

MEMOIRE DE STAGE DE FIN D'ETUDES

Pour l'obtention du

«Mastère professionnel en Nouvelles Technologies des
Télécommunications et Réseaux (N2TR)»

Présenté par :

Anis KADRI

Titre

Développement d'une application web de gestion de Ressources Humaines

Soutenu le : 29/10/2016

Devant le jury :

Président : Mr. Jalel Khediri

Encadreur : Mr. Ezzedine Ben Braiek

Rapporteur : Mr. Hassene Seddik

Année Universitaire : 2015 / 2016

Dédicaces

*A mes chers parents,
A mes adorables soeurs,
A toute ma famille,
A mes chères amies,
Je dédie ce modeste travail,
Qu'ils y trouvent l'expression de ma gratitude et de ma
profonde affection.
Anis KADRI*

Remerciements

C'est avec un grand plaisir que nous réservons ces lignes en signe de gratitude et de reconnaissance à tous ceux qui ont contribué de près ou de loin à l'élaboration de ce travail.

Je tiens en premier lieu à adresser ma reconnaissance et mes remerciements à mon encadreur de mémoire, Monsieur Ezzedine BEN BRAIEK. , pour son encadrement et sa disponibilité, ainsi que pour la richesse et la qualité de son enseignement.

Je remercie également mon encadreur de la société NEXT STEP IT, Monsieur Yassine AYA-CHI pour sa collaboration et ses conseils précieux qui m'ont été d'une grande utilité lors de la réalisation des analyses statistiques.

Nous nous permettons de remercier également ceux qui nous font l'honneur de participer et de siéger dans ce jury et être les rapporteurs de ce modeste travail.

Table des matières

Introduction générale	1
1 Etude préalable	2
1.1 cadre général	2
1.2 Présentation de l'organisme d'accueil	2
1.2.1 Chiffres clés	2
1.2.2 Les services offerts	3
1.2.3 Support et maintenance	4
1.3 Définition de la gestion des ressources humaines	4
1.3.1 Évaluer les besoins en ressources humaines	5
1.3.2 Rédiger les descriptions de tâches	5
1.3.3 Gestion des candidatures	5
1.4 Etude de l'existant pour la gestion des ressources humaines au niveau des candidatures	6
1.5 Problématique	6
1.6 La solution proposée	6
2 Analyse et spécification des besoins	8
2.1 Spécification non formelles des besoins	8
2.1.1 Spécification des besoins fonctionnels	8
2.1.2 Spécification des besoins non fonctionnels	9
2.2 Spécification semi-formelle des besoins	10
2.2.1 Identification des acteurs	10
2.2.2 Diagramme des cas d'utilisation	10
2.2.3 Diagrammes de séquences	13
3 Conception	16
3.1 Architecture globale	16

3.1.1	Présentation de l'architecture client/serveur	16
3.1.2	Les avantages de l'architecture client/seveur	17
3.2	Conception détaillée	17
3.2.1	Vue statique	17
3.2.2	Vue dynamique : diagramme de séquences	19
4	Réalisation	23
4.1	Environnement de travail	23
4.1.1	Environnement matériel	23
4.1.2	Environnement logiciel	23
4.2	Exposition du travail réalisé	25
4.2.1	Interface d'accueil	25
4.2.2	Interface d'authentification	25
4.2.3	Inscription d'un candidat	26
4.2.4	Interface de l'espace administrateur	26
4.2.5	Interface de l'espace recruteur	28
4.2.6	Interface de l'espace candidat	28
	Conclusion générale	31
	Bibliographie	32
	Bibliographie et Netographie	32
	Nétographie	33
	A Annexe	34

Table des figures

1.1	Logo de la société NEXT STEP IT	2
2.1	Diagramme des cas d'utilisation général de l'application	11
2.2	Diagramme du cas d'utilisation : gérer les comptes	11
2.3	Diagramme du cas d'utilisation : s'authentifier	12
2.4	Diagramme du cas d'utilisation : postuler une candidature	12
2.5	Diagramme du cas d'utilisation : gérer les candidatures	13
2.6	Diagramme de séquences relatif à l'administrateur	14
2.7	Diagramme de séquences relatif au recruteur	14
2.8	Diagramme de séquences relatif au candidat	15
3.1	Architecture client/serveur	17
3.2	Diagramme de classes	18
3.3	Diagramme entité-association	19
3.4	Diagramme de séquences de l'authentification	19
3.5	Diagramme de séquences de la gestion des offres	20
3.6	Diagramme de séquences de la postulation	21
3.7	Diagramme de séquences de la recherche d'un candidat	22
4.1	Interface d'accueil	25
4.2	Interface d'authentification	26
4.3	Inscription d'un candidat	26
4.4	Interface de l'espace administrateur	27
4.5	Interface de l'espace administrateur : ajout d'une offre	27
4.6	Interface de l'espace administrateur : modification d'une offre	27
4.7	Interface de l'espace recruteur	28
4.8	Interface de l'espace recruteur : consultation des demandes	28
4.9	Interface de l'espace candidat	29

4.10 Interface de l'espace candidat : consultation des offres.	29
4.11 Interface de l'espace candidat : postulation à une offre.	30
4.12 Espace candidat : Consultation des demandes	30

Introduction générale

Depuis l'apparition de l'internet, bien de fonctionnalités hormis la navigation ont été offertes. Parmi elles, se trouve la possibilité d'interagir avec l'utilisateur en satisfaisant au mieux sa requête grâce à un site web dynamique. De ce fait, toute entité professionnelle telle que les grandes entreprises, cherche à utiliser divers applications web dans le but d'une contribution à une meilleure rentabilité et afin de communiquer avec ses clients non seulement en personne mais aussi à distance.

C'est dans ce cadre que se situe notre travail qui constitue à concevoir et implémenter une application de gestion des ressources humaines de l'entreprise « NEXT STEP IT », qui souffre pendant des périodes de l'année des problèmes liés à la lenteur et au retard au niveau de ses services administratifs. Parmi les activités qui permettent à une organisation ou une entreprise de disposer de ressources nous pouvons identifier : le recrutement, la rémunération, l'évaluation ou l'appréciation, la gestion prévisionnelle des emplois, des compétences et des carrières. . . Dans ce travail, nous nous intéressons particulièrement à la gestion des candidatures. En effet, les différentes composantes de l'entreprise « NEXT STEP IT » gèrent chaque année un très grand nombre de candidatures pour recrutement dans ses différentes filières et ont besoin d'un logiciel de gestion de candidatures pour en automatiser le traitement.

Le présent rapport a pour objectif de donner une idée claire sur le travail effectué pendant la réalisation du projet. Il s'articule autour de quatre chapitres : Le premier chapitre présente l'organisme d'accueil et le cadre générale du sujet à traiter. Le second chapitre permettra de spécifier les différents besoins de l'application, qu'ils soient fonctionnels ou non fonctionnels. Le troisième chapitre présentera les différents aspects conceptuels du projet. Le dernier chapitre traite tous les détails de la réalisation : les outils de travail, les différentes étapes de la réalisation du système ainsi qu'un aperçu sur les résultats obtenus.

Nous finirons par une " Conclusion générale " qui résumera notre travail et qui en exposera des perspectives d'avenir.

Chapitre 1

Etude préalable

Introduction

Dans ce présent chapitre, nous exposons le contexte général de l'élaboration de notre projet. Ainsi, nous présentons dans un premier lieu l'organisme d'accueil et la notion de Gestion des Ressources Humaines. Nous faisons, ensuite, une étude de l'existant afin de dégager par la suite les problèmes que notre application doit pallier et les objectifs auxquels elle doit répondre.

1.1 cadre général

Le présent projet « développement d'une application web de gestion de Ressources Humaines » est réalisé dans le cadre du mémoire de mastère professionnel pour l'année universitaire 2015-2016. Il a été réalisé au sein de la société NEXT STEP IT.

1.2 Présentation de l'organisme d'accueil

1.2.1 Chiffres clés

NEXT STEP IT est une société à responsabilité limitée, créée au cours de l'année 2012, et employant une cinquantaine de personnes.



FIGURE 1.1 – Logo de la société NEXT STEP IT

1.2.2 Les services offerts

Ils se résument principalement en l'audit, l'étude et conseil, l'intégration, l'assistance techniques, le support et la maintenance des solutions de sécurité et de réseau informatique.

1.2.2.1 Audit

Next Step IT propose le service d'audit des infrastructures des systèmes d'informations. Les prestations d'audit ont pour objectif d'établir un état des lieux des infrastructures informatiques existantes afin d'évaluer les besoins et mettre en évidence tous les aspects nécessaires pour optimiser l'environnement et l'organisation. Ainsi, Next Step IT accompagne ses clients pour faire un choix stratégique sur l'évolution de son système d'information en toute connaissance de cause.

1.2.2.2 Étude et conseil

La réussite d'une solution dépend du soin apporté à sa conception. Next Step IT étudie des solutions sur mesure, s'adaptant aux contextes et aux besoins des entreprises. Elle accompagne ses clients dans cette démarche :

- Recueil des besoins.
- Analyse de l'existant et visites des sites.
- Définitions des spécifications techniques, fonctionnalités, performances, interopérabilité, ...etc.
- Identification des contraintes, du budget et des délais.
- Démonstration, mise en place de maquette et accompagnement au test
- Assistance à la rédaction des cahiers des charges.

1.2.2.3 Intégration

Next Step IT est un spécialiste d'intégration de solutions d'infrastructure réseau, système, sécurité et communications. Elle dédie une équipe composée d'un chef de projet, d'un ingénieur commercial et d'ingénieurs et techniciens pour la durée du projet. Toute l'équipe projet de Next Step est certifiée par les constructeurs partenaires. Elle est constituée de techniciens et d'ingénieurs certifiés et formés en permanence sur ses solutions et sur les différentes technologies qui touchent à son métier.

1.2.2.4 Assistance technique

Next Step IT associe la compétence technique de son équipe et les outils informatiques nécessaires afin d'apporter l'assistance technique adéquate et à haute valeur ajoutée à ses clients. Le professionnalisme et l'expertise de Next Step garantissent à ses clients :

- Réactivité : les équipes interviennent directement là où on le souhaite, en fonction des contraintes des clients :
- Efficacité : les techniciens trouvent la solution la plus appropriée dans les meilleurs délais.
- Transparence : Ses interventions font l'objet de rapports et de transfert de compétence.
- Savoir-faire : Le problème du client est pris en charge par une équipe qualifiée, expérimentée et certifiée.

1.2.3 Support et maintenance

L'efficacité du centre du support qui accompagne le client dans la maintenance de ses solutions réseaux garantit la résolution rapide des incidents et la maîtrise des évolutions de la solution. Cette efficacité résulte de :

- La rapidité de prise en compte de la demande,
- La compétence des intervenants qui traitent la demande,
- Les moyens matériels destinés à remplacer les équipements défectueux,
- Un fort partenariat avec les constructeurs.

