Dédicaces

*À ma chère maman, vous m’avez soutenu et encouragé tout au long ces années d’études, à l’âme de mon père 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 Mes chères sœurs Marwa , Amani et à mon frère Mohamed , Ces quelques lignes, ne sauraient traduire le profond amour que je vous porte. Vous êtes ma vraie source d’énergie et de bonheur. C’est grâce à vos encouragements et vos riches conseils je puisse progresser dans ma vie et mes études. Vous*

*êtes une fierté pour moi.*

*À Mes chers amis : Imen, Marwen, Dhafer , Mehdi, Zied, et Amine, 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.*

Remerciements

*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, pour avoir accepté de m’encadrer et qui, avec ses conseils, sa collaboration et sa disponibilité, m’a assuré le bon déroulement de mon projet.*

*J’exprime ma gratitude, également, à tous les enseignants de l’Ecole*

*Polytechnique Sousse qui ont Contribué pour achever ce stade.*

**Résumé**

**L’objectif principal de ce travail de PFE consiste à implémenter une solution pour la bonne gestion de Ressources humaines.**

**Cette application aide et facilite les taches et les missions des différents acteurs de la société : donc elle assure la bonne maitrise et un gain du temps.**

**Le responsable RH peut accéder aux différents états : état des employés, états de congé, accéder : au budget, état de meeting, liste ToDO, pointage et annonces. Ce dernier il est capable de changer et supprimer certain état.**

**Aussi l’employé peut accéder à son profile.**

**Mot clé : gestion de ressources humaines, temps, responsable RH, employé, faciliter, accéder.**

Table des matières

Page

**Contenu II**

**List des Figures………………………………………………………………………VIII**

**List des Tables IX**

**Abréviations 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 11](#_TOC_250080)
     3. BackLog de produit………………………………………………….12
     4. Structure de d découpage du projet………………………………….15
  3. [Architecture globale de la solution 16](#_TOC_250079)
     1. [Architecture physique de l’application 16](#_TOC_250078)
     2. Architecture applicative de la solution………………………………17
  4. [Cadre de réalisation et choix techniques 18](#_TOC_250077)
     1. [Choix technique 18](#_TOC_250076)

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…………………………….24](#_TOC_250054)
     3. [Description textuelle des cas d’utilisation 24](#_TOC_250054)
  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 de récupération de la liste des employés 34](#_TOC_250047)
     2. [Interface de recherche d’un employé 35](#_TOC_250046)
     3. [Interface détails employé 35](#_TOC_250045)
     4. [Interface pour générer un PDF 36](#_TOC_250044)

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 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 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. Répartition 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 son profil » 28
  26. Diagramme de classe du Sprint 1 29
  27. Diagramme de séquence « Consulter liste employés » 30
  28. Diagramme de séquence « Consulter son profil » 33
  29. Interface de récupération de la liste des employés 34
  30. Interface de recherche d’un employé par un seul critère de recherche 35

3.11 Interface détails employé 36

* 1. Interface pour générer un PDF 37
  2. Répartition des cas d’utilisation par acteur 40
  3. Diagramme de cas d’utilisation Sprint 2 41
  4. Diagramme de classe du Sprint 2 43
  5. Diagramme de séquence «S’authentifier» 44
  6. Diagramme de séquence « Ajouter employé » 45
  7. Diagramme de séquence « Modifier employé » 46
  8. Diagramme de séquence «Supprimer employé » 47
  9. Interface d’authentification 48
  10. Interface d’ajout d’un employé (1) 48
  11. Interface d’ajout d’un employé (2) 49
  12. Interface de modifier la fiche d’un employé 49
  13. Interface de suppression d’un employé 50
  14. Répartition des cas d’utilisation par acteur 51
  15. Diagramme de cas d’utilisation Sprint 3 52
  16. Diagramme de classe Sprint 3 52
  17. Diagramme de séquence «Demander un congé» 53
  18. Diagramme de séquence «Consulter assiduité» 64
  19. Diagramme de séquence « Gérer pointage » 65
  20. Interface de demande de congé 66
  21. Interface de consulter assiduités avec un champ de recherche 67
  22. Interface de consulter demande de congé (refuser ou accepter) 68
  23. Interface de ajouter un meeting 59
  24. Interface liste ToDo……………………………………………………………………………………………..59
  25. Interface Budget ………………………………………………………………………………………………. 60
  26. Interface TimeTraking…………………………………………………………………………………….. 60
  27. Interface des annonces ………………………………………………………………. 61
  28. Interface Setting ………………………………………………………………………….62
  29. Fonctionnalités par acteur du sprint 4 65
  30. diagramme de cas d’utilisation Sprint 4 66
  31. Diagramme de classe Sprint 4 68
  32. Diagramme de séquence « Consulter Dashboard » 70
  33. Interface de Consulter dashboard 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 nos jours, l’évolution technologique joue un rôle trop radical dans les divers domaines d’activités et les entreprises, spécialement dans les secteurs liés à la gestion des ressources humaines. Nécessairement, l’objectif actuel des créateurs et des éditeurs de logiciels est de munir à l’utilisateur un produit de qualité qui garantit une certaine satisfaction, une rapidité une modernité : permet de bien comprendre et aide à une accessibilité plus souple 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 fait, ce n’est pas une mission facile de gérer tous les salariés d’une industrie ou une entreprise quelle que soit sa taille : **Grande entreprise, entreprise de taille intermédiaire ou bien petite et moyenne entreprise**.

Nécessairement, la meilleure solution c’est de tirer profit du progrès technologique par la création d’un logiciel de gestion des ressources humaines en facilitant les taches pour les utilisateurs comme les administrateurs ou les responsables des ressources humaines par la sauvegarde des dossiers électroniques des employés, centralisation des informations et un accès rapide et facile pour le bon suivie des tous les données.

Pour aider les administrateurs des ressources humaines nous allons développer un logiciel de gestion des salariés qui peut être aussi consultés par les employés. C’est dans ce cadre que se déroule notre projet de fin d’étude : "**Conception et développement d’une application de gestion des ressources humaines**".

Notre rapport expose les différentes étapes de la réalisation de ce projet s’étalant sur cinq chapitres. Dans le premier chapitre "Etude du projet" nous allons présenter le cadre général du projet et l’organisme d’accueil. Dans Le deuxième chapitre "Analyse et spécification des besoins" nous allons définir la méthodologie du travail choisie : SCRUM. Nous allons expliquer les besoins

fonctionnels par des diagrammes de cas d’utilisation global et les exigences techniques.

Nous allons dévouer les trois derniers chapitres à la réalisation des sprints.

Notre rapport sera arrêté par une conclusion générale et quelques perspectives éventuelles.

CHAPITRE 1

Cadre général du projet

## Introduction

Notre premier chapitre consiste à mettre notre projet dans son contexte général. En premier lieu, nous parlerons de l’organisme d’accueil. En second lieu, une comparaison des solutions existantes aura lieu. Ensuite, nous expliquerons le cadre du projet, les technologies pratiquées et la méthodologie affiliée pour achever 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 présenter les systèmes des gestions des ressources humaines existants afin de déclarer leurs insuffisances et pour énoncer des solutions possibles plus robustes afin d’optimiser leurs actions.

La gestion des ressources humaines est l’ensemble des tâches qui adhèrent de gérer des employés au sein d’une entreprise.

En fait, les entreprises sauvegardent les données de ses employés dans des dossiers appelés « Fiches Employé », ces derniers sont sur des papiers et gérées par des individus. Egalement, les collaborateurs sont concernés aussi de munir des papiers pour leurs sociétés : demande congé, déclaration pour une absence ou un retard . . .

Donc, les missions de l’équipe Ressources Humaines (RH) deviennent plus en plus dures si l’effectif des salariés est massif, notamment dossier englobe des dizaines des papiers. C’est pour cette raison que l’entreprise est doit forcément agrandir l’équipe RH pour le bon contrôler de la situation. Par conséquent ses dépenses seront augmentées.

Aussi, on peut remarquer que l’apparition du la digitalisation résilier beaucoup des conflits des gestions des employés comme l’utilisation des fichiers Excel, néanmoins ce n’était pas la meilleure solution puisque c’est difficile de traiter des données textuelles dans un fichier Excel.

Egalement, nous pouvons citer deux majeur problèmes : celle de la confidentialité puisque nos données sont accessibles par les divers agents de la société. La deuxième c’est que le responsable RH n’a pas une vision complète étant donné que chaque département a ses propres fichiers.

### Solution proposée

Pour résoudre problèmes déjà cités, notre projet présente une solution permettant de :

* + - * Centraliser la gestion des employés
      * Gérer les assiduités
      * 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

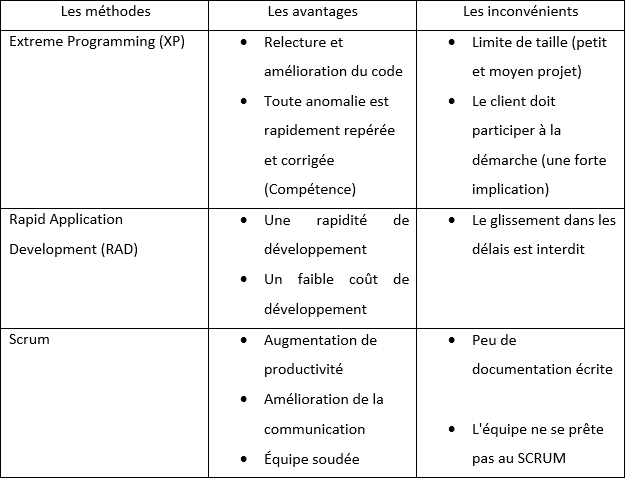
Pour réaliser un projet nous avons besoin de suivre des méthodologies de gestion des projets, absolument, nous allons choisir la méthode selon deux principaux critères : la nature et la taille du projet. Notre choix présente une partie radicale dans la réussite ou l’échec d’un travail.

Si on va aborder des besoins des clients et d’utilisateurs finaux qui sont incomplets ou bien qui changent et évoluent au cours de projet, il est recommandé de se diriger vers une des méthodes itératives, parmi ces derniers on peut citer la fameuse "Agile" qui devise les phases de déroulement des projets (conception, développement, tests, publication) en des petits "sprint" faciles à gérer.

En plus, cette dernière assure la génération des projets de hautes qualités puisqu’ elle se base sur un principe simple : “Gagner plus de contrôle sur le produit final", elle s’adapte aux changements des besoins des clients sans causer la perte du produit.

Plus que ça la méthode Agile nous permet de découper le projet en plusieurs sous-projets, ainsi nous assure des actions correctives sans être risqué de la perte de temps et des coûts, nous savons très bien qu’il impossible de tout estimer et de tout anticiper dès le début. C’est pour cette raison il est conseillé de définir des objectifs à court terme.

Table 1.1 – Tableau comparatif des méthodologies de travail



La méthode la plus adéquate à notre projet c’est Scrum : cette dernière est la meilleur car elle s’adapte avec la complexité du notre projet et sa taille.

Egalement, nous allons exposer l’équipe Scrum ainsi que le cycle de vie de cette méthodologie.

### Processus scrum

C’est le représentant du client dans un projet scrum. C’est dernier représente d’une passerelle entre l’équipe technique et la partie prenante. Il crée « backlog » du produit. Egalement, il est nécessaire la présence des connaissances techniques puisqu’il a pour mission de définir et d’ordonner les « users stories » de chaque sprint.

**Le scrum master** : il joue le rôle de garantir le succès du développement produit

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 :

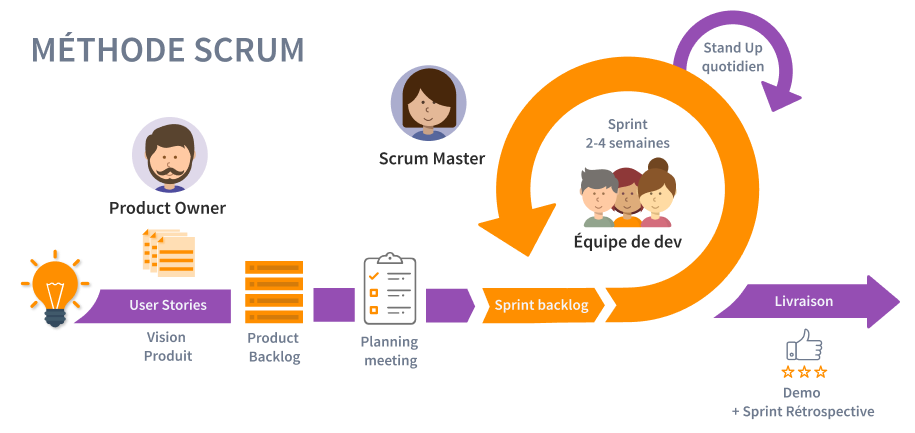


Figure 1.2 – Le principe Scrum

## 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 afin 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.

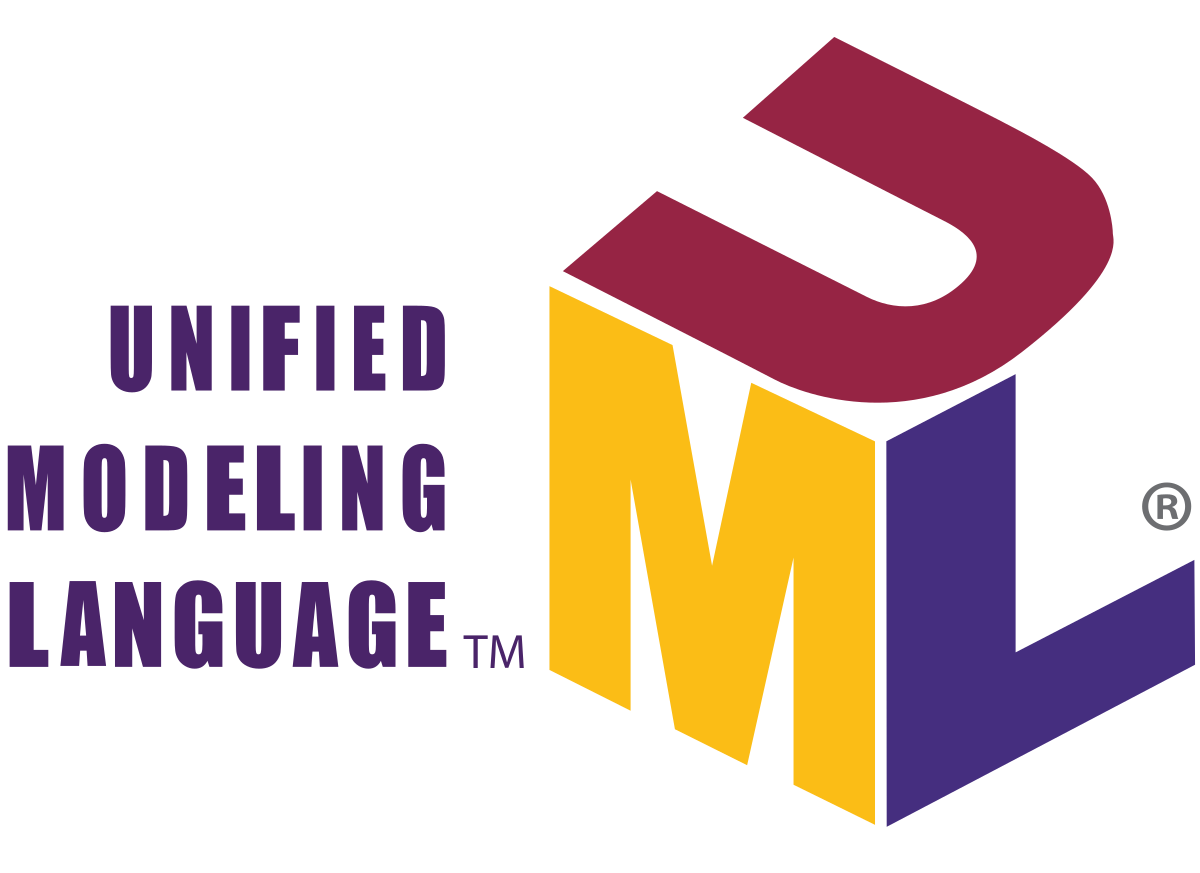


Figure 1.3 – UML logo

**Conclusion**

Dans ce chapitre, nous avons présenté le contexte général du projet, l’organisme d’accueil **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 spécification des besoins.

