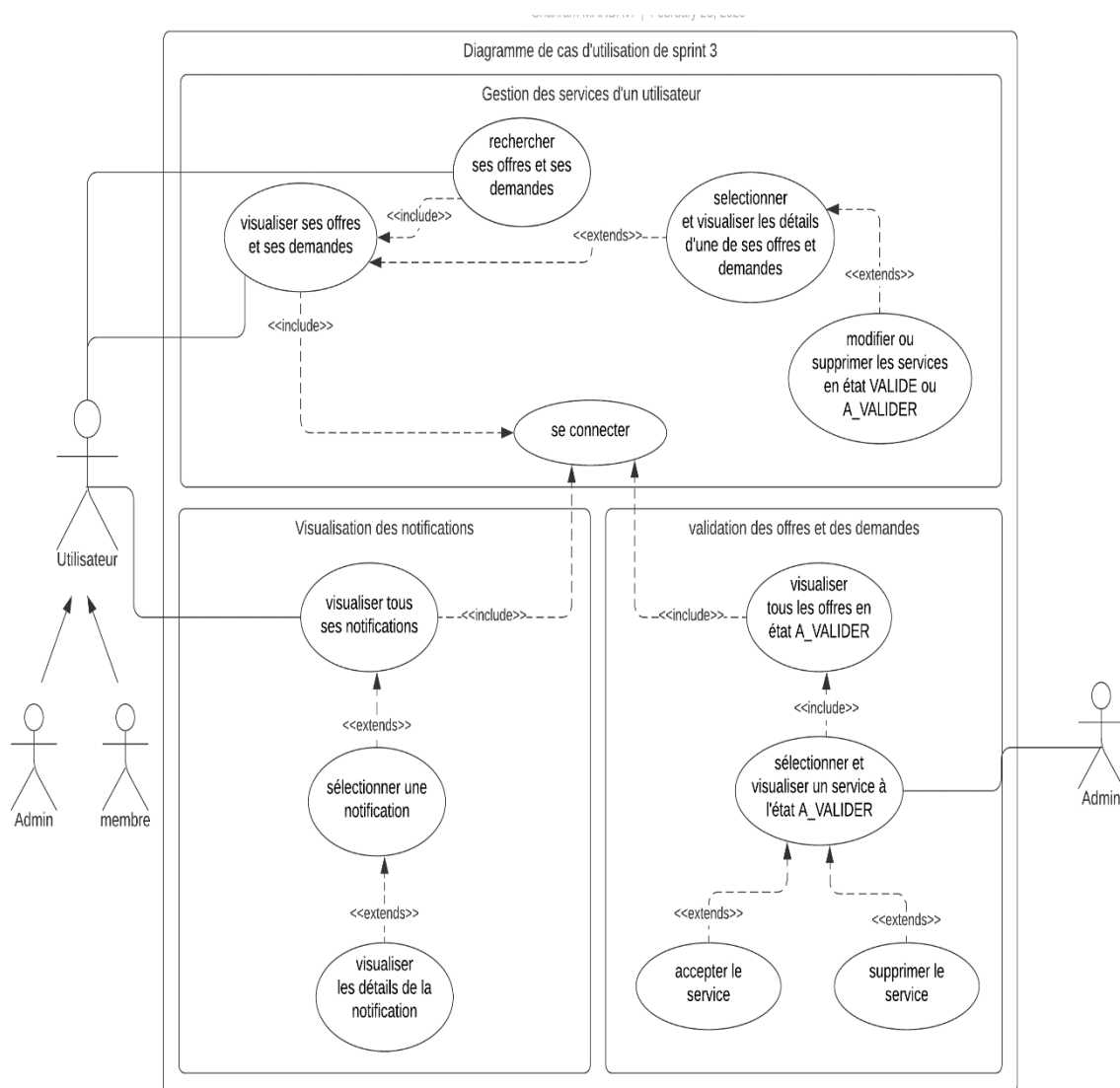


Figure 32 : Diagramme des cas d'utilisation relatif au package "Gestion des notifications et services par administrateurs »

### 5.2.1.1 Les Scénarios

#### 5.2.1.1.1 Scénario « Gestion des services d'un utilisateur »

<b>Cas d'utilisation</b>	<b>Gestion des services d'un utilisateur</b>
<b>Acteur</b>	Membre, Administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit posséder un compte et doit être connecté sur le site.
<b>Post condition</b>	Les services (en état VALIDE et A_VALIDER) ont été modifié/supprimé avec succès.
<b>Scénario nominal</b>	<ol style="list-style-type: none"> <li>1. L'acteur doit se connecter puis se rendre sur la barre de menu</li> <li>2. Il visualise les offres et les demandes de service qu'il a créé.</li> <li>3. Il peut rechercher des offres et des demandes en ajoutant des critères de recherche.</li> <li>4. Il peut également sélectionner un service parmi les services affichés et visualiser les détails de service.</li> <li>5. Une fois la demande sélectionnée, l'utilisateur peut supprimer ou modifier certaines informations de service en état A_VALIDER et VALIDE.</li> </ol>
<b>Exception</b>	<p>E1 : erreur de saisie en cas de modification du service : par exemple les informations saisies ne sont pas au bon format.</p> <p>⇒ Afficher un message d'erreur</p>

