Dédicaces

*À ma chère maman, vous m’avez soutenu et encouragé tout au long ces années d’études, à mon cher papa pour ses sacrifices et pour son soutien financier et moral.*

*C’est grâce à vous que je suis ce que je suis aujourd’hui. Mes parents, que Dieu vous protège et vous accorde santé et plein de bonheur dans la vie.*

*À Ma chère sœur Khaoula, Ces quelques lignes, ne sauraient traduire le profond amour que je te porte. Tu es ma vraie source d’énergie et de bonheur. C’est grâce à tes encouragements et tes riches conseils je puisse progresser dans ma vie et mes études. Tu*

*es une fierté pour moi.*

*À Mes chers amis : Koussai, Khawla , Yara et Ghada, vous êtes ma vraie source de persévérance. Vous avez toujours été présents pour me détresser. Que Dieu vous accorde*

*santé et plein de succés dans la vie.*

Remerciments

*Je tiens à remercier Monsieur Samer HSEN mon encadreur à Gpro-Consulting de m’avoir aidé à mener à Bien mon projet. Je remercie également Monsieur Ghazi*

*ATTROUSSI de m’avoir accueilli au sein de la Société.*

*Je tiens à témoigner toute ma gratitude à mon encadrante à l’Ecole Polytechnique Sousse, Madame Sonia AMAMOU, pour ses précieux conseils et le temps qu’elle a été*

*me consacrer tout au long ma Période de stage.*

*J’exprime ma gratitude, également, à tous les enseignants de l’Ecole Polytechnique Sousse qui ont Contribué pour achever ce stade.*

Table des matières

Page Contenu II

List des Figures VIII

List des Tables IX

Abbreviations IX

Introduction générale XI

1 Cadre général du projet . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

[Introduction 1](#_TOC_250081)

* 1. [Organisme d’accueil GPRO-Consulting 1](#_TOC_250080)
  2. [Etude de l’existant 2](#_TOC_250079)
     1. [Solution proposée 2](#_TOC_250078)
  3. [Méthodologie de travail et de modélisation 3](#_TOC_250077)
     1. [Choix de la méthode 3](#_TOC_250076)
     2. [Processus scrum 4](#_TOC_250075)
  4. [Langage de modélisation 6](#_TOC_250074)

1. Analyse et spécification des besoins . . . . . . . . . . . . . . . . . . . . 7

[Introduction 7](#_TOC_250073)

* 1. [Spécification des besoins 7](#_TOC_250072)
     1. [Identification des acteurs 7](#_TOC_250071)
     2. [Besoins fonctionnels 8](#_TOC_250070)
     3. [Besoins non fonctionnels 9](#_TOC_250069)
  2. [Analyse des besoins 9](#_TOC_250068)
     1. [Diagramme de cas d’utilisation global 9](#_TOC_250067)
     2. [Diagramme de classe global 10](#_TOC_250066)
     3. [BackLog de produit 11](#_TOC_250065)
     4. [Structure de d découpage du projet 14](#_TOC_250064)
  3. [Architecture globale de la solution 14](#_TOC_250063)
     1. [Architecture physique de l’application 14](#_TOC_250062)
     2. [Architecture applicative de la solution 15](#_TOC_250061)
  4. [Cadre de réalisation et choix techniques 16](#_TOC_250060)
     1. [Environnement matériel 16](#_TOC_250059)
     2. [Choix technique 16](#_TOC_250058)

1. Etude et réalisation du sprint 1 22

[Introduction 22](#_TOC_250057)

* 1. [Spécification fonctionnelle 22](#_TOC_250056)
     1. [Répartition des cas d’utilisations par acteur 22](#_TOC_250055)
     2. [Diagramme de cas d’utilisation du sprint 1 23](#_TOC_250054)
     3. [Description textuelle des cas d’utilisation 24](#_TOC_250053)
  2. [Conception 28](#_TOC_250052)
     1. [Diagramme de classe 28](#_TOC_250051)
     2. [Diagramme de séquence 29](#_TOC_250050)
  3. [Réalisation 33](#_TOC_250049)
     1. [Interface d’envoi d’une candidature 33](#_TOC_250048)
     2. [Interface de récupération de la liste des employés 34](#_TOC_250047)
     3. [Interface de recherche d’un employé 35](#_TOC_250046)
     4. [Interface détails employé 35](#_TOC_250045)
     5. [Interface pour générer un PDF 36](#_TOC_250044)
     6. [Génération d’une fiche d’employé 36](#_TOC_250043)
     7. [Interface de consultation de son profil 37](#_TOC_250042)
     8. [Interface de récupération de la liste des candidatures 37](#_TOC_250041)
     9. [Interface de recherche d’une candidature 38](#_TOC_250040)
     10. [Interface de voir détail candidat 38](#_TOC_250039)

1. Etude et réalisation du sprint 2 40

[Introduction 40](#_TOC_250038)

* 1. [Spécifications fonctionnelles 40](#_TOC_250037)
     1. [Répartition des cas d’utilisations par acteur 40](#_TOC_250036)
     2. [Diagramme de cas d’utilisation du sprint 2 41](#_TOC_250035)
     3. [Description textuelle des cas d’utilisation 41](#_TOC_250034)
  2. [Conception 42](#_TOC_250033)
     1. [Diagramme de classe 42](#_TOC_250032)
     2. [Diagramme de séquence 43](#_TOC_250031)
  3. [Réalisation 47](#_TOC_250030)
     1. [Interface d’authentification 47](#_TOC_250029)
     2. [Interface d’ajout d’un employé 48](#_TOC_250028)
     3. [Interface de modifier la fiche d’un employé 49](#_TOC_250027)
     4. [Interface de suppression d’un employé 50](#_TOC_250026)

1. Etude et réalisation du sprint 3 51

[Introduction 51](#_TOC_250025)

* 1. [Spécifications fonctionnelles 51](#_TOC_250024)
     1. [Répartition des cas d’utilisations par acteur 51](#_TOC_250023)
     2. [Diagramme de cas d’utilisation du sprint 3 52](#_TOC_250022)
     3. [Description textuelle des cas d’utilisation 52](#_TOC_250021)
  2. [Conception 54](#_TOC_250020)
     1. [Diagramme de classe 54](#_TOC_250019)
     2. [Diagramme de séquence 55](#_TOC_250018)
  3. [Réalisation 61](#_TOC_250017)
     1. [Interface de demande de congé 61](#_TOC_250016)
     2. [Interface déclaration d’un retard 61](#_TOC_250015)
     3. [Interface déclaration d’une absence 62](#_TOC_250014)
     4. [Interface de consulter assiduité 63](#_TOC_250013)

1. Etude et réalisation du sprint 4 65

[Introduction 65](#_TOC_250012)

* 1. [Spécifications fonctionnelles 65](#_TOC_250011)
     1. [Répartition des cas d’utilisations par acteur 65](#_TOC_250010)
     2. [Diagramme de cas d’utilisation du sprint 4 66](#_TOC_250009)
     3. [Description textuelle des cas d’utilisation 66](#_TOC_250008)
  2. [Conception 67](#_TOC_250007)
     1. [Diagramme de classe 67](#_TOC_250006)
     2. [Diagramme de séquence 68](#_TOC_250005)
  3. [Réalisation 70](#_TOC_250004)
     1. [Interface de Consulter pointage 70](#_TOC_250003)
     2. [Interface de Consulter dashboard 72](#_TOC_250002)

[Conclusion générale et perspective 73](#_TOC_250001)

[Webography 74](#_TOC_250000)

Table des figures

* 1. Gpro-Consulting 1
  2. Le principe Scrum 5
  3. UML logo 6
  4. Identification des acteurs 8
  5. Diagramme de cas d’utilisation global 10
  6. Diagramme de classe global 11
  7. Structure découpage projet 14
  8. Architecture physique de l’application 14
  9. Architecture MVC 15
  10. Logo PostgreSQL 16
  11. Logo IntelliJ 17
  12. Logo WebStorm 17
  13. Logo LaTeX 17
  14. Logo Java 18
  15. Logo HTML 18
  16. Logo CSS 19
  17. Logo Angular 19
  18. Logo Bootstrap 20
  19. Logo Ant Design 20
  20. Logo Spring Boot 21
  21. Répatition des cas d’utilisation par acteur 23
  22. diagramme de cas d’utilisation Sprint 1 24
  23. Raffinement du cas d’utilisation « Chercher Employé » 25
  24. Raffinement du cas d’utilisation « Voir détail employé » 26
  25. Raffinement du cas d’utilisation « Consulter Liste Candidats » 27
  26. Raffinement du cas d’utilisation « Consulter son profil » 28
  27. Diagramme de classe du Sprint 29
  28. Diagramme de séquence « Consulter liste employés » 30
  29. Diagramme de séquence « Consulter liste candidats » 31
  30. Diagramme de séquence « Envoyer Candidature » 32
  31. Diagramme de séquence « Consulter son profil » 33
  32. Interface d’envoi d’une candidature 34
  33. Interface de récupération de la liste des employés 34
  34. Interface de recherche d’un employé par un seul critère de recherche 35
  35. Interface de recherche d’un employé par un plusieurs critères de recherche . 35 3.16 Interface détails employé 36
  36. Interface pour générer un PDF 36
  37. Génération d’une fiche d’employé 37
  38. Interface de consultation de son profil 37
  39. Interface de récupération de la liste des candidatures 38
  40. Interface de recherche d’une candidature par un seul critère de recherche . 38
  41. Interface de recherche d’une candidature par plusieurs critères de recherche 39
  42. Interface de voir détails d’un candidat 39
  43. Répartition des cas d’utilisation par acteur 40
  44. Diagramme de cas d’utilisation Sprint 2 41
  45. Diagramme de classe du Sprint 2 43
  46. Diagramme de séquence «S’authentifier» 44
  47. Diagramme de séquence « Ajouter employé » 45
  48. Diagramme de séquence « Modifier employé » 46
  49. Diagramme de séquence «Supprimer employé » 47
  50. Interface d’authentification 48
  51. Interface d’ajout d’un employé (1) 48
  52. Interface d’ajout d’un employé (2) 49
  53. Interface de modifier la fiche d’un employé 49
  54. Interface de suppression d’un employé 50
  55. Répartition des cas d’utilisation par acteur 51
  56. Diagramme de cas d’utilisation Sprint 3 52
  57. Diagramme de classe Sprint 3 55
  58. Diagramme de séquence «Demander un congé» 56
  59. Diagramme de séquence «Déclarer un retard» 57
  60. Diagramme de séquence «Déclarer une absence» 58
  61. Diagramme de séquence du chatbot 59
  62. Diagramme de séquence «Consulter assiduité» 60
  63. Interface de demande de congé 61
  64. Interface déclaration d’un retard 62
  65. Interface déclaration d’une absence 62
  66. Interface de consulter la liste des absences 63
  67. Interface de consulter la liste des retards 63
  68. Interface de consulter assiduités avec un champs de recherche 64
  69. Fonctionnalités par acteur du sprint 4 65
  70. diagramme de cas d’utilisation Sprint 4 66
  71. Diagramme de classe Sprint 4 68
  72. Diagramme de séquence « Gérer pointage » 69
  73. Diagramme de séquence « Consulter Dashboard » 70
  74. Interface de Consulter pointage 71
  75. Interface de Modifer pointage 71
  76. Interface de Modifer pointage 72

1.1 Tableau comparatif des méthodologies de travail 4

2.1 backlog de notre application 12

**MVC** Model-View-Controller

**RH** Ressources Humaines

**UML** Unified Modeling Language

À l’époquen actuelle, l’évolution technologique joue un rôle très important dans tous les domaines d’activités et les entreprises, particulièrement au cœur des secteurs liés à la gestion des ressources humaines. Conséquemment, l’objectif actuel des créateurs et des éditeurs de logiciels est de fournir à l’utilisateur un produit de qualité qui garantit une rapidité et une modernité attendues : facilité de compréhension et d’accessibilité de l’ensemble des interfaces, réponse rapide en temps réel ou dans un temps raisonnable pour éviter que le client abandonne le système à cause du temps d’attente. En tient compte des utilisateurs qui sollicitent un minimum de documentations pour accoutumer facilement et rapidement les logiciels.

En effet, gérer tous les salariés est une mission dure et complexe pour une industrie ou une entreprise quelle que soit sa taille : **Grande entreprise, entreprise de taille intermédiaire ou bien petite et moyenne entreprise**.

Donc, la solution idéale de nos jours c’est de profiter du progrès technologique par l’adoption d’un logiciel de gestion des ressources humaines qui va faciliter les taches pour les administrateurs et les responsables des ressources humaines par la sauvegarde des dossiers électroniques des employés, centralisation des informations et un accès rapide et facile pour suivre les données des collaborateurs.

Pour simplifier les taches aux administrateurs des ressources humaines nous proposons de développer un logiciel de gestion des salariés qui peut être aussi consultés en toute sécurité par les employés. C’est dans ce cadre que se situe notre projet de fin d’étude : "Conception et développement d’une application de gestion des ressources humaines".

Le présent rapport décrit les différentes étapes de la réalisation de ce projet s’étalant sur cinq chapitres. Commençant par le premier chapitre "Etude du projet" dans lequel nous présenterons le cadre général du projet, ensuite nous exposerons l’organisme d’ac- cueil. Le deuxième chapitre "Analyse et spécification des besoins" définira la méthodologie du travail choisie qui est la méthode SCRUM. Nous détaillerons les besoins fonctionnels clarifiés par des diagrammes de cas d’utilisation global et les exigences techniques de ce

projet. Les trois derniers chapitres seront consacrés à la réalisation des sprints.

Nous clôturons notre rapport par une conclusion générale et quelques perspectives éventuelles.

CHAPITRE 1

Cadre général du projet

## Introduction

Ce chapitre consiste à mettre notre projet dans son contexte général. En premier lieu, nous présenterons l’organisme d’accueil. En second lieu, nous allons faire une comparaison des solutions existantes. Par la suite, nous détaillerons le cadre du projet, les technologies utilisées et la méthodologie adoptée pour accomplir notre projet.

## Organisme d’accueil GPRO-Consulting

GPRO Consulting est une société multi-facettes créée en 2015 qui conçoit et fournit des solutions totales d’entreprise orientées métier, flexibles et développées avec les technologies les plus modernes : java, springboot, angular ... dans les domaines diversifiés.



Figure 1.1 – Gpro-Consulting

## Etude de l’existant

Dans cette partie, nous allons étudier les systèmes des gestions des ressources humaines ordinaires pour mettre l’accent sur leurs insuffisances en premier lieu et de proposer des solutions possibles en second lieu afin d’améliorer leurs actions.

La gestion des ressources humaines ou gestion du capital humain est l’ensemble des tâches qui permettent de gérer des salariés au sein d’une entreprise.

Dans un premier temps, les entreprises optent à stocker les données de ses employés dans des dossiers appelés « Fiches Employé », c’est-à-dire toutes les données sont sur des papiers et gérées par des individus. De même, les collaborateurs sont concernés aussi de fournir des papiers pour leurs sociétés : demande congé, déclaration pour une absence ou un retard . . .

Par conséquent, le travail de l’équipe Ressources Humaines (RH) devient plus en plus pénible si le nombre des salariés est important, car chaque dossier englobe des dizaines des papiers. Pour cela, l’entreprise est obligée d’agrandir l’équipe RH pour mieux contrôler la situation et ça engendre une augmentation de ses dépenses.

Dans un second temps, l’apparition du la digitalisation résoudre beaucoup des pro- blèmes des gestion des employés comme l’utilisation des fichiers Excel, mais ce n’était pas la solution idéale car les données textuelles sont difficilement traitées dans un ficher Ex- cel. Aussi, ce dernier est accessible par différents agents de la société, il s’agit donc d’un problème de confidentialité. De plus, le responsable RH n’a pas une vision globale car chaque département a ses propres fichiers. Les candidats aussi, sont obligés de se déplacer pour déposer leurs candidatures.

### Solution proposée

En réponse aux problèmes évoquées auparavant, notre projet présente une solution permettant de :

* + - * Centraliser la gestion des employés
      * Gérer les assiduités
      * Automatiser le processus d’envoi des candidatures
      * Avoir des données centralisées et dématérialisées
      * Récupérer les données facilement de la base des données en cas d’urgence : sécurité des données.

## Méthodologie de travail et de modélisation

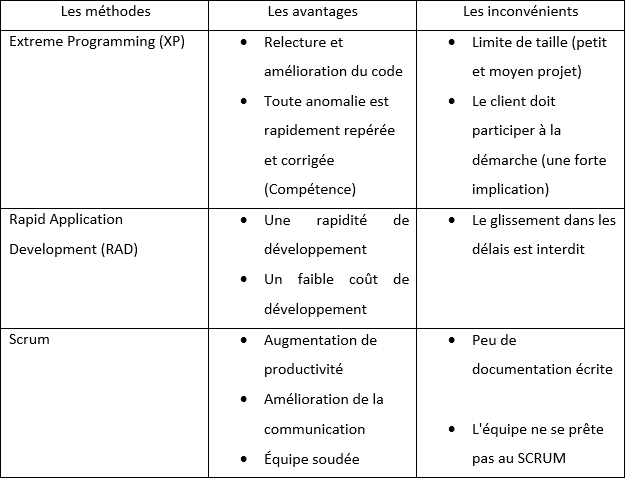
### Choix de la méthode

La réalisation d’un projet nécessite de suivre des méthodologies de gestion des projets, certainement le choix de la méthode compte sur deux principaux critères : la nature et la taille du projet. Ce choix fait une partie essentielle dans la réussite ou l’échec d’un travail. Lorsqu’il s’agit des besoins du clients et d’utilisateurs finaux qui sont incomplets ou bien qui changent et évoluent au cours de projet, il faut s’orienter donc vers une des méthodes itératives, parmi ces derniers on peut distinguer les méthodes "Agile" qui transforme les phases de déroulement des projets (conception, développement, tests, publication) en des petits "sprint" faciles à gérer.