CHAPITRE 2

Analyse et spécification des besoins

## Introduction

Nous développons ce chapitre pour présenter en premier lieu, l’analyse et la spécification des besoins, l’identification 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 trois acteurs :

* + - * **Responsable RH :** il consulte la liste des employés et il peut les gérer (Ajout, Suppression, Modification).
      * **Administrateur :** il a les fonctionnalités principales de gérer les employés.
      * **Employé :** chaque employé peut accéder à sa zone réservée pour : consulter son profil, demander un congé, faire le pointage, accéder aux annonces.

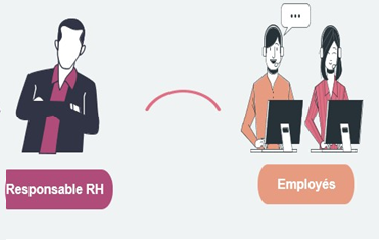


Figure 2.1 – Identification des acteurs

### 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 **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
      * Exporter les informations en forme PDF
      * Gérer les employés : Ajouter, modifier, supprimer un employé
      * Consulter la liste de congé
      * Consulter les demandes de congé : accepter ou refuser
      * Consulter la liste de meeting : ajouter, modifier ou supprimer
      * Accéder la liste à faire
      * Accéder au budget : noter chaque nouvelle dépense
      * Accéder au pointage
      * Publier des annonces
      * Changer ses informations

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 et faire une mise à jour
      * Modifier son mot de passe ou son photo
      * Demander un congé
      * accéder à son état de congé
      * Faire pointage des heures de travail
      * Savoir les règlements de la société.

### 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 moderne et accessible facilement.
      * **Sécurité :** le système doit être sécurisé pour écarter les tentatives d’utilisation inconnue 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

### 2.2.1Diagramme de cas d’utilisation global

Un diagramme de cas d’utilisation global est 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 joue le rôle d’exposer le besoin des utilisateurs d’un système, ils présentent donc une vision orientée utilisateur de ce besoin au contraire d’une vision informatique.[4]

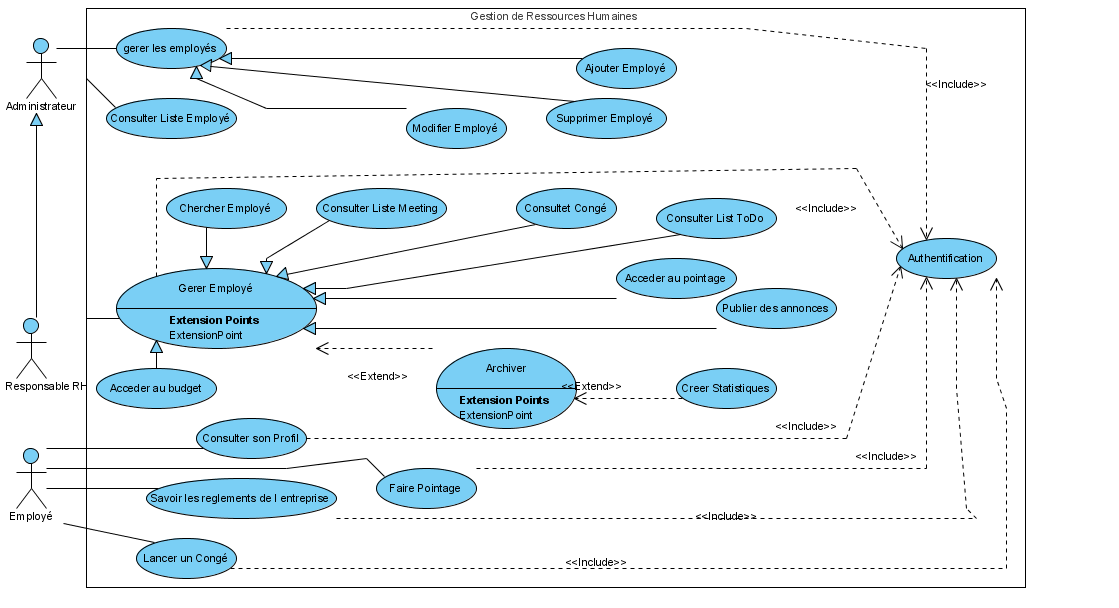


Figure 2.2 – Diagramme de cas d’utilisation global

### 2.2.2 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.

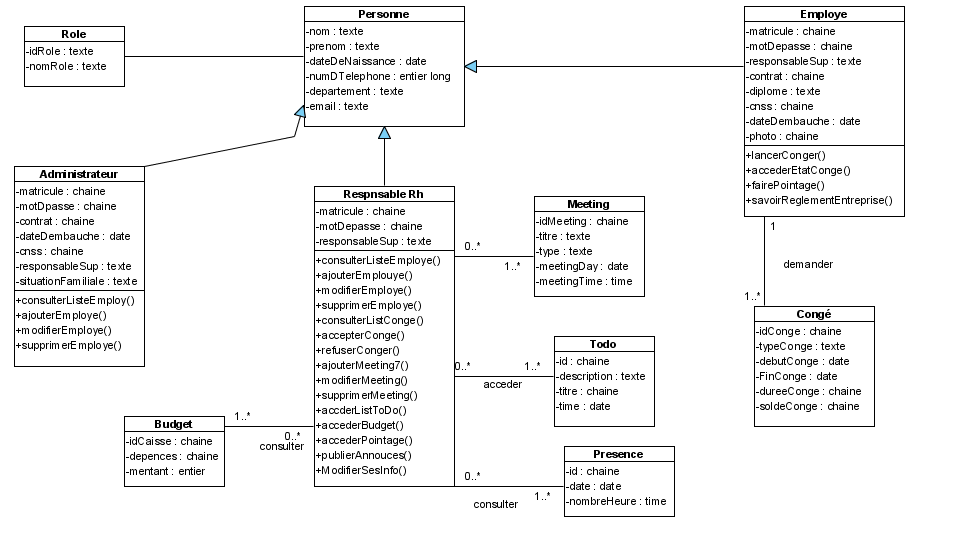


Figure 2.3 – Diagramme de classe global

### 2.2.3BackLog de produit

Le backlog de produit c’est la liste des fonctionnalités d’un produit, il présente les besoins du client que l’équipe doit réaliser, aussi tous les éléments que nécessitent la participation de l’équipe projet. Le product owner s’occupe de classer les éléments par priorité en définissant l’ordre de la réalisation.

Backlog du produit contient les éléments suivants :

* + - * **Features (fonctionnalités) :** représente une partie du projet décomposée en plusieurs user story.
      * **User story (Histoire de l’utilisateur) :** c’est 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.

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

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

|  |  |  |  |
| --- | --- | --- | --- |
| ID-F | Features | ID-US | User Story |
| F1 | ***Gestion des employés*** | 1.1 | En tant que **responsable Rh,** je souhaite consulter la liste des employés |
| 1.2 | En tant que **responsable Rh**, je souhaite faire des recherches multicritères sur les employés |
| 1.3 | En tant que **responsable Rh**, je souhaite ajouter un employé |
| 1.4 | En tant que **responsable Rh**, je souhaite modifier un employé |
| 1.5 | En tant que **responsable Rh**, je souhaite supprimer un employé |
| 1.6 | En tant que **responsable Rh**, je souhaite extraire la fiche d’un employé en format PDF |
| 1.7 | En tant qu’**un administrateur,** je souhaite consulter la liste des employés |
| 1.8 | En tant qu’**un administrateur**, je souhaite faire des recherches multicritères sur les employés |
| 1.9 | En tant qu’**un administrateur**, je souhaite ajouter un employé |
| 1.10 | En tant qu’**un administrateur**, je souhaite modifier un employé |
| 1.11 | En tant qu’**un administrateur,** je souhaite supprimer un employé |
| 1.12 | En tant qu’**un administrateur**, je souhaite extraire la fiche d’un employé en format PDF |
| F2 | ***Gestion des demandes, des déclarations, et des justifications des congés*** | 2.1 | En tant qu**’un employé**, je souhaite demander un congé |
| 2.2 | En tant qu**’un employé**, je souhaite accéder à mon état de congé |
| 2.3 | En tant qu’**un employé,** je souhaite faire pointage des heures de travail |
| 2.4 | En tant qu’**un employé**, je souhaite savoir les règlements de l’entreprise |
| F3 | ***Gestion d’état Employé,***  ***Etat meeting, état à faire, budget, pointage, annonces.*** | 3.1 | En tant que **responsable Rh**, je souhaite consulter la liste de demandes congés. |
| 3.2 | En tant que **responsable Rh**, je souhaite affecter un congé |
| 3.3 | En tant que **responsable Rh**, je souhaite refuser une demande de congé |
| 3.4 | En tant que **responsable Rh**, je souhaite ajouter un meeting. |
| 3.5 | En tant que **responsable Rh**, je souhaite modifier un meeting. |
| 3.6 | En tant que **responsable Rh**, je souhaite supprimer un meeting. |
| 3.7 | En tant que **responsable Rh**, je souhaite accéder à la liste à faire. |
| 3.8 | En tant que **responsable Rh**, je souhaite accéder au budget. |
| 3.9 | En tant que **responsable Rh**, je souhaite noter chaque nouvelle dépense. |
| 3.10 | En tant que **responsable Rh**, je souhaite accéder au pointage. |
| 3.11 | En tant que **responsable Rh**, je souhaite publier des annonces. |
| F4 | ***Gestion de profil*** | 4.1 | En tant que **responsable Rh**, je souhaite consulter et modifier mon profil, changer mon mot de passe ou mon photo. |
| 4.2 | En tant **qu’un administrateur**, je souhaite consulter mon profil, changer mon mot de passe ou mon photo. |
| 4.3 | En tant qu’**un employé**, je souhaite consulter mon profil, changer mon mot de passe ou mon photo |
| F5 | Authentification sécurisée | 5.1 | En tant que **responsable Rh,** je veux m’authentifier de façon sécurisé. |
| 5.2 | En tant qu’**un administrateur**, je veux m’authentifier de façon sécurisé. |
| 5.3 | En tant qu’**un employé**, je veux m’authentifier de façon sécurisé. |

Table 2.1 – backlog de notre application

### Structure de d découpage du projet

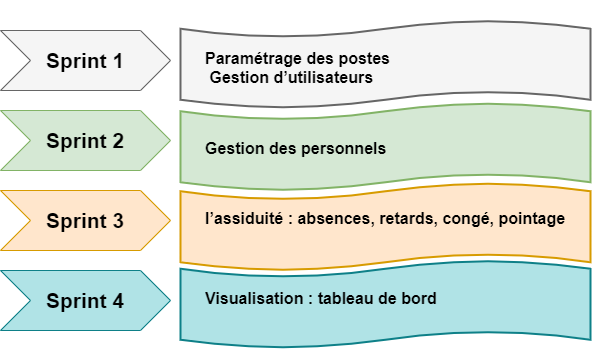
Pour notre projet nous avons décidé de développer quatre sprints comme montré ci- dessus :

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)

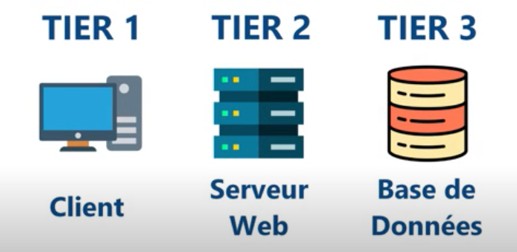


Figure 2.5 – Architecture physique de l’application

**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) :

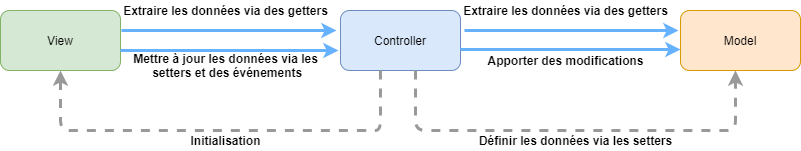


Figure 2.6 – Architecture MVC

**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

### 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.

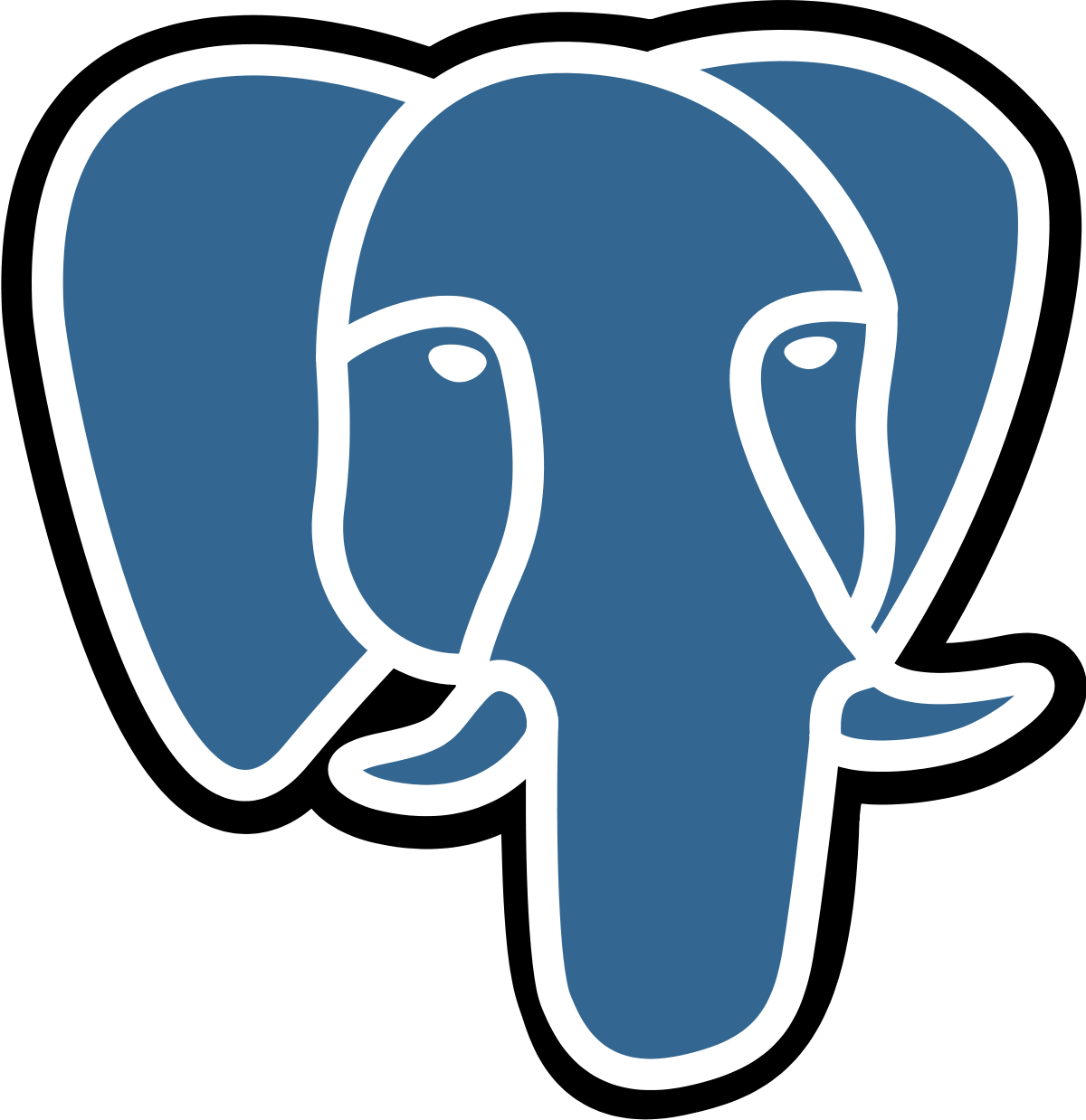


Figure 2.7 – Logo PostgreSQL

#### Editeurs :

**IntelliJ IDEA 2020 [6] :** également appelé « IntelliJ », « IDEA » ou « IDJ » est un environnement de développement intégré (en anglais Integrated Development Environment - 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.