**Tableau 14: Scénario « Gestion des services d'un utilisateur »**

#### 5.2.1.1.2 Scénario « Visualisations des notifications »

<b>Cas d'utilisation</b>	<b>Visualisation des notifications</b>
<b>Acteur</b>	Membre, Administrateur
<b>Pré condition</b>	Le membre ou l'administrateur doit posséder un compte et doit être logué sur le site.
<b>Post condition</b>	Les notifications sont affichées.
<b>Scénario nominal</b>	<ol style="list-style-type: none"> <li>1. L'acteur doit se connecter puis se rendre sur la barre de menu</li> <li>2. Il clique sur l'icône notification.</li> <li>3. Il visualise la liste de toutes ses notifications récentes (qu'il n'a pas encore vues) et anciennes (qu'il a déjà vues), triées par date d'envoi <ol style="list-style-type: none"> <li>a. Il visualise les détails de la notification en la sélectionnant</li> <li>b. Il visualise toutes les notifications</li> </ol> </li> </ol>

**Tableau 15: Scénario « Visualisations des notifications »**

### 5.2.1.1.3 Scénario « Validation des offres et des demandes de service »

<b>Cas d'utilisation</b>	<b>Validation des offres et des demandes de service</b>
<b>Acteur</b>	Administrateur
<b>Pré condition</b>	L'administrateur doit posséder un compte et doit être logué sur le site.
<b>Post condition</b>	Le service est validé ou supprimé
<b>Scénario nominal</b>	<ol style="list-style-type: none"> <li>1. Il doit se connecter puis se rendre sur la barre de menu</li> <li>2. Il visualise toutes les offres et les demandes de service dans l'état A_VALIDER.</li> <li>3. Il sélectionne un service pour visualiser son contenu. <ol style="list-style-type: none"> <li>a. L'administrateur peut supprimer un service</li> <li>b. L'administrateur peut valider ou accepter</li> </ol> </li> <li>4. Une notification sera envoyée au créateur du service.</li> </ol>

Tableau 16: Scénario "Validation des offres et des demandes de service"

### 5.2.2 Diagramme de classe

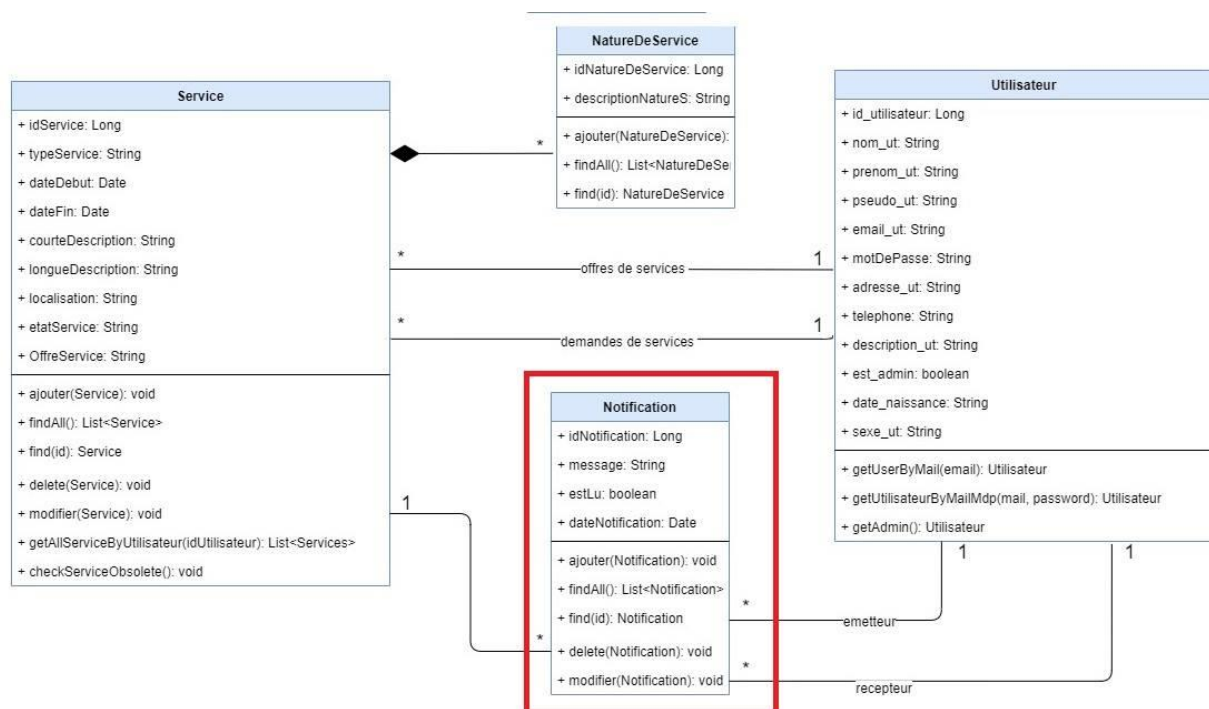


Figure 33 : Diagramme de classe visant la classe "Notification"

Nous allons donc décrire notre classe focalisée dans le diagramme de classe avec deux tableaux. Le premier tableau (tableau 18) présente l'ensemble des attributs utilisés dans notre classe. Nous allons définir le type et la description de chaque attribut.

