**Ministère de l’Enseignement Supérieur et de la Recherche Scientifique**



Rapport de Projet de fin d’année

Spécialité : Génie Logiciel

Intitulé :

Conception et réalisation d’une plateforme d’interaction entre des entreprises et des candidats pour faciliter le recrutement et la recherche d’emploi « Co-Elit »

**Lieu du stage :** Apeiron Technologies

**Réalisé par :** Mohamed Belhassen ZINELABIDINE

Fahmi BOUMAIZA

**Encadré par :** M. Wael SAKHRI

**Année Universitaire 2017 / 2018**

Remerciements

Glossaire

**CV** Curriculum Vitae

TABLE DES MATIERES

Glossaire

Introduction Générale

1 Etude préalable

Introduction

* 1. Présentation du sujet
  2. Etude de l’existant
  3. Solution proposée
  4. Spécification des besoins
     1. Besoins fonctionnels
     2. Besoins non fonctionnels
  5. Méthodologie Adoptée
     1. SCRUM
     2. Organisation

Conclusion

1. Sprint 0 : Phase de préparation

Introduction

* 1. Planification des sprints
  2. Environnement et outils de travail
     1. Environnement matériel (chelou)
     2. Environnement logiciel
        1. JIRA
        2. BITBUCKET
        3. SOURCE TREE
        4. INTELLIJ IDEA
  3. Langages et framework
     1. Langage java
     2. JDL
     3. JDL Studio
     4. Spring
     5. Angular
     6. JHipster
  4. Patrons de conception
     1. Patron de conception MVC
  5. Architecture de l’application
  6. Diagrammes Génériques
     1. Diagramme de cas d’utilisation générale
     2. Diagramme de package(chelou à changer en sequence ?)
     3. Diagramme de classe

Conclusion

1. Chapitre 3 : Etude Conceptuelle

Inctroduction

* 1. Organisation
  2. Analyse
     1. Diagramme de cas d’utilisation
     2. Diagramme de sequence système
     3. Diagramme d’interation
  3. Tast Board
  4. BurnDown Chart
  5. Realisation

Conclusion

Conclusion Générale

Bibliographie

TABLE DES FIGURES

Figure 1.1 : Processus Scrum …………………………………………………………5

LISTE DES TABLEAUX

3.1 Descriptif de cas d’utilisation gérer question……………………………………………………………………………15

INTRODUCTION GENERALE

L'informatique est l'art, la technique ou la science qui consiste à manipuler des informations à l'aide d'un outil, l'ordinateur. L'informatique a pour objet de définir des algorithmes qui permettent de modifier la vision que l'on a d'un problème, ou d'extraire d'une grande quantité d'informations mal structurées, de nouvelles connaissances plus utiles.

L'informatique d'entreprise est parfois vendue à des utilisateurs professionnels sous la forme d'une plate-forme complète pouvant être largement appliquée à l'échelle d'une organisation, puis personnalisée par les utilisateurs dans chaque domaine. Cela signifie que les analyses, les rapports, la gestion de la base de données et d'autres applications sont standard dans le système, tandis que les packages d'application utilisés et les données accessibles dans chaque zone seront différents. En ce sens, l'informatique d'entreprise part de la recherche de solutions logicielles uniques à des problèmes métier spécifiques, tels que les logiciels d'inventaire ou de comptabilité. Au lieu de cela, l'informatique d'entreprise est destinée à offrir des solutions intégrées à ces problèmes.

De nos jours, le recrutement en ligne utilise le pouvoir d'Internet pour faire correspondre les gens aux emplois. En général, il s'agit de faire la publicité des postes vacants sur les sites d'emploi ou les sites Web d'entreprise. À ce niveau très basique, il est particulièrement efficace pour obtenir un niveau de réponse élevé. Bien qu'il puisse générer des centaines d'applications de plus que la publicité imprimée traditionnelle, simplement attirer plus de candidats n'est qu'une partie du travail.

La vraie force et le véritable pouvoir du recrutement en ligne, lorsqu'ils sont correctement réalisés, résident dans l'utilisation de la technologie Internet non seulement pour attirer des candidats, mais aussi pour y faire face. Dans ce sens, il s'agit aussi de rationaliser le processus de recrutement - les départements RH sont tellement occupés qu'ils peuvent offrir un meilleur service de recrutement à leurs collègues des finances, du marketing, des ventes et de la fabrication. De plus, cela libère plus de temps pour des tâches à plus forte valeur ajoutée.