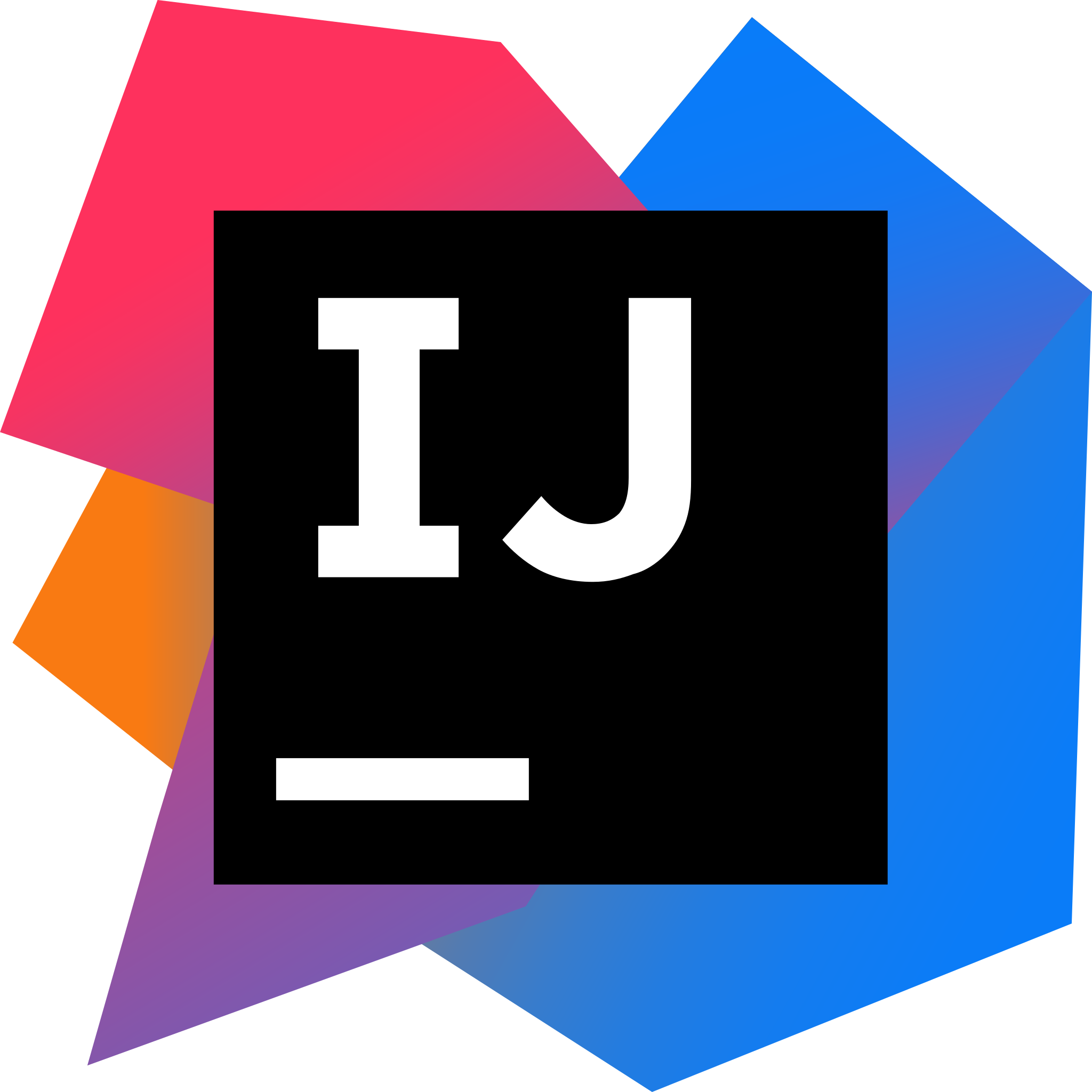


Figure 2.8 – Logo IntelliJ

**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 :**

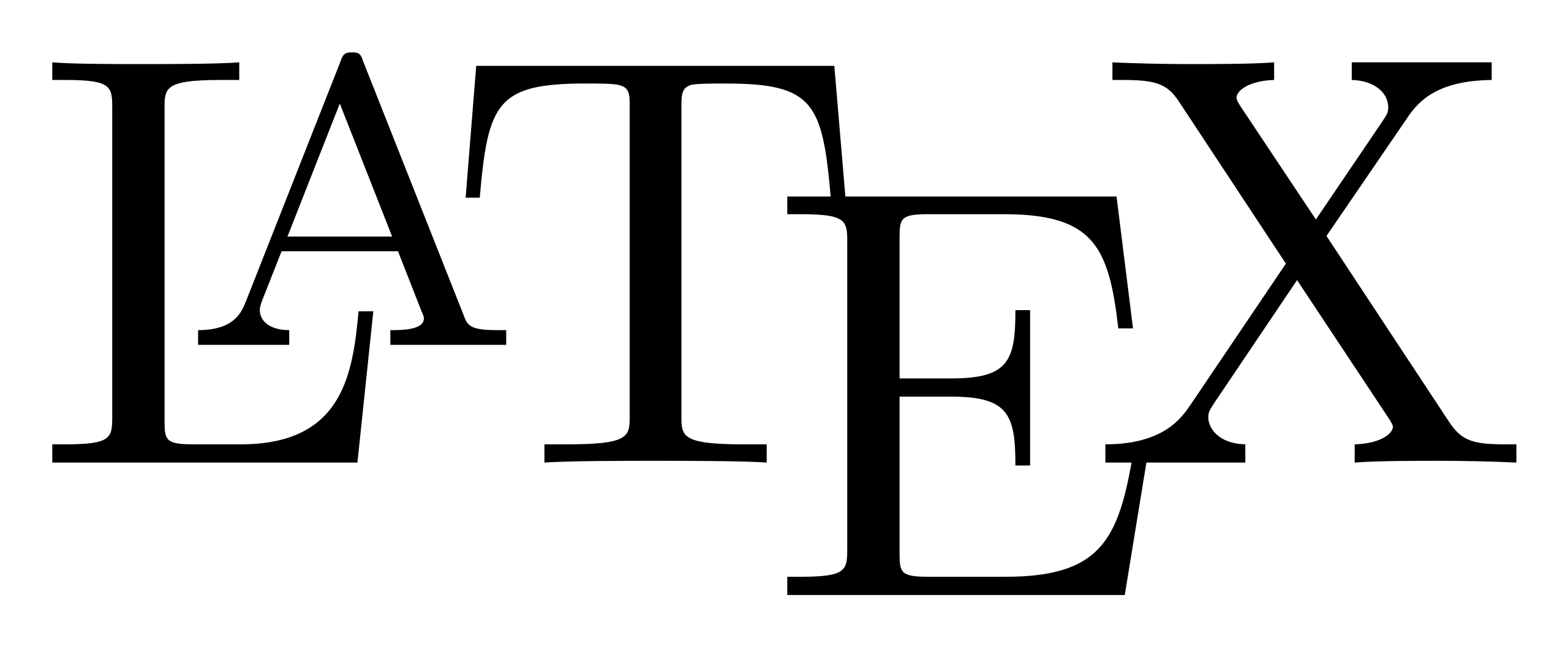


Figure 2.10 – Logo LaTeX

**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 performances 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 disposition 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 codirigé par l’équipe du projet « Angular » à Google et par une communauté de particuliers 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 architecture 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 formulaires, 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 ensemble de composants de haute qualité.

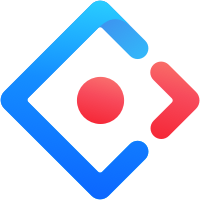


Figure 2.16 – Logo Ant Design

**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 différentes 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’infrastructure 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 configuration 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 application.

CHAPITRE 3

Etude et réalisation du sprint 1

## Introduction

Après avoir analysé les besoins de notre application et présenté nos choix technologiques, 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 diagramme 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.

|  |  |
| --- | --- |
| **Acteurs** | **Cas d’utilisation** |
| Responsable RH | * Consulter la liste des Employés * Faire des recherches multicritères des employés * Voir détails employés * Exporter les informations en PDF |
| Administrateur | * Consulter la liste des Employés * Faire des recherches multicritères des employés * Voir détails employés |
| Employé | * Consulter son profile |

Figure 3.1 – Répartition 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.

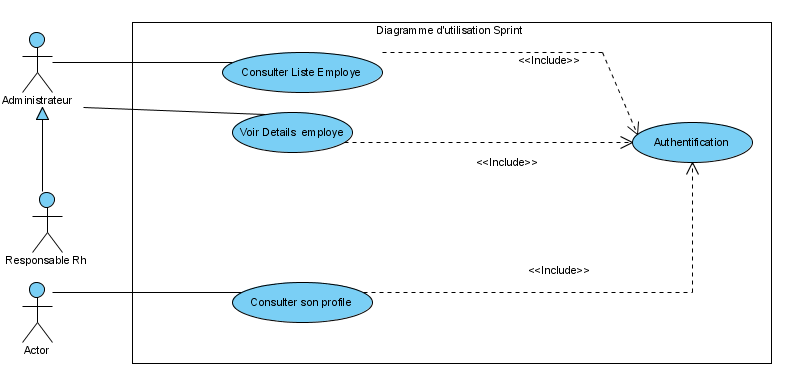


Figure 3.2 – Diagramme de cas d’utilisation Sprint 1

### Description textuelle des cas d’utilisation

Description textuelle du cas d’utilisation **« 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 être affiché.

Raffinement du cas d’utilisation « Chercher Employé »

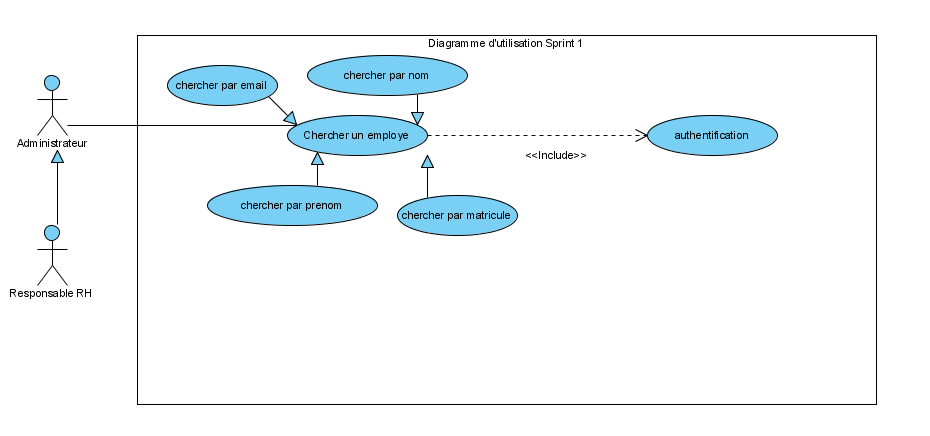


Figure 3.3 – Raffinement du cas d’utilisation « Chercher Employé »

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 y a zone de recherche multicritères

—l’acteur peut effectuer des recherches sur la liste des employés.

**Raffinement du cas d’utilisation «Voir détail employé »**

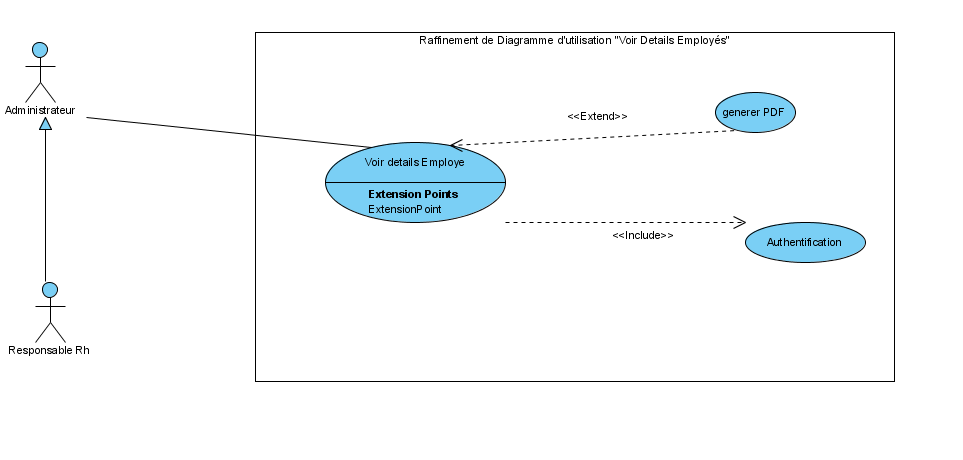


Figure 3.4 – Raffinement du cas d’utilisation « Voir détail employé »

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 champ intitulé "détails employé", l’acteur va cliquer sur le bouton et une fiche d’employé va être affichée.

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

**Raffinement du cas d’utilisation « Consulter son profil »**

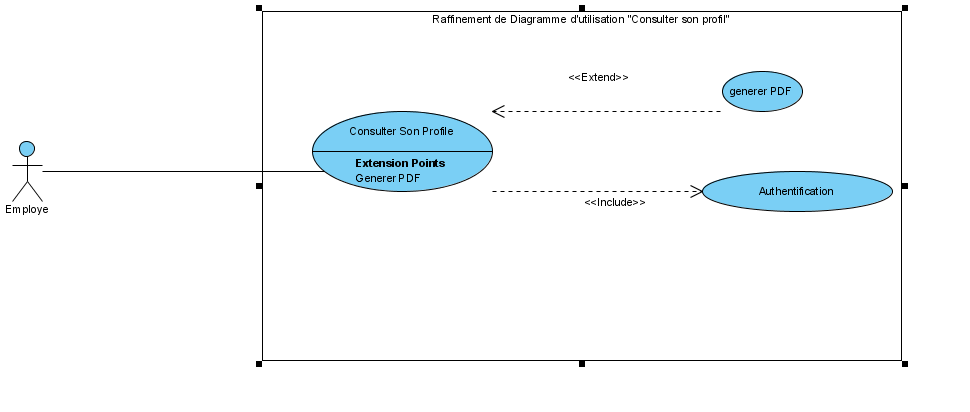


Figure 3.5 – Raffinement du cas d’utilisation « Consulter son profil »

Description textuelle du cas d’utilisation **« Consulter son profil » :**

**Acteurs :** Employé

**Précondition :** s’authentifier

**Postcondition :** Employé authentifié

**Scénario principal :**

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

—Le profil va être 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.

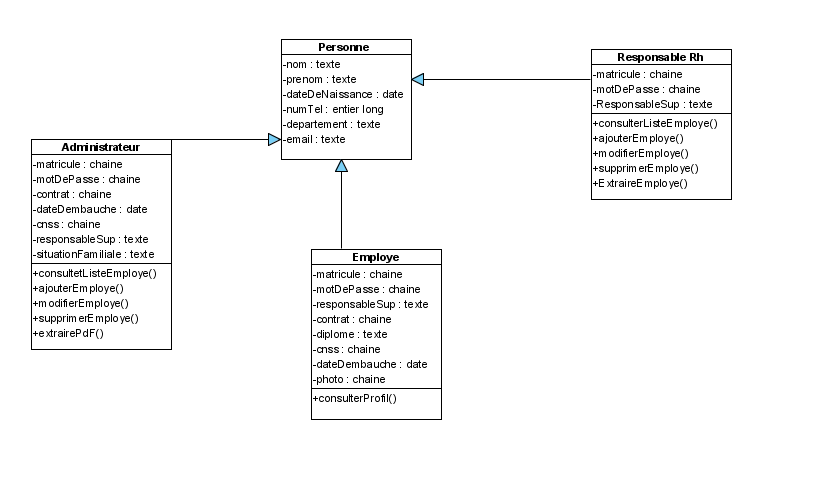


Figure 3.6 – Diagramme de classe du Sprint

### Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les acteurs 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’interface 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.