1.3 Définition de la gestion des ressources humaines

La gestion des ressources humaines se définit comme « un ensemble de pratiques ayant pour objectif de mobiliser et de développer les ressources humaines pour une plus grande efficacité et efficience, en soutien de la stratégie d'une organisation (association, entreprise, administration publique, etc.)[1]. La gestion des ressources humaines comprend des fonctions administratives et opérationnelles. Gérer les ressources humaines d'une entreprise ou d'un organisme veut dire voir aux besoins en matière de personnel autant du côté administratif que du côté opérationnel. La gestion des ressources humaines comprend donc les tâches de :

- recrutement et embauche du personnel ;
- orientation et formation du personnel ;
- respect des lois et des normes gouvernementales ;
- encadrement dans les fonctions ;
- évaluation de la performance du personnel.

1.3.1 Évaluer les besoins en ressources humaines

La planification des ressources humaines peut être un exercice simple qui épargne des frustrations et qui rapporte de la productivité dans des périodes achalandées. Pour évaluer les besoins en ressources humaines, il vaut mieux commencer par réviser la dernière année. L'utilisation d'un tableau peut faciliter cette tâche. Il s'agit d'énumérer le nombre d'employés au cours des diverses périodes de l'année. En ce qui a trait aux tâches, il est important de noter les lacunes en main d'œuvre et aussi les besoins en compétences nécessaires pour réaliser les tâches. Pour poursuivre dans la planification, il faut déterminer les besoins anticipés selon les attentes pour la prochaine année. Une expansion amène un besoin de ressources humaines supplémentaires qui pourrait comporter des compétences spécifiques.

Durant la phase de planification, il est nécessaire de se questionner et d'auto-évaluer la vision à court et à long terme comme producteur ou propriétaire d'entreprise. Quelles sont les forces et les faiblesses par rapport aux plans pour l'entreprise ? La planification des ressources humaines est une étape qui devrait se répéter annuellement.

1.3.2 Rédiger les descriptions de tâches

La rédaction des descriptions de postes permet de communiquer clairement aux employés les attentes et les exigences face au travail qu'ils devront accomplir. Il est important de revoir la législation du travail et les normes salariales lors de la rédaction des descriptions de tâches afin de s'assurer de les respecter.

1.3.3 Gestion des candidatures

Lorsque la description de tâche est complète, une offre d'emploi doit être préparée avec le titre de l'emploi et les grandes lignes du poste à combler. Le quotidien des services de recrutement est souvent rythmé par le flux inexorable de candidatures et de CV, en réponse à annonce ou en candidature spontanée. D'où la nécessité de concevoir des applications pour fluidifier le marché de l'emploi et offrir plus de visibilité au CV d'un candidat, en simplifiant la démarche de dépôt de candidature.

Notre application est concentrée sur cette rubrique de gestion des ressources humaines.

1.4 Etude de l'existant pour la gestion des ressources humaines au niveau des candidatures

La gestion des ressources humaines comprend des fonctions administratives et opérationnelles. Dans ce sujet on va étudier la tâche de recrutement de candidatures. La gestion des candidatures au sein de l'entreprise « NEXT STEP IT » se déroule actuellement de la manière suivante :

- Les dossiers de candidatures sont traités de manière papier et numérique (au format PDF) ou uniquement sous format papier ;
- Les données issues de ces dossiers sont entrées dans un tableau Microsoft Excel et un premier avis est donné par le responsable qui examine l'intégralité des dossiers ;
- Les meilleurs candidats sont sélectionnés parmi le flux des CV reçus ;
- Des entretiens sont planifiés avec les candidats qui ont été choisis et qui vont être informés par un email.

1.5 Problématique

Le processus de recrutement, mentionné ci-dessus, leur permet de traiter la tâche de recrutement d'une manière fiable mais non efficace. Des solutions indépendantes sont utilisées afin de satisfaire les besoins du personnel, mais ces outils ne sont pas toujours efficaces du fait qu'ils sont divisés sur plus qu'un service indépendant et contenant une grande partie traitée manuellement non centralisée et non partagée à l'accès du système informatique.

En général, les méthodes classiques de recrutement consistent à consulter les agences de recrutement locales ou les médias traditionnels. Le recrutement à travers ces méthodes est loin de satisfaire objectivement les besoins des employeurs et des employés. Les employés qualifiés n'étant pas toujours informés sur les offres disponibles sur le marché du travail. Il a été nécessaire de trouver une méthode rapide et efficace : d'où l'extension du recrutement à travers le net.

1.6 La solution proposée

C'est pourquoi ce projet d'une application Web couplée à une base de données nous a été proposé, afin de gérer aussi bien la postulation aux offres des emplois que la sélection des meilleurs candidats. Cette application doit respecter les conditions suivantes :

- Centraliser l'information et les données concernant le recrutement,
- Diminuer le temps de saisie des candidatures,

- Diminuer le temps de suivi d'une candidature,
- Partager plus facilement l'information des candidatures avec les responsables de services.

Conclusion

Dans ce qui précède, nous avons situé le projet dans son cadre général et introduit l'état de l'art, puis nous avons essayé d'analyser l'existant afin d'éclaircir les objectifs de notre application. En se basant sur cette étude, nous spécifierons dans le chapitre suivant les différentes fonctionnalités de notre projet.

Chapitre 2

Analyse et spécification des besoins

Introduction

Dans ce chapitre, nous allons nous focaliser sur les fonctionnalités offertes par notre application ; cela en recensant les différents besoins fonctionnels et non fonctionnels qu'elle doit satisfaire et en présentant, à l'aide de la méthodologie UML (Unified Modeling Language), les diagrammes des cas d'utilisation.

2.1 Spécification non formelles des besoins

Dans cette partie et dans le but d'identifier les services que doit fournir notre application, nous allons détailler les exigences spécifiques qui consistent en des besoins fonctionnels et des besoins non fonctionnels.

2.1.1 Spécification des besoins fonctionnels

L'application doit pouvoir gérer trois types d'utilisateurs ; l'administrateur, le recruteur et les candidats.

Par rapport à l'administrateur, notre application doit répondre aux exigences suivantes :

- L'administrateur doit avoir un identifiant et un mot de passe à saisir pour se connecter.
- Il faut pouvoir visualiser l'ensemble des candidatures dans une même liste avec la possibilité d'appliquer des filtres sur celle-ci ;
- Il doit exister un espace emploi sur le site Web permettant l'administrateur de publier les différentes offres avec la possibilité de modifier, ajouter ou supprimer une offre.
- L'administrateur doit être capable d'identifier facilement les candidatures issues des annonces publiées ;

- L'administrateur doit être capable d'ajouter un questionnaire spécifique à chaque annonce pour qualifier les profils reçus et traiter en priorité les candidatures les plus pertinentes.

Par rapport au recruteur, notre application doit répondre aux exigences suivantes :

- Le recruteur doit avoir un identifiant et un mot de passe à saisir pour se connecter.
- Il faut pouvoir visualiser l'ensemble des candidatures ;
- Le recruteur doit être capable de bénéficier d'une traçabilité complète de chaque candidature : origine de la candidature, autres offres auxquelles le candidat a répondu ;
- Une fois les informations des candidatures sont étudiées, le recruteur doit être capable de retenir ou rejeter une candidature selon sa pertinence.

Par rapport aux candidats, notre application doit répondre aux exigences suivantes :

- Les candidats doivent être capable de créer un compte sur le site et avoir ainsi un identifiant et un mot de passe à saisir pour se connecter.
- L'application doit permettre aux candidats de postuler directement sur le site, sans passer par un email.
- Une fois le candidat a postulé, il doit être capable de suivre ou annuler ses candidatures.
- Chaque candidature est définie par son type, son statut (ouverte, demande d'informations, rejetée, etc.). Elle possède également un numéro de référence unique qui doit permettre de retrouver rapidement les documents (non numériques) qui lui sont liés ;
- Les saisies des candidatures doivent se faire via un formulaire avec des champs clefs obligatoires.

2.1.2 Spécification des besoins non fonctionnels

Afin d'assurer un bon fonctionnement de l'application et de garantir la satisfaction de l'utilisateur, des contraintes doivent être prises en compte tout au long du développement du projet :

- **Ergonomie et convivialité** : L'application doit fournir une interface simple et élégante pour l'utilisateur afin de faciliter l'exploitation des services de l'application ;
- **Portabilité, maintenance et évolution** : L'application doit être multiplateforme c'est-à-dire fonctionnelle sur n'importe quel système d'exploitation. Le code doit être bien lisible, compréhensible et modulaire pour garantir la souplesse, l'évolution et la maintenance de la solution et ainsi répondre aux changements du marché.
- **Rapidité et robustesse** : L'application doit assurer la rapidité de réponse et la fiabilité du résultat procuré : les candidats doit être capable de postuler facilement aux offres et l'administrateur doit être capable de sélectionner facilement, efficacement et d'une manière fiable les candidatures les plus pertinentes.

- **Sécurité** : L'application doit tenir compte de confidentialité des données des utilisateurs.

2.2 Spécification semi-formelle des besoins

Afin d'obtenir de manière plus formelle une vue globale sur les exigences de l'application et ainsi réussir une bonne spécification des besoins, ces derniers doivent être modélisés. Cette partie présente alors une modélisation de ces besoins en ayant recours aux concepts fondamentaux d'UML (Unified Modeling Language), à savoir le diagramme de cas d'utilisation et le diagramme de séquences.

2.2.1 Identification des acteurs

Avant d'entamer la présentation des diagrammes, il faut identifier les acteurs qui sont en interaction avec l'application. Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié. Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages susceptibles d'être porteurs de données.

Notre système nécessite l'intervention de trois acteurs principaux : le recruteur qui reçoit les candidatures, l'administrateur qui gère les compte et la mise à jour des offres d'emploi et les candidats qui sont à la recherche d'un emploi.

2.2.2 Diagramme des cas d'utilisation

Les cas d'utilisation permettent d'exprimer les besoins des utilisateurs d'un système. Le diagramme des cas d'utilisation permet donc d'identifier les possibilités d'interaction entre le système et les acteurs. Le use case, qui présente l'ensemble des fonctionnalités offerte par l'application pour nos utilisateurs(admin, recruteur et candidat), est donné par la figure 2.1.

Ce Diagramme met en évidence les fonctionnalités générales attendues de notre application. La gestion des candidature doit essentiellement tenir en compte une bonne gestion des offres et des comptes, une vue globale sur les transactions effectués par les candidats. Notre application permet aussi aux candidats de remplir une demande de candidature en ligne. Nous allons faire, dans ce qui suit, une description détaillée des cas d'utilisation les plus pertinents.