De même pour deuxième tableau (tableau 19) pour la définition des différentes méthodes.

Attributs		
Nom	Type	Description
<b>idNotification</b>	Long	L'identifiant de la notification
<b>message</b>	String	Le message de la notification envoyée
<b>estLu</b>	Boolean	Permet de distinguer entre les notifications visualisées et non visualisées
<b>dateNotification</b>	Date	Indique la date d'envoi de la notification

**Tableau 18 : Attributs de la classe "Notification"**

Méthodes		
Nom	Type	Description
<b>ajouter(Notification)</b>	void	Ajouter une notification
<b>delete(Notification)</b>	void	Supprimer la notification sélectionné
<b>findAll()</b>	List< Notification >	Afficher la liste de tous les notifications
<b>find(id)</b>	Notification	Chercher une notification avec sa référence

**Tableau 19 : Méthodes de la classe « Notification »**

## 5.3 Conception

Cette partie a pour objectif de définir la solution conceptuelle de ce sprint. Nous allons présenter en premier lieu une vue dynamique grâce au diagramme de séquences pour les scénarios d'exécution les plus importants et les descriptions D'IHM et descriptions du mécanisme de gestion des erreurs.

### 5.3.1 Diagrammes de séquences

#### 5.3.1.1 Diagramme de séquence relatif à la gestion des services d'un utilisateur

sd Gestion Services

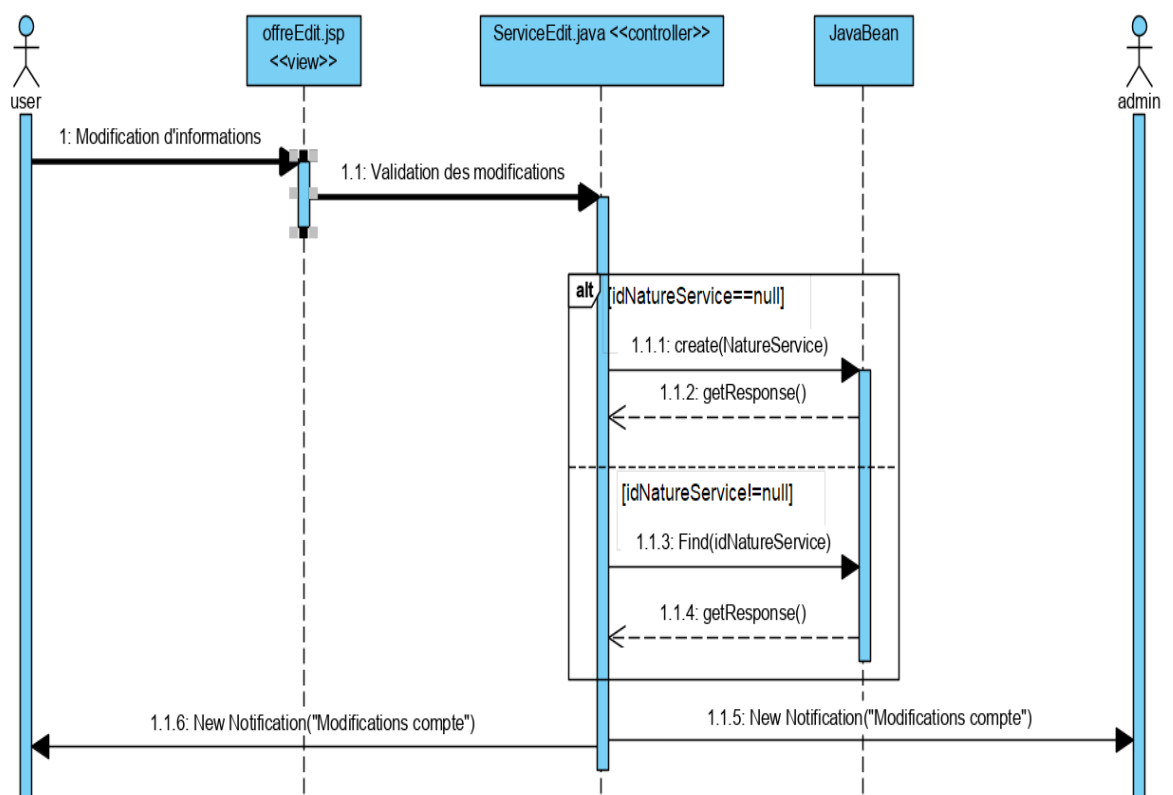


Figure 34: Diagramme de séquence relatif à la gestion des services d'un utilisateur

L'utilisateur consulte la page de ses services. Il sélectionne un service puis il aura la possibilité de supprimer ou modifier un service. Dans le cas d'une modification il sera redirigé vers « ServiceEdit » puis il valide les informations saisies. Pour la suppression, le contrôleur « ServiceDelete » sera appelé.