.

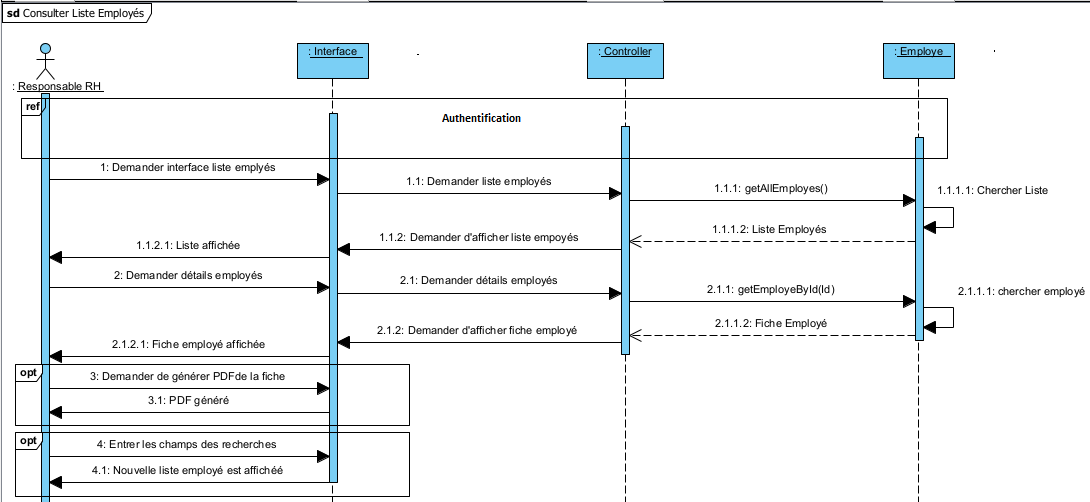
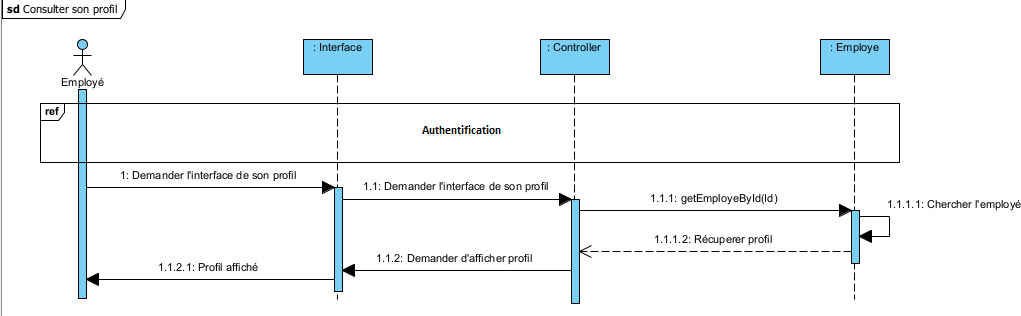


Figure 3.7 – Diagramme de séquence « Consulter liste employés »

* 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 3.8 – Diagramme de séquence « Consulter son profil»

## Réalisation

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

### 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.

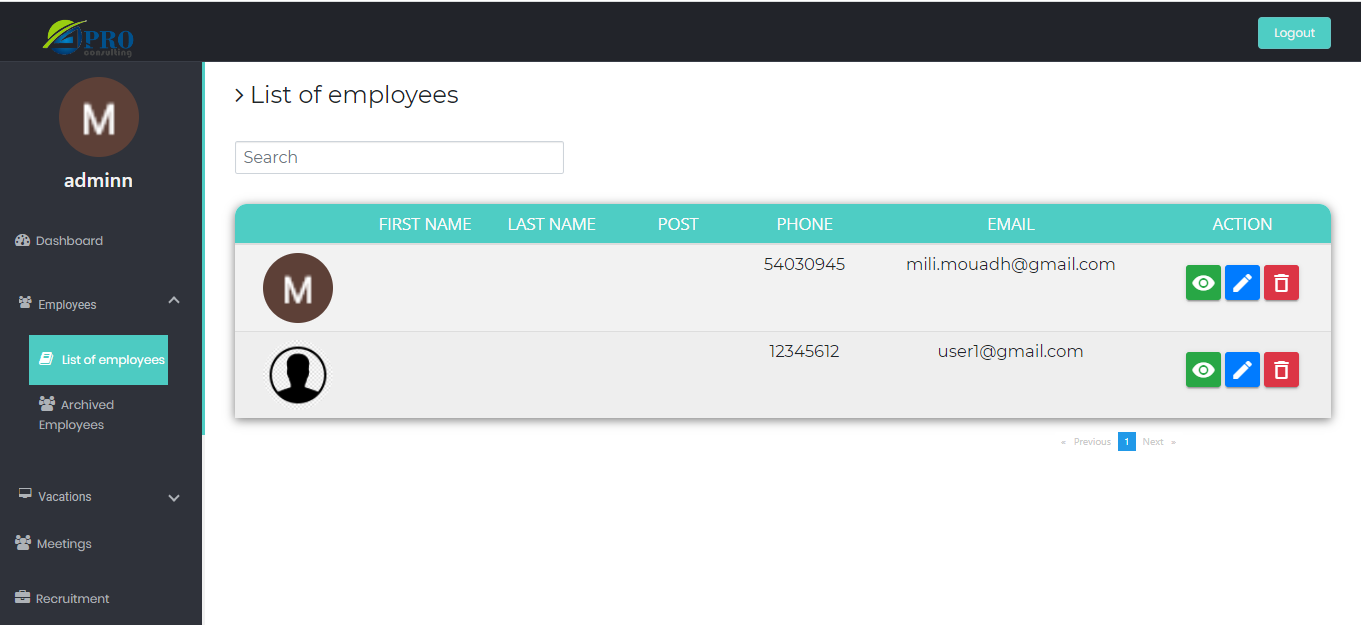


Figure 3.9 – Interface de récupération de la liste des employés

### Interface de recherche d’un employé

On peut effectuer des recherches sur la liste des employés

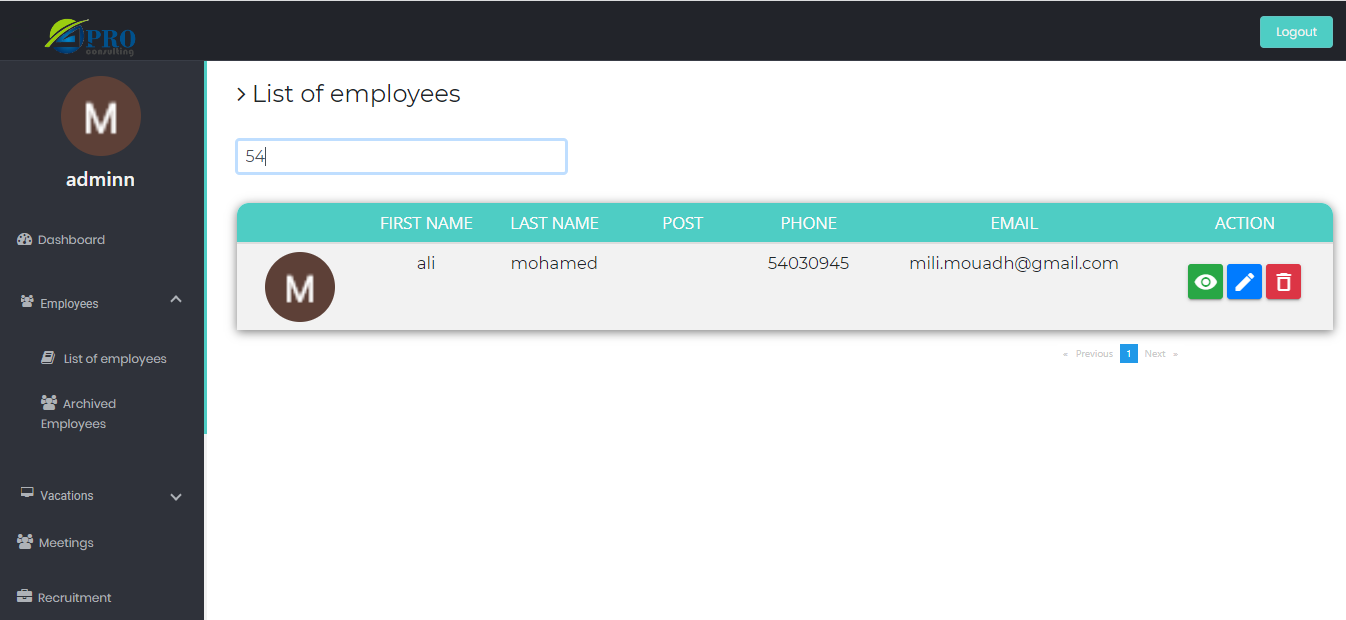


Figure 3.10 – Interface de recherche d’un employé par un seul critère de recherche

### Interface détails employé

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

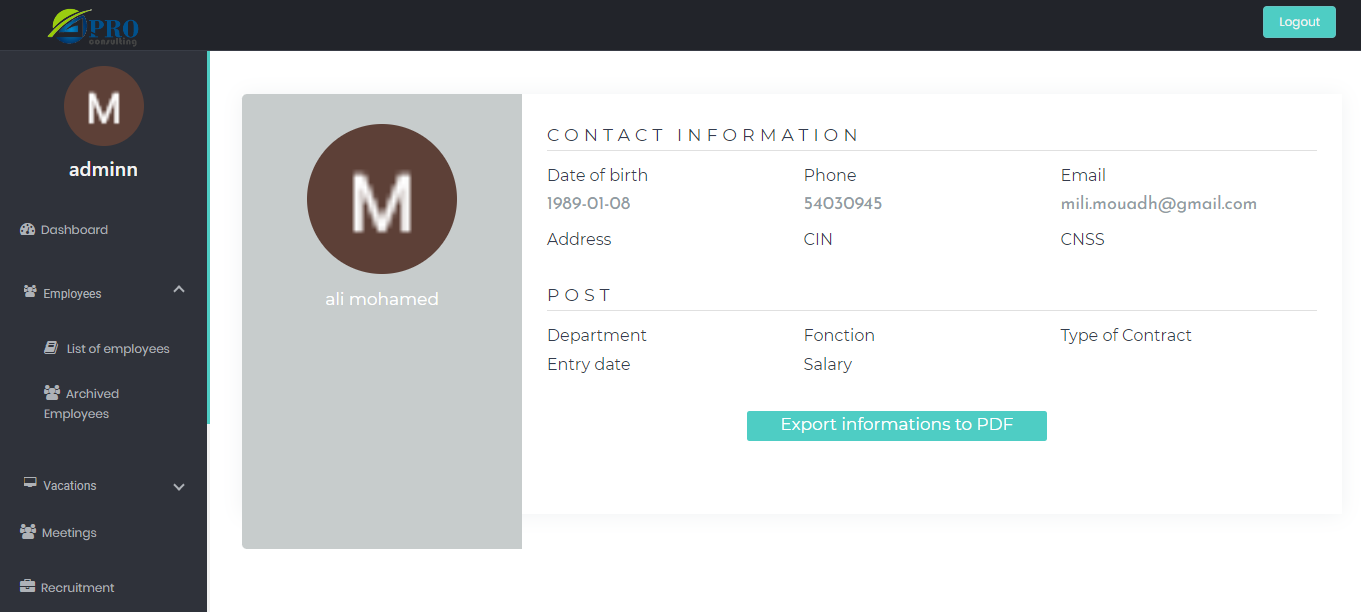


Figure 3.11 – Interface détails employé

### Interface pour générer un PDF

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

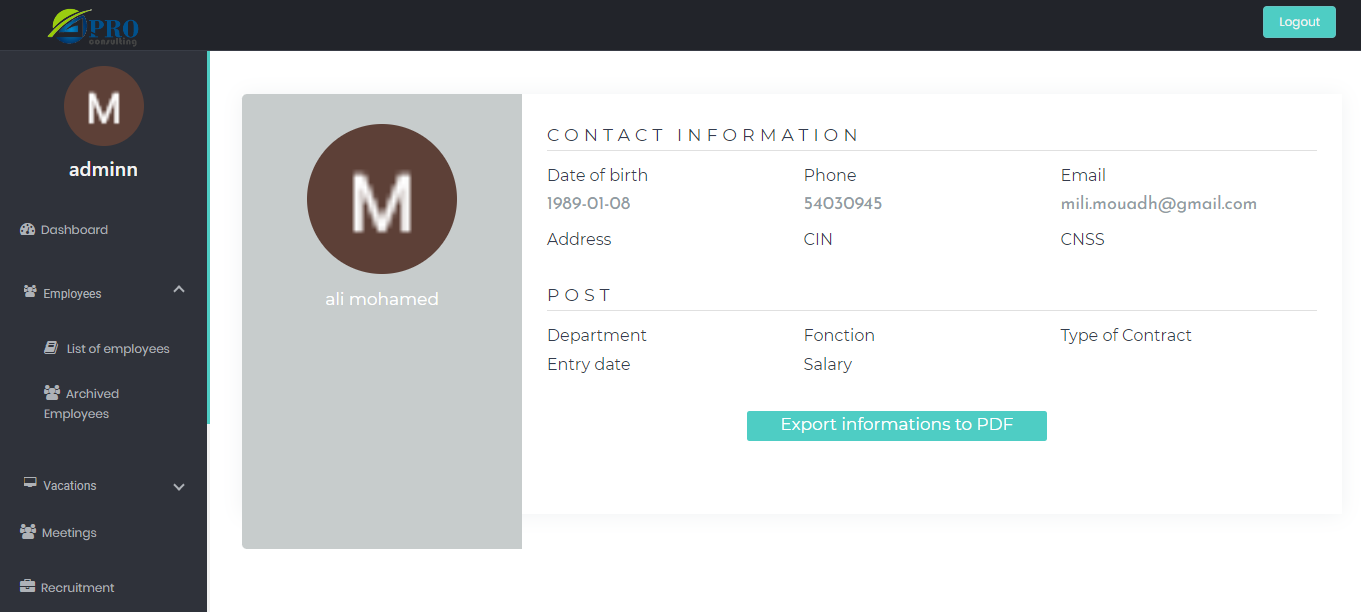


Figure 3.12 – Interface pour générer un PDF

**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.

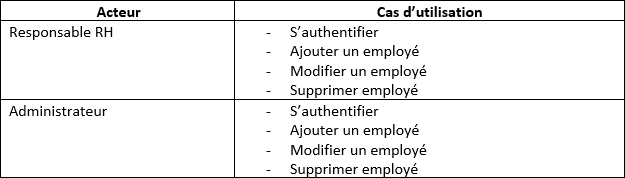


Figure 4.1 – Répartition des cas d’utilisation par acteur

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

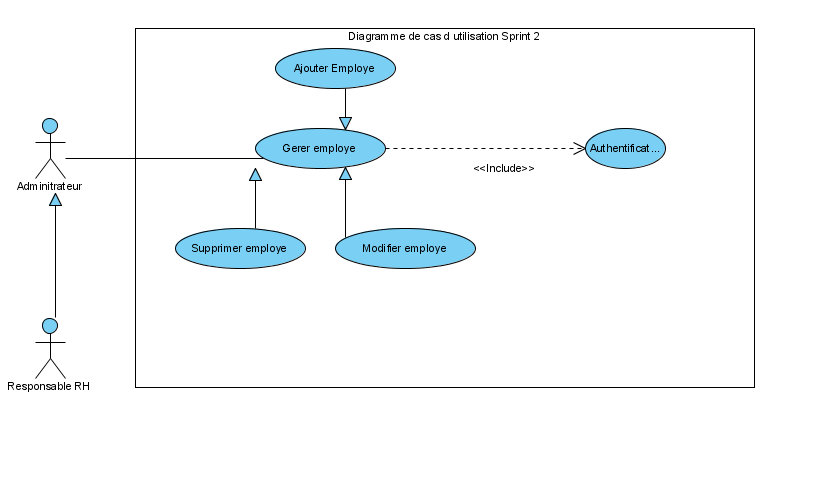
****

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 être 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 être 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 être 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.