2.2.2.1 Cas d'utilisation : gérer les comptes

La figure 2.2 montre le diagramme relatif au cas d'utilisation : gérer les comptes. En choisissant de s'inscrire, le candidat remplit un formulaire. L'administrateur décide alors si ce candidat sera ajouté ou non. L'administrateur a la possibilité aussi de supprimer un candidat, si celui-ci

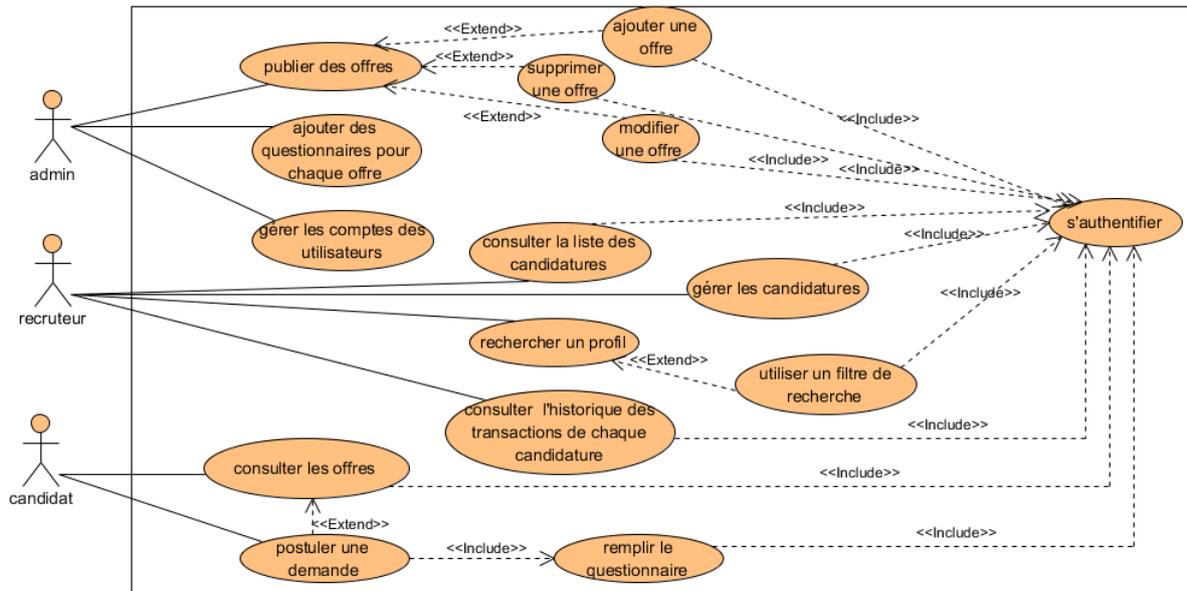


FIGURE 2.1 – Diagramme des cas d’utilisation général de l’application

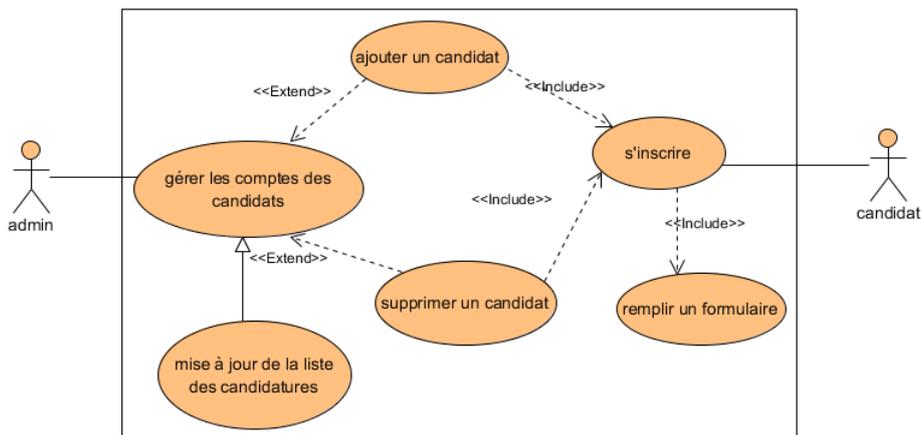


FIGURE 2.2 – Diagramme du cas d’utilisation : gérer les comptes

n’est plus actif ou s’il décide de se désinscrire du site. Dans le cas d’ajout ou de suppression, la liste des candidats est mise à jour.

2.2.2.2 Cas d’utilisation : s’authentifier

La figure 2.3 montre le diagramme relatif au cas d’utilisation : s’authentifier. Grâce à l’authentification, chaque utilisateur pourra accéder à certains volets de l’application selon la nature du privilège qu’il possède. Chacun des utilisateurs a un identifiant et un mot de passe qu’il doit saisir chaque fois qu’il désire s’authentifier.

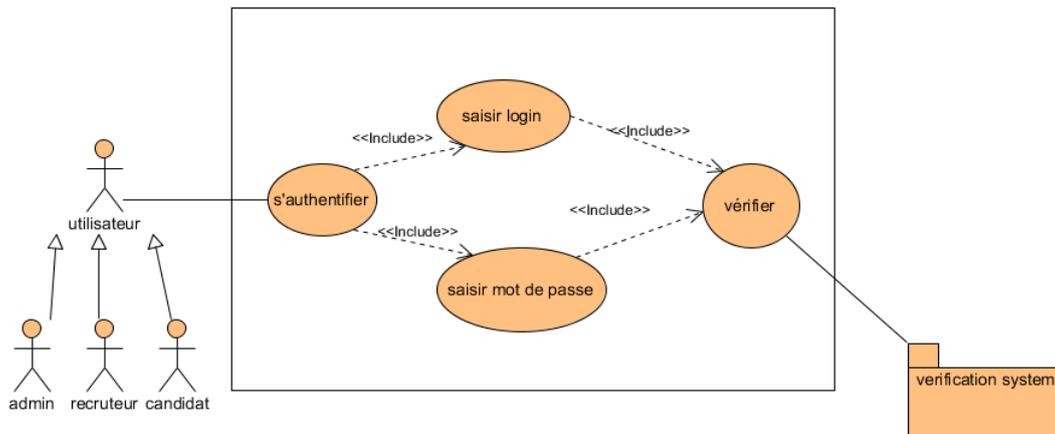


FIGURE 2.3 – Diagramme du cas d'utilisation : s'authentifier

2.2.2.3 Cas d'utilisation : postuler une candidature

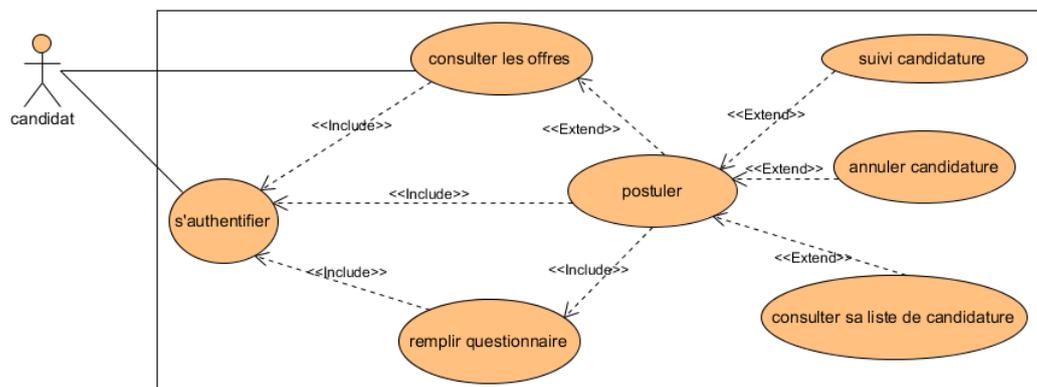


FIGURE 2.4 – Diagramme du cas d'utilisation : postuler une candidature

La figure 2.4 montre le diagramme relatif au cas d'utilisation : postuler une candidature. Une fois authentifié, la candidat a la possibilité de postuler une candidature directement après la consultation des offres d'emploi ajoutées par l'administrateur. Pour pouvoir postuler, un formulaire spécifique à chaque offre doit être rempli par le candidat. Ce dernier a le droit d'annuler ou de suivre sa candidature.

2.2.2.4 Cas d'utilisation : gérer les candidatures

La figure 2.5 montre le diagramme relatif au cas d'utilisation : gérer les candidatures. Le recruteur, préalablement authentifié, est responsable de l'ouverture et de la clôture des candidatures. Il a le droit de consulter les profils des candidats ainsi que les offres auxquels ils ont répondu. Avec la consultation des candidatures, le recruteur doit être capable de qualifier les profils reçus et apporter sa décision sur chaque candidature (refus, acceptation, mise en attente).

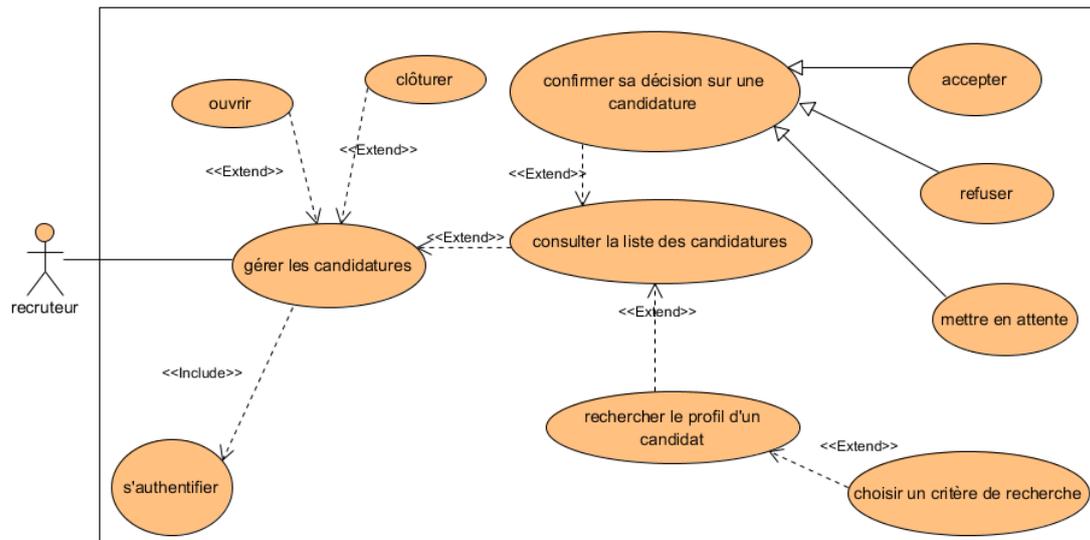


FIGURE 2.5 – Diagramme du cas d'utilisation : gérer les candidatures

2.2.3 Diagrammes de séquences

Pour illustrer les diagrammes des cas d'utilisation, nous allons introduire dans ce qui suit les diagrammes de séquences relatifs à notre application. Ce diagramme sert à représenter les successions particulières d'enchaînement d'un scénario en utilisant des conventions graphiques. Le scénario présenté par la figure 2.6 décrit les actions qu'affecte l'administrateur pour jouir des différentes fonctionnalités offertes par l'application à savoir :

- accéder à l'espace admin.
- consulter la liste des candidats.
- ajouter un candidat.
- supprimer un candidat.
- consulter la liste des offres.
- ajouter une offre.
- ajouter le formulaire correspondant à l'offre ajoutée.
- supprimer une offre.
- modifier une offre.