Dans ce cadre, nous avons développé un programme d'application sur mesure pour les recruteurs, ce qui permettra d'économiser du temps, des efforts et de l'argent. Ils peuvent automatiser le processus de présélection en définissant des questions «tueur» (que seuls les meilleurs candidats répondront correctement), profilage et notation, tests psychométriques et analyses CV automatiques pour rechercher des mots clés tels que les qualifications et l'expérience.

Les résultats de notre travail sont présentés dans ce rapport qui comportera trois chapitres :

Le premier chapitre « **Etude préalable** » nous mettons l’accent sur le champ d’étude de notre application qui est le recrutement en ligne. Nous procédons par la suite à une synthèse des solutions existantes sur le marché en discutant les avantages et les inconvénients de chacune d’elles. Puis nous proposons les différentes solutions aux problèmes relevés.

Le second chapitre « **Phase de préparation** » nous introduisons une introduction à l’environnement et aux outils de travail.

Le troisième chapitre « **Etude Conceptuelle** » sera consacré à la méthodologie de conception générale et détaillée.

CHAPITRE 1

ETUDE PREALABLE ET ANALYSE DES BESOINS

# 1.1 Introduction

Dans ce chapitre nous procéderons à la présentation de l’organisme d’accueil ainsi que la présentation du projet. Nous passerons ensuite à la problématique et nous entamerons cette partie avec le choix méthodologique.

# Organisme d’accueil

Apeiron Tech. est une start-up tunisienne spécialisée dans le génie logiciel et le développement d'applications. Elle se localise à Sousse-Tunisie mais sa culture est sans limite, et sa pensée est universelle. Elle comporte une équipe multiculturelle d'experts et d'ingénieurs. Leur gamme de services est fournie dans le monde entier.

# Présentation du sujet

Apeiron Tech souhaite développer un système d’aide à la décision pour la gestion des recrutements qui se base sur les données fournies par la plateforme. Ce système sera non seulement bénéfique pour les entreprises qui auront la possibilité de mettre en place et personnaliser leur propre processus de recrutement à distance, voir même former les candidats en mettant en place un système de formation, mais aussi pour les candidats qui pourraient proposer leurs services ou bien participer aux différentes compagnes de recrutement.

# Problématique

L’objectif d’une plateforme de recrutement en ligne sera d’atteindre la bonne cible au bon moment et de maximiser la qualité des recrues. Dans ce contexte, notre objectif serait d’apporter un trafic de qualité sur la plateforme. Pour cette raison, nous développerons une plateforme web, facile à utiliser et répondant aux attentes des clients.

# Choix méthodologique

Pour aboutir à un travail dans les normes, il nous faut un choix méthodologique bien réfléchi en fonction des besoins. Nous présenterons pour débuter la définition d’une méthode agile.

## Méthode agile

Le terme "agile" définit une approche de gestion de projet qui prend le contre-pied des approches traditionnelles prédictives et séquentielles de type cycle en V ou waterfall (en cascade). La notion même de "gestion de projet" est remise en question au profit de "gestion de produit". De façon à raisonner davantage "produit" que "projet". Après tout l'objectif d'un projet consiste bien à donner naissance à un produit.