En outre, la méthode Agile garantit la génération des projets de hautes qualités car elle se base sur un principe simple : “Gagner plus de contrôle sur le produit final", par sa capacité à l’adaptation aux changements des besoins du clients sans causer la perte du produit.

Aussi la méthode Agile divise le projet en plusieurs sous-projets, ainsi assumer des actions correctives sans être piégé dans la perte de temps et des coûts, car il est difficile de tout prévoir et de tout anticiper dès le début. Donc il est recommandé de fixer des objectifs à court terme.

Table 1.1 – Tableau comparatif des méthodologies de travail



Après avoir analysé les différentes méthodes, nous allons adopter la méthode Scrum puisqu’elle convient avec le degré de complexité du notre projet et sa taille.

Par la suite, nous présenterons l’équipe Scrum ainsi que le cycle de vie de cette méthodo- logie.

### Processus scrum

C’est le représentant du client dans un projet scrum. Il joue le rôle d’une passerelle entre l’équipe technique et la partie prenante. Il a comme mission de créer le « backlog » du produit. Aussi, il doit avoir des connaissances techniques car il est chargé de définir et d’ordonner les « users stories » de chaque sprint.

**Le scrum master** : Sa responsabilité est de mener les développements produit vers le

succès et de gérer les empêchements qui menacent l’avancement de l’équipe pendant les sprints.

**Scrum team** : Est guidée par le Scrum Master, ceux sont les membres chargées de la

réalisation du sprint (développeurs, architectes, testeurs. . . ).

#### Nous enchaînons par le principe de scrum qui est le suivant :

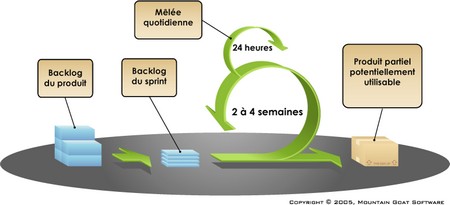


Figure 1.2 – Le principe Scrum

* + - * Le product owner définit les besoins fonctionnels sous forme de "User Story" appelé :

« Product Backlog ».

* + - * User Story : est une description en langage naturel d’un besoin exprimé par l’utilisa- teur.
      * Pour le sprint planning, l’équipe décomposent les éléments du product backlog en éléments plus petit d’une journée ou moins, Ce plannnig est limité de 8 heures pour un sprint d’un mois. Il faut que les objectifs à atteindre soient clairement exposés.
      * L’équipe va travailler sur un sprint entre 2 à 4 semaine et une fois par jour, il faut faire un « daily scrum » pendant cette période. L’objectif de cette réunion qui dure 15 min est de partager l’état d’avancement du sprint et de mentionner les obstacles rencontrés.
      * À la fin de chaque sprint, il faut avoir un livrable. Aussi, une réunion de revue de sprint sera organisée pour évaluer les résultats du sprint et ajuster le plan de la release c’est-à-dire, mentionner des éléments qu’ont pu être ajoutés ou supprimés.

## Langage de modélisation

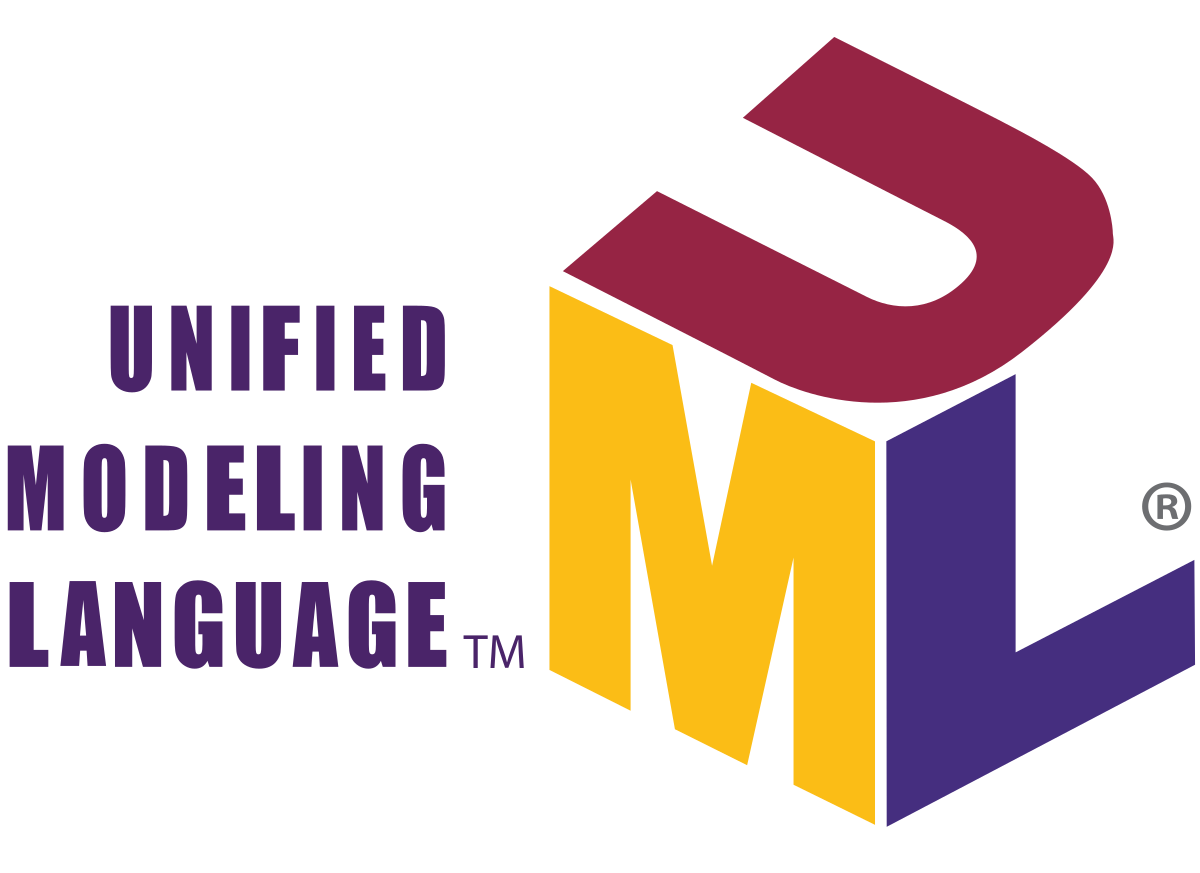
Nous choisissons le langage Unified Modeling Language (UML) pour la modélisation et la planification de notre projet. Il permet aux développeurs de réaliser des différents diagrammes a fin de représenter graphiquement les objets et les besoins, concevoir et décrire un système d’une manière compréhensible pour améliorer la communication avec les parties prenantes.

Figure 1.3 – UML logo

**Conclusion**

Dans ce chapitre, nous avons présenté le contexte général du projet, l’organisme d’ac- cueil **GPRO-Consulting** et le choix de la méthodologie de développement. De plus, nous avons traité l’étude et le critique de l’existant.

Dans le chapitre suivant, nous présenterons l’architecture générale du projet et la spéci- fication des besoins.

CHAPITRE 2

Analyse et spécification des besoins

## Introduction

Nous développons ce chapitre pour présenter en premier lieu, l’analyse et la spécifi- cation des besoins, l’identifications des acteurs ainsi que les besoins fonctionnels et non fonctionnels qui touchent les objectifs généraux et les besoins supplémentaires dont nous ajouterons pour achever notre solution. En second lieu, nous présenterons le diagramme de cas d’utilisation globale et le Backlog de produit.

## Spécification des besoins

### Identification des acteurs

La première étape dans l’analyse et la spécification des besoins c’est l’indentification des acteurs. Un acteur est une personne en position d’utiliser le système pour des besoins définis.

Dans notre application, nous pouvons identifier quatre acteurs :

* + - * **Candidat :** qui peut envoyer sa candidature via un formulaire.
      * **Responsable RH :** il consulte la liste des employés et il peut les gérer (Ajout, Suppression, Modification), aussi il gère les candidatures par l’acceptation ou le refus.
      * **Administrateur :** il a les fonctionnalités principales de gérer les employés.
      * **Employé :** chaque employé peut accéder à sa zone réserver pour : consulter son profil, demander un congé, déclarer une absence ou un retard.



Figure 2.1 – Identification des acteurs

### Besoins fonctionnels

Les besoins fonctionnels représentent les fonctionnalités offertes par l’application aux acteurs. De ce fait, nous montrons les fonctionnalités attendues de chaque acteur.

L’application doit assurer aux candidats les fonctionnalités suivantes :

* + - * Accéder à l’application sans authentification
      * Envoyer une candidature (cv + ses données)

L’application doit assurer au responsable RH les fonctionnalités suivantes :

* + - * S’authentifier d’une façon sécurisée.
      * Consulter son profil : voir et mettre à jour ses données
      * Consulter la liste des employés
      * Gérer les employés : Ajouter, modifier, supprimer un employé
      * Gérer l’état d’un employé : absence, congé, retard
      * Consulter les publications des autres utilisateurs : voir les autres publications.
      * Consulter la liste des candidatures : accepter ou refuser

L’application doit assurer à l’administrateur les fonctionnalités suivantes :

* + - * S’authentifier d’une façon sécurisée.
      * Consulter son profil : voir ses données
      * Consulter la liste des employés
      * Gérer les employés : Ajouter, modifier, supprimer un employé

L’application doit assurer à l’employé les fonctionnalités suivantes :

* + - * S’authentifier d’une façon sécurisée.
      * Consulter son profil : voir ses données
      * Demander un congé
      * Déclarer un retard
      * Déclarer une absence
      * Envoyer un justificatif (absence / retard)

### Besoins non fonctionnels

Les besoins non fonctionnels décrivent les exigences internes auxquelles est attaché le système pour sa réalisation et son bon fonctionnement.

* + - * **Ergonomie :** la facilité de compréhension l’ensemble des pages du site doit être mo-

derne et accessible facilement.

* + - * **Sécurité :** le système doit être sécurisé pour écarter les tentatives d’utilisation incon- nue tout en garantir une sécurité au niveau de la gestion des accès aux données.
      * **Efficacité et rapidité :** réponse rapide en temps réel ou dans temps raisonnable pour éviter que les utilisateurs abandonnent le système à cause du temps d’attente.

## Analyse des besoins

### Diagramme de cas d’utilisation global

Un diagramme de cas d’utilisation global présente un aperçu général et total du projet de son fonctionnement.

Un diagramme de cas d’utilisation capture le comportement d’un système, d’un sous- système, d’une classe ou d’un composant tel qu’un utilisateur extérieur le voit. Il scinde la fonctionnalité du système en unités cohérentes, les cas d’utilisation, ayant un sens pour les acteurs. Les cas d’utilisation permettent d’exprimer le besoin des utilisateurs d’un système, ils sont donc une vision orientée utilisateur de ce besoin au contraire d’une

vision informatique.[4]

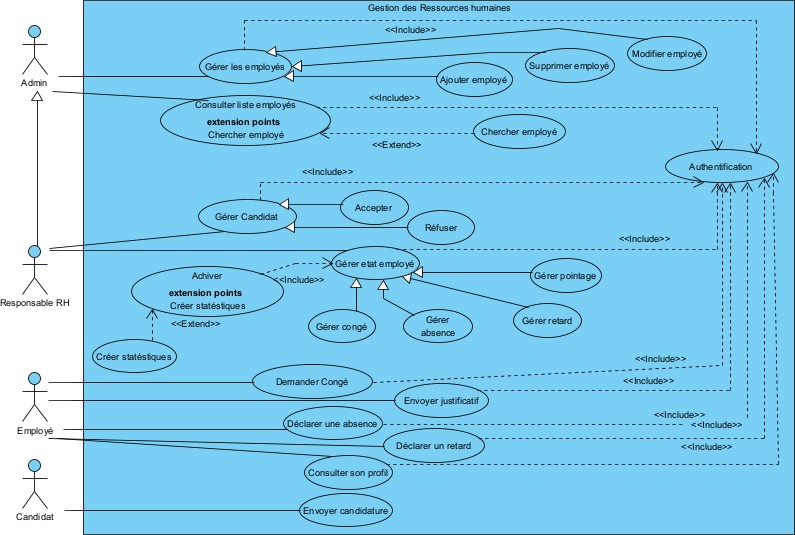


Figure 2.2 – Diagramme de cas d’utilisation global

### Diagramme de classe global

Le diagramme de classes décrit la structure du système en modélisant ses objets et les relations entre eux, ses attributs et les interactions entre ses classes.

Ci-dessous nous présentons le diagramme de classe globale de notre application.

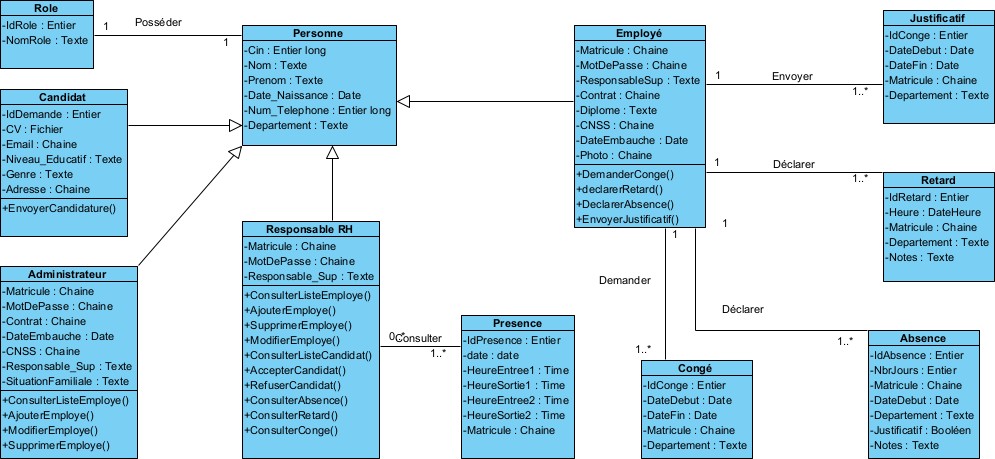


Figure 2.3 – Diagramme de classe global

### BackLog de produit

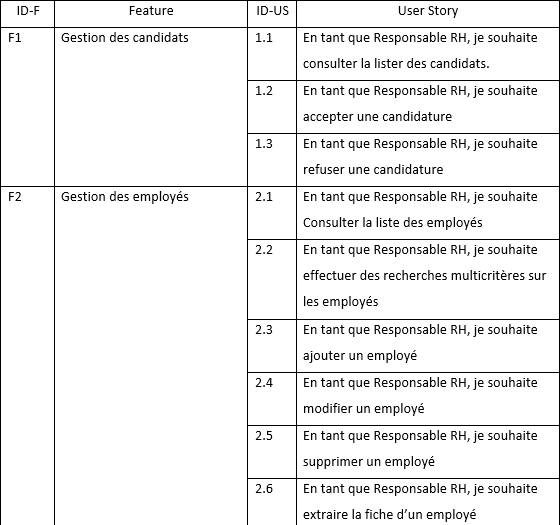
Le backlog de produit représente la liste des fonctionnalités d’un produit, il permet d’englober tous les besoins du client que l’équipe doit réaliser, ainsi que tous les éléments nécessitant la participation de l’équipe projet. Le product owner se charge de classer les éléments par priorité en définissant l’ordre de la réalisation.

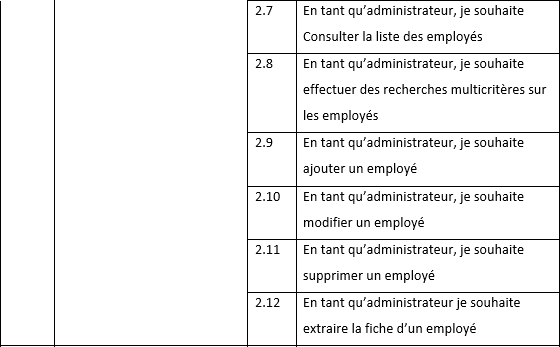
Backlog du produit contient les éléments suivants :

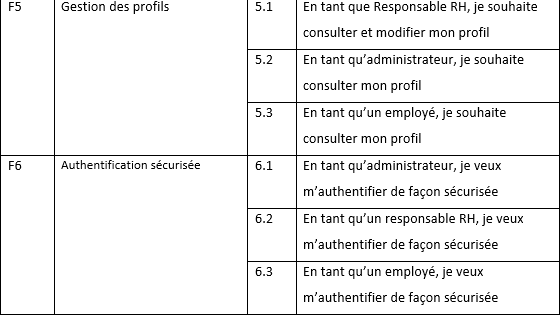
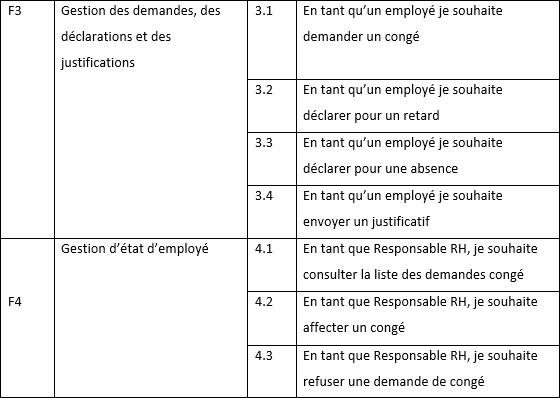
* + - * **Features (fonctionnalités) :** est une partie du projet décomposée en plusieurs user story.
      * **User story (Histoire de l’utilisateur) :** il s’agit d’une demande faisable écrite en langage naturel, doit être compris par tous les acteurs du projet et priorisé selon leur valeur pour déterminer l’ordre de la réalisation.