### 5.3.1.2 Diagramme de séquence relatif à la validation des services

sd Acceptaion Offre Service

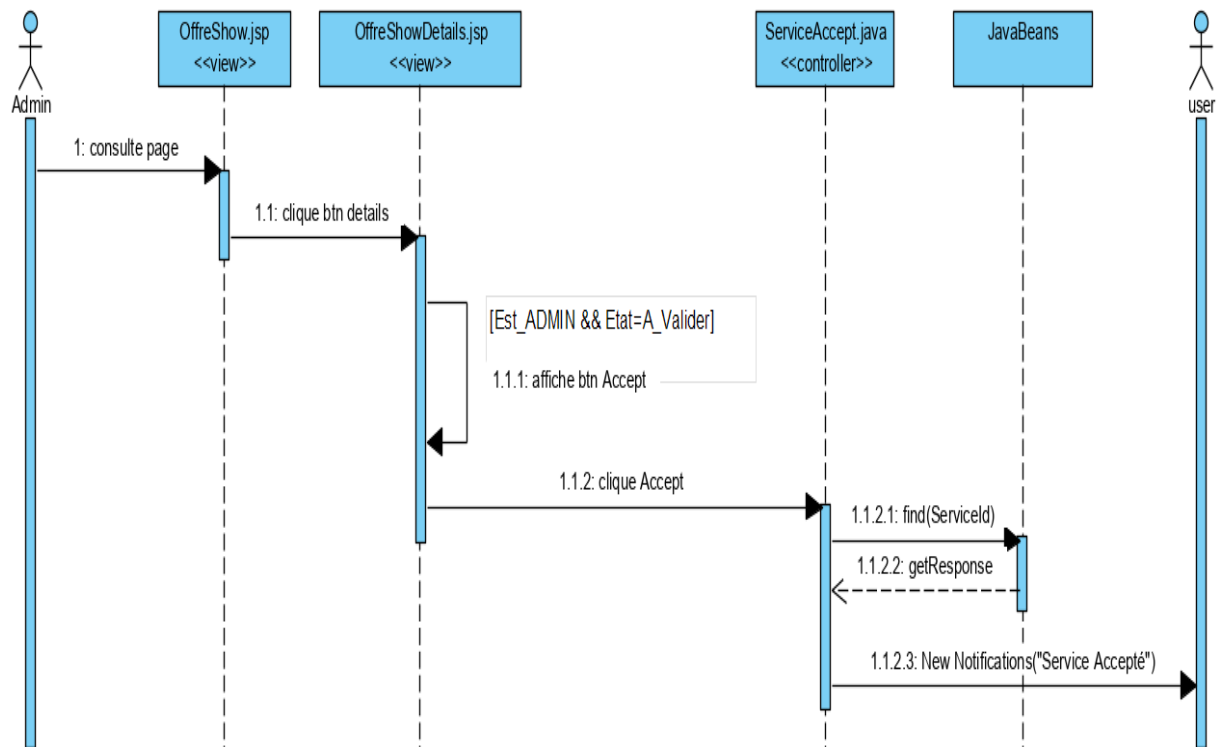


Figure 35 : Diagramme de séquence relatif à la validation des services

L'administrateur consulte la page *OffreShow.jsp* et clique sur le bouton *details* d'une offre (ou demande). Si l'offre (ou la demande) est dans l'état « A\_Valider » alors le bouton *accepter* s'affichera. L'administrateur clique sur le bouton *accepter*. Le controller *ServiceAccept.java* récupère l'objet *Service* correspondant et envoie une notification à l'utilisateur qui a offert (ou demandé) le service.

### 5.3.2 Diagramme d'activité relatif à la visualisation des notifications et gestion des services

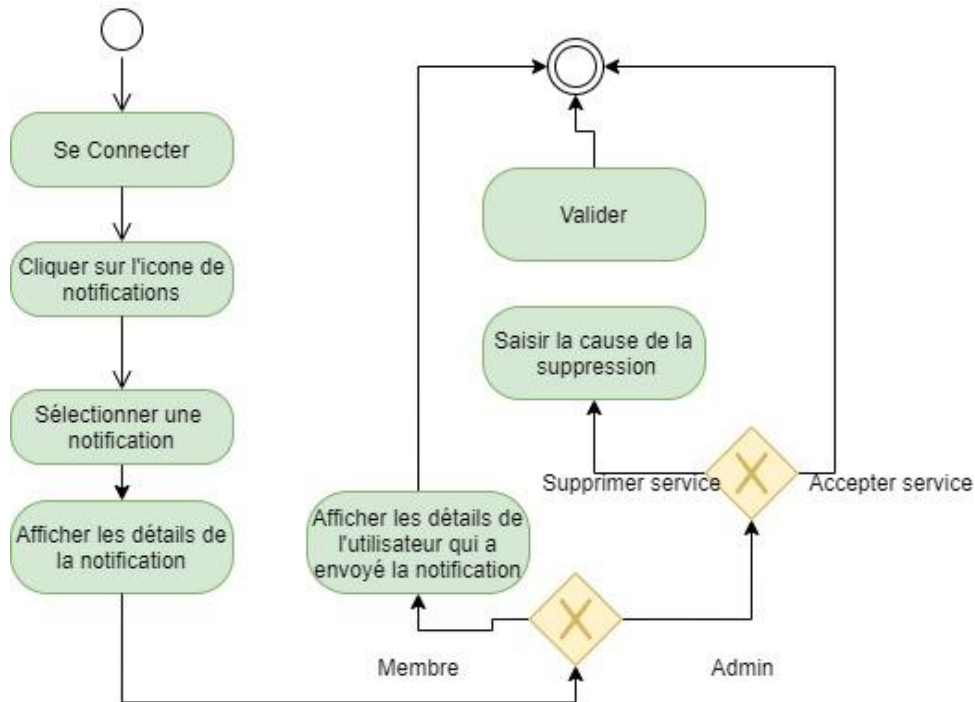


Figure 36 : Diagramme d'activité relatif à la visualisation des notifications et le gestion des services