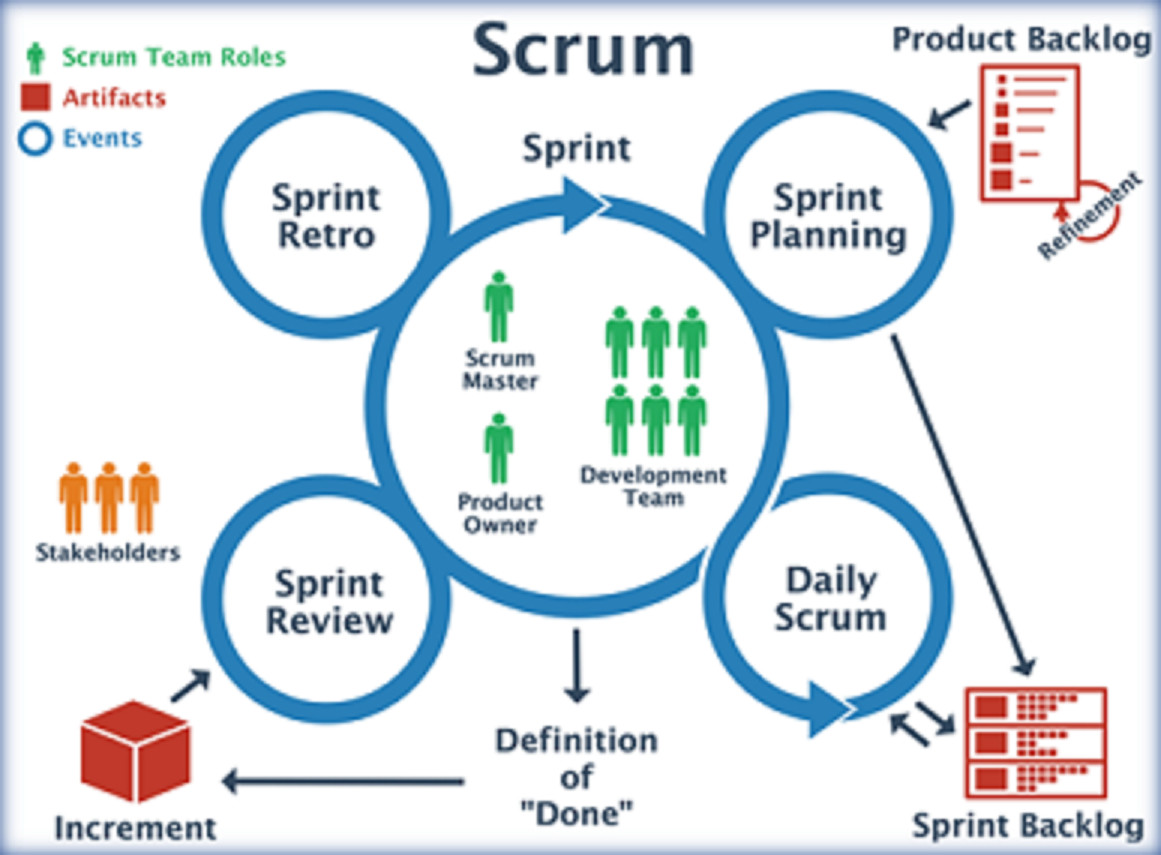
Une approche dite "traditionnelle" attend généralement du client une expression détaillée et validée du besoin en entrée de réalisation, laissant peu de place au changement. La réalisation dure le temps qu'il faut et le rendez-vous est repris avec le client pour la recette. Cet effet tunnel peut être très néfaste et conflictuel, on constate souvent un déphasage entre le besoin initial et l'application réalisée. On se rapporte alors aux spécifications validées et au contrat. Certains projets se terminent dans la douleur (surtout dans le cadre d'un contrat au forfait classique) au risque de compromettre la relation client. De plus il n'est pas rare que certaines fonctionnalités demandées se révèlent finalement inutiles à l'usage alors que d'autres, découvertes en cours de route, auraient pu donner plus de valeur au produit.

L'approche Agile propose au contraire de réduire considérablement voire complètement cet effet tunnel en donnant davantage de visibilité, en impliquant le client du début à la fin du projet et en adoptant un processus itératif et incrémental. Elle considère que le besoin ne peut être figé et propose au contraire de s'adapter aux changements de ce dernier. Mais pas sans un minimum de règles. [1]

## Méthode agile adoptée : SCRUM

Scrum est un cadre de travail permettant de répondre à des problèmes complexes et changeants, tout en livrant de manière productive et créative des produits de la plus grande valeur possible. [2]

Ce cadre est constitué d'une définition des rôles, de réunions et d'artefacts comme le montre la figure ci-dessous :



**Figure 1.1** : Processus SCRUM

Scrum définit 3 rôles :​

* **Le « Product Owner »** qui porte la vision du produit à réaliser (représentant généralement le client).
* **Le « Scrum Master »** garant de l'application de la méthodologie Scrum.
* **L'équipe de développement** qui réalise le produit.