Table 2.1 – backlog de notre application Le tableau ci-dessous représente la backlog de notre application :

* + - * **ID-F :** un identifiant de feature
      * **ID-US :** identifiant de chaque user story







### Structure de d découpage du projet

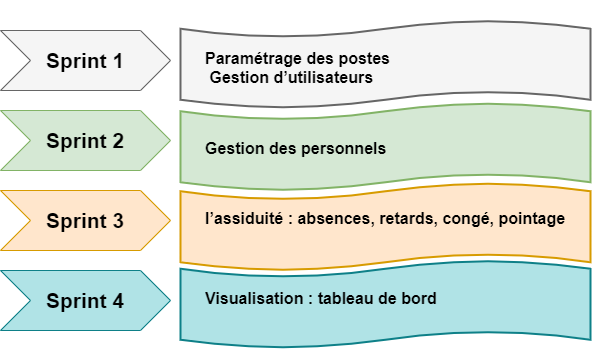
Pour notre projet nous avons décidé de développer quatre sprints comme montré ci- dessus :

Figure 2.4 – Structure découpage projet

## Architecture globale de la solution

### Architecture physique de l’application

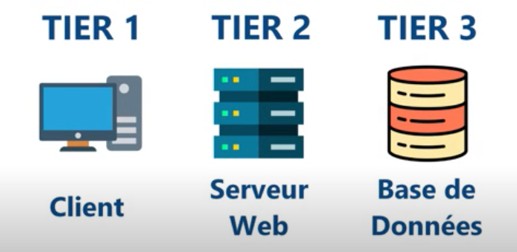
Au niveau de notre projet nous avons choisi une architecture basée sur trois couches : couche présentation, couche de traitement et couche d’accès aux données. (Démontrée dans la figure ci-dessous)

Figure 2.5 – Architecture physique de l’application

* + - * **Un client :** il s’agit d’un navigateur web permettant l’utilisateur de dialoguer avec l’application via internet.
      * **Un serveur d’application :** héberge les couches de notre application.
      * **Un serveur de base de données :** correspondant aux données qui sont conservées sur la durée

### Architecture applicative de la solution

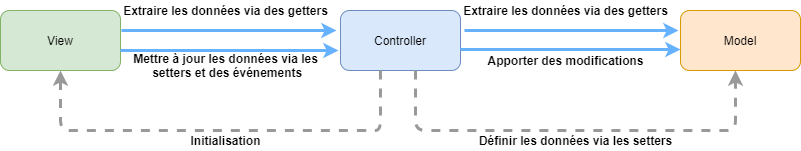
L’architecture logicielle présente la division logique d’une application et la façon de rassembler les composants par le type de fonction et le traitement effectué. Pour notre application nous avons adopté l’architecture Model-View-Controller (MVC) :

Figure 2.6 – Architecture MVC

* + - * **Modèle :** cette partie gère les données de votre site. Son rôle est d’aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu’elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL.
      * **Vue :** cette partie se concentre sur l’affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu’elle doit afficher.
      * **Contrôleur :** cette partie gère la logique du code qui prend des décisions. C’est en quelque sorte l’intermédiaire entre le modèle et la vue : le contrôleur va demander au

modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. le contrôleur est le chef d’orchestre : c’est lui qui reçoit la requête du visiteur et qui contacte d’autres fichiers (le modèle et la vue) pour échanger des informations avec eux.[9]

## Cadre de réalisation et choix techniques

### Environnement matériel

* + - * Ordinateur : ASUS X556UV
      * Processeur : Intel Core i7-6500U
      * Mémoire physique : 8 GO
      * Carte graphique : Nvidia GeForce GT 920M Système d’exploitation : windows 10

### Choix technique

#### Outils de développement :

**PostgreSQL [10] :** est un système de gestion de base de données relationnelle et objet (SGBDRO). C’est un outil libre disponible selon les termes d’une licence de type BSD. Ce système est concurrent d’autres systèmes de gestion de base de données, qu’ils soient libres (comme MariaDB et Firebird), ou propriétaires (comme Oracle, MySQL, Sybase, DB2, Informix et Microsoft SQL Server). Comme les projets libres Apache et Linux, Post- greSQL n’est pas contrôlé par une seule entreprise, mais est fondé sur une communauté mondiale de développeurs et d’entreprises.

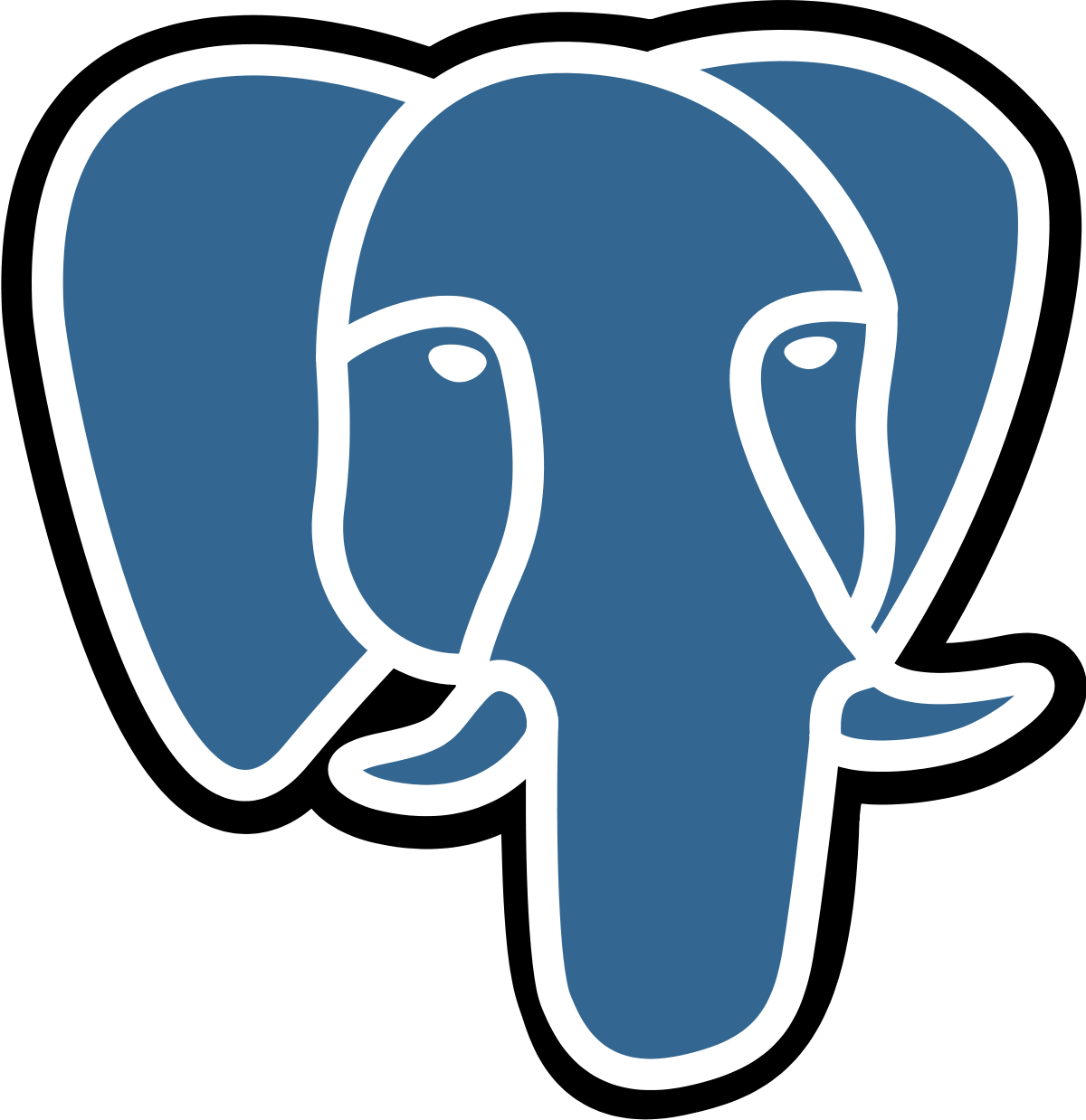


Figure 2.7 – Logo PostgreSQL

#### Editeurs :

**IntelliJ IDEA 2020 [6] :**également appelé « IntelliJ », « IDEA » ou « IDJ » est un environnement de développement intégré (en anglais Integrated Development Environ- ment - IDE) de technologie Java destiné au développement de logiciels informatiques. Il est développé par JetBrains (anciennement « IntelliJ ») et disponible en deux versions, l’une communautaire, open source, sous licence Apache 2 et l’autre propriétaire, protégée par une licence commerciale. Tous deux supportent les langages de programmation Java,

Kotlin, Groovy et Scala.

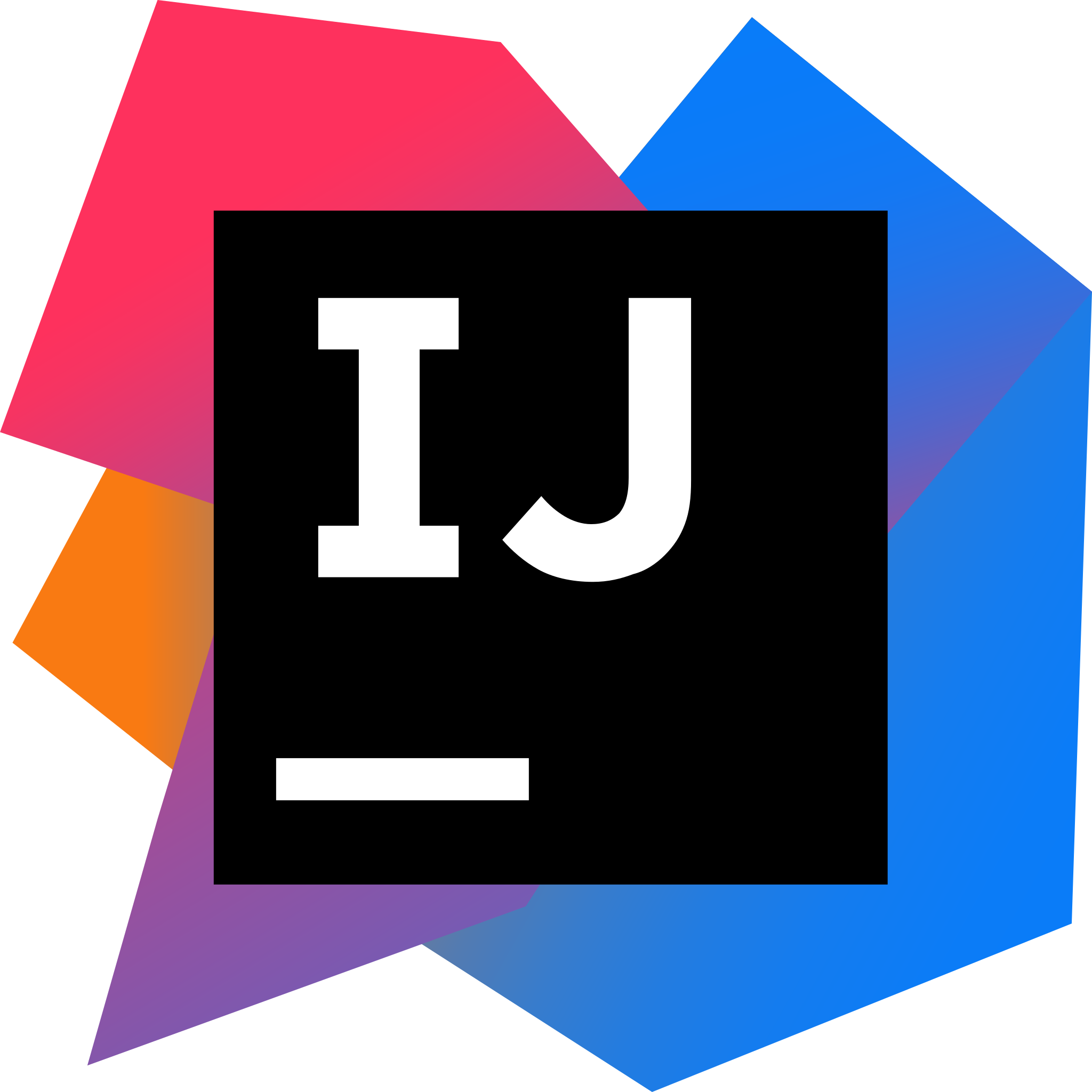


Figure 2.8 – Logo IntelliJ

**WebStorm 2020 [13] :** WebStorm est un IDE pour les langages Web (HTML, CSS et JavaScript), développé par l’entreprise JetBrains et basé sur la plateforme IntelliJ IDEA.



Figure 2.9 – Logo WebStorm

**LaTeX :** est un langage permet de rédiger des documents dont la mise en page est réalisée automatiquement. **Langages de développement :**

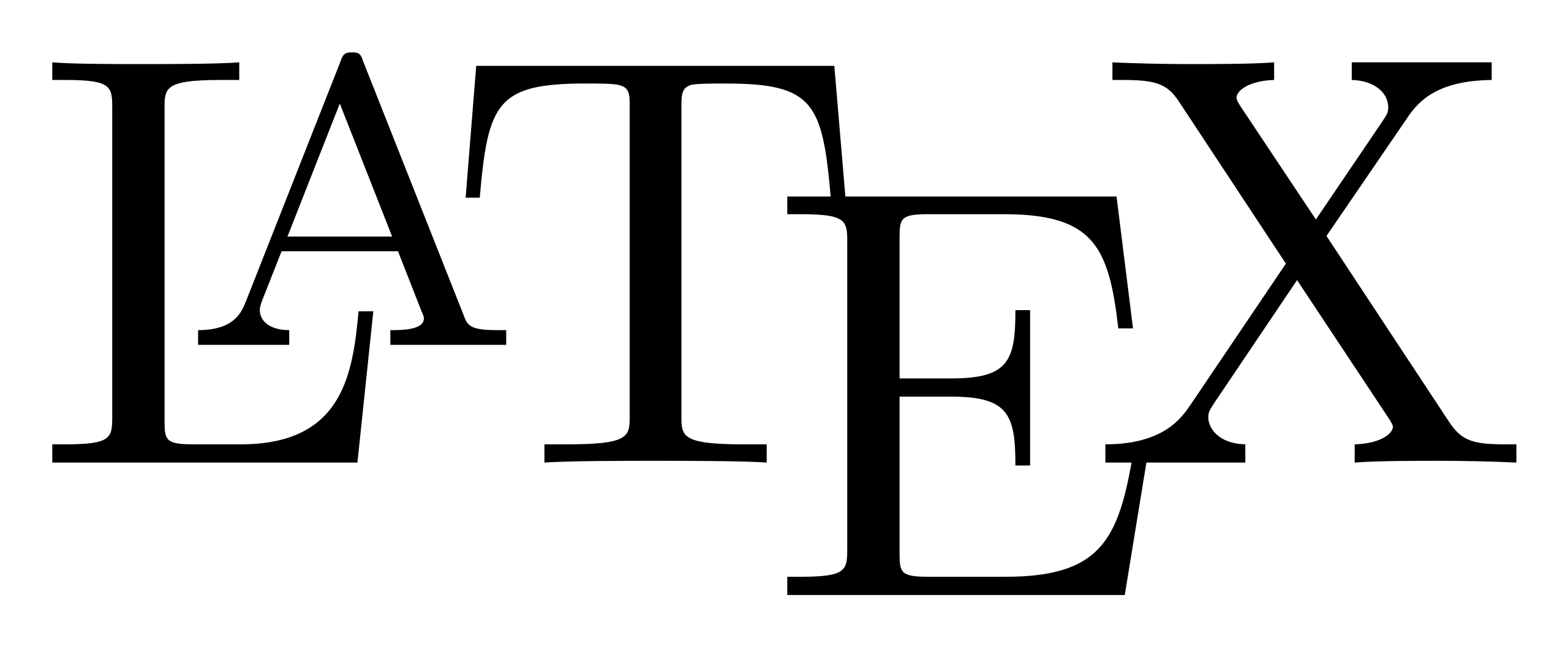


Figure 2.10 – Logo LaTeX

**JAVA 8 :**Java est un langage de programmation orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld.[7] Java 8 est la dernière version de Java et offre de nouvelles fonctionnalités, des perfor- mances accrues et des corrections de bug pour améliorer l’efficacité de développement et

d’exécution des programmes Java. La nouvelle version de Java est d’abord mise à disposi- tion des développeurs afin qu’ils disposent du temps adéquat pour effectuer les opérations de test et de certification ; les utilisateurs finals pourront ensuite la télécharger sur le site Web java.com. [8]

Figure 2.11 – Logo Java

**HTML [5] :**Le HyperText Markup Language, généralement abrégé HTML ou dans sa dernière version HTML5, est le langage de balisage conçu pour représenter les pages web.