La figure 2.7 illustre le scénario effectué par le recruteur afin de :

- accéder à l'espace recruteur.
- consulter la liste des candidats.
- ajouter un candidat.
- supprimer un candidat.
- étudier les candidatures.

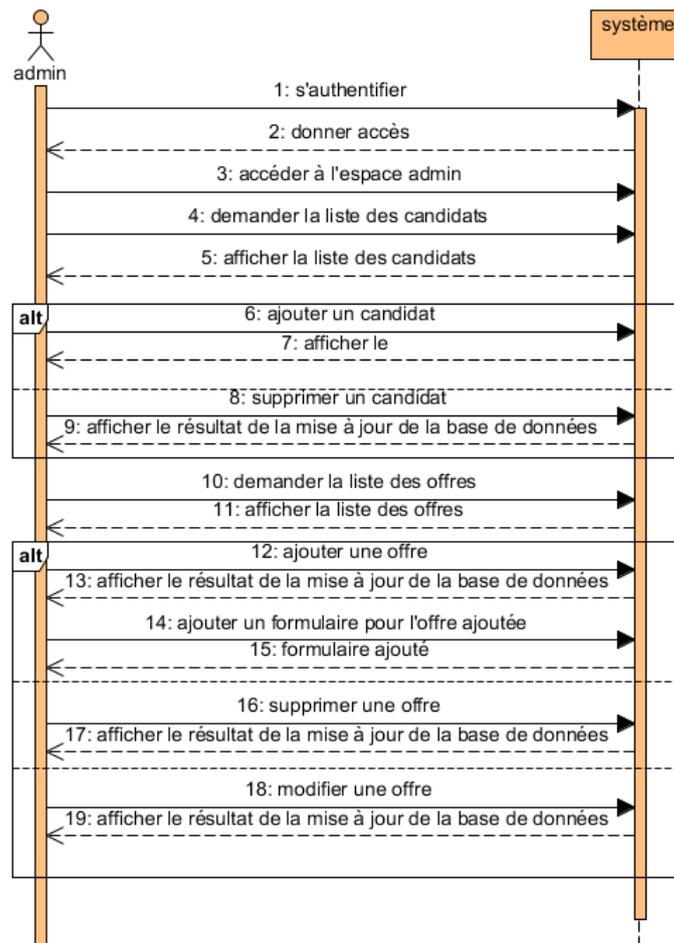


FIGURE 2.6 – Diagramme de séquences relatif à l’administrateur

- apporter sa décision.

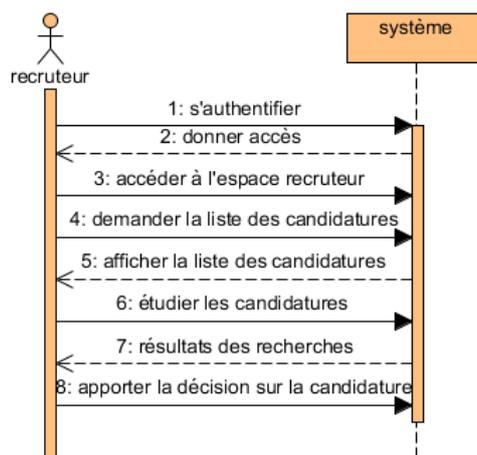


FIGURE 2.7 – Diagramme de séquences relatif au recruteur

La figure 2.8 illustre le scénario permettant le candidat de postuler à une offre d’emploi :

- accéder à l’espace candidat.

- consulter la liste des offres d'emploi.
- choisir une offre pour y postuler.
- remplir un formulaire.
- confirmer la postulation.

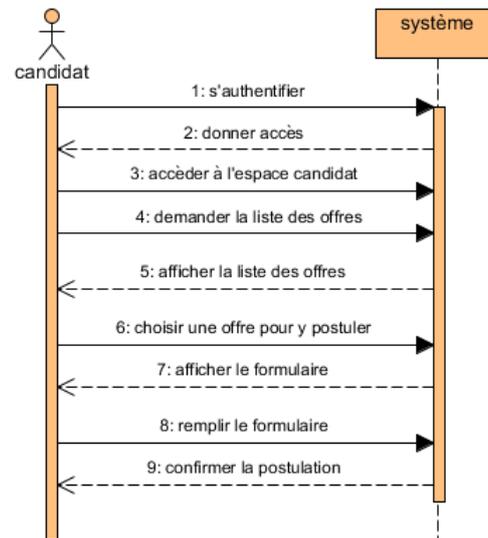


FIGURE 2.8 – Diagramme de séquences relatif au candidat

Conclusion

Dans ce chapitre nous avons tout d'abord décortiqué les besoins en termes d'exigences fonctionnelles et non fonctionnelles de l'application. Ces besoins sont par la suite modélisés au moyen des diagrammes UML (Unified Modeling Language). Grâce à cette analyse, nous sommes prêts à passer à la description détaillée de l'aspect architectural de l'application dans le chapitre suivant.

Chapitre 3

Conception

Introduction

Après avoir fixé les besoins et les objectifs dans le chapitre précédent, nous nous focalisons sur l'aspect architectural de l'application. Cette phase a pour but de concevoir les schémas généraux qui permettent la modélisation et la description d'une manière non ambiguë du fonctionnement désiré de l'application. Dans ce chapitre deux vues conceptuelles seront décrites. La première donne une vue globale de l'architecture tandis que la deuxième offre une vue détaillée en se basant sur le langage UML (Unified Modeling Language).

3.1 Architecture globale

Afin de réussir l'étape de conception, il est nécessaire de suivre un contexte conceptuel qui répond aux spécificités et aux besoins fondamentaux de l'application, menant ainsi à la satisfaction de ses utilisateurs. L'architecture client/serveur est l'architecture qui paraît le mieux adapté à notre application. Nous allons entamer cette partie par la définition de cette architecture.

3.1.1 Présentation de l'architecture client/serveur

Cette architecture est basée sur l'utilisation de deux types de logiciels, à savoir un logiciel serveur et un logiciel client s'exécutant normalement sur deux machines différentes.

On appelle logiciel serveur un programme qui offre un service sur le réseau. Le serveur accepte des requêtes, les traite et renvoie le résultat au demandeur.

On appelle logiciel client un programme qui utilise le service offert par un serveur. Le client envoie une requête et reçoit la réponse.

La seule obligation de coopération entre le client et le serveur est le respect du protocole qui

assure la communication inter-processus : HTTP pour le Web, POP/IMAP/SMTP pour le courrier électronique, SNMP pour l'administration de réseau.

La figure 3.1 illustre l'architecture client/serveur adapté pour cette application.

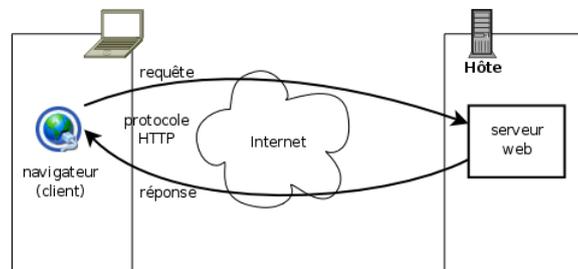


FIGURE 3.1 – Architecture client/serveur

3.1.2 Les avantages de l'architecture client/serveur

Pour le développement de notre application nous avons opté pour l'architecture client/serveur pour plusieurs raisons à savoir :

- Modèle adéquat pour la distribution des données.
- Prend avantage des fonctionnalités réseaux.
- Facile d'ajouter des serveurs supplémentaires ou de mettre à jour les serveurs existants.
- intégrité : les données du serveur sont gérées sur le serveur de façon centralisée. Les clients restent individuels et indépendants.
- Partage des ressources : un serveur traite plusieurs clients en même temps et contrôle leurs accès aux ressources.

3.2 Conception détaillée

Dans ce qui suit, nous détaillons la conception à l'aide de la méthodologie UML.

3.2.1 Vue statique

3.2.1.1 Diagramme de classes

Le diagramme de classes exprime la structure statique du système en termes de classes et de relations entre ces classes. Il fait abstraction des aspects dynamiques et temporels. La figure 3.2 montre le diagramme de classes qui collaborent au fonctionnement de notre application.

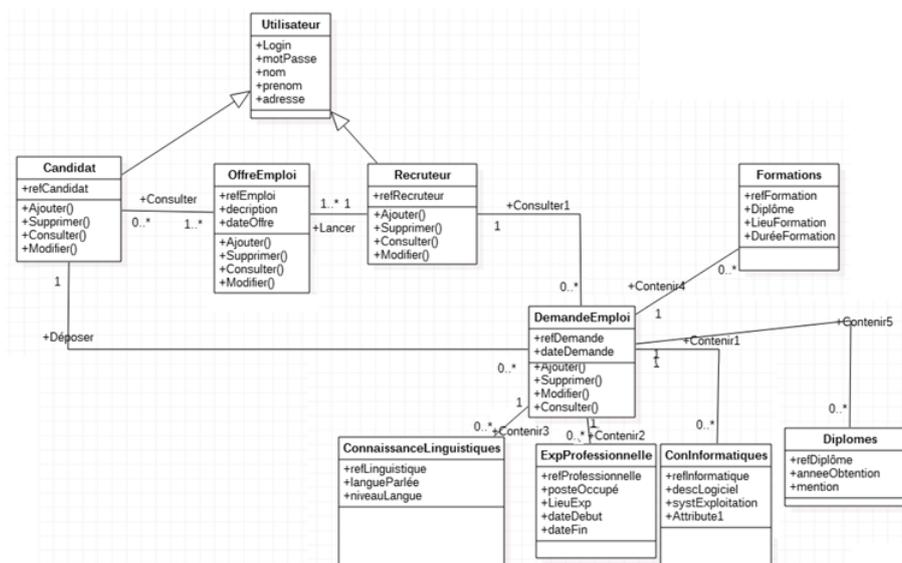


FIGURE 3.2 – Diagramme de classes

Ces classes sont décrites comme suit :

- La classe **Administrateur** : Cette classe a comme attributs la clé primaire qui identifie l'administrateur (**ref_Administrateur : Int**), le login (**login : String**) et le mot de passe (**motpasse : String**). L'administrateur doit s'authentifier pour pouvoir consulter, modifier, ajouter ou supprimer une offre.
- La classe **Recruteur** : Cette classe a comme attributs la clé primaire qui identifie le recruteur (**ref_recruteur : Int**), le nom et le prénom (**nomprenom : string**), le login (**login : String**) et le mot de passe (**motpasse : String**). Le recruteur doit s'authentifier pour pouvoir consulter la liste des demandes d'emploi.
- La classe **Candidat** : Cette classe a comme attributs la clé primaire qui identifie le candidat (**ref_candidat : Int**), le nom et le prénom (**nomprenom : string**), l'adresse (**adresse : String**), le login (**login : String**) et le mot de passe (**motpasse : String**).
- La classe **Demande** : Cette classe a comme attributs la clé primaire qui identifie la demande (**ref_demande : Int**), la date de la demande, les connaissances du candidat, etc.
- La classe **Offre** : Cette classe a comme attributs la clé primaire qui identifie l'offre de l'emploi (**ref_emploi : Int**), la date de l'offre, une description de l'offre, etc.