La vie d'un projet Scrum est rythmée par un ensemble de réunions clairement définies et strictement limitées dans le temps (timeboxing) :

* **Planification du Sprint** (Sprint = itération) : au cours de cette réunion, l'**équipe de développement** sélectionne les éléments prioritaires du « **Product Backlog** » (liste ordonnancée des exigences fonctionnelles et non fonctionnelles du projet) qu'elle pense pouvoir réaliser au cours du sprint (en accord avec le « **Product Owner** »).
* **Revue de Sprint** : au cours de cette réunion qui a lieu à la fin du sprint, l'**équipe de développement** présente les fonctionnalités terminées au cours du sprint et recueille les feedbacks du **Product Owner** et des utilisateurs finaux. C'est également le moment d'anticiper le périmètre des prochains sprints et d'ajuster au besoin la planification de release (nombre de sprints restants).
* **Rétrospective de Sprint** : la rétrospective qui a généralement lieu après la revue de sprint est l'occasion de s'améliorer (productivité, qualité, efficacité, conditions de travail, etc.) à la lueur du "vécu" sur le sprint écoulé (principe d'**amélioration continue**).
* **Mêlée quotidienne** : il s'agit d'une réunion de synchronisation de l'équipe de développement qui se fait debout (elle est aussi appelée "stand up meeting") en 15 minutes maximum au cours de laquelle chacun répond principalement à 3 questions : « Qu'est-ce que j'ai terminé depuis la dernière mêlée ? Qu'est-ce que j'aurai terminé d'ici la prochaine mêlée ? Quels obstacles me retardent ? » [3]

## **Conclusion**

Dans ce premier chapitre nous avons mis le sujet dans son cadre général. Nous allons commencer un deuxième chapitre intitulé «Etude Préalable et Analyse des Besoins» dans lequel nous allons préciser nos besoins après une étude analytique d’un exemple de plateforme de recrutement en ligne en déduisant ses avantages et ses défaillances que nous allons dépasser dans notre projet.

CHAPITRE 2

PHASE DE PREPARATION

# 2.1 Introduction

Dans le premier chapitre, nous avons opté pour l’utilisation de la méthodologie Scrum pour la conception de notre application. Avant de se lancer dans la réalisation de tout projet, il est indispensable de bien étudier et analyser des projets similaires pour profiter des avantages et éviter les malveillances dans le projet présent.

Pour cela une étude préalable de l’existant est primordiale. Nous procéderons par la suite à une planification des sprints ainsi que la définition des outils choisis pour le développement de l’application.

# 2.2 Etude de l’existant

Tout d’abord avant d’entamer la phase de planification du travail, il faut appréhender suffisamment l’environnement existant.

## 2.2.1 Critique de l’existant

Pour espérer décrocher un poste de travail, le candidat doit s’armer d’un CV riche en informations. Les sociétés de leur côté s’appuient sur ce dernier afin de présélectionner les candidats pour un éventuel entretien.

Premièrement, pour une entreprise, la présélection du candidat est conditionnée entre autres par les informations écrites sur papier par ce dernier. Sans preuve du contraire lors d’un entretien, le candidat est la personne qu’il a décrite sur son CV. Dans ce genre de situation, le gaspillage des ressources humaines est inévitable.

En autre, un des inconvénients des entretiens, que le candidat n’est pas libre de choisir l’heure et l’endroit, en effet ces derniers seront imposés par l’organisme d’accueil.

## 2.2.2 Solution proposée

Après un examen approfondi de l’existant, nous pouvons constater que l’intérêt de notre plateforme est considérable. En effet, le processus de recrutement prend souvent du temps et dépend de la disponibilité du chargé de recrutement et de la sincérité et la volonté du candidat. La plateforme permettra aux candidats de passer par une série d’étapes comportant une série de tests ou de formations imposés par l’entreprise en question et finir par avoir une chance d’être embauché et cela sans avoir à se déplacer, donc à distance.

# 2.3 Planification des sprints

Dans Scrum, la réunion de planification du sprint est suivie par le Product Owner, ScrumMaster et toute l'équipe Scrum. Les parties prenantes extérieures peuvent assister à l'invitation de l'équipe, bien que cela soit rare dans la plupart des entreprises.

Pendant la réunion de planification du sprint, le propriétaire du produit décrit les fonctionnalités les plus prioritaires pour l'équipe. L'équipe pose suffisamment de questions pour pouvoir transformer une tâche importante du Product Backlog en tâches plus détaillées du Sprint Backlog. Deux artefacts définis résultent d'une réunion de planification des Sprints :

* Un objectif de sprint
* Un Sprint Backlog

Un objectif de sprint est une description courte, en une ou deux phrases, de ce que l'équipe prévoit d'accomplir pendant le sprint. Il est écrit en collaboration par l'équipe et le propriétaire du produit. Voici des exemples de buts de sprint sur notre application :

* En tant que candidat je peux gérer mon profil, notamment ajouter, supprimer et mettre à jour mes préférences.
* En tant qu’entreprise je peux mettre place une série de tests.