### 5.3.3 Description de l'IHM

Lorsqu'un utilisateur crée une offre ou une demande de service, une notification sera envoyée à l'administrateur.

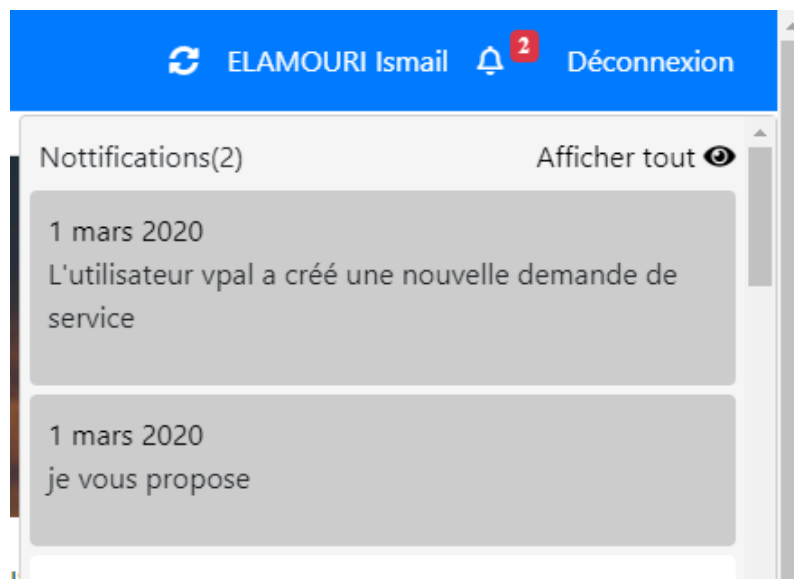


Figure 37: IHM de notification envoyé à l'administration



Dans cet exemple nous remarquons que l'administrateur a deux notifications non lues. En cliquant sur la première notification cette page sera affichée.

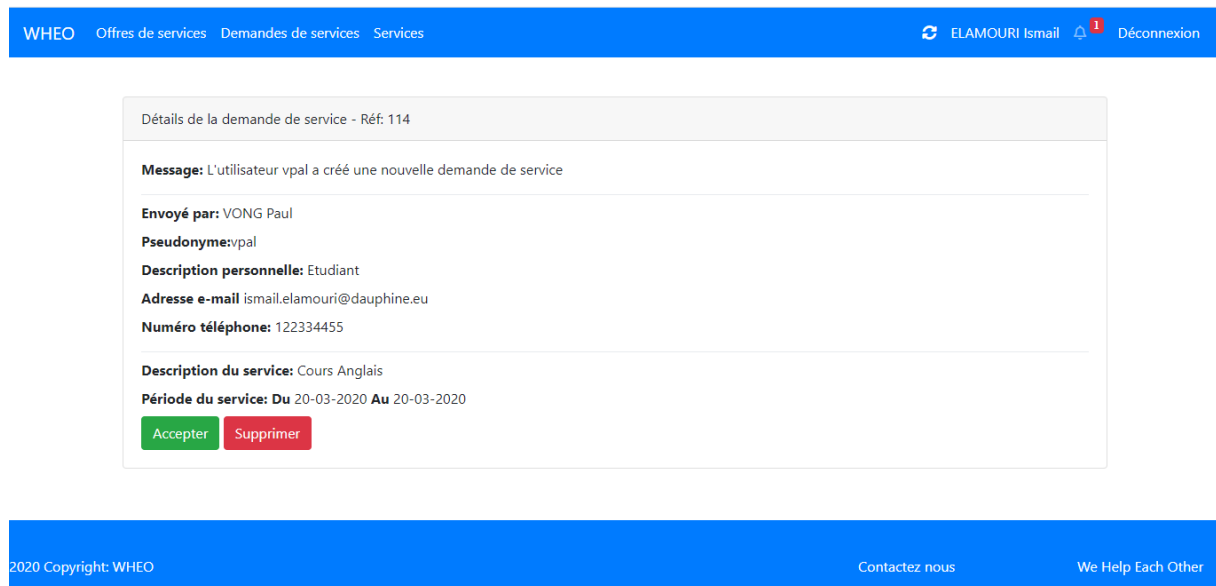


Figure 38: IHM de détails de la notification

Après avoir visualisé la notification, le nombre a été décrémenté et l'administrateur a été redirigé vers une page affichant les détails du service, de la notification ainsi que le créateur.

Dans ce cas l'administrateur a le choix d'accepter ou de supprimer le service et une notification sera envoyé au créateur. Si l'administrateur accepte le service, ce dernier passe à l'état valide sinon une notification sera envoyée au créateur contenant le motif de la suppression.