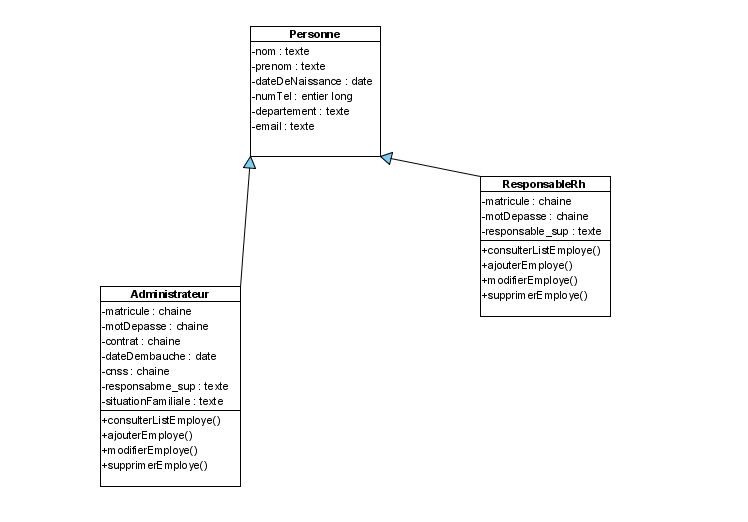


Figure 4.3 – Diagramme de classe du Sprint 2

### Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les acteurs 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»

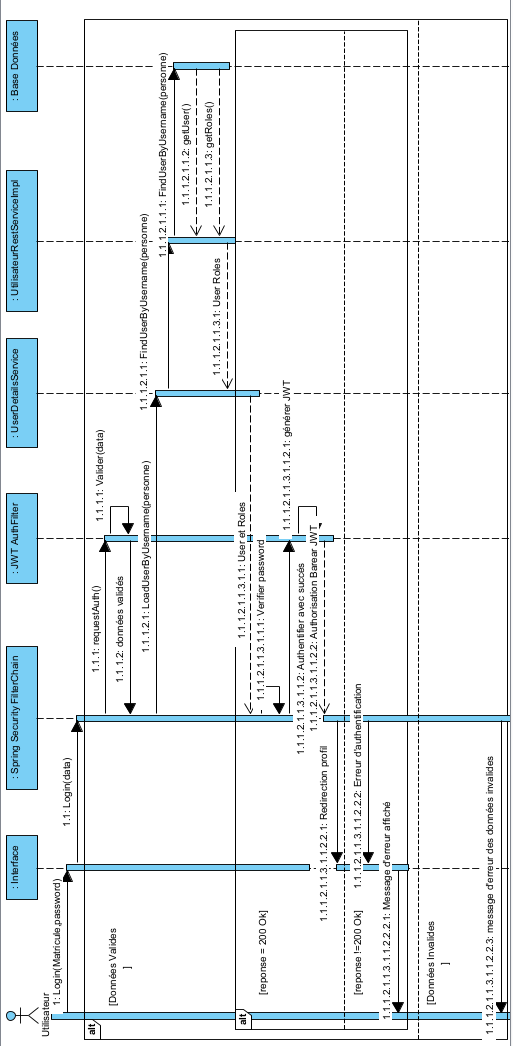


Figure 4.4 – Diagramme de séquence «S’authentifier»

* + Diagramme de séquence « Ajouter employé »

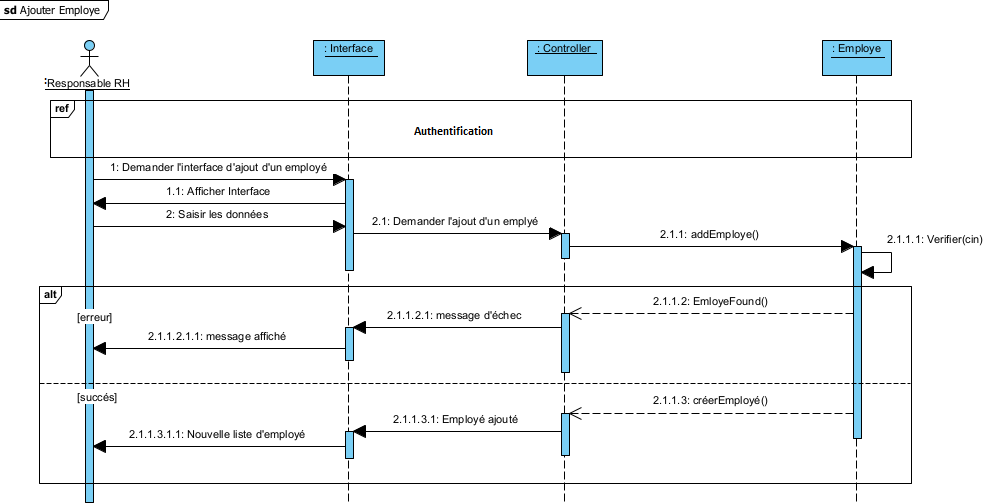
Le responsable RH demande l’interface d’ajout d’un employé, il va accéder à cette interface par le bouton « Ajouter employé » du menu sidebar. Un formulaire d’ajout d’employés va être affiché. C’est le même scénario pour un administrateur.

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». Un formulaire de modification d’employé va être affiché. C’est le même scénario pour un administrateur.

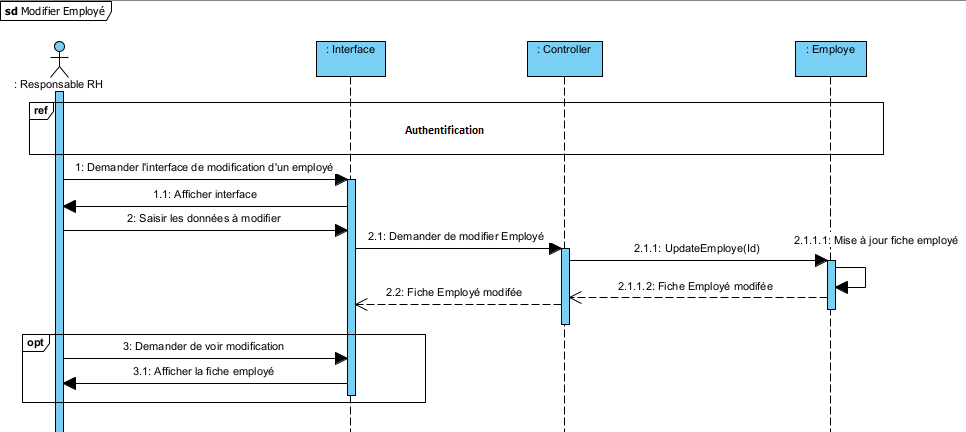


Figure 4.6 – Diagramme de séquence « Modifier employé »

* + Diagramme de séquence «Supprimer employé »

Le responsable RH demande l’interface fiche d’employé, il va accéder à cette interface .Il va cliquer sur le bouton « Supprimer » se trouve à droite de la fiche. C’est le même scénario pour un administrateur.

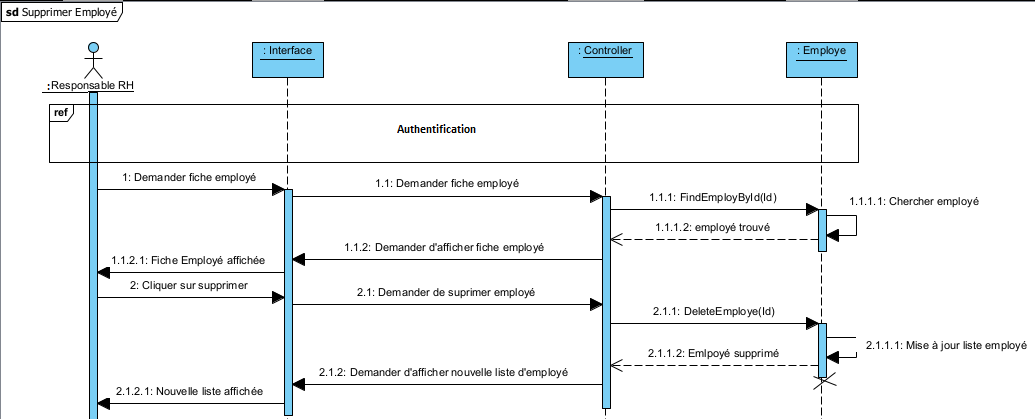


Figure 4.7 – Diagramme de séquence «Supprimer employé »

## 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

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

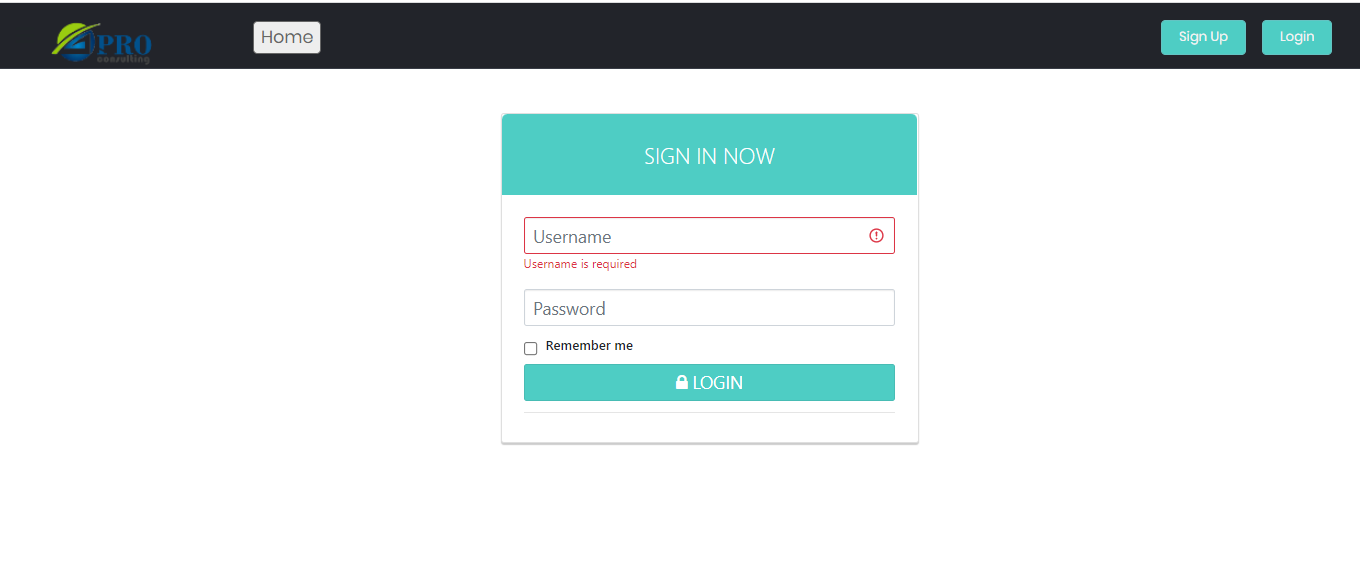


Figure 4.8 – Interface d’authentification

### 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 « sign up ».

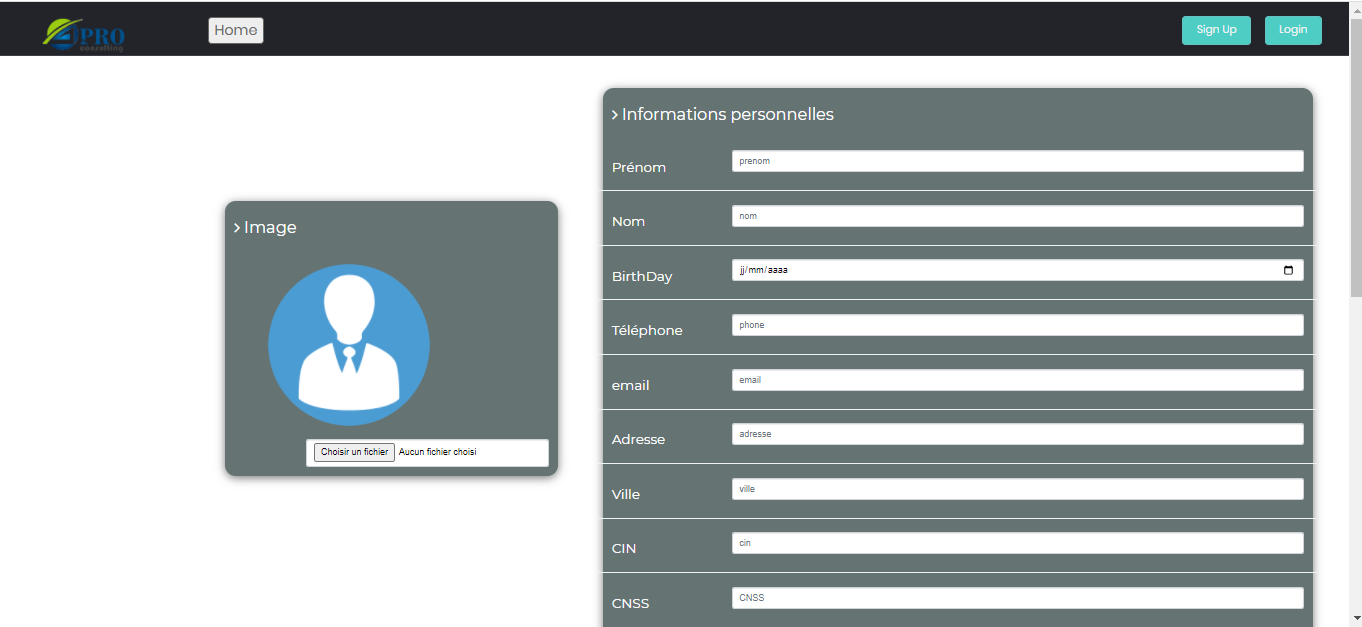


Figure 4.9 – Interface d’ajout d’un employé (1)

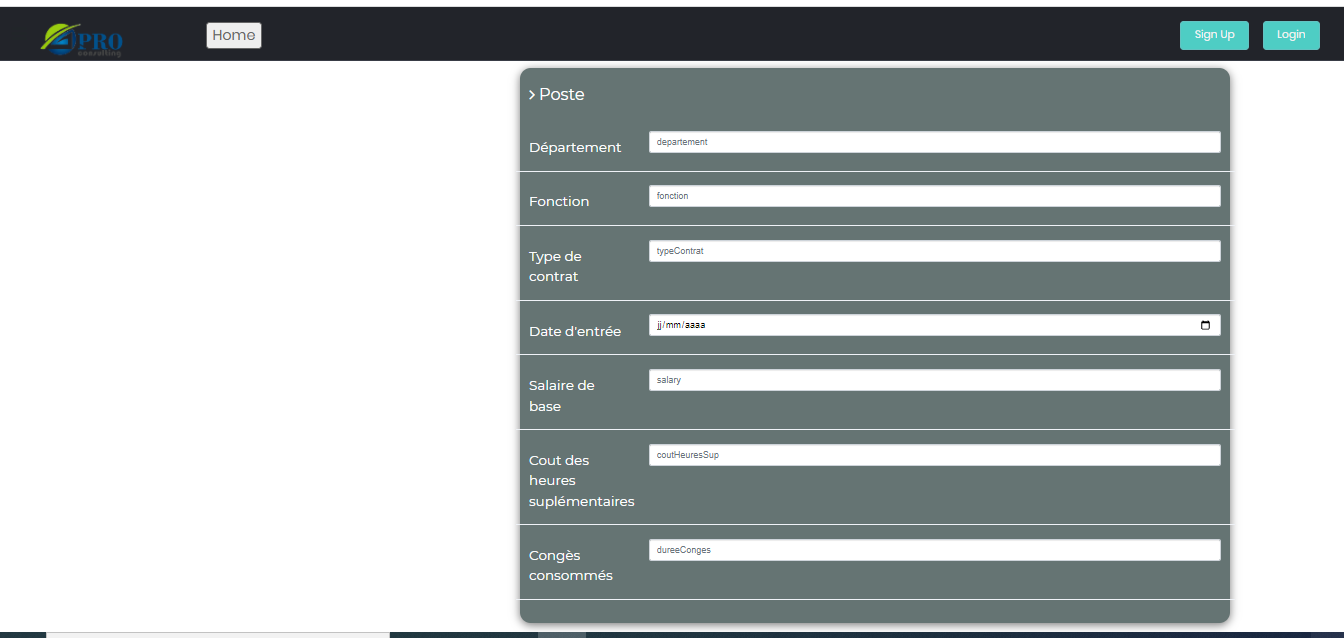


Figure 4.10 – Interface d’ajout d’un employé (2)