L'objectif du sprint peut être utilisé pour signaler rapidement à ceux qui sont en dehors du sprint. Il y a toujours des parties prenantes qui veulent savoir sur quoi travaille l'équipe, mais qui n'ont pas besoin d'en savoir plus sur chaque élément du Product Backlog (user story).

Le succès du sprint sera ensuite évalué lors de la réunion de révision de sprint par rapport à l'objectif du sprint plutôt que par rapport à chaque élément spécifique sélectionné dans le Product Backlog.

Le Sprint Backlog est l'autre sortie de la planification de sprint. Un Sprint Backlog est une liste des éléments du Product Backlog que l'équipe s'engage à fournir, ainsi que la liste des tâches nécessaires à la livraison de ces éléments de Product Backlog. Chaque tâche sur l'arriéré de sprint est également généralement estimée.

Un point important à réitérer ici est que c'est l'équipe qui choisit la quantité de travail qu'ils peuvent faire dans le prochain sprint. Le propriétaire du produit n'a pas à dire : "Il nous reste quatre sprints, donc vous devez faire le quart de tout ce dont j'ai besoin." Nous pouvons espérer que l'équipe en fera autant (ou plus), mais c'est à l'équipe de déterminer ce qu'elle peut faire au sprint.

# 2.4 Environnement et outils de travail

## 2.4.1 JIRA

Jira Software est un outil de gestion de projet Agile qui prend en charge toute méthodologie Agile, qu'il s'agisse d'un scrum, d'un kanban ou de votre propre création. Grâce à un seul outil, vous pouvez planifier, suivre et gérer tous vos projets de développement logiciel Agile, des tableaux aux rapports Agile. Choisissez un framework pour découvrir comment Jira Software peut aider votre équipe à livrer des logiciels de qualité supérieure, et ce, plus rapidement.[4]

2.4.2 BITBUCKET

Bitbucket est un service web d'hébergement et de gestion de développement logiciel utilisant les logiciels de gestion de versions Git et Mercurial. [5]

2.4.3 SOURCE TREE 

Sourcetree simplifie la façon dont vous interagissez avec vos référentiels Git afin que vous puissiez vous concentrer sur le codage. Visualisez et gérez vos référentiels grâce à l'interface Git simple de Sourcetree.[6]



2.4.4 INTELLIJ IDEA

IntelliJ IDEA (couramment appelé IntelliJ) est un Environnement de développement intégré (IDE) Java développé et édité par JetBrains.

IntelliJ est un IDE complet misant sur la productivité avec des systèmes d'auto-complétion intelligente, d'analyse de code en temps réel, de refactoring avancé ; l'intégration d'outils de tests et de debugging ; et une pléthore de raccourcis clavier permettant de réaliser presque n'importe quelle tâche rapidement sans jamais lever les mains du clavier pour utiliser la souris.[7]

# 2.5 Langages et Framework



## 2.5.1 Langage JAVA

Java est un langage de programmation et une plate-forme informatique qui ont été créés par Sun Microsystems en 1995. [8]



2.5.2 JDL (JHipster Domain Language)

Le JDL est un langage de domaine spécifique à JHipster où nous avons ajouté la possibilité de décrire toutes vos entités et leurs relations dans un seul fichier (ou plus d'un) avec une syntaxe simple et conviviale. [9]



2.5.3 JDL Studio

Un outil pour dessiner des diagrammes UML impertinents basés sur la syntaxe JDL. Fournit un retour instantané et a un style personnalisable. [10]



2.5.4 Spring

En informatique, Spring est un framework libre pour construire et définir l'infrastructure d'une application java4, dont il facilite le développement et les tests. [11]



2.5.5 Angular

Angular est une plateforme de développement qui permet de créer des applications web dynamiques et immersives. [12]



2.5.6 JHipster

JHipster est une plate-forme de développement pour générer, développer et déployer des applications Web Spring Boot + Angular / React et des microservices Spring. [13]

# Conclusion

Dans ce chapitre nous avons présenté les différents langages et framework utilisés lors du développement du projet. Dans le chapitre qui suit, nous allons citer les besoins fonctionnels et les besoins non fonctionnels qui pour pouvoir se concentrer sur les problèmes à résoudre pendant la réalisation de notre projet.