3.2.1.2 Diagramme entité association

Le diagramme de la figure 3.3 illustre les tables construits ainsi que les relations qui existent entre eux.

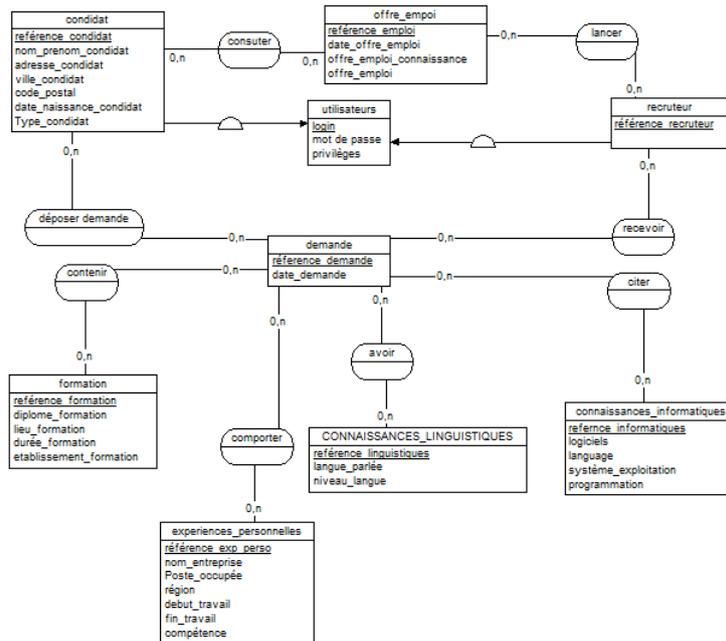


FIGURE 3.3 – Diagramme entité-association

3.2.2 Vue dynamique : diagramme de séquences

Dans cette section, nous présentons les diagrammes de séquences associés à notre application. Les diagrammes de séquences montrent les collaborations entre les objets selon un point de vue temporel en mettant l'accent sur la chronologie des envois de messages. Ainsi, Les diagrammes de séquences présentent une vue dynamique du système et la communication entre les couches. Dans ce qui suit nous présentons les diagrammes les plus importants de notre application.

Authentification

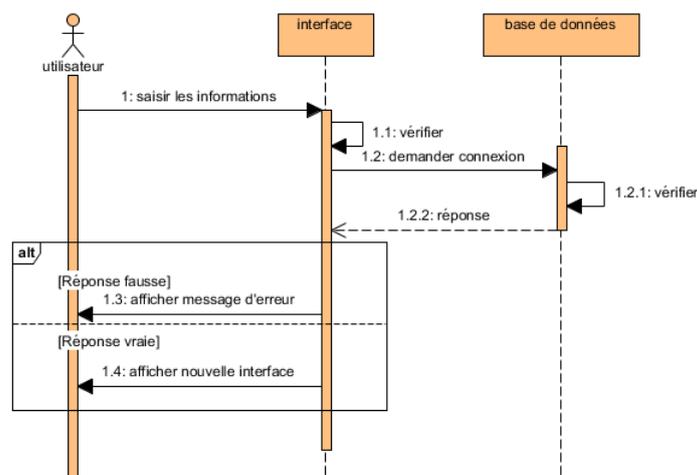


FIGURE 3.4 – Diagramme de séquences de l'authentification

Le diagramme de séquence présenté par la figure 3.4 est relatif au scénario de l'authentification qui est détaillé comme suit :

- Acteurs : les utilisateurs de l'application : le recruteur, l'administrateur et les candidats.
- But : s'authentifier pour profiter des différentes fonctionnalités offertes par l'application.
- Pré-conditions : L'utilisateur doit avoir un compte déjà créé.
- Enchaînement :
 - L'utilisateur demande l'accès au système.
 - L'utilisateur saisit les information : son identifiant et son mot de passe.
 - Le système vérifie l'existence de l'utilisateur dans la base de données.
 - Si l'utilisateur existe, le système affiche l'interface qui lui est relatif.
- Enchaînement alternatif : Si les champs ne sont pas valides, le système affiche un message d'erreur.
- Post-condition : L'utilisateur doit avoir l'accès aux fonctionnalités.

Gestion des offres

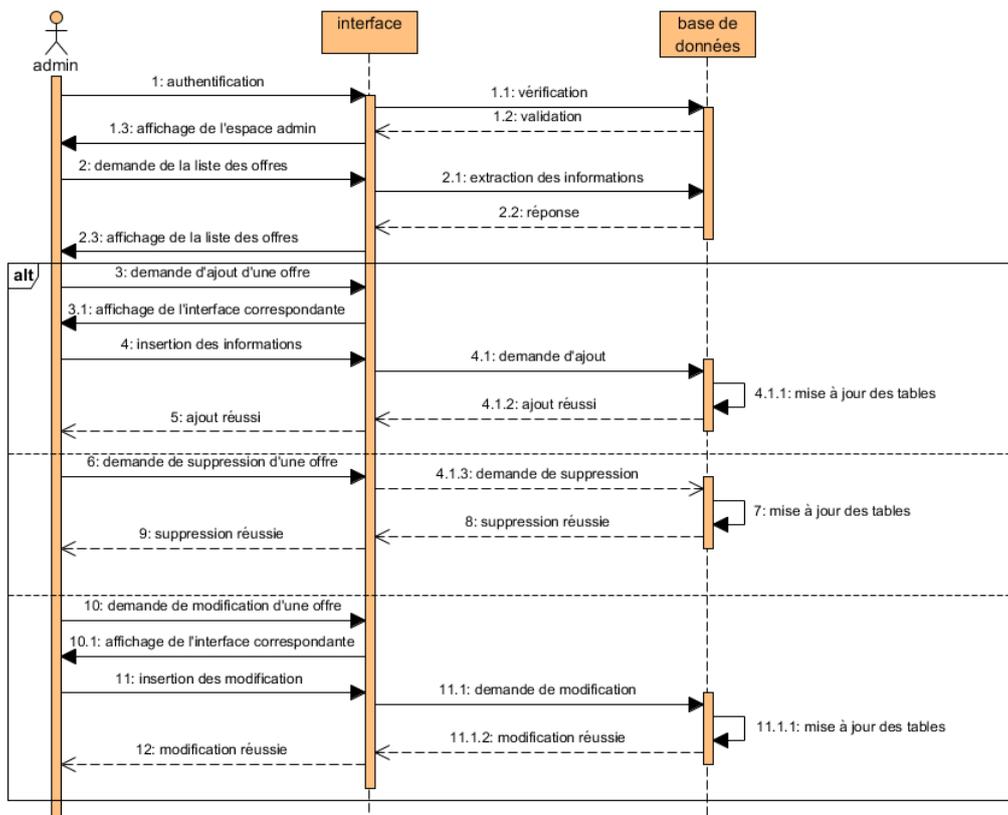


FIGURE 3.5 – Diagramme de séquences de la gestion des offres

Le scénario illustré par la figure 3.5 correspond à la gestion des offres. Il se déroule comme

suit : Après authentification, l'interface de l'espace administrateur est affiché. Si l'administrateur demande la liste des offres déjà existantes, le système extrait cette dernière de la base de donnée et l'affiche par la suite. L'administrateur possède le droit de porter ses modifications sur la liste. En effet, il a la possibilité d'ajouter, supprimer ou modifier une offre. Dans les trois cas, la base de donnée est mise à jour selon les changements subis.

Postulation

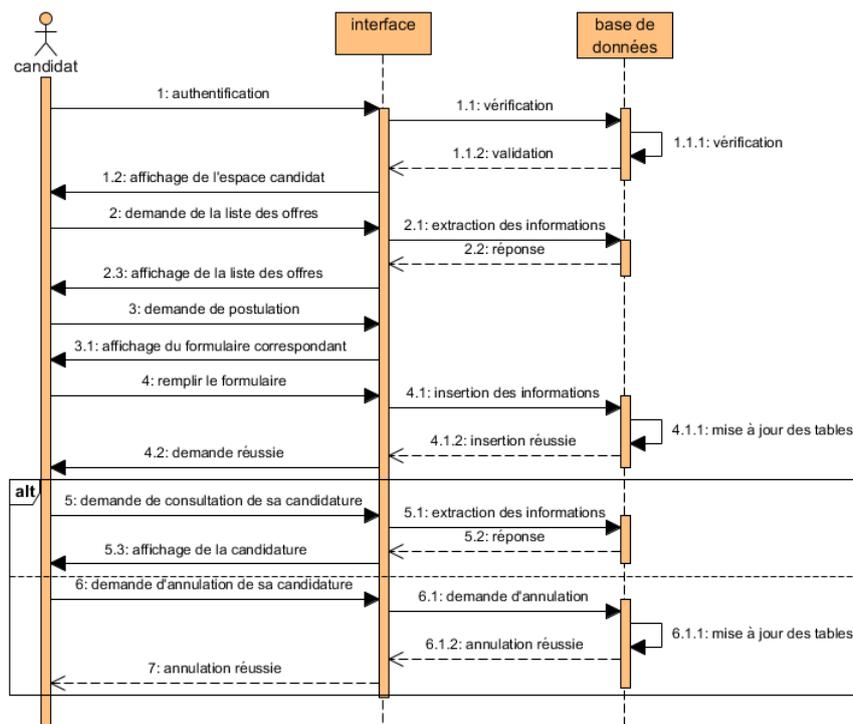


FIGURE 3.6 – Diagramme de séquences de la postulation

La figure 3.6 illustre le diagramme de séquence correspondant à la postulation. L'interface du candidat est affiché une fois celui-ci s'est authentifié. Pour pouvoir postuler, le candidat demande de consulter la liste des offres disponibles. Après extraction des informations à partir de la base de données, une nouvelle interface affichant les offres est lancée. Si le candidat décide de postuler pour une offre, un formulaire spécifique à l'offre s'affiche. Après le remplissage du formulaire par le candidat, la liste des candidature de la base de données est mise à jour. Après postulation, le candidat possède le droit de suivre ou d'annuler sa candidature.