Figure 2.12 – Logo HTML

**CSS [3] :**Les feuilles de style en cascade1, généralement appelées CSS de l’anglais Cascading Style Sheets, forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium (W3C).

Introduit au milieu des années 1990, CSS devient couramment utilisé dans la conception de sites web et bien pris en charge par les navigateurs web dans les années 2000.



Figure 2.13 – Logo CSS

#### Choix des frameworks de front-end :

**Angular 11 [1] :**Angular (communément appelé "Angular 2+" ou "Angular v2 et plus")2,3 est un cadriciel (framework) côté client, open source,basé sur TypeScript, et co-dirigé par l’équipe du projet « Angular » à Google et par une communauté de particu- liers et de sociétés. Angular est une réécriture complète de AngularJS, cadriciel construit par la même équipe. Il permet la création d’applications Web et plus particulièrement de ce qu’on appelle des « Single Page Applications » : des applications web accessibles via une page web unique qui permet de fluidifier l’expérience utilisateur et d’éviter les chargements de pages à chaque nouvelle action. Le Framework est basé sur une architec- ture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités. Un type d’architecture qui a largement fait ses preuves et qui permet une forte maintenabilité et une amélioration du travail collaboratif.



Figure 2.14 – Logo Angular

**Bootstrap 4 [2] :** Bootstrap est une collection d’outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d’applications web. C’est un ensemble qui contient des codes HTML et CSS, des formu- laires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions

JavaScript en option. C’est l’un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.

Figure 2.15 – Logo Bootstrap

**Ant Design :** est un langage de conception d’interface utilisateur, il fournit un en- semble de composants de haute qualité.

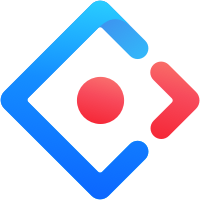


Figure 2.16 – Logo Ant Design

**Choix des frameworks de back-end :** Les architectures micro-services permettent de développer des petits morceaux d’un projet. Chaque micro-service est indépendant des autres. D’où, cette architecture garantit une maintenance facile, une grande tolérance aux pannes, aussi, les développeurs d’un même projet peuvent utiliser des technologies diffé- rentes au niveau de chaque micro-service. Avec Spring boot la création des micro-services se fait rapidement grâce au Spring Initializr.

**Spring-Boot :** Spring est un framework open source pour construire et définir l’infra- structure d’une application Java3, dont il facilite le développement et les tests. [11] Publiée en 2012, Spring Boot est une solution de « convention plutôt que configuration

» destinée à l’infrastructure logicielle Java Spring qui réduit la complexité de la configu- ration de nouveaux projets Spring. À cette fin, Spring Boot définit une configuration de base incluant des directives pour l’utilisation de l’infrastructure logicielle ainsi que toutes

les bibliothèques de prestataires tiers pertinentes, ce qui permet de faciliter autant que possible la création de nouveaux projets. Cette méthode simplifie considérablement la création d’applications indépendantes prêtes pour la production. [12]

Figure 2.17 – Logo Spring Boot

**Conclusion**

Dans ce chapitre, nous avons présenté les besoins fonctionnels et non fonctionnels de notre projet. Ensuite, nous avons défini les acteurs et les différents cas d’utilisations.

Dans le chapitre suivant, nous allons mettre l’accent sur le premier Sprint de notre appli- cation.

CHAPITRE 3

Etude et réalisation du sprint 1

## Introduction

Après avoir analysé les besoins de notre application et présenté nos choix techno- logiques, nous passons à la réalisation de notre projet. Notre premier chapitre permet d’identifier les différents postes et les fonctionnalités de chaque acteur.

Nous présenterons donc, la spécification fonctionnelle, la conception ainsi que la réalisation de cette partie.

## Spécification fonctionnelle

Pour chaque Sprint en va présenter la spécification fonctionnelle à travers un dia- gramme de cas d’utilisation. Car ce dernier permet de donner un aspect global du système et de définir les liens entre les acteurs et leurs différentes fonctionnalités.

### Répartition des cas d’utilisations par acteur

Le tableau ci-dessous présente toutes les fonctionnalités de chaque acteur du sprint 1.

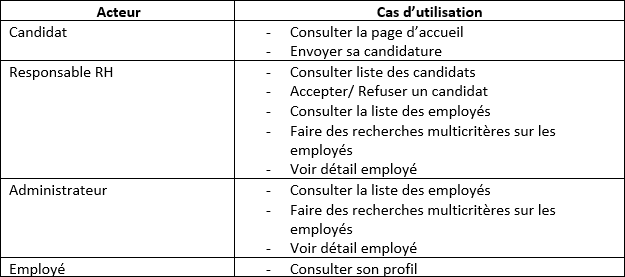


Figure 3.1 – Répatition des cas d’utilisation par acteur

### Diagramme de cas d’utilisation du sprint 1

Ce diagramme nous montre une vue globale du sprint 1. L’administrateur hérite les fonctionnalités « Consulter liste employés », « Chercher un employé » et « Voir détail employé » de responsable RH.

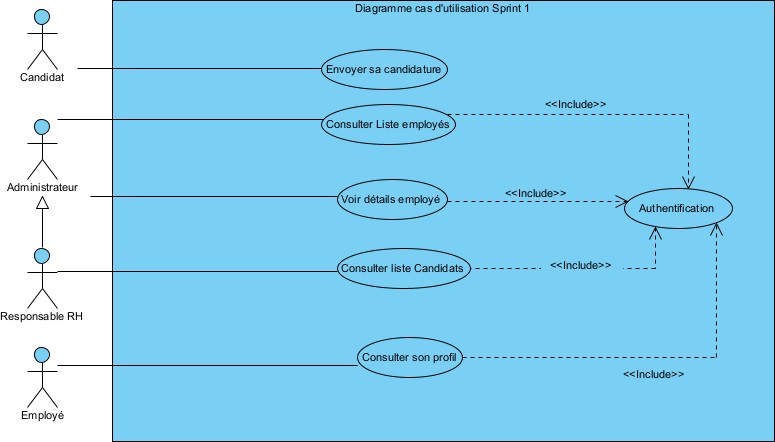


Figure 3.2 – diagramme de cas d’utilisation Sprint 1

### Description textuelle des cas d’utilisation

* + - * Description textuelle du cas d’utilisation**« Envoyer sa Candidature » : Acteurs :** Candidat

**Précondition :** Accéder au formulaire **Postcondition :** Candidature envoyé **Scénario principal :**

—Le candidat se trouve sur la page d’acceuil

—Le candidat cliquer sur le bouton s’inscrire

—Le candidat remplir le formulaire de la candidature

—Le candidat va caliquer sur le bouton envoyer

* + - * Description textuelle du cas d’utilisation **« Consulter Liste Employés » : Acteurs :** Responsable RH, Administrateur

**Précondition :** s’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—L’acteur va sélectionner le bouton "liste employés" du menu sidebar

—Liste employés va etre affiché.

Raffinement du cas d’utilisation « Chercher Employé »

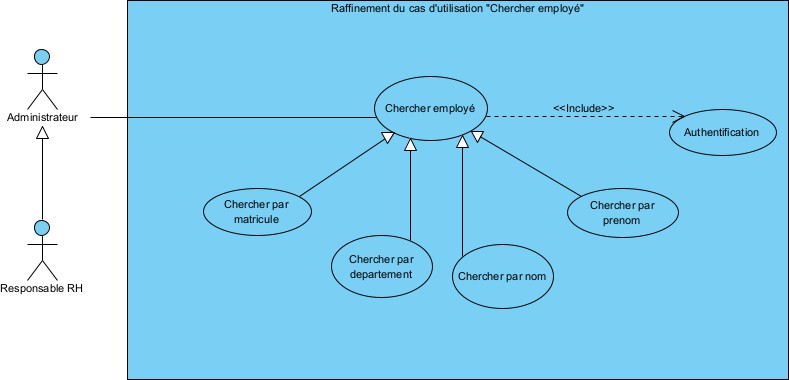


Figure 3.3 – Raffinement du cas d’utilisation « Chercher Employé »

* + - * Description textuelle du cas d’utilisation **« Chercher Employé » : Acteurs :** Responsable RH, Administrateur

**Précondition :** s’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—Dans l’interface "liste d’employés", il ya zone de recherche multicritères

—l’acteur peut effectuer des recherches sur la liste des employés. Raffinement du cas d’utilisation «Voir détail employé »

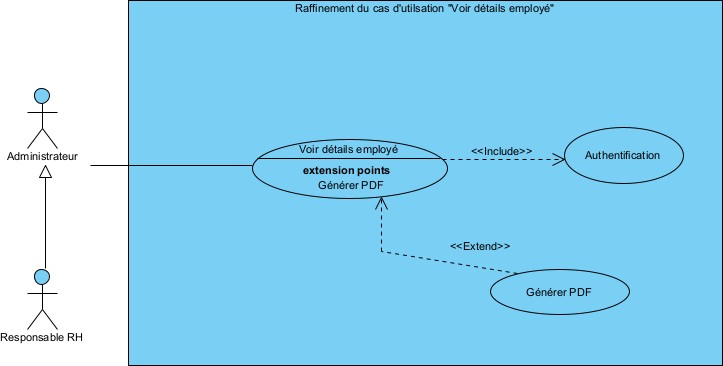


Figure 3.4 – Raffinement du cas d’utilisation « Voir détail employé »

* + - * Description textuelle du cas d’utilisation **«Voir détail employé» : Acteurs :** Responsable RH, Administrateur

**Précondition :** s’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—La liste des employés s’affiche sous forme d’un tableau

—Le tableau contient un champs intitulé "détails employé", l’acteur va cliquer sur le bouton et une fiche d’employé va etre afffiché.

—L’acteur a la possiblité de générer un PDF pour une fiche d’employé.

Raffinement du cas d’utilisation « Consulter Liste Candidats »

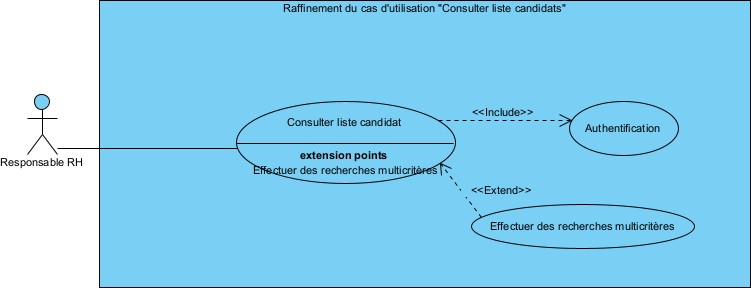


Figure 3.5 – Raffinement du cas d’utilisation « Consulter Liste Candidats »

* + - * Description textuelle du cas d’utilisation **« Consulter liste des candidats » : Acteurs :** Responsable RH

**Précondition :** s’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—Le responsable RH va sélectionner le bouton "Candidats" du menu sidebar

—Ensuite une liste des candidats va etre affiché.

—Dans l’interface "liste candidats", il ya une zone de recherche multicritères

—L’acteur donc prut effectuer des recherches sur la liste des candidats.

* + - * Description textuelle du cas d’utilisation **«Voir détail candidat» : Acteurs :** Responsable RH

**Précondition :** s’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—Le responsable RH peut cliquer sur un bouton pour voir les détails d’une candidature.

Raffinement du cas d’utilisation « Consulter son profil »

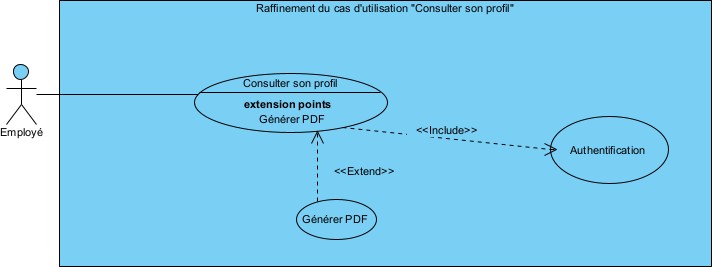


Figure 3.6 – Raffinement du cas d’utilisation « Consulter son profil »

* + - * Description textuelle du cas d’utilisation **« Consulter son profil » : Acteurs :** Employé

**Précondition :** s’authentifier **Postcondition :** Employé authentifié **Scénario principal :**

—L’ employé va selectionner le bouton "Profil" du menu sidebar.

—Le profil va etre affiché

—L’employé a la possibilité de générer un PDF pour sa fiche d’employé.

## Conception

Nous avons détaillé le diagramme de cas d’utilisation par des descriptions textuelles du quelques cas pour mieux clarifier le fonctionnement de ce sprint.

Maintenant nous allons présenter les diagrammes de classe et diagramme de séquence pour modéliser les objets et les relations entre eux ainsi que représenter les interactions entre les acteurs et le système.

### Diagramme de classe

Le diagramme de classe permet de décrire et de présenter les classes et les relations entre eux ainsi de donner une vue statique d’un modèle, ou une partie d’un modèle comme

la montre la figure ci-dessous.

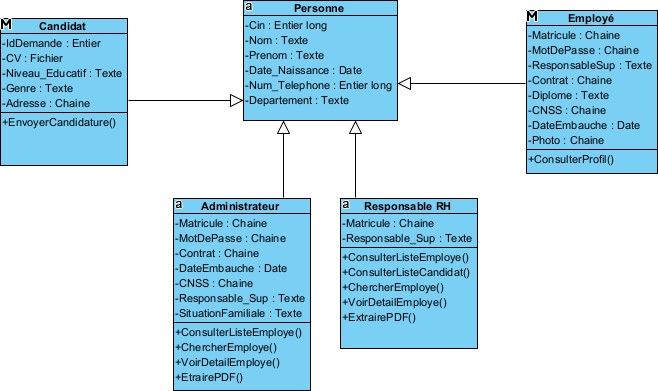


Figure 3.7 – Diagramme de classe du Sprint

### Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les ac- teurs et notre système selon un ordre chronologique. Ci-dessous nous allons présenter les diagrammes de séquences en détail du notre premier sprint.

* Diagramme de séquence « Consulter liste employés » Le responsable RH demande l’in- terface de la liste d’employés, il va accéder à cette interface par le bouton « liste employé » du menu sidebar. Une liste d’employés va être récupérée de la base de données et affichée. Pour chaque employé de la liste on peut accéder à sa fiche par le bouton « voir détails

». Au-dessus de la liste d’employés, on trouve des champs de recherches multicritères, Là où on peut effectuer des filtres sur les employés et la liste va être modifiée pour chaque recherche.

C’est le même scénario pour un administrateur.

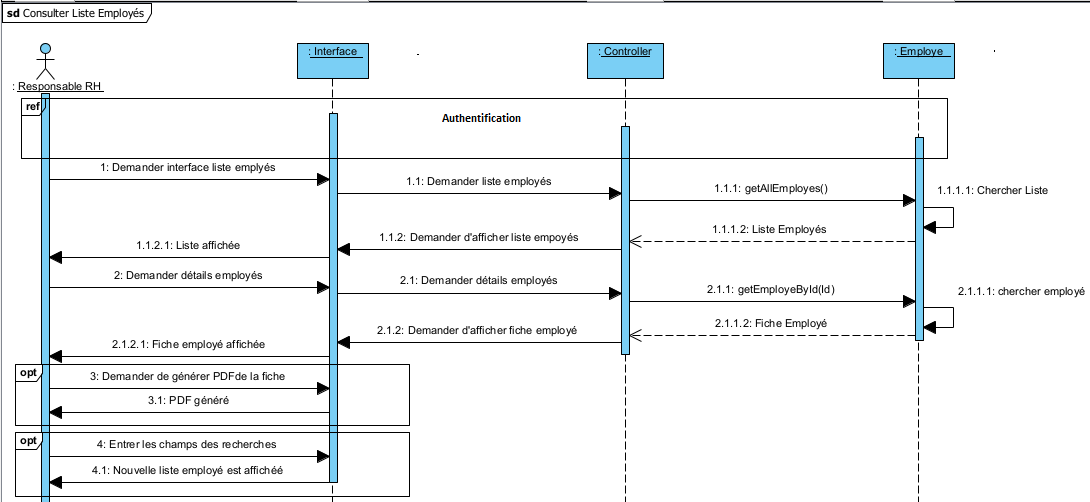


Figure 3.8 – Diagramme de séquence « Consulter liste employés »

* Diagramme de séquence « Consulter liste candidats » Le responsable RH demande l’interface de la liste des candidats, il va accéder à cette interface par le bouton « Candi- dats » du menu sidebar. Une liste des candidats va être récupérée de la base de données et affichée. Au-dessus de la liste des candidats, on trouve des champs de recherches mul- ticritères, Là où on peut effectuer des filtres sur les candidats et la liste va être modifiée pour chaque recherche.