Chapitre 3 Etude conceptuelle

Introduction

Une étude conceptuelle bien élaborée est très essentielle pour la réalisation d’une application de haute qualité. C’est dans ce contexte que nous devons éclaircir en un premier temps la réalisation des étapes principales du développement d’un projet afin que ce dernier puisse respecter les attentes du client.

Comme première étape nous allons décrire les différents patrons de conception utilisés dans la réalisation de notre site web, puis nous allons présenter les différents types de diagrammes qui vont démontrer notre choix conceptuel.

1. Patrons de conception

Les patrons de conception ont contribué énormément à incruster chez les développeurs les bonnes pratiques basées sur l'expérience. Ils proposent des solutions à des problèmes qui peuvent être difficilement résolus par un seul composant, d’où la nécessité d’impliqué plusieurs rôles qui peuvent être joués par plusieurs composants d'un logiciel. [14]

* + 1. Patron de conception MVC
* **Le Controller :**

○ C’est lui qui valide la requête, remplit le modèle, et redirige vers la bonne vue.  
○ Supporte l’injection de dépendance : c’est ainsi qu’il accède à la  
couche Service.

* **La vue :**

○ C’est typiquement un composant « Angular » qui va être chargée d’afficher le modèle dans notre *Single Page Application* (SPA).

* Le modèle :   
  ○ Un Java Bean ou un ensemble de Java Beans (Collection), provenant de la couche service et étant affiché par un composant « Angular ».  
  ○ Dans l’architecture présente, ce Java Bean est généralement une entité Hibernate, ou un ensemble d’entités Hibernate.  
  ○ Il peut être validé par Bean Validation.
  + 1. Patron de conception DTO

Un objet de transfert de données (DTO en anglais) est un patron de conception utilisé dans les architectures logicielles objet.

Son but est de simplifier les transferts de données entre les sous-systèmes d'une application logicielle. Les objets de transfert de données sont souvent utilisés en conjonction des objets d'accès aux données.

L’apport de ce patron de conception est de limiter le nombre d’appel de méthode pour récupérer les différentes données requises du coté serveur à travers une interface qui regroupe plusieurs entités vers l’objet de transfert de données correspondant. [15]

1. Conception générale

Au cours de cette partie, nous proposons de détailler la conception de notre site web en allant du plus général vers le plus particulier.

* + 1. Identification des acteurs

Un acteur c’est l’idéalisation d’un rôle joué par une personne externe, un processus ou une chose qui interagit avec un système.

Les acteurs de notre projet :

* **Les candidats :** ils représentent les personnes à la recherche d’un emploi qui vont entamer une série d’épreuves et personnaliser leur profil pour augmenter leur chance d’embauche au sein d’une entreprise.
* **Les entreprises :** ils représentent les organisations qui vont établir un contact avec un candidat qui vas passer par leur propre processus de recrutement.
* L’administrateur : pour les sites web on l’appelle généralement « le webmaster ». C’est celui qui assure le dynamisme du site et veille sur sa performance.
  + 1. Identification des cas d’utilisations

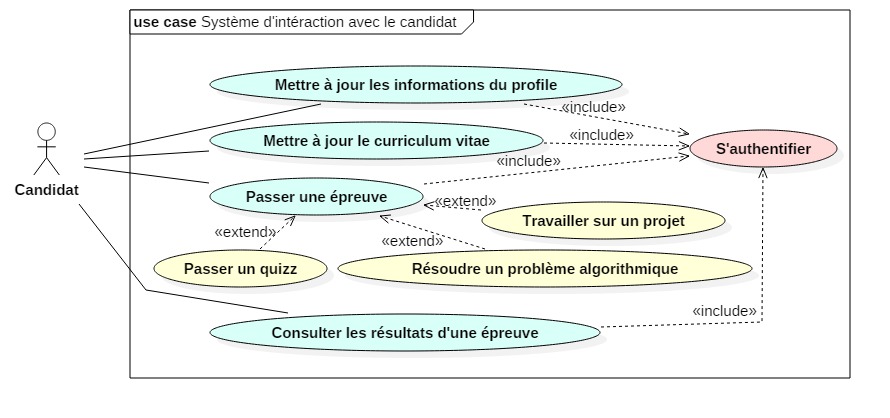
Durant cette partie, on va présenter les cas d’utilisations des acteurs qui interagissent avec le système.