Recherche

Le diagramme de séquence de la figure 3.7 est relatif au scénario de recherche du profil d'un candidat.

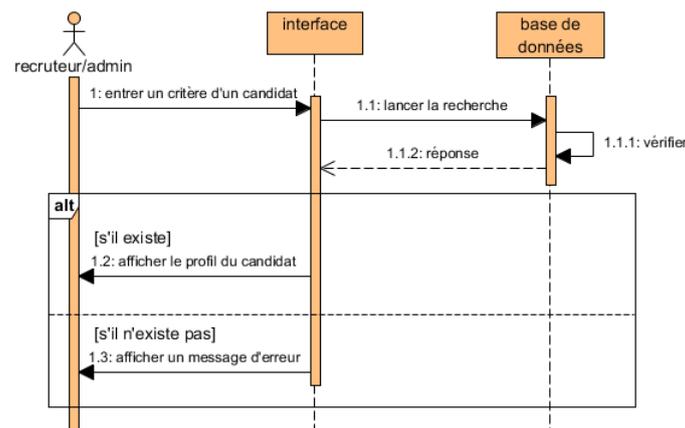


FIGURE 3.7 – Diagramme de séquences de la recherche d'un candidat

Avec la pré-condition d'authentification, l'utilisateur (l'administrateur ou le recruteur) entre un critère selon lequel il désire faire la recherche d'un candidat. Le système envoie une requête à la base de donnée pour extraire les informations nécessaires. Si le critère est vérifié, le profil correspondant est affiché. Sinon un message d'erreur est affiché.

Conclusion

A travers ce chapitre, nous avons présenté notre conception proposée pour l'application. Nous avons fourni, dans un premier lieu, une conception globale. Par la suite, nous avons détaillé la conception à travers les diagrammes UML qui nous ont permis de dégager la vue statique ainsi que la vue dynamique de l'application. Afin d'illustrer la concrétisation de cette conception, nous présenterons dans le chapitre suivant une description de l'état de la réalisation de notre projet.

Chapitre 4

Réalisation

Introduction

Cette partie contient le dernier volet de ce rapport. Elle a pour objectif d'exposer le travail achevé. Dans un premier temps, nous présentons l'environnement matériel, logiciel et les différents outils de développement utilisés. Dans un second temps, nous illustrons la réalisation de notre travail par des imprimes écran des interfaces les plus importantes de notre application.

4.1 Environnement de travail

Nous présentons dans cette section l'environnement matériel mis à la disposition du présent projet, ainsi que l'environnement logiciel utilisé pour le développement et la mise en place de notre application.

4.1.1 Environnement matériel

Pour la réalisation de ce projet, nous avons disposé d'un ordinateur TOSHIBA caractérisé par :

- **Processeur** : Intel(R) Core(TM) I3 2.4 GHz
- **Mémoire** : 8 Go de RAM .
- **Disque dur** : 750 Go .
- **Système d'exploitation** : windows 10.

4.1.2 Environnement logiciel

Dans ce qui suit, nous présentons l'environnement logiciel utilisé pour mener à terme ce sujet.

4.1.2.1 Adobe Dreamweaver

Adobe Dreamweaver (anciennement Macromedia Dreamweaver) est un éditeur du site web de type WYSIWYG. Il fut l'un des premiers éditeurs HTML, mais également l'un des premiers à intégrer un gestionnaire du site (CyberStudio GoLive étant le premier). Ces innovations l'imposèrent rapidement comme l'un des principaux éditeurs du site web, aussi bien utilisable par le néophyte que par le professionnel. Dreamweaver est édité par la société Adobe Systems et fait partie de la suite de développement Studio 8 de l'éditeur, qui comprend Macromedia Flash, Macromedia Fireworks (édition graphique) et Macromedia Coldfusion (serveur). Macromedia, qui éditait Dreamweaver auparavant, a été racheté par Adobe en décembre 2005.

Il offre, aujourd'hui, la possibilité de concevoir des feuilles de style. Les liaisons avec des bases de données ont également été améliorées ainsi que le chargement des fichiers sur les serveurs d'hébergement. Il propose, en outre, l'utilisation de modèles imbriqués de pages web, selon un format propriétaire. Depuis la version MX, il peut être utilisé avec des langages web dynamiques (ASP, PHP) à l'aide d'outils relativement simples d'utilisation. Il permet ainsi de développer des applications dynamiques sans connaissance préalable des langages de programmation.

4.1.2.2 Mysql

MySQL est un système de gestion de base de données (SGBD). C'est un serveur de bases de données relationnelles Open Source. Il stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table. Cela améliore la rapidité et la souplesse de l'ensemble. Il est très utilisé dans les projets libres et dans le milieu industriel. MySQL est très souvent utilisées avec PHP qui est un langage de programmation web côté serveur, ce qui veut dire que c'est le serveur qui va interpréter le code PHP (langage de scripts) et générer du code HTML qui pourra être interprété par le navigateur. Pour la gestion de ces derniers nous nous sommes servis de WampServer qui est un environnement serveur qui permet de développer les applications PHP et qui contient les dernières versions du serveur Web ultra connu Apache, PHP, MySQL et Phpmyadmin ;

4.1.2.3 PHP

PHP a été inventé à l'origine pour le développement d'applications web dynamiques qui constituent encore le cas d'utilisation le plus courant et son point fort. Cependant, les évolutions qui lui ont été apportées jusqu'à aujourd'hui assurent à PHP une polyvalence non négligeable. PHP est par exemple capable d'interagir avec Java, de générer des fichiers PDF, d'exécuter des commandes Shell, de gérer des objets (au sens programmation orientée objet), de créer des

images ou bien de fournir des interfaces graphiques au moyen de PHP GTK.

4.2 Exposition du travail réalisé

Dans cette partie, nous présentons notre travail en exposant des captures d'écran des interfaces les plus importantes de notre application.

4.2.1 Interface d'accueil



FIGURE 4.1 – Interface d'accueil

L'interface d'accueil de l'application contient le menu principal du site Web. Des liens hypertextes donnent la possibilité à l'utilisateur d'accéder directement aux rubriques qui l'intéressent. Ces pages constituent également des menus qui peuvent être subdivisés en sous-menus. Si l'utilisateur clique sur le lien "Recrutement" de l'interface d'accueil, il est dirigé vers une interface dont les rubriques offerts par cette interface sont : espace administrateur, espace recruteur, espace candidat et un lien pour pouvoir revenir à la page d'accueil.

4.2.2 Interface d'authentification

Quelque soit le type de l'utilisateur (administrateur, recruteur ou candidat), il doit s'authentifier avant d'accéder aux différentes fonctionnalités. L'authentification est assurée par l'interface illustrée par la figure 4.2. Dans cette interface, l'utilisateur saisit son identifiant et son mot de

passé. S'il ne dispose pas d'un compte, il doit s'inscrire en cliquant sur le bouton "Nouvelle inscription".



FIGURE 4.2 – Interface d'authentification

4.2.3 Inscription d'un candidat

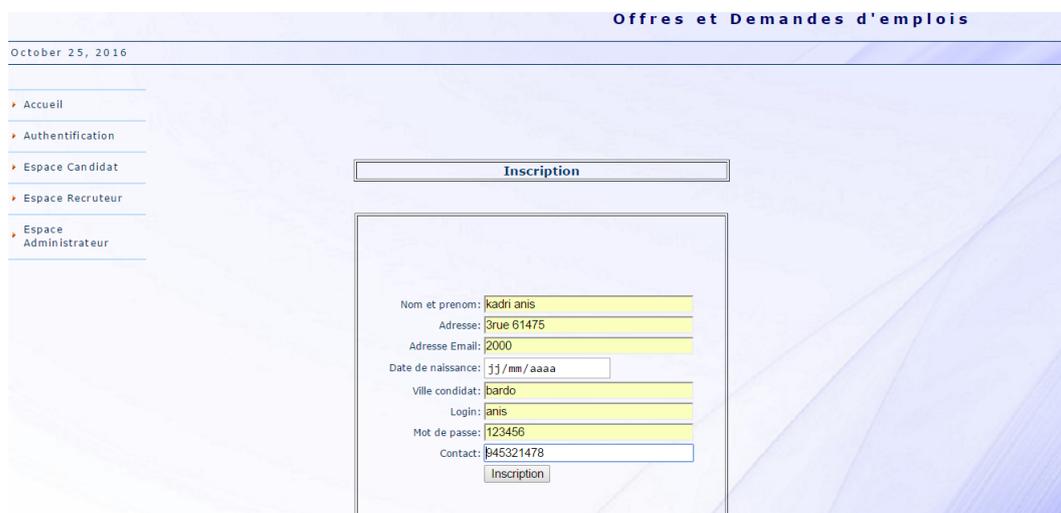


FIGURE 4.3 – Inscription d'un candidat

Si un candidat désire s'inscrire, un formulaire qu'il doit remplir est affiché comme le montre l'interface de la figure 4.3.

4.2.4 Interface de l'espace administrateur

Après authentification, l'administrateur peut jouir des fonctionnalités via l'"espace administrateur" présenté par la figure 4.4. Les interfaces des figures 4.5 et 4.6 présentent l'ajout et la

modification d'une offre d'emploi par l'administrateur.