### 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 « save ».

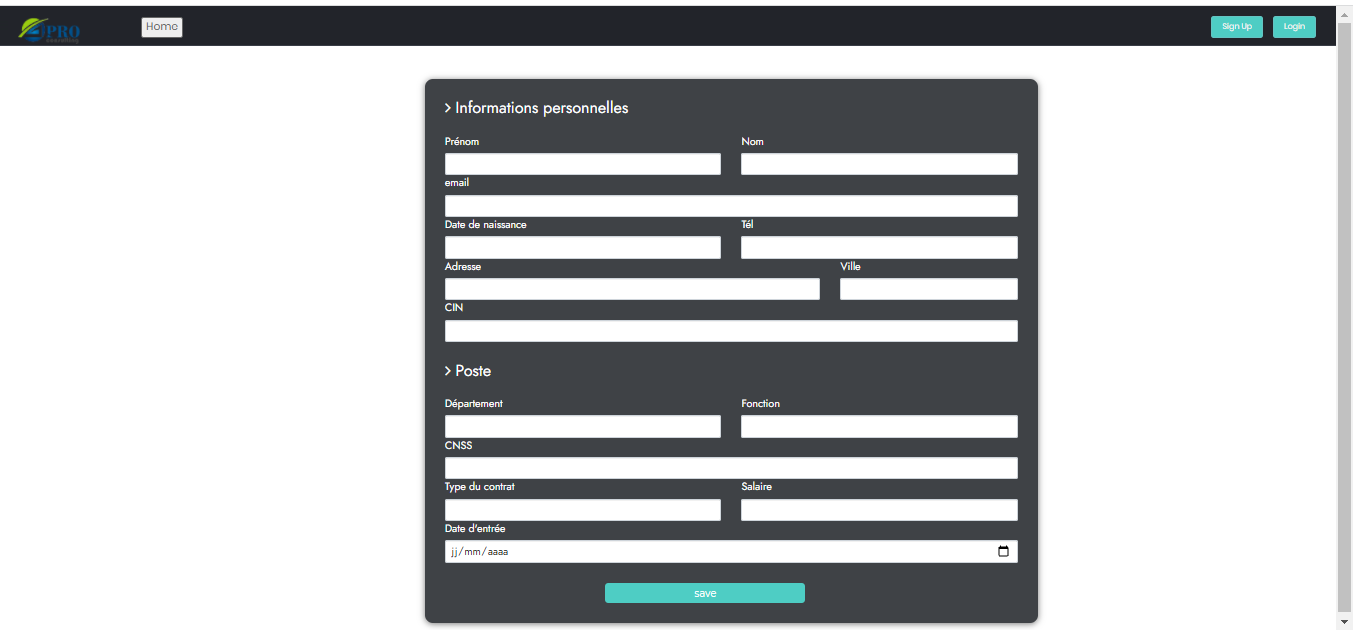


Figure 4.11 – Interface de modifier la fiche d’un employé

### Interface de suppression d’un employé

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

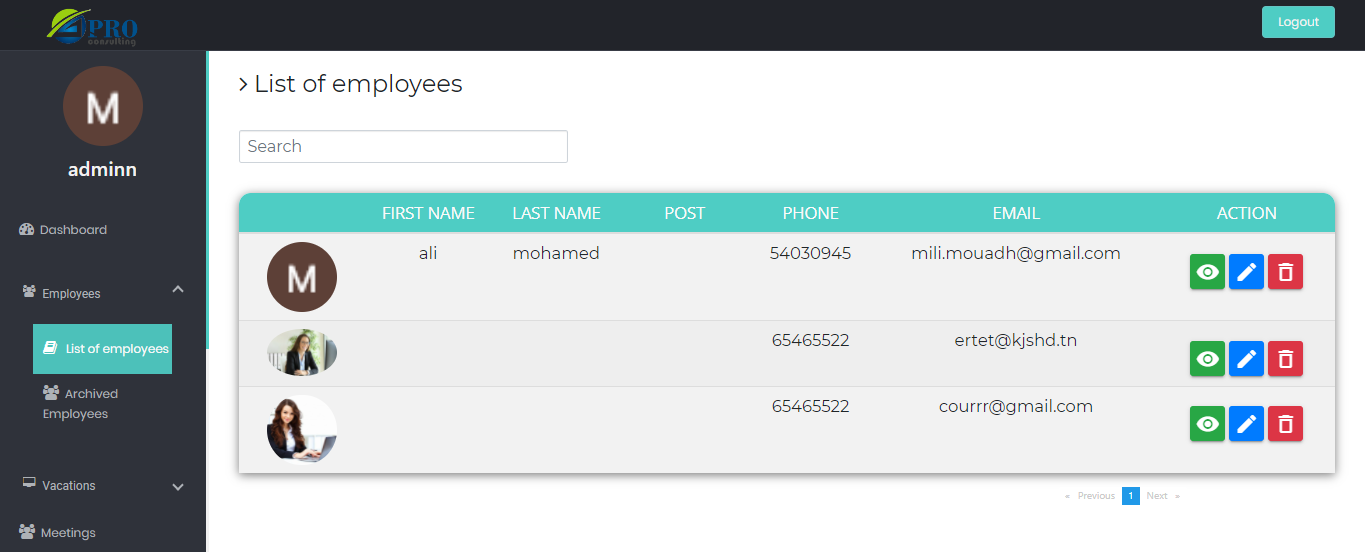


Figure 4.12 – Interface de suppression d’un employé

**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**

|  |  |
| --- | --- |
| **Acteurs** | **Cas d’utilisation** |
| Employé | * S’authentifier * Demander un congé * Consulter liste de ses congés * Accéder au pointage |
| Responsable RH | * S’authentifier * Gérer état employé (congé) * Gérer Meeting * Gérer liste ToDo * Gérer Budget * Gérer pointage * Gérer les annonces |

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

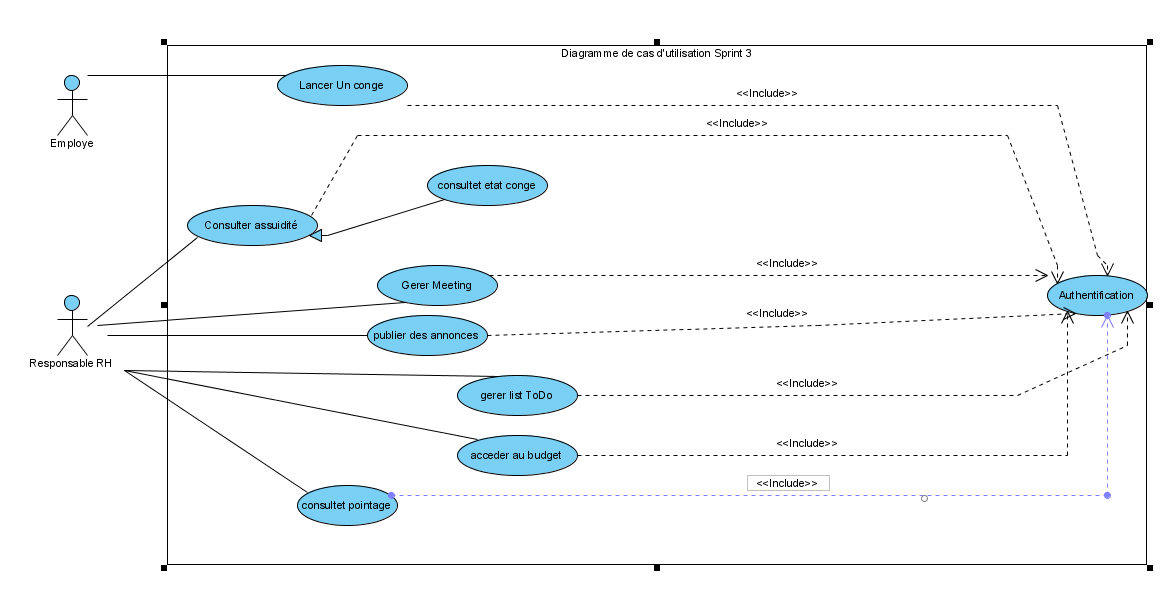
****

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 via notre application ,il peut aussi acceder à sa liste de congé . Le responsable RH a une vision globale donc sur tous type de demandes ou déclarations.il peut, aussi, consulter la liste de pointage et la modifier en cas d’une faute au niveau du pointage. En plus, il peut accéder à la liste de meeting, liste Todo, et les modifier. Egalement il peut accéder au budget et ajouter de dépenses. Ce dernier il a un autre rôle c’est publier des annonces.

### Description textuelle des cas d’utilisation

* + - * Description textuelle du cas d’utilisation **« Demander un congé » :**

**Acteurs :** Employé

**Précondition :** S’authentifier

**Postcondition :** Acteur authentifié

**Scénario principal :**

—L’employé va sélectionner le bouton "request for vacation" du menu sidebar

* + - * Description textuelle du cas d’utilisation **« consulter état employé » :**

**Acteurs :** Responsable RH

**Précondition :** S’authentifier

**Postcondition :** Responsable RH authentifié

**Scénario principal :**

— Responsable RH va consulter la liste de congé.

## 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.

## 5.2.1 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.

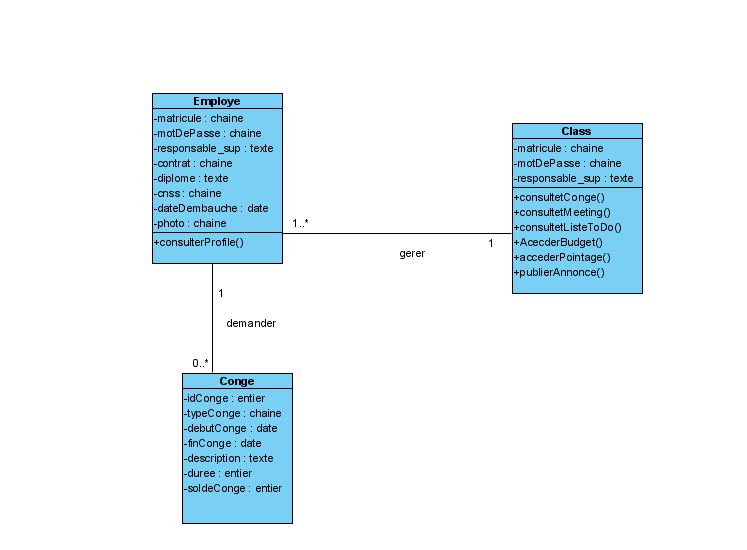


Figure 5.3 – Diagramme de classe Sprint 3

### 5.2.2 Diagramme de séquence

Les diagrammes de séquences présentent graphiquement les interactions entre les acteurs 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.

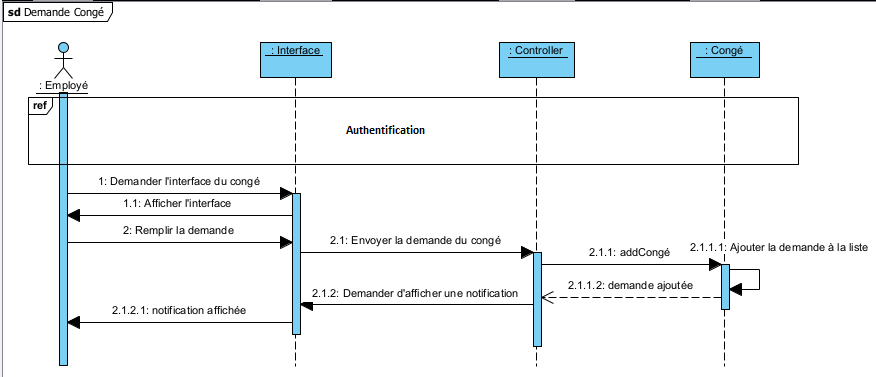


Figure 5.4 – Diagramme de séquence «Demander un congé»

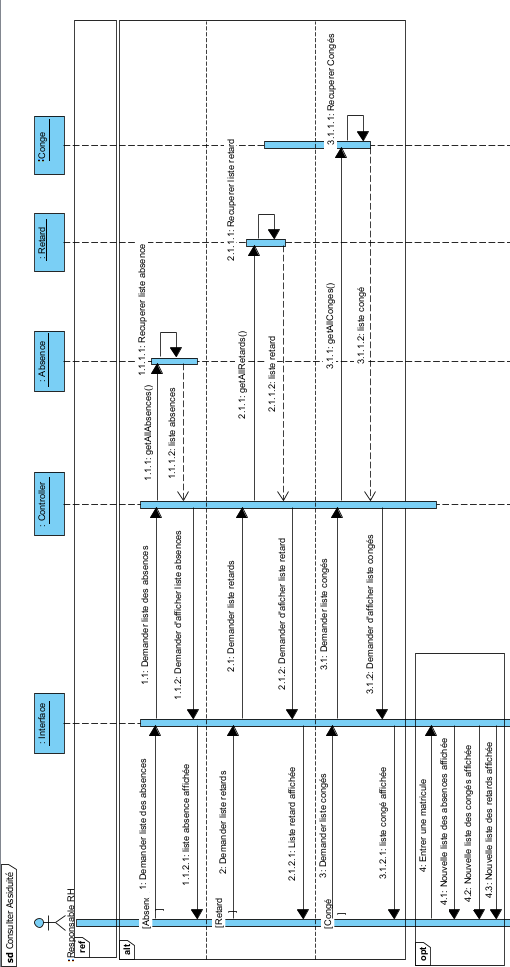


Figure 5.5– Diagramme de séquence «Consulter assiduité»

* + Diagramme de séquence «Consulter assiduité»

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

« Review of vacation» 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.