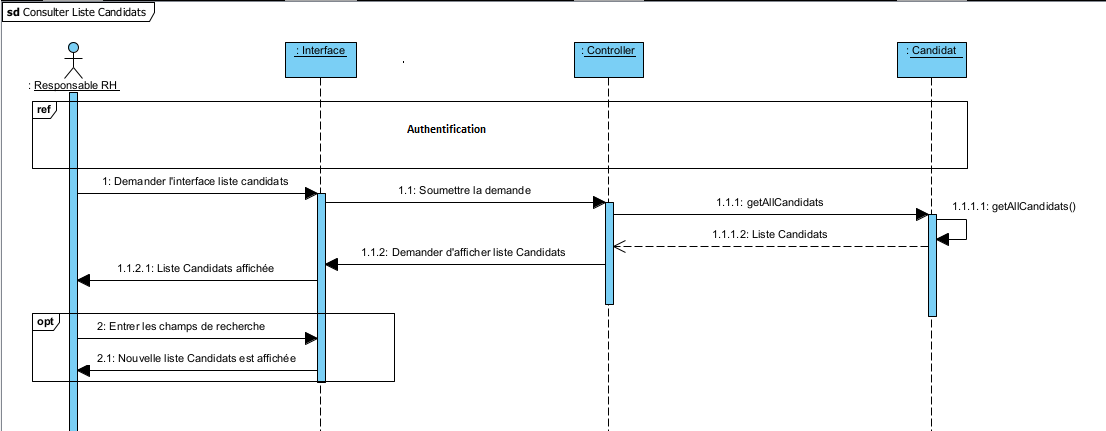


Figure 3.9 – Diagramme de séquence « Consulter liste candidats »

* Diagramme de séquence « Envoyer Candidature » Un candidat demande l’interface d’envoi d’une candidature, il va accéder à cette interface par le bouton « S’inscrire » de la page Home. Le candidat doit remplir tous les champs du formulaire et cliquer sur « Envoyer ». Si un champ n’est pas rempli le bouton « Envoyer » reste désactivé.

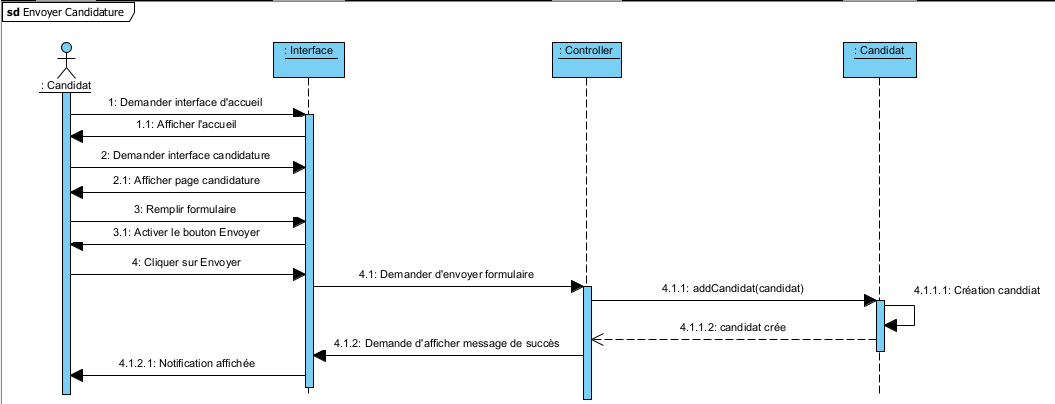


Figure 3.10 – Diagramme de séquence « Envoyer Candidature »

* Diagramme de séquence « Consulter son profil »

L’employé va accéder à son profil par le bouton « Profil » du menu sidebar de son compte. Sa fiche est affichée.

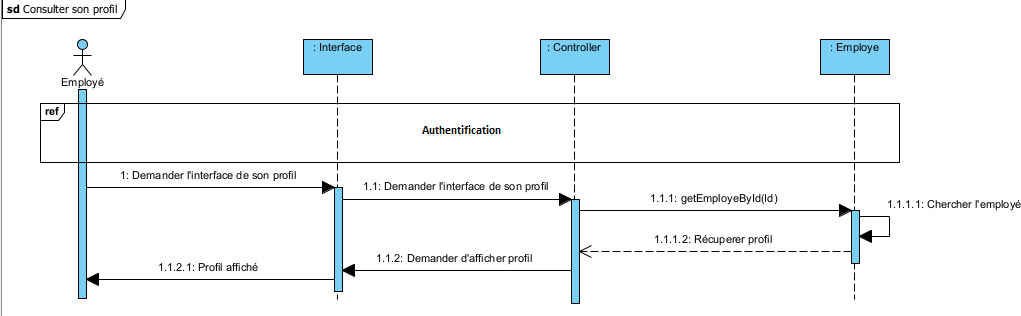


Figure 3.11 – Diagramme de séquence « Consulter son profil »

## Réalisation

Nous présentons dans cette partie les interfaces du notre premier sprint.

### Interface d’envoi d’une candidature

Le candidat doit remplir tous les champs de ce formulaire et puis cliquer sur « Envoyer

». Si un champ n’est pas rempli le bouton « Envoyer » reste désactivé.

Le bouton « Annuler » permet de restaurer les valeurs par défaut de formulaire.

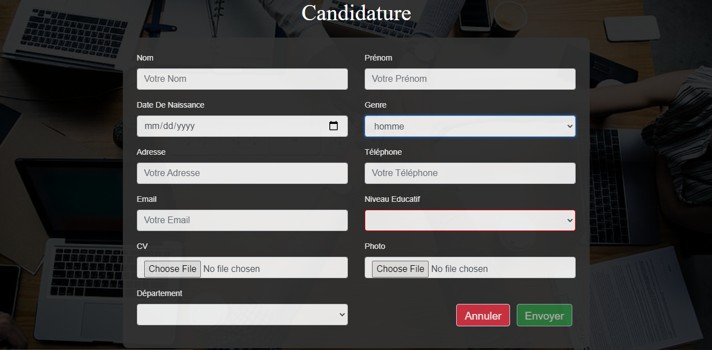


Figure 3.12 – Interface d’envoi d’une candidature

### Interface de récupération de la liste des employés

Le bouton « Liste employés » du menu sidebar permet d’afficher l’interface de la liste d’employé ou on trouve aussi les champs de recherches au-dessus de tableau des employés.

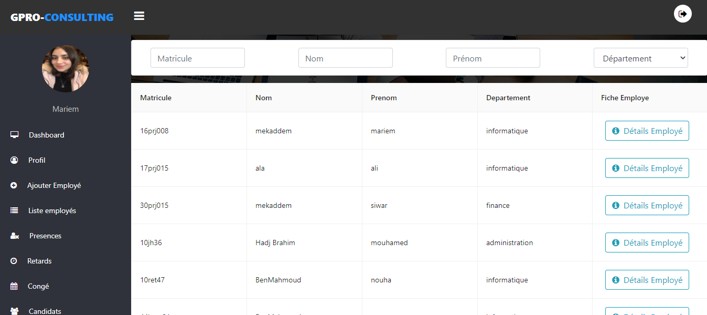


Figure 3.13 – Interface de récupération de la liste des employés

### Interface de recherche d’un employé

On peut effectuer des recherches sur la liste des employés

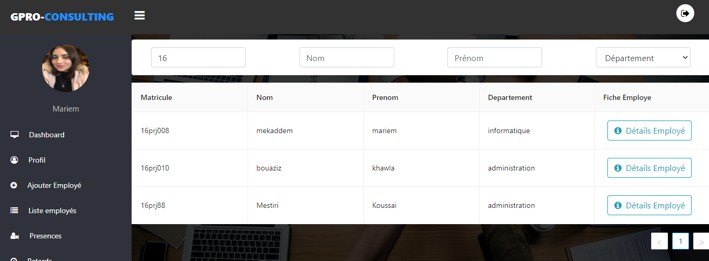


Figure 3.14 – Interface de recherche d’un employé par un seul critère de recherche

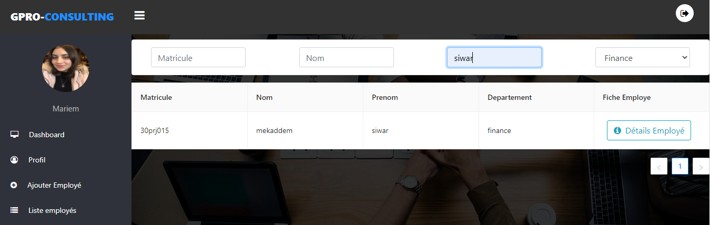


Figure 3.15 – Interface de recherche d’un employé par un plusieurs critères de recherche

### Interface détails employé

Le bouton « Détails Employé » dans le tableau permet d’afficher la fiche d’un employé.

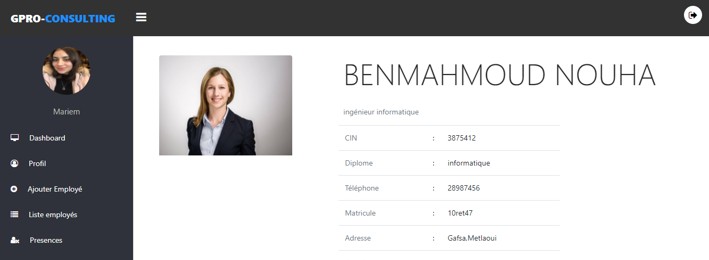


Figure 3.16 – Interface détails employé

### Interface pour générer un PDF

Le bouton « PDF » permet de générer une fiche pdf pour la fiche d’un employé.

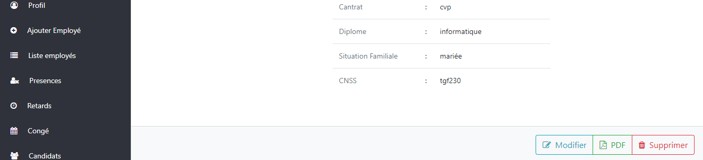


Figure 3.17 – Interface pour générer un PDF

### Génération d’une fiche d’employé

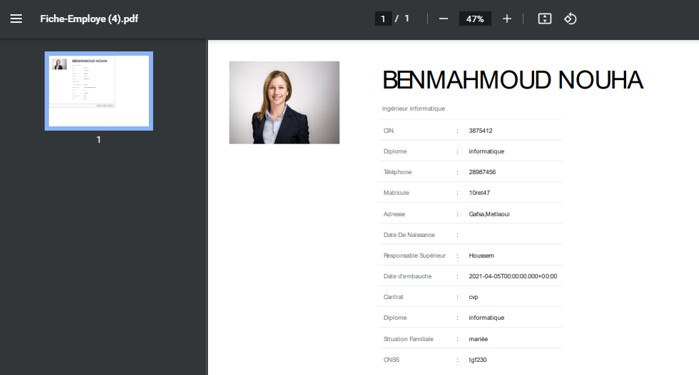


Figure 3.18 – Génération d’une fiche d’employé

### Interface de consultation de son profil

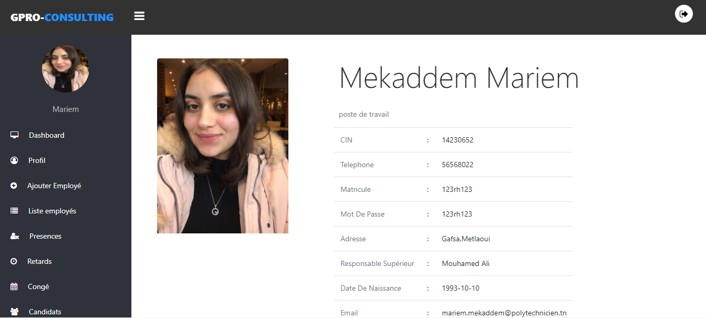


Figure 3.19 – Interface de consultation de son profil

### Interface de récupération de la liste des candidatures

Le bouton « Candidats » du menu sidebar permet d’afficher l’interface de la liste des candidatures où on trouve aussi les champs de recherches au-dessus de tableau des Candidats.

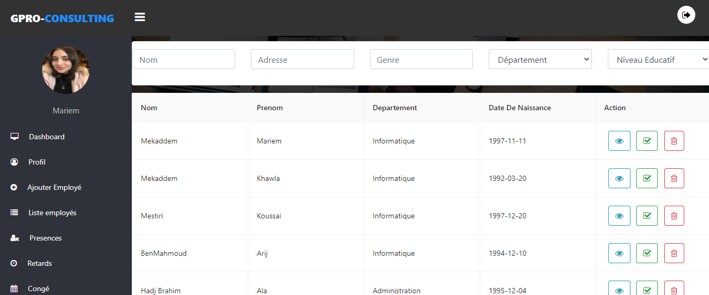


Figure 3.20 – Interface de récupération de la liste des candidatures

### Interface de recherche d’une candidature

On peut effectuer des recherches sur la liste des candidats

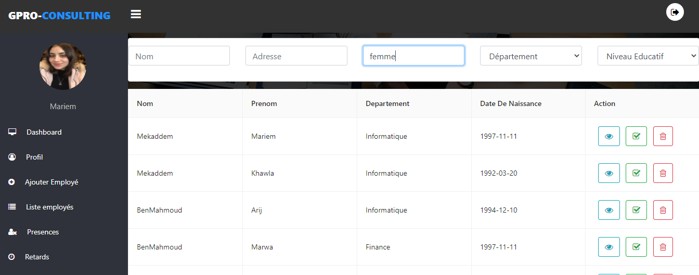


Figure 3.21 – Interface de recherche d’une candidature par un seul critère de recherche

### Interface de voir détail candidat

Le bouton en bleu permet de voir les détails d’un candidat sous forme d’un modal qui va être affiché.

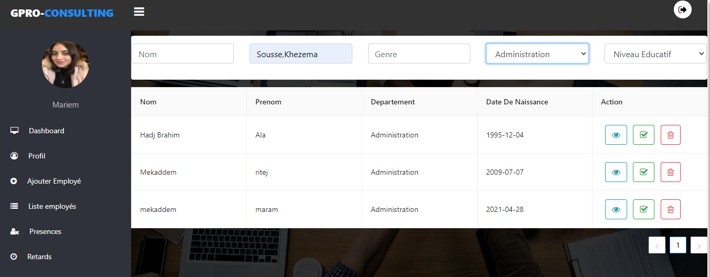


Figure 3.22 – Interface de recherche d’une candidature par plusieurs critères de recherche

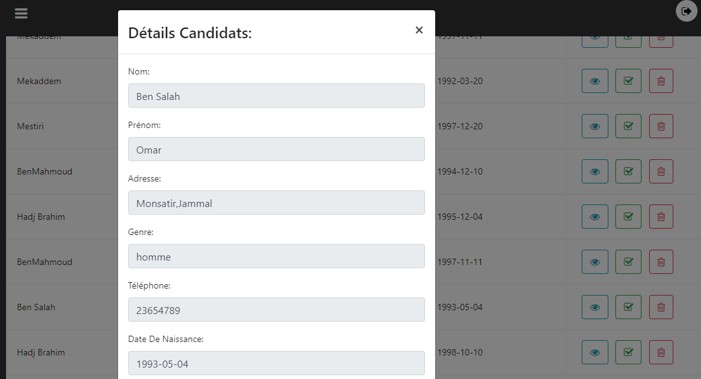


Figure 3.23 – Interface de voir détails d’un candidat

**Conclusion**

Dans ce chapitre nous avons présenté notre premier sprint l’analyse, la conception et la réalisation. Le chapitre suivant sera une présentation de notre deuxième sprint.

CHAPITRE 4

Etude et réalisation du sprint 2

## Introduction

Dans le chapitre précédent, nous avons étudié et réalisé notre premier sprint. Dans ce chapitre, nous allons présenter le deuxième sprint : La partie de gestion des personnels.

## Spécifications fonctionnelles

À travers un diagramme de cas d’utilisation, nous allons présenter la spécification fonctionnelle de ce sprint comme pour le premier sprint.

### Répartition des cas d’utilisations par acteur

La description ci-dessous présente toutes les fonctionnalités de chaque acteur du notre deuxième sprint.

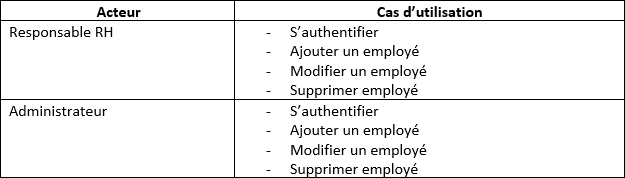


Figure 4.1 – Répartition des cas d’utilisation par acteur

### Diagramme de cas d’utilisation du sprint 2

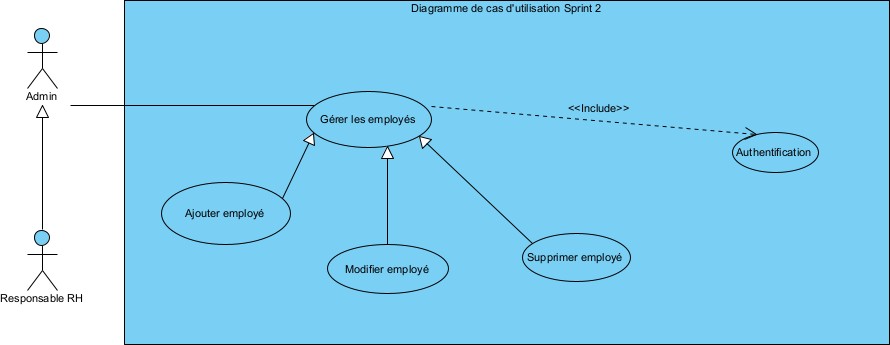


Figure 4.2 – Diagramme de cas d’utilisation Sprint 2

Ce diagramme nous montre une vue globale du sprint 2. L’administrateur hérite les fonctionnalités « Ajouter employés », « Modifier employé » et « Supprimer employé » de responsable RH.