Figure 3.1 **-** Diagramme de cas d’utilisation du candidat

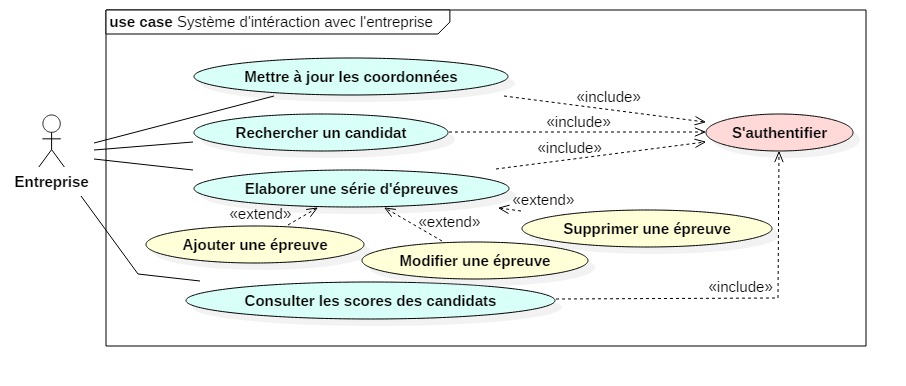
Le candidat peut mettre à jour ses informations personnelles ainsi que son curriculum vitae, il peut aussi passer des tests publiés par des entreprises et consulter son score. Toutes ces actions requièrent une authentification par un email et un mot de passe comme il est indiqué dans la figure 3.1 ci-dessus.

Figure 3.2 **-** Diagramme de cas d’utilisation d’une entreprise

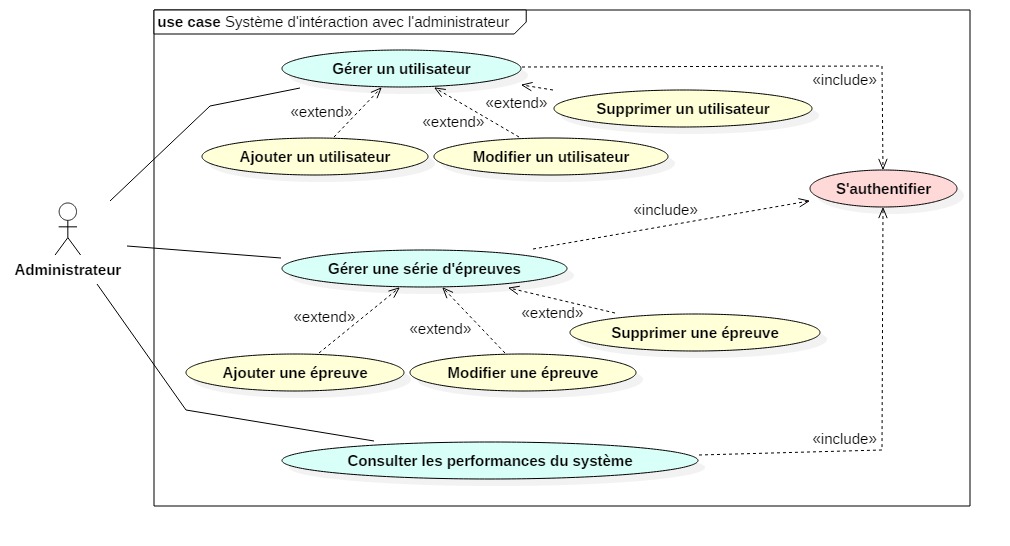
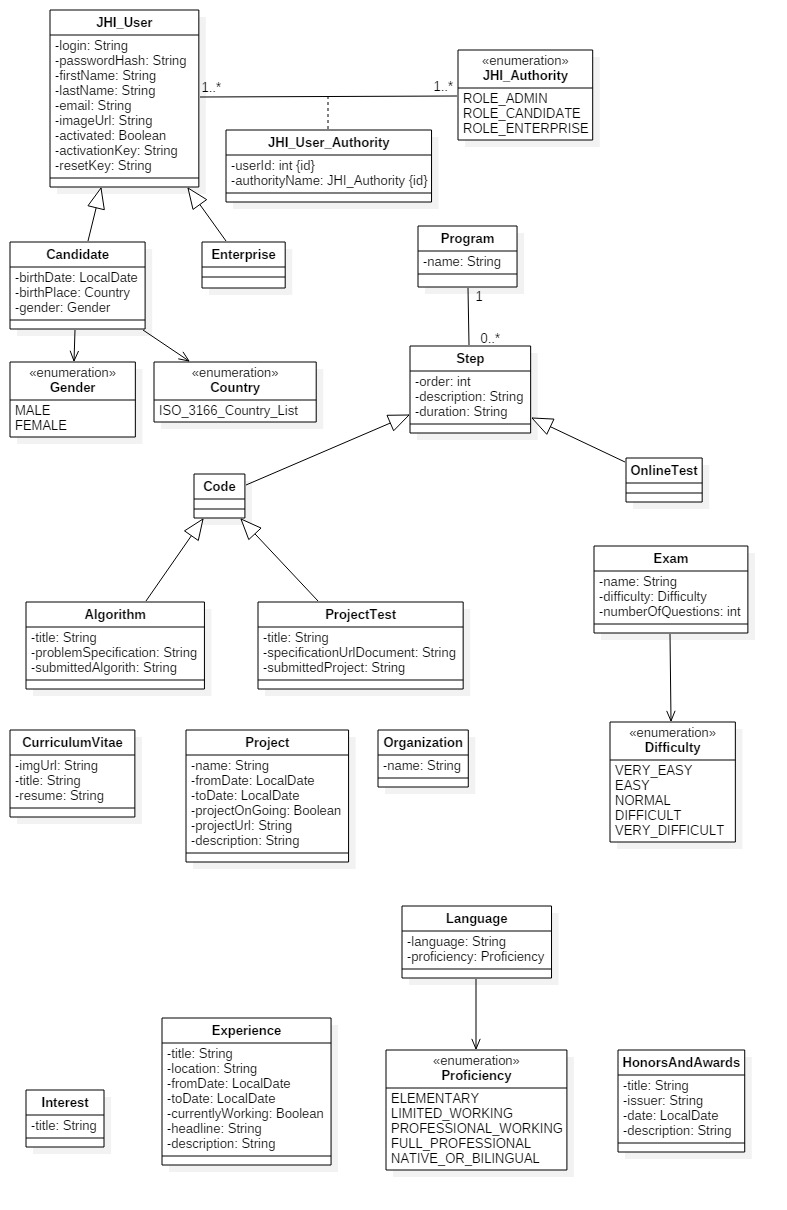
La figure 3.2 ci-dessus montre les différents cas d’utilisation possible d’une entreprise avec notre système comme la mise à jour de ses coordonnées ainsi que toutes les actions nécessaires pour le processus d’évaluation d’un candidat.