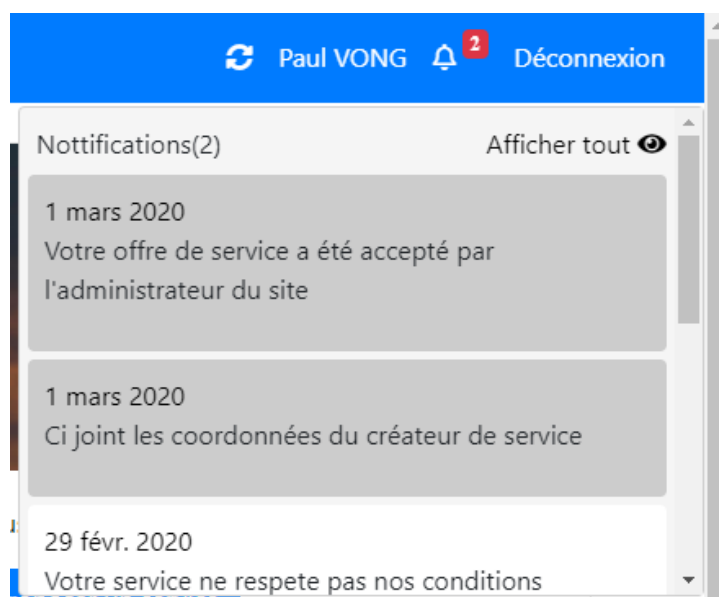


Figure 39: IHM de notification d'acceptation d'un service par l'administrateur

Dans la figure ci-dessus nous remarquons la différence entre les notifications visualisées et non visualisée. La première notification affiche que le service crée a été accepté par l'administrateur. La deuxième notification affiche les coordonnées de l'utilisateur à qui nous avons proposé un service. La troisième notification montre que le service a été supprimé par l'administrateur. En cliquant sur la première notification l'interface suivante sera affichée

Détails de l'offre de service - Réf: 115

**Message:** Votre offre de service a été accepté par l'administrateur du site

**Envoyé par:** Ismail ELAMOURI  
**Pseudonyme:** elamis  
**Description personnelle:** Etudiant en M2 miage à l'université Paris-Dauphine  
**Adresse e-mail:** elamouriismail@gmail.com  
**Numéro téléphone:** 699667601

**Description du service:** Cours Anglais  
**Période du service:** Du 11-03-2020 Au 11-03-2020

2020 Copyright: WHEO Contactez nous We Help Each Other

**Figure 40: IHM de détails de la notification envoyée par l'administrateur**

De même pour la deuxième et troisième notification.

### 5.3.3.1 Page de visualisation des services par utilisateur

Mes demandes et offres de services

ID	Description du service	Etat	Offre / demande	Action
109	Pele	OBSOLETE	Demande de service	
110	Garde	OBSOLETE	Offre de service	
114	Cours Anglais	A_VALIDER	Demande de service	
115	Cours Anglais	VALIDE	Offre de service	

Showing 1 to 4 of 4 rows

2020 Copyright: WHEO Contactez nous We Help Each Other

**Figure 41: Page de visualisation des services par utilisateur**

A partir de cette interface l'utilisateur peut consulter ses propres services sous forme d'un tableau où il peut distinguer les offres des demandes et aussi effectuer des recherches sur un service quelconque. Il peut aussi afficher le détail de chaque service en cliquant sur l'icône « Œil ».

Si le service est à l'état « VALIDE » ou « A\_VALIDER », l'utilisateur peut modifier ou supprimer le service sélectionné à partir de cette interface ci-dessous.

The screenshot shows a web application interface for managing services. At the top, there is a blue navigation bar with the text 'WHEO Offres de services Demandes de services Services' and a user profile 'Paul VONG' with a 'Déconnexion' link. Below the navigation bar, there is a light gray box titled 'Détails de la demande de service - Réf: 114'. Inside this box, the following information is displayed: 'Proposé par: vpal', 'Type de service: Service à offrir', 'Nature de service: Cours', 'Période: Du 20-03-2020 Au 20-03-2020', 'Description: Cours Anglais', 'Description détaillée: Donner cours de 8h à 9h', and 'Localisation: France, paris, 75011'. At the bottom of the box, there are two buttons: 'Modifier' (green) and 'Supprimer' (red). Below the box, there is a blue footer bar with the text '2020 Copyright: WHEO', 'Contactez nous', and 'We Help Each Other'.

Figure 42: IHM pour modifier ou supprimer un service

## 5.4 Evaluation

Pendant la phase de développement, nous avons été confrontés à certaines difficultés comme celle de l'affichage des notifications envoyés à l'administrateur. Les notifications restent présentes, même après la suppression du service par l'utilisateur.

## CONCLUSION

Dans ce chapitre nous avons réalisé les processus notifications des utilisateurs, gestion des services d'un utilisateur, visualisation des notifications, validations des services et vérification de l'obsolescence du service. Nous avons réussi à développer la totalité de l'application après une conception des tâches présentées dans le cahier de charge afin de livrer un produit complet et fonctionnel.