FIGURE 4.4 – Interface de l'espace administrateur

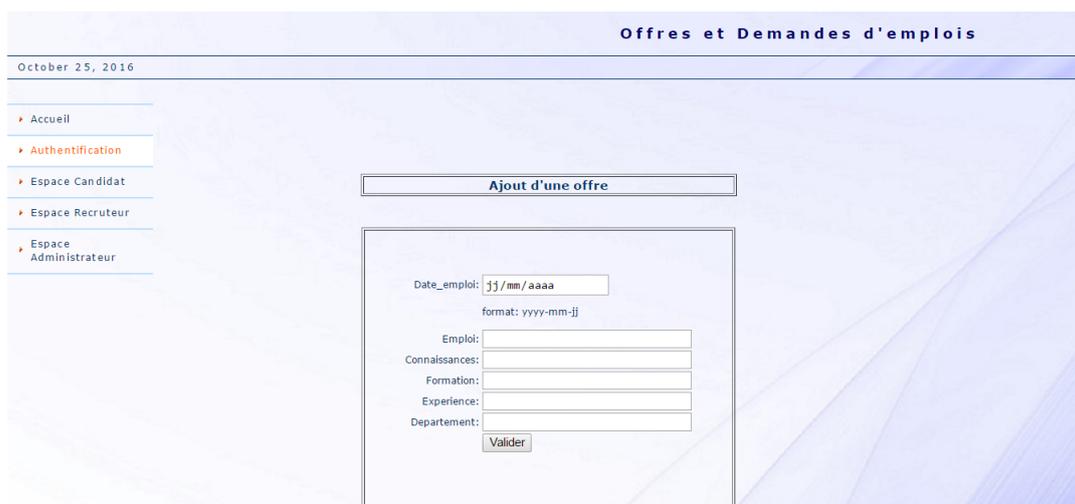


FIGURE 4.5 – Interface de l'espace administrateur : ajout d'une offre

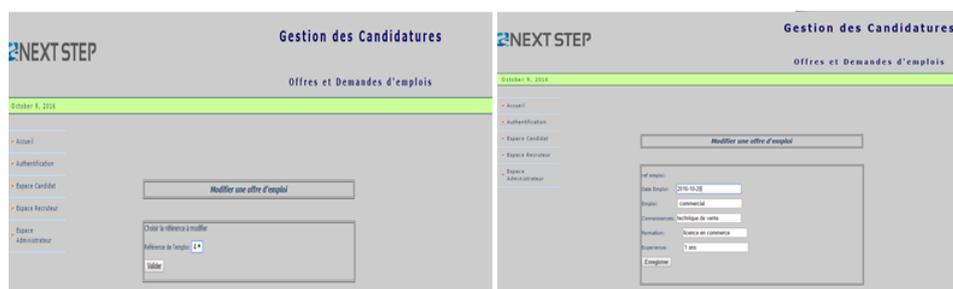


FIGURE 4.6 – Interface de l'espace administrateur : modification d'une offre

4.2.5 Interface de l'espace recruteur

Dans l'espace recruteur illustré par la figure 4.7, ce dernier dispose des services suivants : recherche d'un candidat et consultation des candidatures (figure 4.8) avec un champ pour écrire sa décision pour chaque candidature.



FIGURE 4.7 – Interface de l'espace recruteur



FIGURE 4.8 – Interface de l'espace recruteur : consultation des demandes

4.2.6 Interface de l'espace candidat

La figure 4.9 montre l'interface de l'espace candidat. Si le candidat clique sur le rubrique "consultation des offres" du menu, l'interface de la figure 4.10 est affichée. Dans cette dernière il peut accéder aux différentes offres disponibles. S'il souhaite postuler pour une offre, il n'a qu'à cliquer sur le bouton "postuler". Dans ce cas l'interface, présentée par la figure 4.11, est lancée

où figurent un formulaire composé par des onglets dont le candidat doit remplir les différents champs. Le candidat peut également consulter la liste de ses demandes et savoir la décision de l'administration comme le montre la figure 4.12.



FIGURE 4.9 – Interface de l'espace candidat

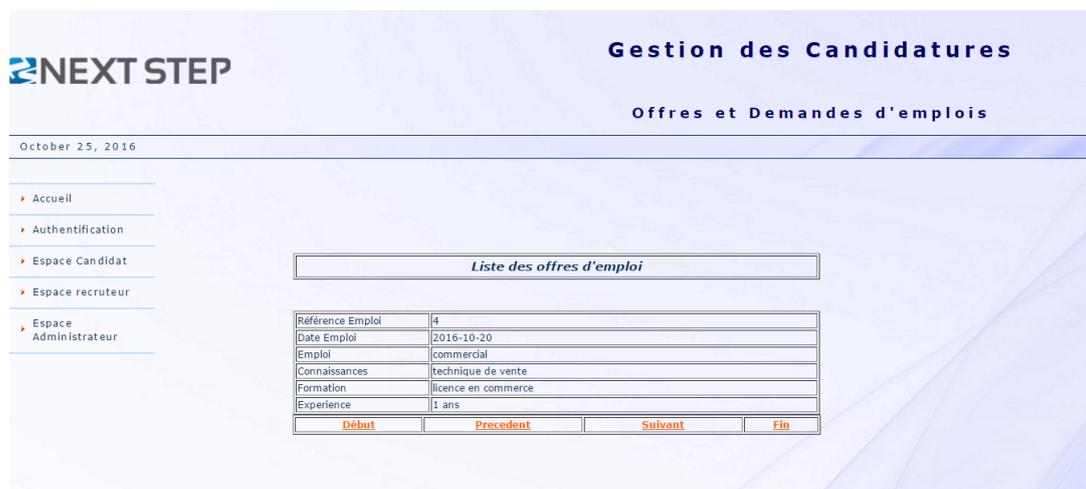


FIGURE 4.10 – Interface de l'espace candidat : consultation des offres.



FIGURE 4.11 – Interface de l'espace candidat : postulation à une offre.



FIGURE 4.12 – Espace candidat : Consultation des demandes

Conclusion

A ce stade, nous atteignons la fin de l'étude du projet. Dans ce dernier chapitre, nous avons à la fois décrit les environnements matériel et logiciel sur lesquels nous avons construit notre application. Ensuite, nous avons illustré les fonctionnalités importantes de l'application en fournissant quelques interfaces graphiques de l'application. A présent, nous passerons, dans la partie suivante, à la conclusion globale de notre projet.

Conclusion générale

Dans notre projet nous avons mis en œuvre une application web permettant la gestion des ressources humaines et plus particulièrement la gestion des candidatures pour la société "NEXT STEP IT". L'objectif est de fluidifier le marché de l'emploi et offrir plus de visibilité au CV d'un candidat, en simplifiant la démarche de dépôt de candidature.

Pour aboutir à ce résultat, nous avons tout d'abord commencé par une étude de l'art avec une étude de l'existant. Ensuite, nous avons analysé et spécifié les besoins nécessaires pour le développement de notre projet : nous avons pu dégager les fonctionnalités principales que l'application doit intégrer ainsi que les fonctionnalités secondaires qui répondent aux besoins non fonctionnels qu'elle doit satisfaire. Par la suite, nous avons procédé à l'étape de conception : nous avons commencé par l'architecture adoptée pour aboutir après à une conception détaillée qui met l'accent sur l'aspect statique et dynamique de l'application. Enfin, nous avons présenté les environnements matériels et logiciels utilisés ainsi qu'une illustration des différentes interfaces graphiques de notre application.

Ce projet nous a permis de s'adapter, de s'améliorer dans le développement des applications web, d'enrichir et d'approfondir nos connaissances techniques. En outre, il nous a été bénéfique dans la familiarisation avec la conduite de projet informatique de façon méthodique et organisée.

Bibliographie

- [1] Fannader, Rémy ; Leroux, Hervé : UML Principes de modélisation. Dunod, Paris 1999.
- [2] Kettani, Nasser et al. : De Merise à UML. Eyrolles, Paris 1998.

Nétographie

[n1] *<http://www.manager-go.com/>*

[n2] *<http://valorisationcapitalhumain.ca>*

[n3] *<http://www.piloter.org>*

[n4] *<https://grh.ooreka.fr>*

Annexe**Base de données****Liste des attributs**

Nom	Code	Type
adresse_candidat	ADRESSE_CONDISAT	TXT50
code_postal	CODE_POSTAL	N8
Compétence	COMPETENCE	TXT30
date_demande	DATE_DEMANDE	D
date_naissance_candidat	DATE_NAISSANCE_CONDISAT	D
date_offre_emploi	DATE_OFFRE_EMPLOI	D
debut_travail	DEBUT_TRAVAIL	D
diplome_formation	DIPLOME_FORMATION	TXT50
durée_formation	DUREE_FORMATION	TXT35
etablissement_formation	ETABLISSEMENT_FORMATION	TXT35
fin_travail	FIN_TRAVAIL	D
Language	LANGUAGE	TXT30
langue_parlée	LANGUE_PARLEE	TXT30
lieu_formation	LIEU_FORMATION	TXT50
Logiciels	LOGICIELS	TXT30
Login	LOGIN	TXT30
mot de passe	MOT_DE_PASSE	TXT30
niveau_langue	NIVEAU_LANGUE	I
nom_entreprise	NOM_ENTREPRISE	TXT25
nom_prenom_candidat	NOM_PRENOM_CONDISAT	TXT50
offre_emploi	OFFRE_EMPLOI	TXT30
offre_emploi_connaissance	OFFRE_EMPLOI_CONNAISSANCE	TXT40
Poste_occupée	POSTE_OCCUPEE	TXT30

Nom	Code	Type
Privilèges	PRIVILEGES	I
Programmation	PROGRAMMATION	TXT30
référence_condidat	REFERENCE_CONDIDAT	LI
référence_demande	REFERENCE_DEMANDE	LI
référence_demande	REFERENCE_CV	LI
référence_emploi	REFERENCE_EMPLOI	LI
référence_exp_perso	REFERENCE_EXP_PERSO	LI
référence_formation	REFERENCE_FORMATION	LI
référence_linguistiques	REFERENCE_LINGUISTIQUES	LI
référence_recuteur	REFERENCE_RECRUTEUR	LI
refernce_informatiques	REFERNCE_INFORMATIQUES	LI
Région	REGION	TXT25
système_exploitation	SYSTEME_EXPLOITATION	TXT30
Type_condidat	TYPE_CONDIDAT	TXT30
ville_condidat	VILLE_CONDIDAT	TXT25

Liste des entités

Nom	Code
Condidat	CONDIDAT
connaissances_informatiques	CONNAISSANCES_INFORMATIQUES
CONNAISSANCES_LINGUISTIQUES	CONNAISSANCES_LINGUISTIQUES
Demande	DEMANDE
experiences_personnelles	EXPERIENCES_PERSONNELLES
Formation	FORMATION
offre_emploi	OFFRE_EMPOI
Recuteur	RECRUTEUR
Utilisateurs	UTILISATEURS

Liste des associations

Nom	Code	Nombre	
avoir	AVOIR	0	
citer	CITER	0	
comporter	COMPORTE	0	
consuter	CONSUTER	0	
contenir	CONTENIR	0	
déposer demande	DEPOSER_DEMANDE	0	
lancer	LANCER	0	
recevoir	RECEVOIR	0	