Figure 3.3 **-** Diagramme de cas d’utilisation d’un administrateur

L’administrateur peut gérer les utilisateurs (candidats ou entreprises) ainsi que la constitution des épreuves, il peut aussi consulter les performances du système en terme de mémoire utilisée. Toutes ces actions requirent une authentification comme il est indiqué dans la figure 3.3 ci-dessus.

* + 1. Diagramme de classe



BIBLIOGRAPHIE

[1] Méthode agile : <https://www.agiliste.fr/introduction-methodes-agiles/>

[2] Méthode agile SCRUM :<https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-FR.pdf>

[3] Le cadre méthodologique SCRUM : <https://www.agiliste.fr/introduction-methodes-agiles/>

[4] JIRA : <https://fr.atlassian.com/software/jira/agile>

[5] BITBUCKET : <https://fr.atlassian.com/software/bitbucket>

[6] SOURCETREE : <https://www.sourcetreeapp.com/>

[7] INTELLIJ IDEA : <https://doc.ubuntu-fr.org/utilisateurs/blackeco/brouillon-intellij>

[8] JAVA : <https://www.java.com/fr/download/faq/whatis_java.xml>

[9] JDL : <https://www.jhipster.tech/jdl/>

[10] JDL Studio : <https://start.jhipster.tech/jdl-studio/>

[11] Spring framework : <https://fr.wikipedia.org/wiki/Spring_(framework)>

[12] Angular : <https://openclassrooms.com/courses/developpez-avec-angular>

[13] JHipster : <https://www.jhipster.tech/>

[14] Frank Buschmann, Kevin Henney et Douglas C. Schmidt, “Pattern-oriented software architecture: On patterns and pattern languages”

[15] Data Transfer Object : https://martinfowler.com/eaaCatalog/dataTransferObject.html