### Conclusion générale

Dans le cadre de notre projet, nous avons conçu et développé un site web d'une association. Le présent document détaille toutes les étapes de la mise en place. Nous avons l'opportunité d'appliquer la méthodologie Scrum pour construire notre application. Le travail que nous avons achevé repose principalement sur la plateforme J2EE et profite de la puissance des Frameworks Ejb, JPALink et Bootstrap.

Nous avons commencé par présenter le contexte général de notre projet. Pour cela, nous avons spécifié les besoins fonctionnels et non fonctionnels auxquels notre application doit répondre. Par la suite nous avons présenté, l'environnement de notre travail, les outils et les technologies utilisés tout au long de notre stage. Nous nous sommes aussi intéressées à l'étude conceptuelle de notre projet dans laquelle nous avons dégagé dans une première partie l'architecture trois tiers de notre application. Nous avons détaillé dans une deuxième partie notre conception à travers les diagrammes de cas d'utilisation, de séquences, d'activité et de classes. Finalement, nous avons exposé notre application par quelques scénarios d'exécution et une simulation d'exécution d'un processus à travers des captures d'écran.

Ce projet nous a permis d'acquérir une expérience enrichissante. Nous avons eu la chance de maîtriser de plus des nouvelles technologies plus qu'intéressantes.

## Annexes

### [A1] : Les méthodes agiles

« Une méthode agile est une approche itérative et incrémentale, qui est menée dans un esprit collaboratif avec juste ce qu'il faut de formalisme. Elle génère un produit de haute qualité tout en prenant en compte l'évolution des besoins des clients [B1]. »

La famille «agile» offre plusieurs méthodologies dont nous citons les plus connues [B2][N1] :

a) XP (eXtreme Programming) : est une méthode agile qui priorise l'excellence technique.

Elle comporte un ensemble de pratiques interdépendantes qui demandent beaucoup de rigueur, afin de puiser le maximum d'efficacité des équipes de développement.

La méthode XP est notamment basée sur les concepts suivants:

- Les équipes de développement travaillent sur des cycles courts d'une à deux semaines maximum avec le client.
- Les livraisons de versions du logiciel interviennent très tôt et à une fréquence élevée pour maximiser l'impact des retours utilisateurs.
- L'équipe de développement travaille en collaboration totale sur la base de binômes.
- Le code est testé et nettoyé tout au long du processus de développement.
- Des indicateurs permettent de mesurer l'avancement du projet.

b) Crystal Clear : est une plate-forme méthodologique. La famille de méthode Crystal comporte quatre membres, qui ne sont pas encore tous documentés. Plusieurs principes doivent être partagés par l'équipe entière :

- Le nombre de membres de l'équipe est limité à six personnes.
- Tous les membres de l'équipe doivent travailler dans une même pièce.
- Les schémas de modélisation doivent être réalisés en groupe.
- Les livraisons des parties exécutables le plus fréquemment possible.
- La collaboration avec le client.

c) Scrum : est un processus de contrôle empirique de développement logiciel. Il permet aux équipes de produire des logiciels de manière itérative incrémentale. Dans le cadre de notre projet, nous avons adopté la méthode SCRUM comme une méthodologie de travail.

### [A2] : MVC

C'est une méthode d'organisation et de structuration des applications logicielles. Ce design est basé sur la décomposition de l'application en trois éléments indispensables : modèle, vue et contrôleur.

MVC permet de décomposer le problème en modules dont chacun est responsable d'un traitement spécifique à savoir :

- Couche présentation : C'est la couche chargée de la présentation des informations. Il s'agit de l'interface graphique de la plateforme que l'utilisateur peut accéder à travers un navigateur Web.
- Couche métier: Elle contient la logique de l'application. Les communications entre cette couche et l'interface se font via des requêtes. Les différents services de l'application seront implémentés au niveau de cette couche.
- Couche persistance: c'est la couche d'accès aux données. En fait, elle fournit les informations exploitées par la logique applicative. Autrement dit, c'est la base de données contenant les tables nécessaires au déroulement de l'application.

Le MVC très pratique, peut se révéler lourd à mettre en place. Ceci à cause de la multitude de contrôleurs à implémenter. Afin de simplifier la réalisation d'un tel modèle, une nouvelle version a été introduite : **le MVC**. C'est exactement le même modèle de conception à la différence qu'il n'y a plus qu'un seul contrôleur qui se charge de rediriger la requête vers le bon traitement.