### Description textuelle des cas d’utilisation

* + - * Description textuelle du cas d’utilisation **« Ajouter employé » : Acteurs :** Responsable RH, Administrateur

**Précondition :** S’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—L’acteur va sélectionner le bouton "Ajouter employé" du menu sidebar

—Un formulaire d’ajout va etre affiché

* + - * Description textuelle du cas d’utilisation **« Modifier employé » : Acteurs :** Responsable RH, Administrateur

**Précondition :** S’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—L’acteur va sélectionner le bouton "Modifier" qui se trouve en bas de la fiche employé

—Un formulaire de modification va etre affiché

* + - * Description textuelle du cas d’utilisation **« Supprimer employé » : Acteurs :** Responsable RH, Administrateur

**Précondition :** S’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—L’acteur va sélectionner le bouton "Supprimer employé" du menu sidebar

—Ensuite l’employé va etre supprimé

## Conception

Nous avons détaillé le diagramme de cas d’utilisation par des descriptions textuelles du quelques cas pour mieux clarifier le fonctionnement de ce sprint.

Maintenant nous allons présenter les diagrammes de classe et diagramme de séquence pour modéliser les objets et les relations entre eux ainsi que représenter les interactions entre les acteurs et le système.

### Diagramme de classe

Le diagramme de classe permet de décrire et de présenter les classes et les relations entre eux ainsi de donner une vue statique d’un modèle, ou une partie d’un modèle comme la montre la figure ci-dessous.

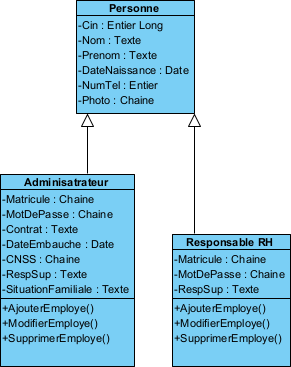


Figure 4.3 – Diagramme de classe du Sprint 2

### Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les ac- teurs et notre système selon un ordre chronologique. Ci-dessous nous allons présenter les diagrammes de séquences en détail du notre premier sprint.

* Diagramme de séquence «S’authentifier»

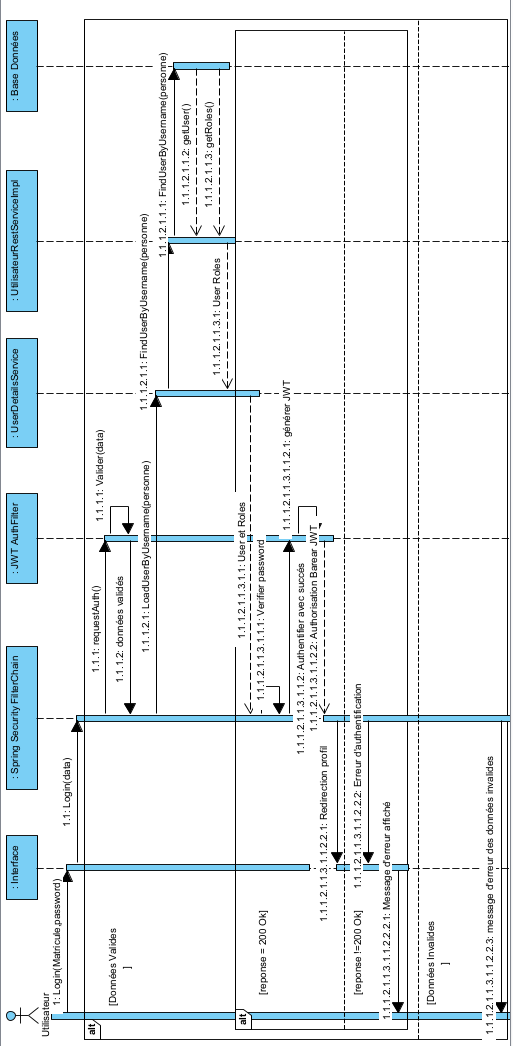


Figure 4.4 – Diagramme de séquence «S’authentifier»

* + Diagramme de séquence « Ajouter employé »

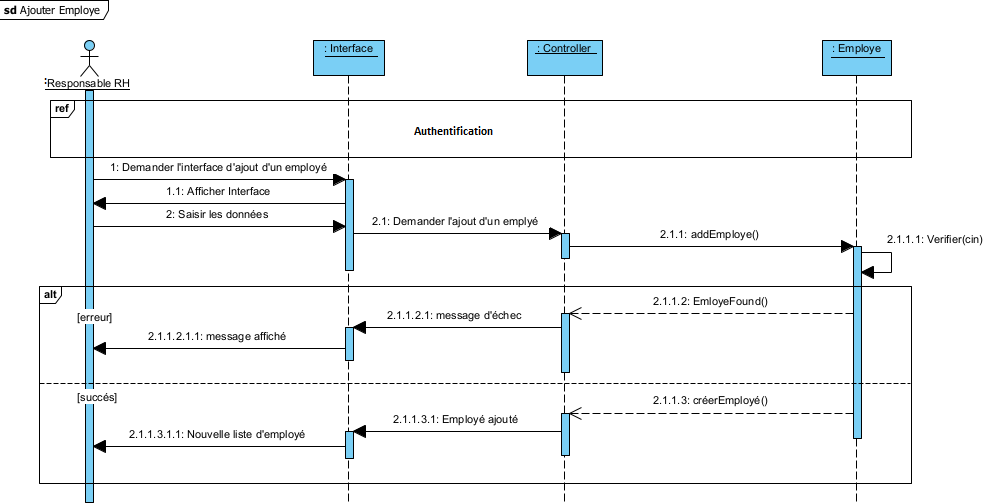
Le responsable RH demande l’interface d’ajout d’un employé, il va accéder à cette interface par le bouton « Ajouter employé » du menu sidebar. Un formulaire d’ajout d’employés va être affiché. C’est le même scénario pour un administrateur.

Figure 4.5 – Diagramme de séquence « Ajouter employé »

* + Diagramme de séquence « Modifier employé »

Le responsable RH demande l’interface de modification de la fiche d’un employé, il va accéder à cette interface par le bouton « Modifier» qui se trouve en bas de la fiche. Un formulaire de modification d’employé va être affiché. C’est le même scénario pour un administrateur.

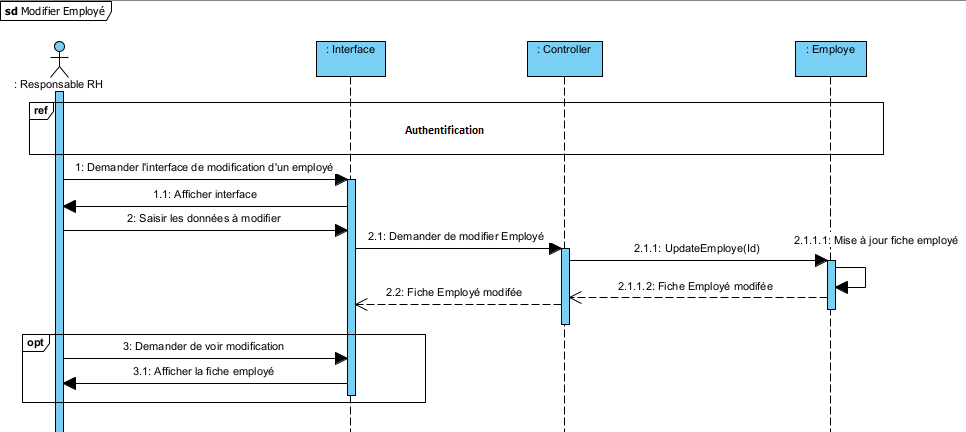


Figure 4.6 – Diagramme de séquence « Modifier employé »

* + Diagramme de séquence «Supprimer employé »

Le responsable RH demande l’interface fiche d’employé, il va accéder à cette interface par le bouton «Voir détails» qui se trouve dans la liste des employés. Il va cliquer sur le bouton « Supprimer » se trouve en bas de la fiche . C’est le même scénario pour un administrateur.

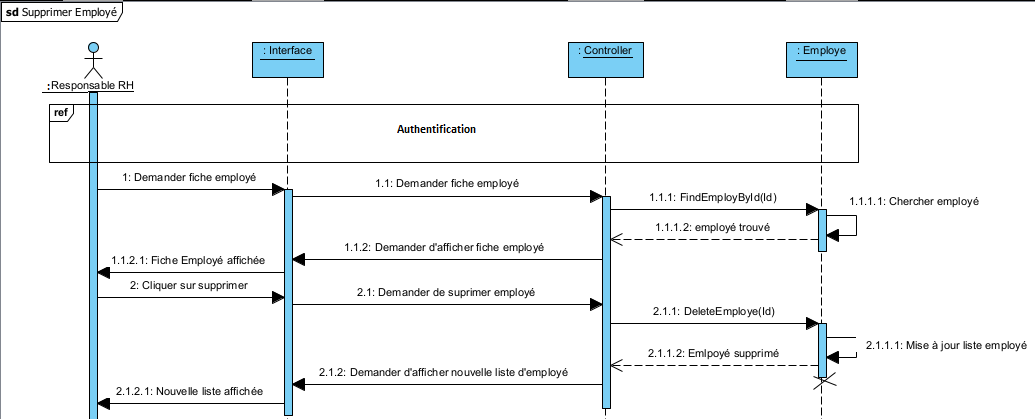


Figure 4.7 – Diagramme de séquence «Supprimer employé »

## Réalisation

Nous présentons dans cette partie les interfaces du notre deuxième sprint.

### Interface d’authentification

L’acteur doit saisir sa matricule et son mot de passe, puis il va cliquer sur le bouton

« Se connecter ». Si les données sont incorrectes il doit les saisir du nouveau.

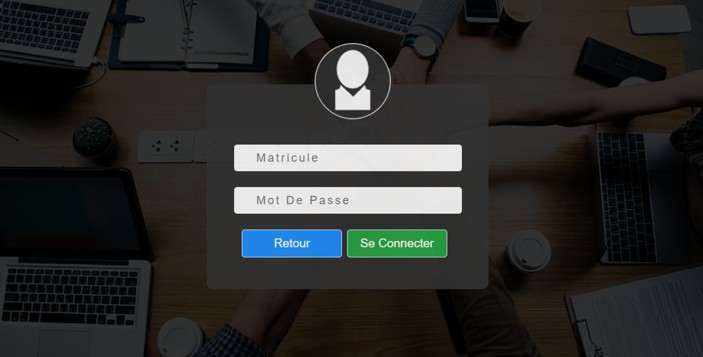


Figure 4.8 – Interface d’authentification

### Interface d’ajout d’un employé

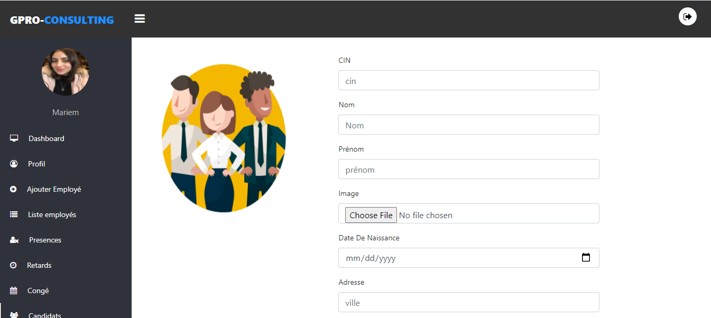
Le responsable RH doit remplir ce formulaire pour ajouter un nouvel employé, puis il va cliquer sur le bouton « Ajouter ». Le bouton « Annuler » permet de restaurer les valeurs par défaut de formulaire.

Figure 4.9 – Interface d’ajout d’un employé (1)

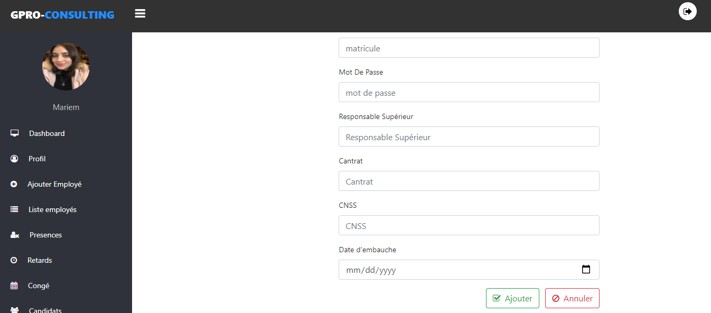


Figure 4.10 – Interface d’ajout d’un employé (2)

### Interface de modifier la fiche d’un employé

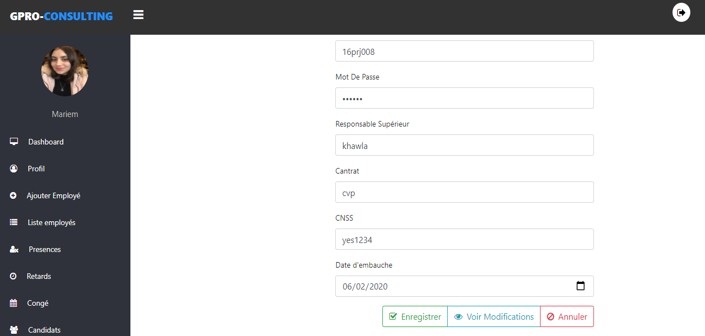
Le responsable RH va saisir de nouveau les champs à modifier dans ce formulaire, puis il va cliquer sur le bouton « Enregistrer ». Le bouton « Annuler » permet de revenir à la liste des employés.

Figure 4.11 – Interface de modifier la fiche d’un employé

### Interface de suppression d’un employé

Le responsable RH doit cliquer sur le bouton « Supprimer ».

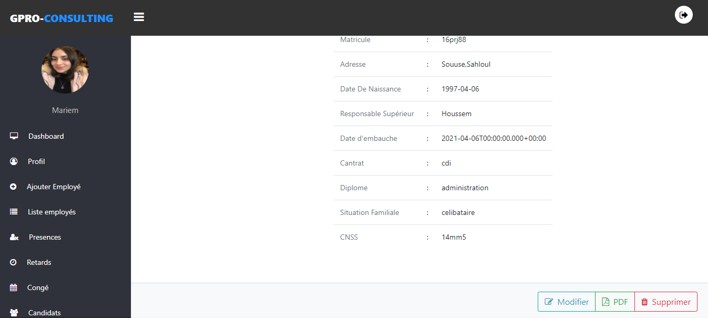


Figure 4.12 – Interface de suppression d’un employé

**Conclusion**

Dans ce chapitre nous avons présenté notre deuxième sprint l’analyse, la conception et la réalisation. Le chapitre suivant sera une présentation de notre troisième sprint.

CHAPITRE 5

Etude et réalisation du sprint 3

## Introduction

Dans le chapitre précédent, nous avons étudié et réalisé notre deuxième sprint : la gestion des personnels. Dans ce chapitre, nous allons présenter notre troisième sprint : La partie gestion d’assiduité.

## Spécifications fonctionnelles

À travers un diagramme de cas d’utilisation, nous allons présenter la spécification fonctionnelle de ce sprint.

### Répartition des cas d’utilisations par acteur

Le tableau ci-dessous présente toutes les fonctionnalités de chaque acteur du sprint 3.

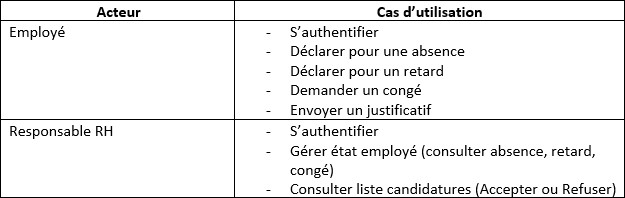


Figure 5.1 – Répartition des cas d’utilisation par acteur

### Diagramme de cas d’utilisation du sprint 3

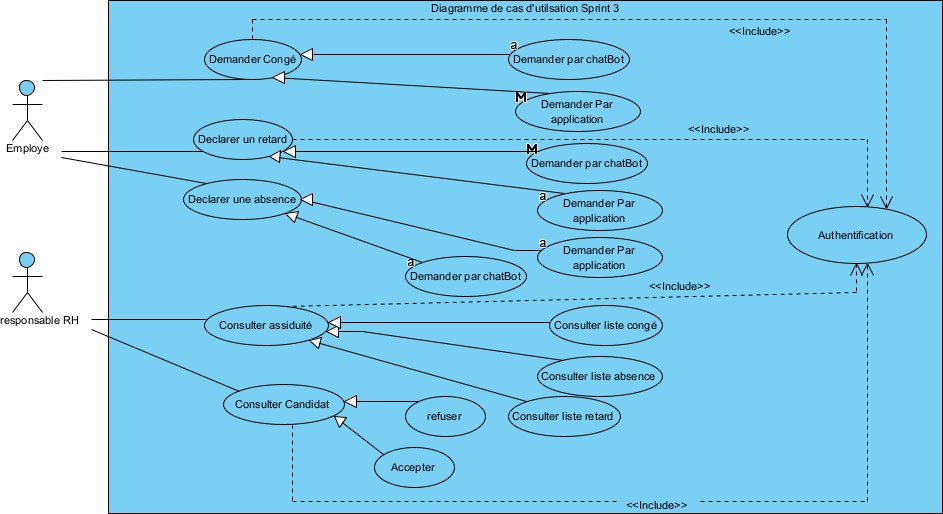


Figure 5.2 – Diagramme de cas d’utilisation Sprint 3

Ce diagramme nous montre une vue globale du sprint 3. L’employé peut demander des congés, déclarer des absences ou des retards via notre application. Le responsable RH a une vision globale donc sur tous type de demandes ou déclarations.