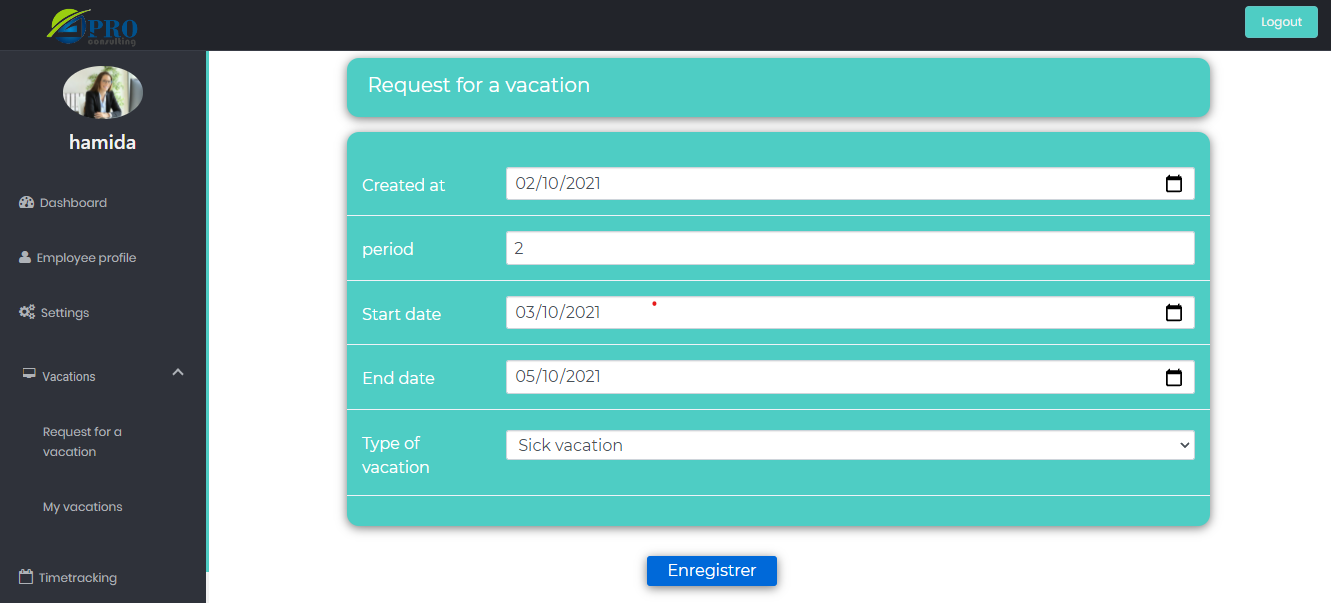


Figure 5.6 – Interface de demande de congé

L’employé peut aussi consulter la liste de ses congés :

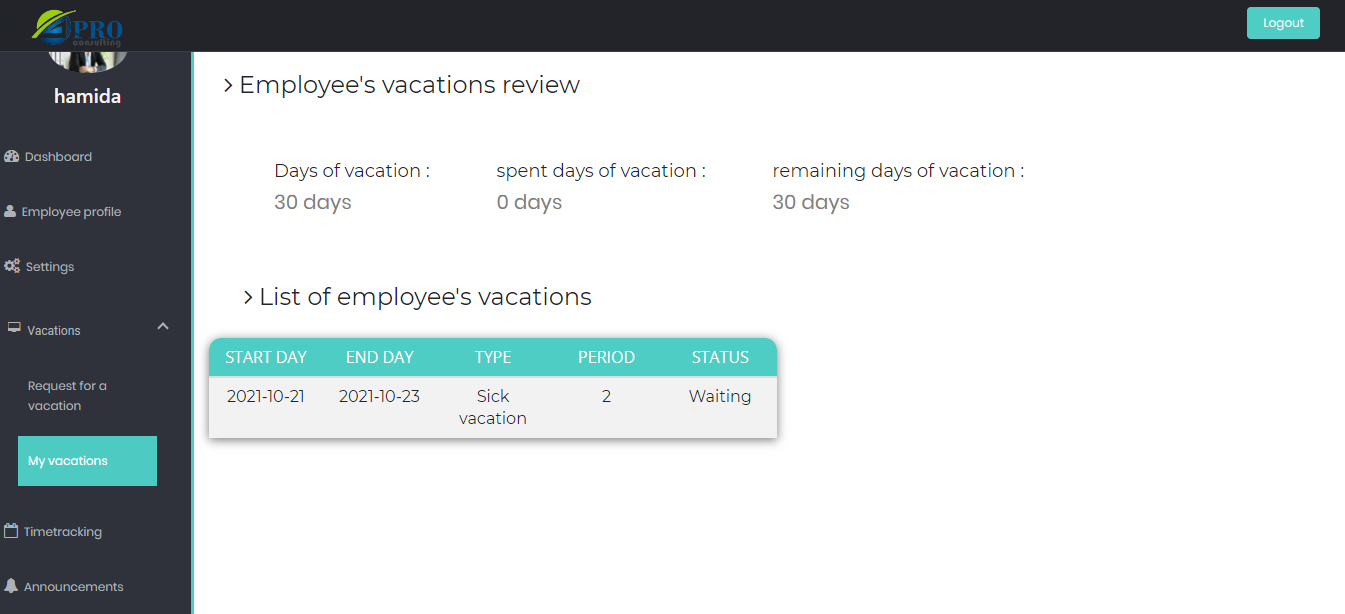


Figure 5.7 – Interface liste de congé

### Interface de consulter assiduité

Le responsable RH peut accéder à l’interface « employees vacations review » pour consulter la liste de demandes du congé. Il peut aussi effectuer des recherches sur ces tables. Egalement le responsable RH peut accepter ou refuser un congé.

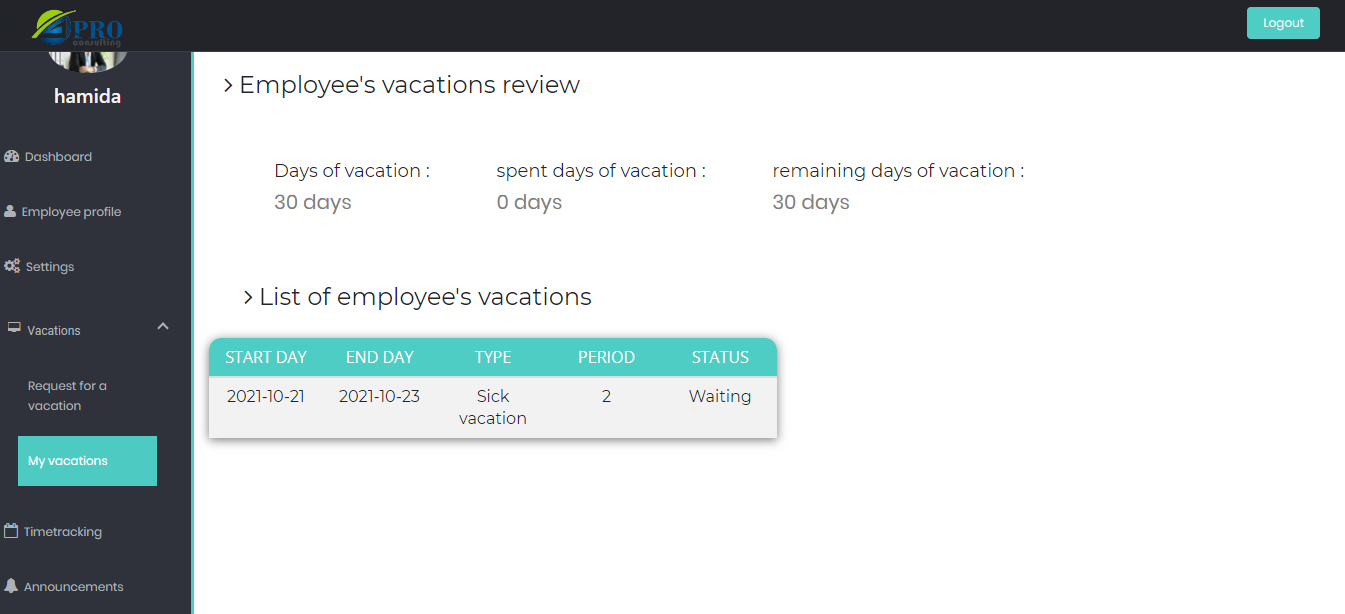


Figure 5.8 – Interface de consulter assiduités avec un champs de recherche

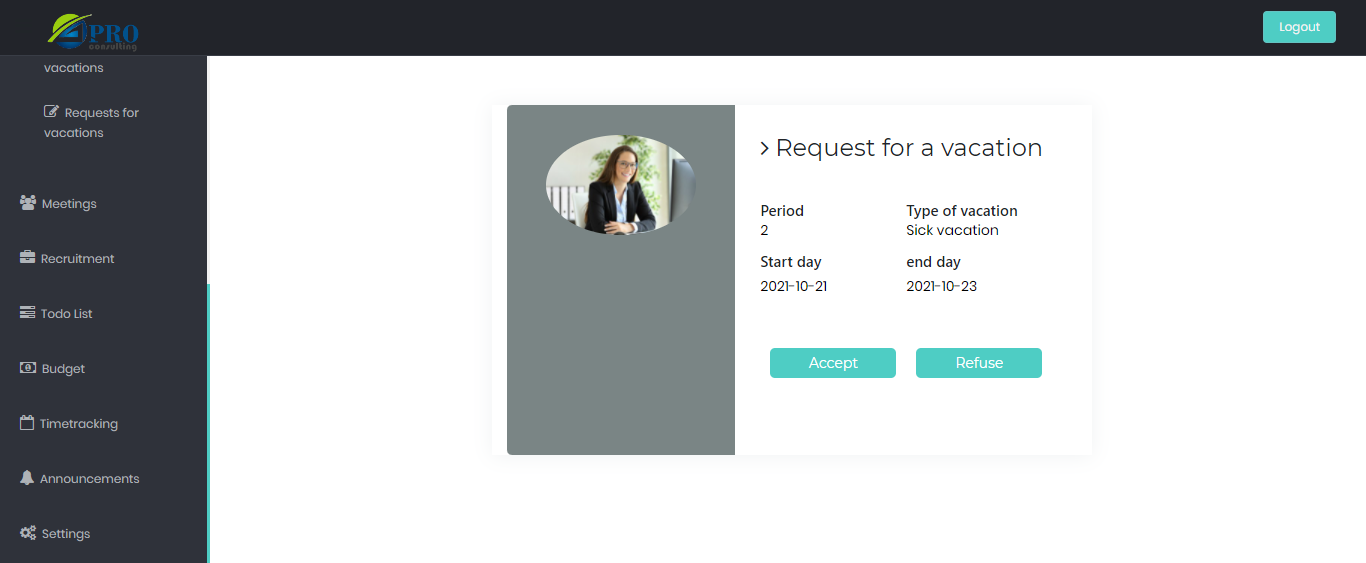


Figure 5.9 – Interface de consulter demande de congé (refuser ou accepter)

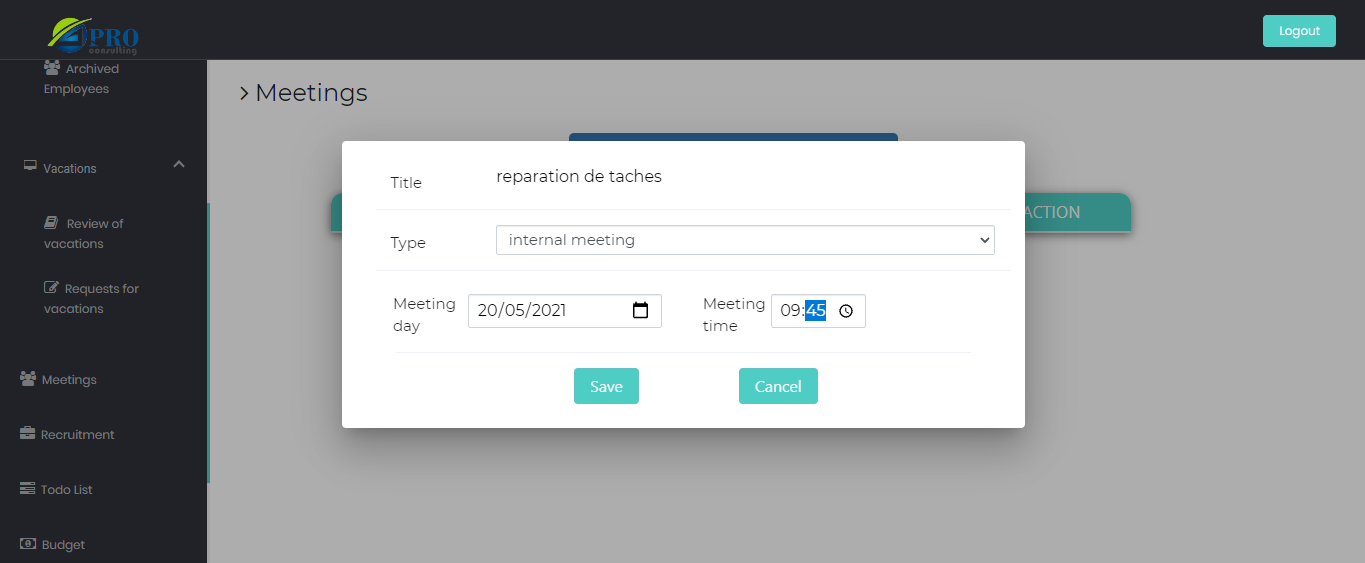


Figure 5.10 – Interface de ajouter un meeting

Le responsable RH peut accéder ajouter, modifier et supprimer un meeting.

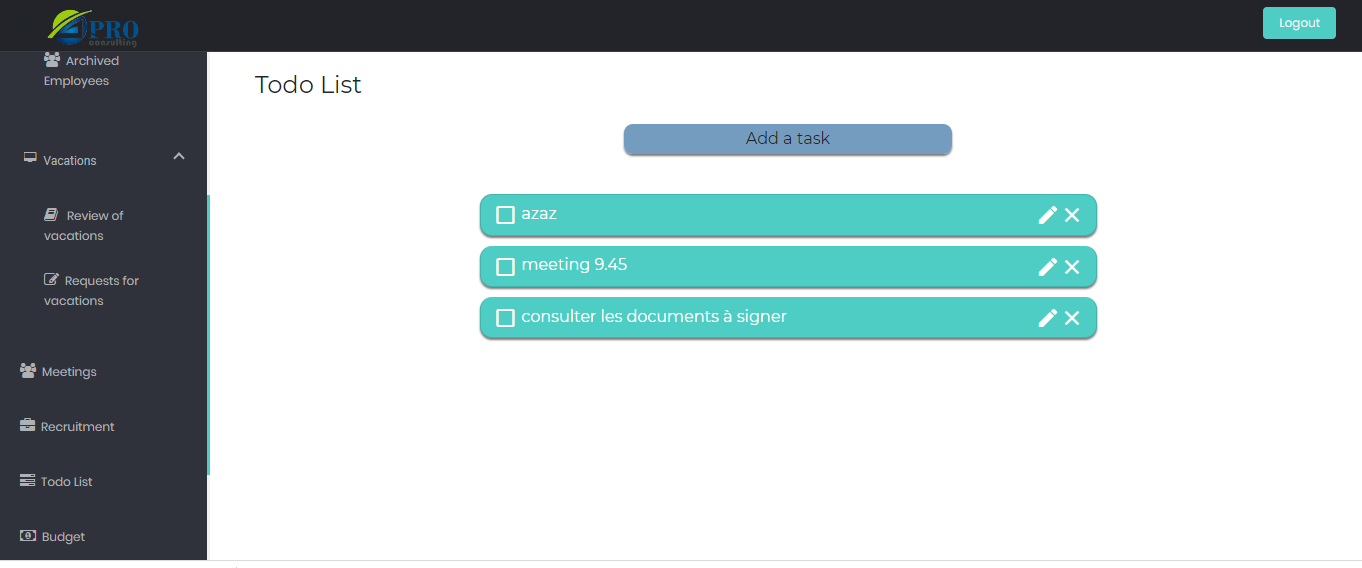


Figure 5.11 – Interface liste ToDo

Le responsable RH peut accéder ajouter, modifier et supprimer liste ToDo

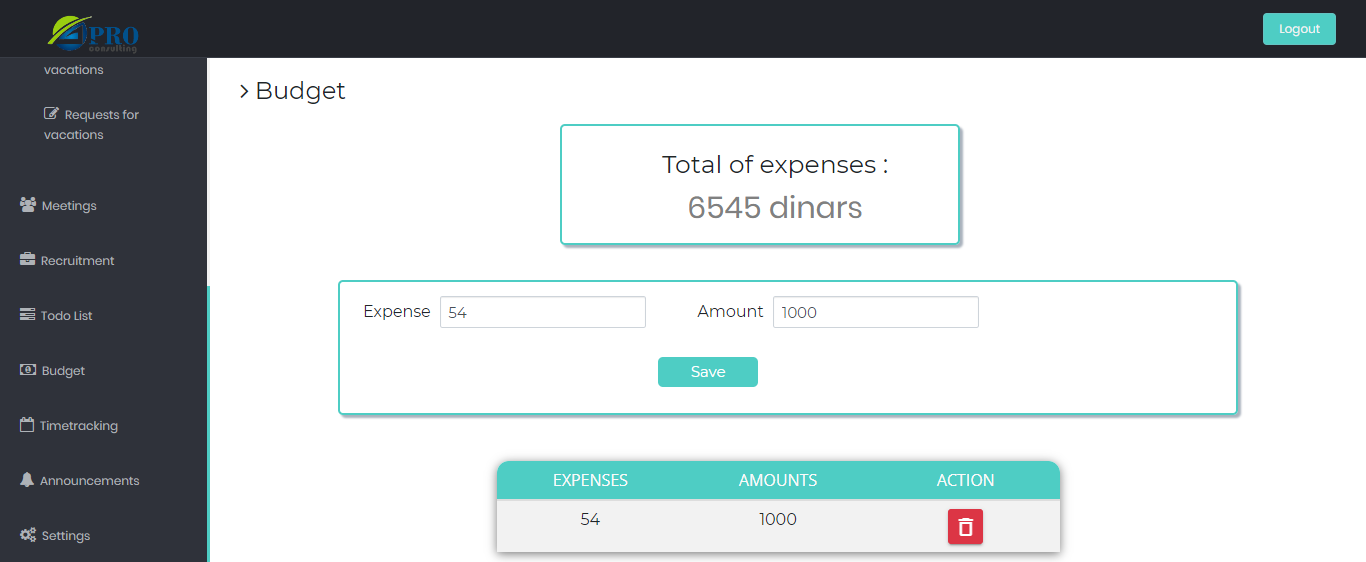
.