### [A3] : EclipseLink

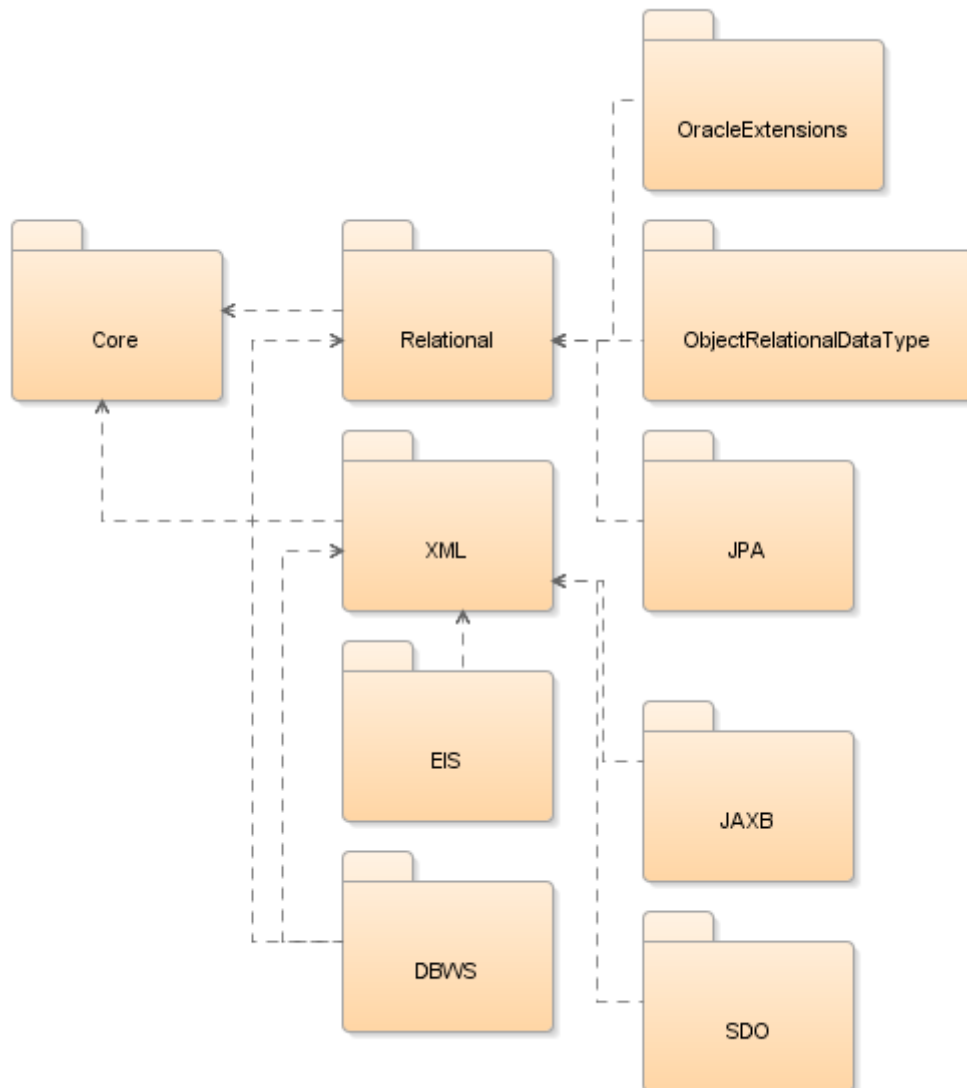
- **Présentation générale**

**EclipseLink** est un framework open source de mapping objet-relationnel pour les développeurs Java. Il fournit une plateforme puissante et flexible permettant de stocker des objets Java dans une base de données relationnelle et/ou de les convertir en documents XML.



- **Architecture d'EclipseLink**

**EclipseLink** permet d'assurer la persistance des objets de l'application dans un entrepôt de données. Ceci est dans la majorité des cas une base de données relationnelle, mais il peut être aussi un fichier XML. Le mapping des objets est effectuée par Hibernate en se basant sur des fichiers de configuration en format texte ou souvent XML. La figure suivante décrit l'architecture du Framework Hibernate.



**Figure 43: Architecture d' EclipseLink**

Le tableau ci-dessous décrit les différents éléments de l'architecture d'Hibernante :

Élément	Description
<b>Core</b>	Core descripteur et mapping des fonctionnalités and et l'accès au donnés
<b>Relational</b>	Relationnel descripteur et mapping accès à la base de données
<b>OR data-type</b>	descripteur et mappage des données relationnel objet.
<b>Oracle</b>	Extensions de base de données Oracle.
<b>EIS</b>	systèmes d'information d'entreprise, accès et adaptateurs de données JCA.
<b>XML</b>	XML descriptor et mappings, accès aux données XML.
<b>JAXB</b>	Java API pour XML Binding facade and schema processing.
<b>JPA</b>	Java Java Persistence API facade and meta données processing.
<b>SDO</b>	SDO - Service Data Objects implementation.
<b>DBWS</b>	DBWS - Traitement des web services de bases de données

Tableau 20: Élément de l'architecture d'EclipseLink



## Bibliographie

[B1] Veronique Messenger Rota "Gestion de projet : Vers les méthodes agiles" (21 février 2013).

[B2] Jean-Louis Bénard, directeur technique de Business Interactif : Méthode Agiles 2001-V1.1.

[B3] Martin Mayer « Implémentation de la méthodologie SCRUM dans les grandes entreprises », département de génie logiciel et des IT à l'école de technologies supérieure Québec. (Avril 2010).

## Netographie

[N1] : <http://www.dsi.cnrs.fr/methodes/gestion-projet/methodologie/bi-methodes-agiles.pdf>

[N2]: <http://blog.dcube.fr/blog/2014/04/28/scrum-vs-cycle-en-v-2/>

[N3] : <http://getbootstrap.com/getting-started/>, "Présentation de Bootstrap"

[N4] : <https://jquery.com/>, "Présentation de JQuery"