### Description textuelle des cas d’utilisation

* + - * Description textuelle du cas d’utilisation **« Demander un congé » par(application/ chatbot) :**

**Acteurs :** Employé

**Précondition :** S’authentifier **Postcondition :** Acteur authentifié **Scénario principal :**

—L’employé va sélectionner le bouton "demande congé" du menu sidebar

—Il va remplir sa demander à travers un formulaire

* + - * Description textuelle du cas d’utilisation **« Déclarer un retard » par(application**

**/ chatbot) : Acteurs :** Employé

**Précondition :** S’authentifier **Postcondition :** Employé authentifié **Scénario principal :**

—L’employé va sélectionner le bouton "Déclarer une absence" du menu sidebar

—Il va remplir sa déclaration à travers un formulaire

* + - * Description textuelle du cas d’utilisation **« Déclarer une absence» par ( applica- tion/ chatbot) :**

**Acteurs :** Employé

**Précondition :** S’authentifier **Postcondition :** Employé authentifié **Scénario principal :**

—L’employé va sélectionner le bouton "Déclarer une absence" du menu sidebar

—Il va remplir sa déclaration à travers un formulaire

* + - * Description textuelle du cas d’utilisation **« Consulter état employé» : Acteurs :** Responsable RH

**Précondition :** S’authentifier **Postcondition :** Responsable RH authentifié **Scénario principal :**

—Le responsable RH va consulter la liste des retards.

—Le responsable RH va consulter la liste des absences.

—Le responsable RH va consulter la liste des congés.

* + - * Description textuelle du cas d’utilisation **« Consulter candidats» : Acteurs :** Responsable RH

**Précondition :** S’authentifier **Postcondition :** Responsable RH authentifié **Scénario principal :**

—Le responsable RH va consulter la liste des candidats et après avoir consulter les détails de chaque candidat.

—il peut accepter ou refuser une candidature en cliquant soit sur le buton d’acceptation

ou le bouton de suppression, en cas d’acceptation un e-mail va être envoyer automa- tique au candidat.

## Conception

Nous avons détaillé le diagramme de cas d’utilisation par des descriptions textuelles du quelques cas pour mieux clarifier le fonctionnement de ce sprint.

Maintenant nous allons présenter les diagrammes de classe et diagramme de séquence pour modéliser les objets et les relations entre eux ainsi que représenter les interactions entre les acteurs et le système.

### Diagramme de classe

Le diagramme de classe permet de décrire et de présenter les classes et les relations entre eux ainsi de donner une vue statique d’un modèle, ou une partie d’un modèle comme la montre la figure ci-dessous.

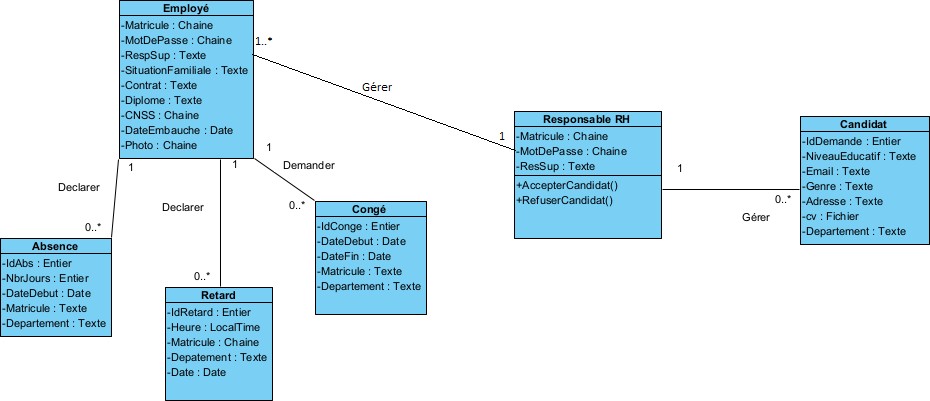


Figure 5.3 – Diagramme de classe Sprint 3

### Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les ac- teurs et notre système selon un ordre chronologique. Ci-dessous nous allons présenter les diagrammes de séquences en détail du notre premier sprint.

* Diagramme de séquence «Demander un congé»

L’employé peut accéder à l’interface de demande de congé en cliquant sur le buton « Demander un congé » du menu sidebar. Il va remplir la demande et l’envoyer.

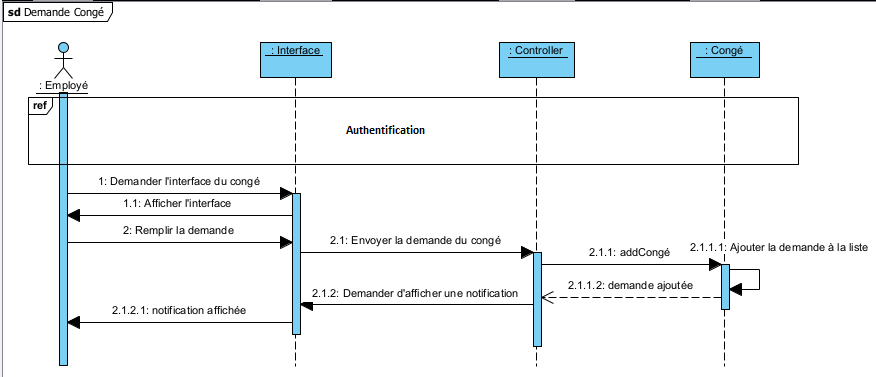


Figure 5.4 – Diagramme de séquence «Demander un congé»

* + Diagramme de séquence «Déclarer un retard»

L’employé peut accéder à l’interface de déclaration d’un retard en cliquant sur le buton

« Déclarer un retard» du menu sidebar. Il va remplir la déclaration et l’envoyer.

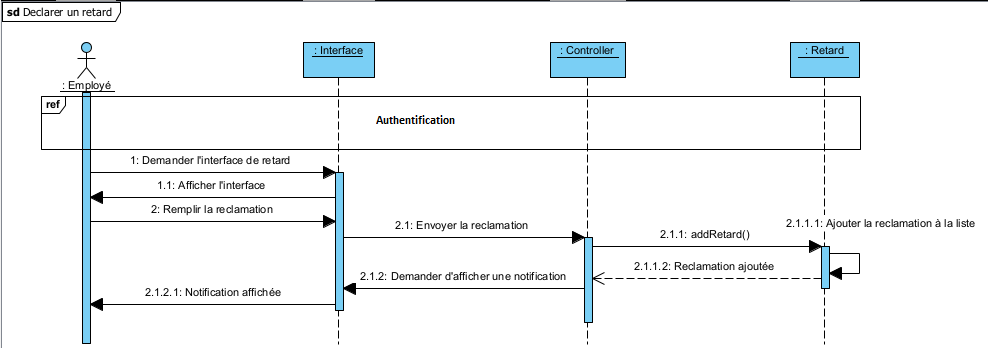


Figure 5.5 – Diagramme de séquence «Déclarer un retard»

* + Diagramme de séquence «Déclarer une absence»

L’employé peut accéder à l’interface de déclaration d’une absence en cliquant sur le buton

« Déclarer une absence» du menu sidebar. Il va remplir la déclaration et l’envoyer.

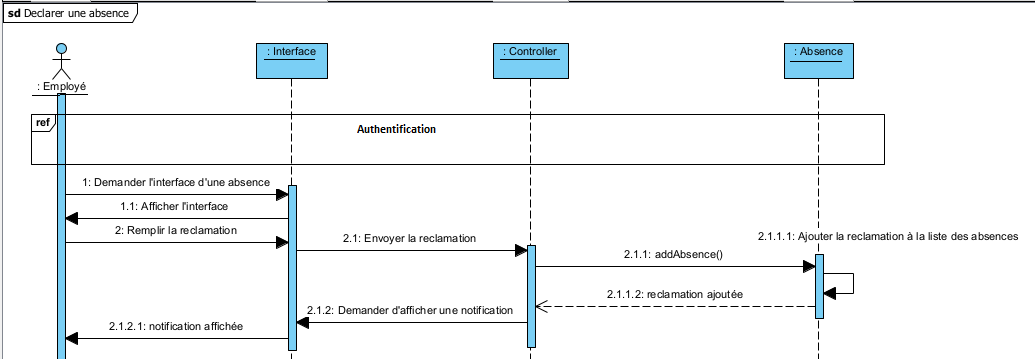


Figure 5.6 – Diagramme de séquence «Déclarer une absence»

* + Diagramme de séquence du chatbot

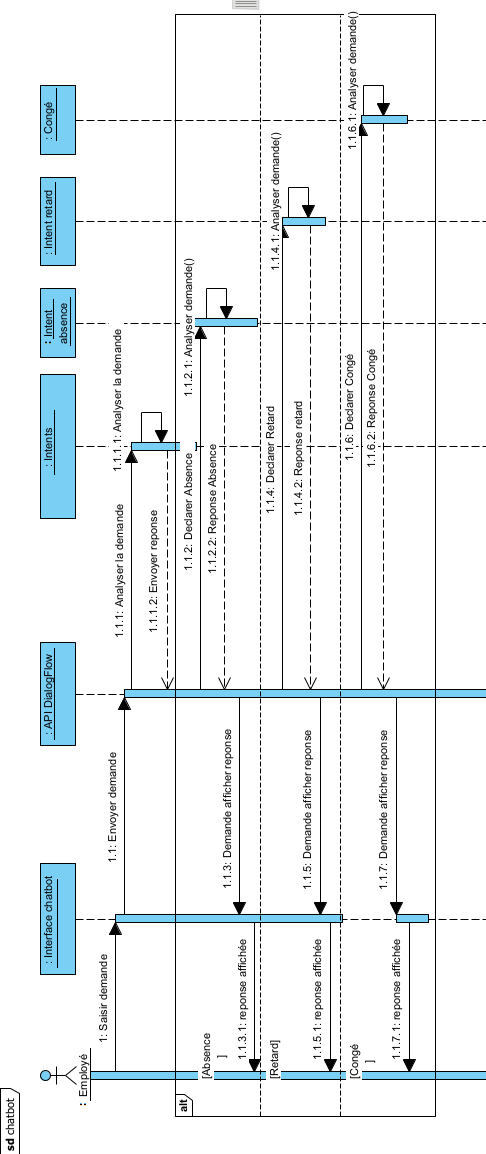


Figure 5.7 – Diagramme de séquence du chatbot

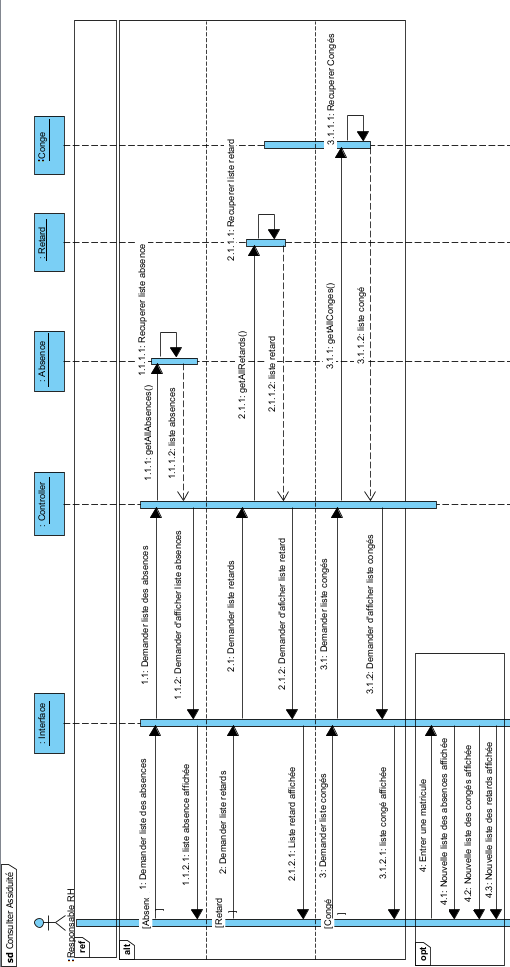


Figure 5.8 – Diagramme de séquence «Consulter assiduité»

* + Diagramme de séquence «Consulter assiduité»

Le responsable RH peut consulter les demandes et les déclarations en cliquant sur le buton

« Assiduité » du menu sidebar.

## Réalisation

Nous présentons dans cette partie les interfaces du notre troisième sprint.

### Interface de demande de congé

L’employé doit remplir sa demande à travers ce formulaire.

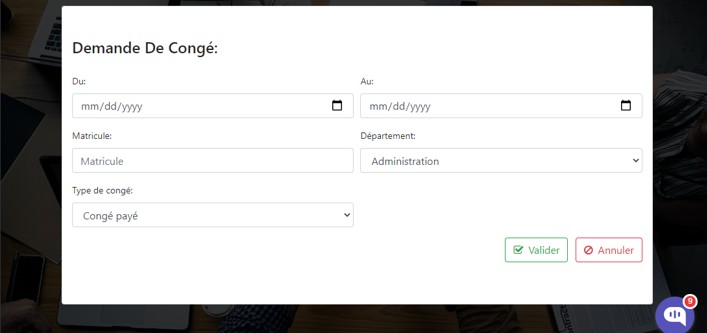


Figure 5.9 – Interface de demande de congé

### Interface déclaration d’un retard

L’employé doit remplir sa déclaration pour son retard à travers ce formulaire.

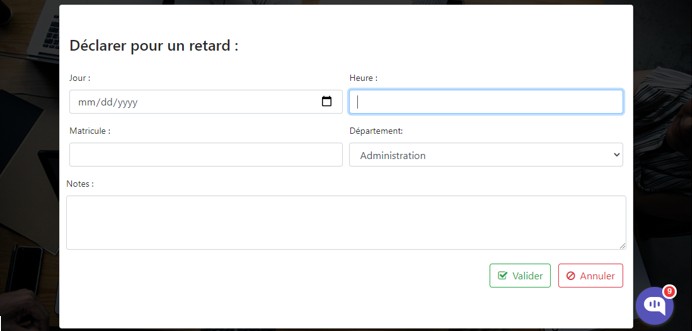


Figure 5.10 – Interface déclaration d’un retard

### Interface déclaration d’une absence

L’employé doit remplir sa déclaration pour son absence à travers ce formulaire.

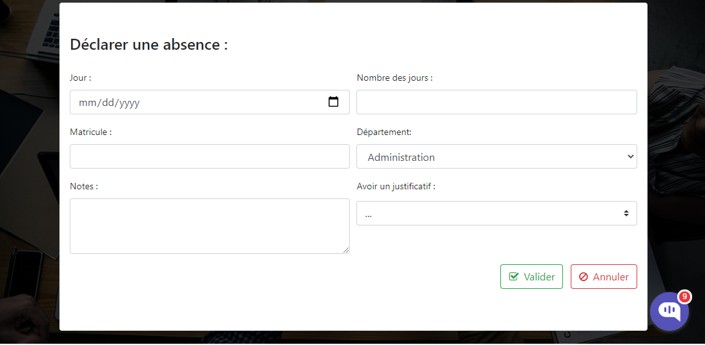


Figure 5.11 – Interface déclaration d’une absence

### Interface de consulter assiduité

Le responsable RH peut naviguer simplement entre les différentes tables pour consulter les différents demandes et réclamations. Il peut aussi effectuer des recherches sur ces tables.

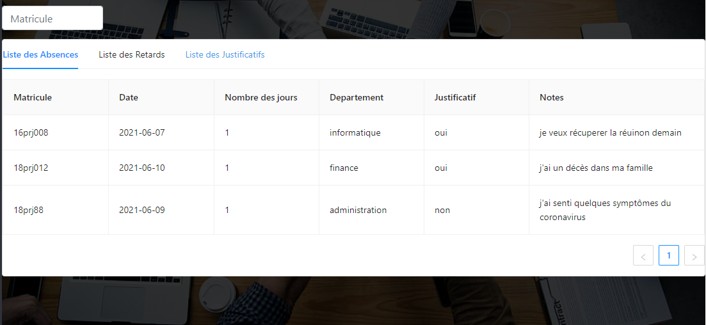


Figure 5.12 – Interface de consulter la liste des absences

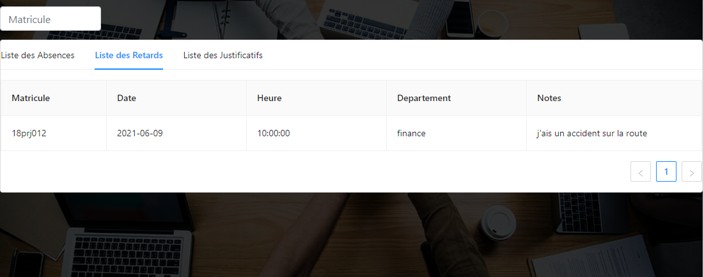


Figure 5.13 – Interface de consulter la liste des retards

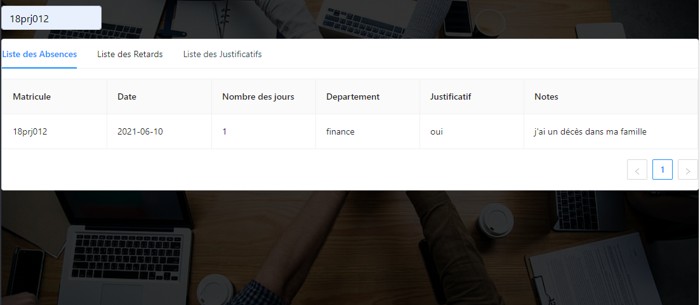


Figure 5.14 – Interface de consulter assiduités avec un champs de recherche

**Conclusion**

Dans ce chapitre nous avons présenté le troisième sprint l’analyse, la conception et la réalisation. Le chapitre suivant sera une présentation de notre dernier sprint.