Annexe03: Tables

Code sql pour générer la base de données

```
%% =====  
%%   Nom de la base   :   MODELE_1  
%%   Nom de SGBD     :   Sybase SQL Anywhere  
%%   Date de création :   18/06/2016   00:03  
%% =====  
  
%% =====  
%%   Table : FORMATION  
%% =====  
create table FORMATION  
(  
    REFERENCE_FORMATION      LI                not null,  
    DIPLOME_FORMATION        long varchar      ,  
    LIEU_FORMATION           long varchar      ,  
    DUREE_FORMATION          long varchar      ,  
    ETABLISSEMENT_FORMATION  long varchar      ,  
    primary key (REFERENCE_FORMATION)  
);  
  
%% =====  
%%   Table : EXPERIENCES_PERSONNELLES  
%% =====  
create table EXPERIENCES_PERSONNELLES  
(  
    REFERENCE_EXP_PERSO      LI                not null,  
    NOM_ENTREPRISE          long varchar      ,  
    POSTE_OCCUPEE          long varchar      ,  
    REGION                  long varchar      ,  
    DEBUT_TRAVAIL           date              ,
```

```

    FIN_TRAVAIL          date          ,
    COMPETENCE          long varchar  ,
    primary key (REFERENCE_EXP_PERSO)
);

%% =====
%%  Table : CONNAISSANCES_LINGUISTIQUES
%% =====

create table CONNAISSANCES_LINGUISTIQUES
(
    REFERENCE_LINGUISTIQUES  LI          not null,
    LANGUE_PARLEE           long varchar ,
    NIVEAU_LANGUE           integer     ,
    primary key (REFERENCE_LINGUISTIQUES)
);

%% =====
%%  Table : CONNAISSANCES_INFORMATIQUES
%% =====

create table CONNAISSANCES_INFORMATIQUES
(
    REFERNCE_INFORMATIQUES  LI          not null,
    LOGICIELS              long varchar ,
    LANGUAGE                long varchar ,
    SYSTEME_EXPLOITATION    long varchar ,
    PROGRAMMATION          long varchar ,
    primary key (REFERNCE_INFORMATIQUES)
);

%% =====
%%  Table : OFFRE_EMPLOI
%% =====

```

```
create table OFFRE_EMPLOI
(
  REFERENCE_EMPLOI      LI          not null,
  DATE_OFFRE_EMPLOI    date          ,
  OFFRE_EMPLOI_CONNAISSANCE long varchar ,
  OFFRE_EMPLOI          long varchar ,
  primary key (REFERENCE_EMPLOI)
);
```

```
%% =====
```

```
%% Table : DEMANDE
```

```
%% =====
```

```
create table DEMANDE
(
  REFERENCE_CV          LI          not null,
  DATE_DEMANDE          date          ,
  primary key (REFERENCE_CV)
);
```

```
%% =====
```

```
%% Table : UTILISATEURS
```

```
%% =====
```

```
create table UTILISATEURS
(
  LOGIN                 long varchar  not null,
  MOT_DE_PASSE          long varchar  ,
  PRIVILEGES            integer       ,
  primary key (LOGIN)
);
```

```
%% =====
```

```
%% Table : RECRUTEUR
```

```
%% =====
create table RECRUTEUR
(
    LOGIN                long varchar        not null,
    REFERENCE_RECRUTEUR  LI                  not null,
    MOT_DE_PASSE         long varchar        ,
    PRIVILEGES           integer              ,
    primary key (LOGIN, REFERENCE_RECRUTEUR)
);
```

```
%% =====
%% Table : CONDIDAT
%% =====
create table CONDIDAT
(
    LOGIN                long varchar        not null,
    REFERENCE_CONDIDAT  LI                  not null,
    MOT_DE_PASSE         long varchar        ,
    PRIVILEGES           integer              ,
    NOM_PRENOM_CONDIDAT long varchar        ,
    ADRESSE_CONDIDAT    long varchar        ,
    VILLE_CONDIDAT      long varchar        ,
    CODE_POSTAL          numeric(8)         ,
    DATE_NAISSANCE_CONDIDAT date            ,
    TYPE_CONDIDAT        long varchar        ,
    primary key (LOGIN, REFERENCE_CONDIDAT)
);
```

```
%% =====
%% Table : CONTENIR
%% =====
create table CONTENIR
```

```
(
  REFERENCE_CV          LI          not null,
  REFERENCE_FORMATION  LI          not null,
  primary key (REFERENCE_CV, REFERENCE_FORMATION)
);
```

```
%% =====
```

```
%% Table : COMPORTE
```

```
%% =====
```

```
create table COMPORTE
```

```
(
  REFERENCE_CV          LI          not null,
  REFERENCE_EXP_PERSO  LI          not null,
  primary key (REFERENCE_CV, REFERENCE_EXP_PERSO)
);
```

```
%% =====
```

```
%% Table : AVOIR
```

```
%% =====
```

```
create table AVOIR
```

```
(
  REFERENCE_CV          LI          not null,
  REFERENCE_LINGUISTIQUES LI      not null,
  primary key (REFERENCE_CV, REFERENCE_LINGUISTIQUES)
);
```

```
%% =====
```

```
%% Table : CITER
```

```
%% =====
```

```
create table CITER
```

```
(
  REFERENCE_CV          LI          not null,
```

```
REFERENCE_INFORMATIQUES      LI                not null,  
primary key (REFERENCE_CV, REFERENCE_INFORMATIQUES)  
);
```

```
%% =====  
%% Table : DEPOSER_DEMANDE  
%% =====
```

```
create table DEPOSER_DEMANDE  
(  
    LOGIN                long varchar        not null,  
    REFERENCE_CONDISAT   LI                 not null,  
    REFERENCE_CV         LI                 not null,  
    primary key (LOGIN, REFERENCE_CONDISAT, REFERENCE_CV)  
);
```

```
%% =====  
%% Table : RECEVOIR  
%% =====
```

```
create table RECEVOIR  
(  
    LOGIN                long varchar        not null,  
    REFERENCE_RECRUTEUR  LI                 not null,  
    REFERENCE_CV         LI                 not null,  
    primary key (LOGIN, REFERENCE_RECRUTEUR, REFERENCE_CV)  
);
```

```
%% =====  
%% Table : LANCER  
%% =====
```

```
create table LANCER  
(  
    REFERENCE_EMPLOI     LI                 not null,
```

```
LOGIN                long varchar        not null,
REFERENCE_RECRUTEUR  LI                    not null,
primary key (REFERENCE_EMPLOI, LOGIN, REFERENCE_RECRUTEUR)
);

%% =====
%%  Table : CONSUTER
%% =====

create table CONSUTER
(
  LOGIN                long varchar        not null,
  REFERENCE_CONDIDAT  LI                    not null,
  REFERENCE_EMPLOI    LI                    not null,
  primary key (LOGIN, REFERENCE_CONDIDAT, REFERENCE_EMPLOI)
);

alter table RECRUTEUR
  add foreign key FK_RECRUTEU_HERITAGE__UTILISAT (LOGIN)
    references UTILISATEURS (LOGIN) on update restrict on delete restrict;

alter table CONDIDAT
  add foreign key FK_CONDIDAT_HERITAGE__UTILISAT (LOGIN)
    references UTILISATEURS (LOGIN) on update restrict on delete restrict;

alter table CONTENIR
  add foreign key FK_CONTENIR_LIEN_91_DEMANDE (REFERENCE_CV)
    references DEMANDE (REFERENCE_CV) on update restrict on delete restrict;

alter table CONTENIR
  add foreign key FK_CONTENIR_LIEN_92_FORMATIO (REFERENCE_FORMATION)
    references FORMATION (REFERENCE_FORMATION) on update restrict on delete
restrict;
```

```
alter table COMPORTE

    add foreign key FK_COMPORTE_LIEN_93_DEMANDE (REFERENCE_CV)

        references DEMANDE (REFERENCE_CV) on update restrict on delete restrict;

alter table COMPORTE

    add foreign key FK_COMPORTE_LIEN_94_EXPERIEN (REFERENCE_EXP_PERSO)

        references EXPERIENCES_PERSONNELLES (REFERENCE_EXP_PERSO) on update restrict
on delete restrict;

alter table AVOIR

    add foreign key FK_AVOIR_LIEN_95_DEMANDE (REFERENCE_CV)

        references DEMANDE (REFERENCE_CV) on update restrict on delete restrict;

alter table AVOIR

    add foreign key FK_AVOIR_LIEN_96_CONNAISS (REFERENCE_LINGUISTIQUES)

        references CONNAISSANCES_LINGUISTIQUES (REFERENCE_LINGUISTIQUES) on update
restrict on delete restrict;

alter table CITER

    add foreign key FK_CITER_LIEN_100_DEMANDE (REFERENCE_CV)

        references DEMANDE (REFERENCE_CV) on update restrict on delete restrict;

alter table CITER

    add foreign key FK_CITER_LIEN_101_CONNAISS (REFERNCE_INFORMATIQUES)

        references CONNAISSANCES_INFORMATIQUES (REFERNCE_INFORMATIQUES) on update
restrict on delete restrict;

alter table DEPOSER_DEMANDE

    add foreign key FK_DEPOSER__LIEN_103_CONDIDAT (LOGIN, REFERENCE_CONDIDAT)

        references CONDIDAT (LOGIN, REFERENCE_CONDIDAT) on update restrict on delete
restrict;

alter table DEPOSER_DEMANDE

    add foreign key FK_DEPOSER__LIEN_104_DEMANDE (REFERENCE_CV)
```

```
references DEMANDE (REFERENCE_CV) on update restrict on delete restrict;
```

```
alter table RECEVOIR
```

```
add foreign key FK_RECEVOIR_LIEN_106_RECRUTEU (LOGIN, REFERENCE_RECRUTEUR)
references RECRUTEUR (LOGIN, REFERENCE_RECRUTEUR) on update restrict on
delete restrict;
```

```
alter table RECEVOIR
```

```
add foreign key FK_RECEVOIR_LIEN_107_DEMANDE (REFERENCE_CV)
references DEMANDE (REFERENCE_CV) on update restrict on delete restrict;
```

```
alter table LANCER
```

```
add foreign key FK_LANCER_LIEN_115_OFFRE_EM (REFERENCE_EMPLOI)
references OFFRE_EMPOI (REFERENCE_EMPLOI) on update restrict on delete
restrict;
```

```
alter table LANCER
```

```
add foreign key FK_LANCER_LIEN_116_RECRUTEU (LOGIN, REFERENCE_RECRUTEUR)
references RECRUTEUR (LOGIN, REFERENCE_RECRUTEUR) on update restrict on
delete restrict;
```

```
alter table CONSUTER
```

```
add foreign key FK_CONSUTER_LIEN_113_CONDIDAT (LOGIN, REFERENCE_CONDIDAT)
references CONDIDAT (LOGIN, REFERENCE_CONDIDAT) on update restrict on delete
restrict;
```

```
alter table CONSUTER
```

```
add foreign key FK_CONSUTER_LIEN_114_OFFRE_EM (REFERENCE_EMPLOI)
references OFFRE_EMPOI (REFERENCE_EMPLOI) on update restrict on delete
restrict;
```