Figure 5.12 – Interface Budget

Le responsable RH peut accéder au budget et ajouter des dépenses.

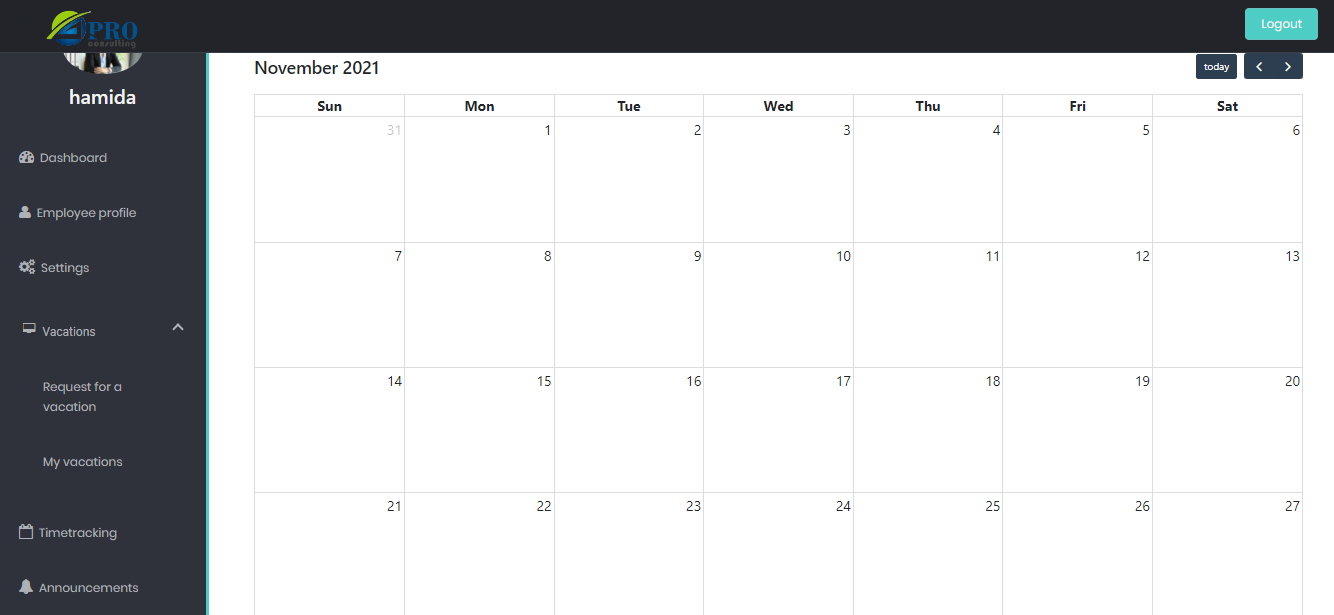


Figure 5.13 – Interface TimeTraking

Le responsable RH peut accéder à l’interface TimeTraking qui liste le bilan des heures de travail pointées par les employés.

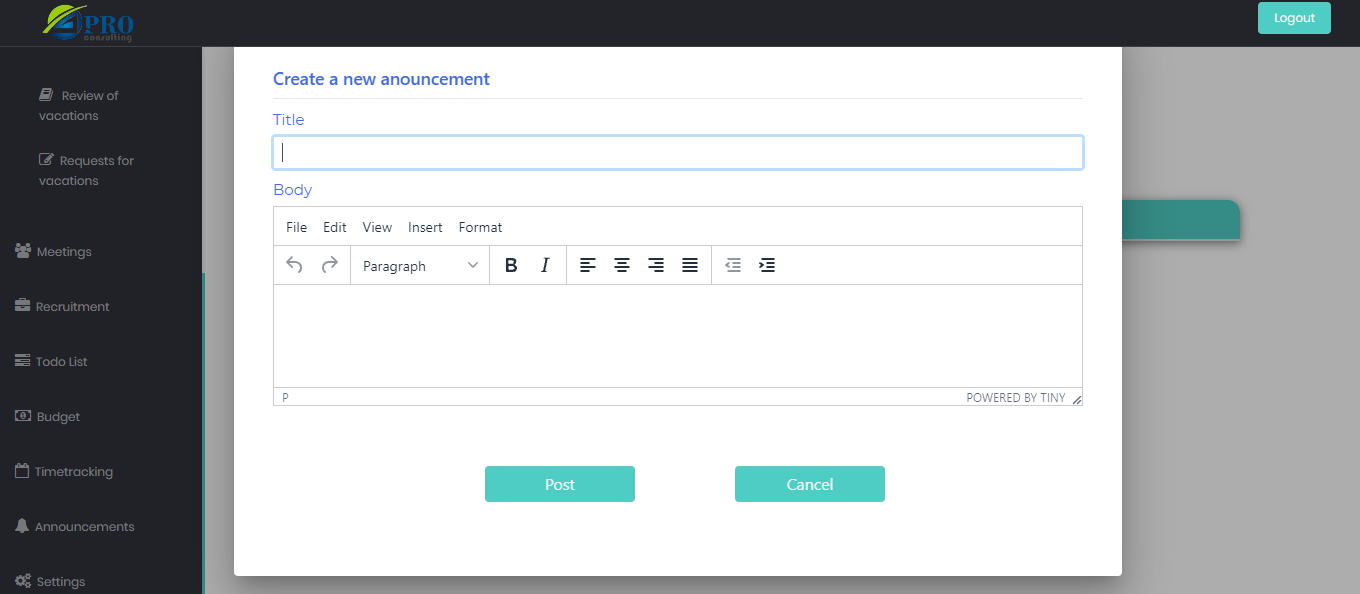
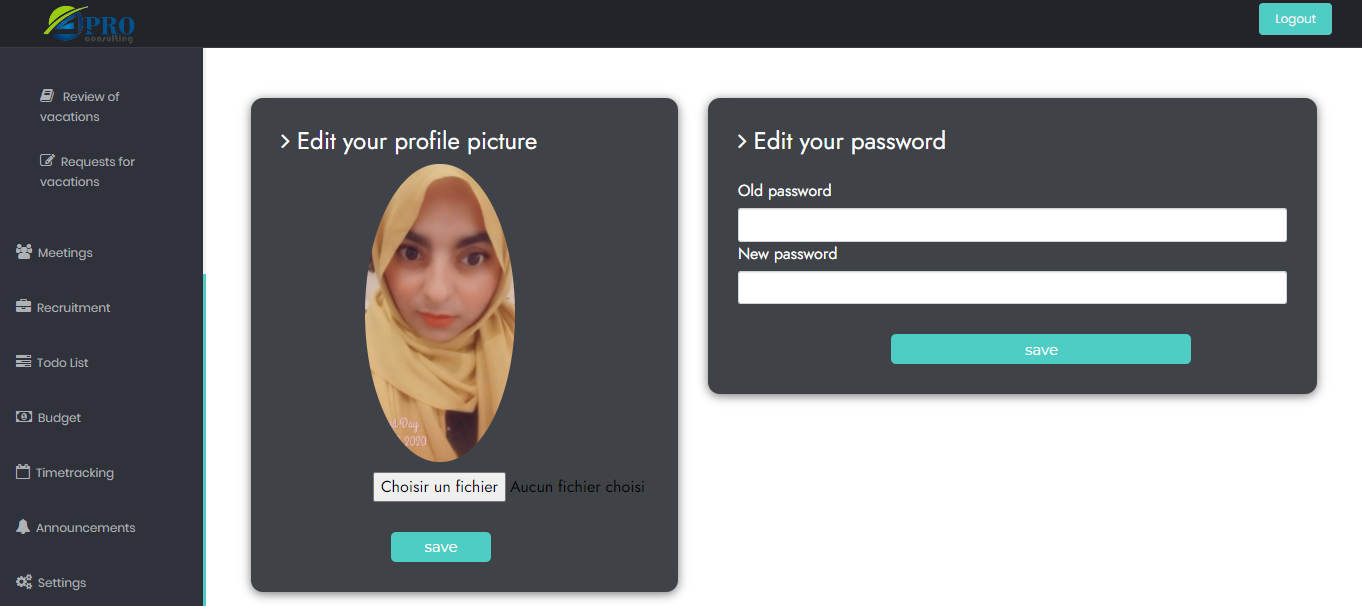


Figure 5.14 – Interface des annonces

Le responsable RH peut à l’interface « «annonces » qui lui permet de publier des nouvelles annonces.



Le responsable RH peut à l’interface « «Setting » qui lui permet de faire un mise a jour de ses informations de compte

Figure 5.15 – Interface Setting

**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.

|  |  |
| --- | --- |
| **Acteur** | **Cas d’utilisation** |
| * Responsable RH | * S’authentifier * Consulter dashboard |

Figure 6.1 – Fonctionnalités par acteur du sprint 4

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

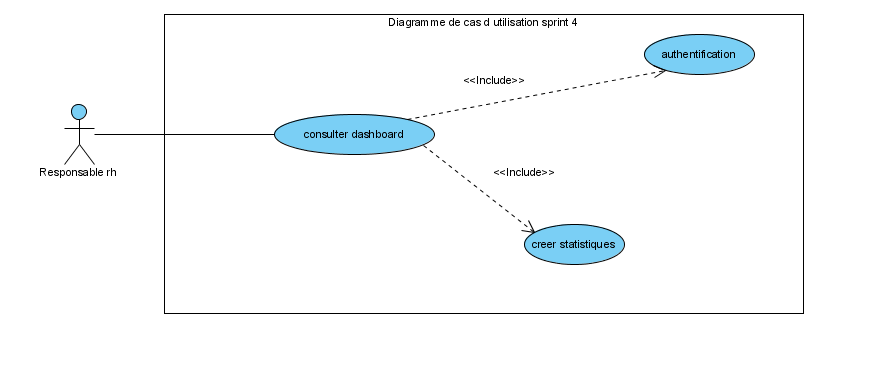
****

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 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 **« 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.

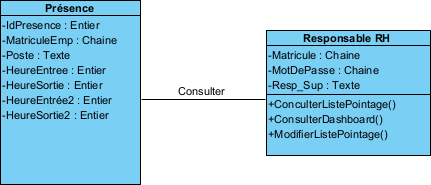


Figure 6.3 – diagramme de classe Sprint 4

### Diagramme de séquence

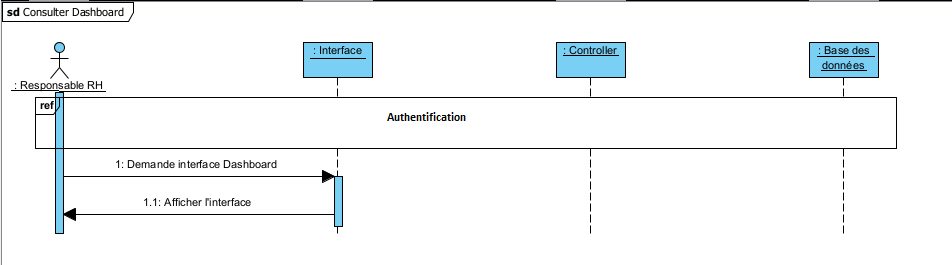
Le responsable RH peut accéder à l’interface du Dashboard, une représentation graphique des données de la société va être affichée.

Figure 6.4 – Diagramme de séquence « Consulter Dashboard »

## Réalisation

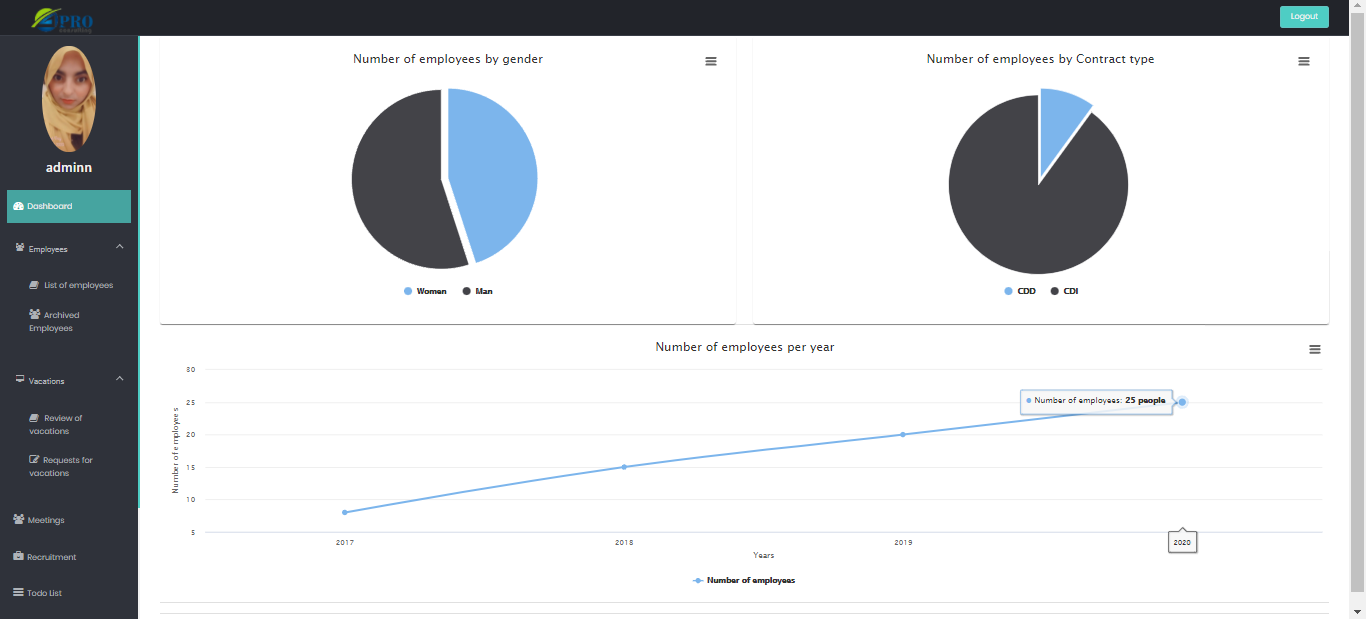
Nous présentons dans cette partie l’interfaces du notre sprint.

### 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 6.5 – Interface de Consulter dashboard



**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’implémentation 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 termes d’amélioration, nous pouvons à titre d’exemple, ajouter des fonctionnalités 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.