CHAPITRE 6

Etude et réalisation du sprint 4

## Introduction

Dans le chapitre précédent, nous avons étudié et réalisé notre deuxième sprint : la gestion des personnels. Dans ce chapitre, nous allons présenter notre troisième sprint : La partie gestion d’assiduité.

## Spécifications fonctionnelles

À travers un diagramme de cas d’utilisation, nous allons présenter la spécification fonctionnelle de ce sprint.

### Répartition des cas d’utilisations par acteur

Le tableau ci-dessous présente toutes les fonctionnalités de chaque acteur du sprint 4.

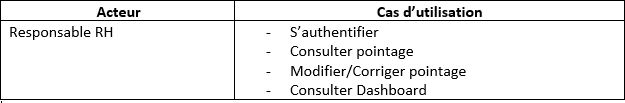


Figure 6.1 – Fonctionnalités par acteur du sprint 4

### Diagramme de cas d’utilisation du sprint 4

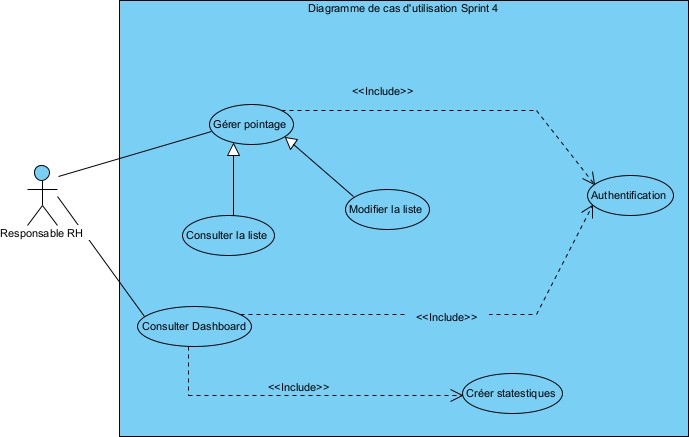


Figure 6.2 – diagramme de cas d’utilisation Sprint 4

Ce diagramme nous montre une vue globale du sprint 4. Le responsable RH peut consulter la liste de pointage et la modifier en cas d’une faute au niveau du pointage. Aussi, il peut consulter un Dashboard pour voir des informations présentées graphiquement et d’avoir une vue générale du l’entreprise.

### Description textuelle des cas d’utilisation

* + - * Description textuelle du cas d’utilisation **« Gérer pointage » : Acteurs :** Responsable RH

**Précondition :** S’authentifier **Postcondition :** Responsable RH authentifié **Scénario principal :**

—Le responsable RH a la possibilité de consulter la liste de pointage en cliquant sur le bouton « présences » du menu sidebar.

—et aussi de la corriger en cas d’erreur du pointage en cliquant sur le bouton « modifier

» qui se trouve au niveau du tableau de pointage.

* + - * Description textuelle du cas d’utilisation **« Consulter Dashboard » : Acteurs :** Responsable RH

**Précondition :** S’authentifier **Postcondition :** Responsable RH authentifié **Scénario principal :**

—A travers une simple présentation graphique le responsable RH peut voir quelques statistiques sur un ensemble des informations de la société en cliquant sur le buton « Dashboard » du menu sidebar.

## Conception

Nous avons détaillé le diagramme de cas d’utilisation par des descriptions textuelles du quelques cas pour mieux clarifier le fonctionnement de notre sprint.

Maintenant nous allons présenter les diagrammes de classe et diagramme de séquence pour modéliser les objets et les relations entre eux ainsi que représenter les interactions entre les acteurs et le système.

### Diagramme de classe

Le diagramme de classe permet de décrire et de présenter les classes et les relations entre eux ainsi de donner une vue statique d’un modèle, ou une partie d’un modèle comme la montre la figure ci-dessous.

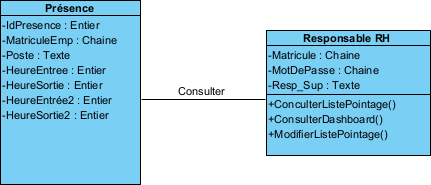


Figure 6.3 – Diagramme de classe Sprint 4

### Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les ac- teurs et notre système selon un ordre chronologique. Ci-dessous nous allons présenter les diagrammes de séquences en détail du notre premier sprint.

* Diagramme de séquence « Gérer pointage »

Le responsable RH peut accéder à l’interface de gestion de pointage, une liste de pointage va être affichée. Il a la possibilité de corriger un pointage en cas d’erreur au niveau de saisie.

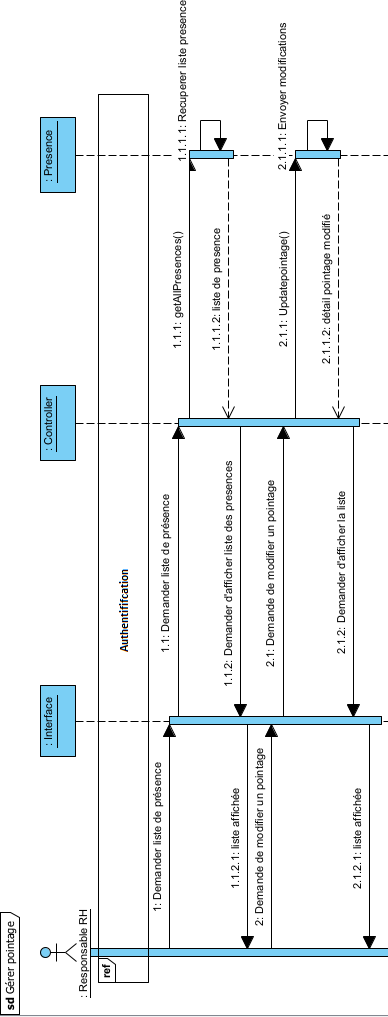


Figure 6.4 – Diagramme de séquence « Gérer pointage »

* + Diagramme de séquence « Consulter Dashboard »

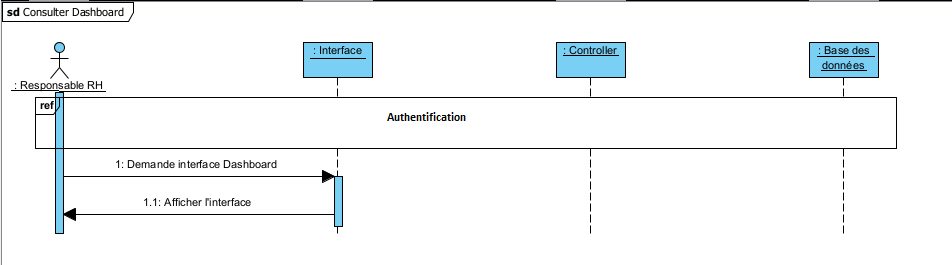
Le responsable RH peut accéder à l’interface du Dashboard, une représentation graphique des données de la société va être affichée.

Figure 6.5 – Diagramme de séquence « Consulter Dashboard »

## Réalisation

Nous présentons dans cette partie les interfaces du notre sprint.

### Interface de Consulter pointage

Le responsable RH peut accéder à l’interface de gestion de pointage en cliquant du le bouton « Présence » du menu Sidebar, une liste de pointage va être affichée.

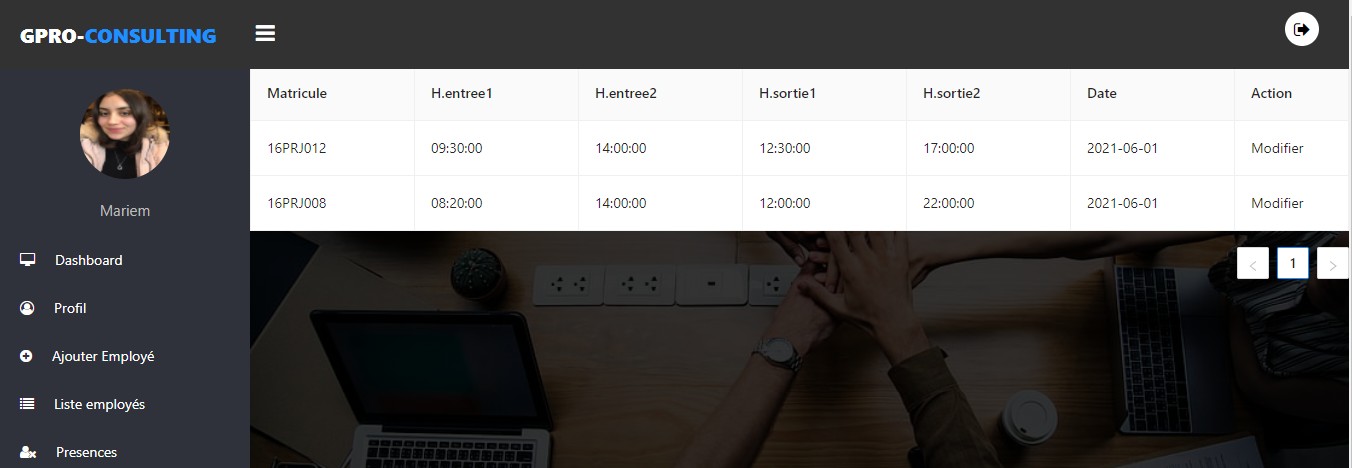


Figure 6.6 – Interface de Consulter pointage

Le responsable RH à la possibilité de modifier les champs en cliquant sur « Modifier

» dans la liste de pointage.

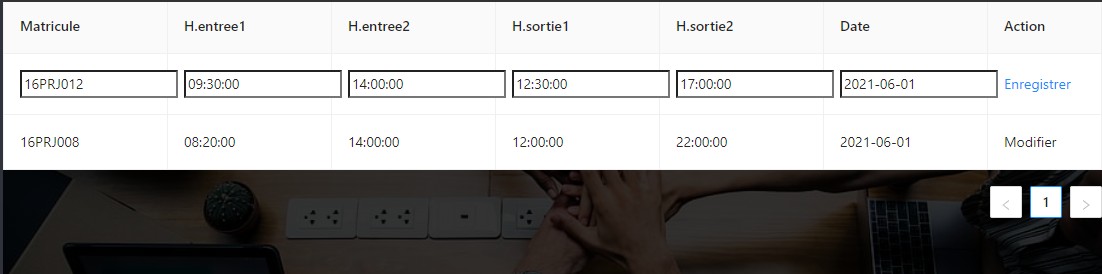


Figure 6.7 – Interface de Modifer pointage

### Interface de Consulter dashboard

Le responsable RH peut accéder à l’interface Dashboard en cliquant du le bouton

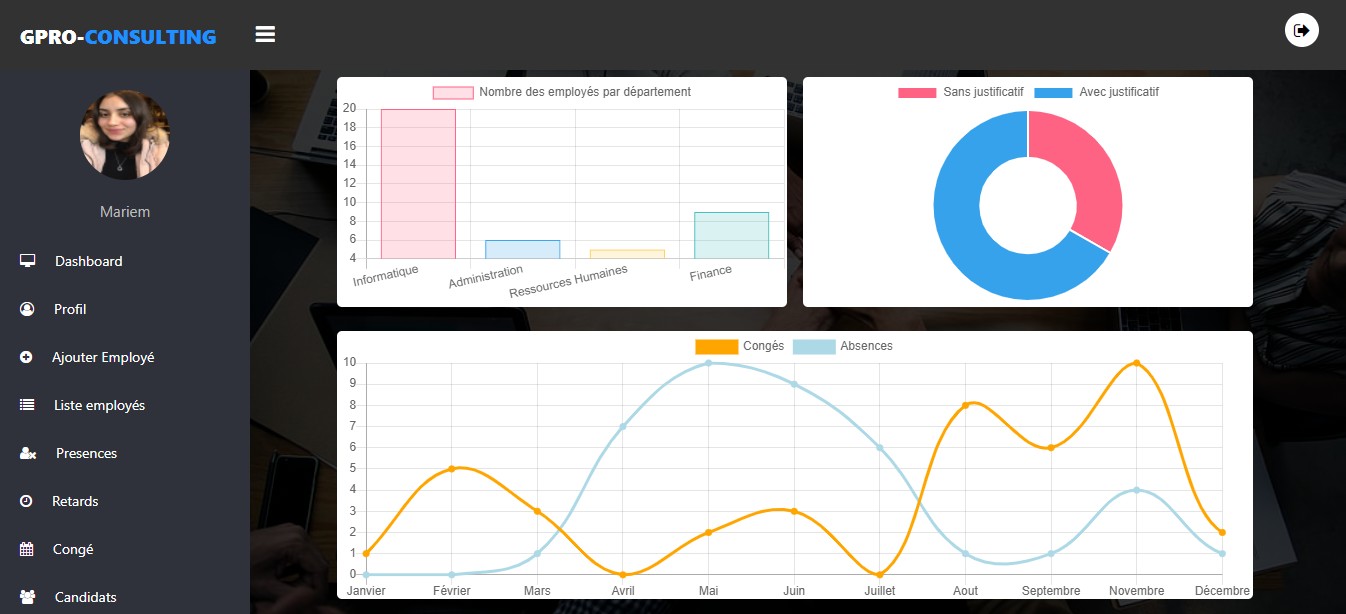
« Dashboard » du menu Sidebar, des informations sous un format graphique vont être affichées.

Figure 6.8 – Interface de Modifer pointage

**Conclusion**

Dans ce dernier chapitre nous avons présenté le troisième sprint l’analyse, la conception et la réalisation.

## Conclusion générale et perspective

L’objectif de notre projet de fin d’étude est de développer une application web de gestion des ressources humaines. Le présent rapport couvre l’étude, la conception et l’im- plémentation de notre solution. Nous avons opté la méthodologie SCRUM pour réaliser notre projet.

Nous avons commencé par étudier le contexte général de notre application. Ensuite, nous avons réparti notre projet en quatre sprints en respectant les priorités de nos Besoins fonctionnels. Tout au long de la période de stage, nous avons consacré plus de temps à découvrir et à maîtriser les outils et les technologies au niveau de l’application.

En terme d’amélioration, nous pouvons à titre d’exemple, ajouter des fonctionnalités à notre chabot en intégrant des modules d’intelligences pour rendre le flux de l’information automatisé dans notre application. C’est-à-dire toute action faite par l’employé

(une demande de congé, une déclaration d’un retard ou d’une absence) va être traitée par le chabot : l’affectation d’un congé, l’analyse d’un justificatif . . . Et le résultat de ce traitement va être envoyer sous forme d’un historique des actions au responsable RH. Ce dernier il a la possibilité de modifier ces actions.

Ce projet était une occasion pour enrichir et développer mes compétences au sein d’un environnement professionnel.

# Webography

1. *Angular*. Juin 2021. url : https://fr.wikipedia.org/wiki/Angular.
2. *Bootstrap*. Juin 2021. url : https : / / fr . wikipedia . org / wiki / Bootstrap \_ (framework).
3. *CSS*. Juin 2021. url : https://fr.wikipedia.org/wiki/Feuilles\_de\_style\_ en\_cascade.
4. *diagramme de cas d’utilisation*. Juin 2021. url : https:// laurent- audibert. developpez.com/.
5. *HTML*. Juin 2021. url : https://fr.wikipedia.org/wiki/Hypertext\_Markup\_ Language.
6. *IntelliJ IDEA*. Juin 2021. url : https://fr.wikipedia.org/wiki/IntelliJ\_ IDEA.
7. *Java*. Juin 2021. url : https://fr.wikipedia.org/wiki/Java\_(langage).
8. *Java*. Juin 2021. url : https://[www.java.com/fr/download/help/java8\_fr.](http://www.java.com/fr/download/help/java8_fr) html).
9. *MVC*. Juin 2021. url : https://openclassrooms.com.
10. *PostreSQL*. Juin 2021. url : https://fr.wikipedia.org/wiki/PostgreSQL.
11. *Spring Boot*. Juin 2021. url : https : / / fr . wikipedia . org / wiki / Spring \_ (framework).
12. *Spring Boot*. Juin 2021. url : : 20https:/[/www.ionos.fr/digitalguide/sites-](http://www.ionos.fr/digitalguide/sites-) internet/developpement-web/tutoriel-spring-boot/.
13. *WebStorm IDEA*. Juin 2021. url : https://fr.wikipedia.org/wiki/WebStorm.