

HEYRENDT Maëlle
LE VAN Clélia
SUEUR Nicolas
TISSERAND Mickaël

Rapport de projet

Création d'une application de gestion de congés

Groupe 1
Chef de projet : Clélia Le Van
Année 2019 - 2020

Partie UML	4
Analyse des différents acteurs et besoins utilisateurs	4
Gérer une demande de congés	6
Déposer une demande de congés	7
Administrer une demande de congés par un membre du service RH	9
Gérer une fiche employé	10
Afficher les informations relatives aux demandes de congés	12
Diagramme de classes global de l'application	12
Partie Java Edition Entreprise	15
Mise en place d'une base de données via MySQL	15
Les différentes pages de l'application	18
2.2.1. Différents rôles de l'application de gestion de congés	18
2.2.2. Description des différentes pages	19
Gestion du CSS	20
Mise en place du calendrier des congés	20
Calendrier des employés	21
Calendrier des chefs d'équipe	22
Gestion des statistiques des équipes	23
Jeux de tests	28
3.1 Connexion et affichage de la page d'accueil	29
3.1.1 La vue Employé	29
3.1.2 La vue Employé RH	30
3.1.3 La vue Chef d'équipe	31
3.1.4 La vue Responsable RH	33
3.2 Gestion des employés	34
3.2.1 Création d'un employé	34
3.2.2 Suppression d'un employé	34
3.2.3 Modification d'un employé par un employé RH	35
3.2.4 Modification d'un employé via son profil	36
3.3 Gestion relatives aux demandes de congés	37
3.3.1 Création d'une demande de congés	37
3.3.2 Modification d'une demande côté employé	38
3.3.3 Modification d'une demande côté rh	39
3.3.4. Affichage des statistiques de demandes de congés	40
3.4 Gestion des teams	40
3.4.1 Création d'une team	40
Conclusion	42
Annexes	43

Le Langage de Modélisation Unifié (*Unified Modeling Language* en anglais) est un langage de modélisation graphique permettant d'expliquer les modèles objets à l'aide de différents diagrammes. Depuis sa standardisation en 1997, l'*UML* constitue une étape primordiale lors de la phase d'analyse de la conception d'une application orientée objet. Au cours du module d'*UML*, nous avons pu nous familiariser avec ce langage et ainsi mieux appréhender l'importance d'une étude *UML*.

Ainsi, afin de mettre en pratique nos connaissances acquises en *Java* et en *UML*, nous avons été chargés de concevoir une application de type gestion de congés (que nous avons nommée *DaysOffManager*). Cette application se doit d'être développée à l'aide de *Java Enterprise Edition* (ie *Java EE*). Cette plateforme est une norme qui spécifie l'infrastructure de gestion des applications et les interfaces de programmation (ie API) des services utilisées pour concevoir ces applications.

Avant même de passer au développement de l'application, nous avons donc effectué une étude *UML*, et ainsi, nous avons pu faire ressortir les acteurs de l'application et les différents cas d'utilisation. Cette étude et les modifications apportées lors du développement de l'application vous sont présentées en première partie de ce rapport.

De plus, nous avons pu passer à l'implémentation de l'application à l'aide de *Java EE*. La deuxième partie de ce rapport est donc consacrée à la création de la base de données, les principales fonctionnalités de l'application ainsi que les différents modes d'utilisation.

En dernière partie de ce rapport, vous retrouverez un jeu de tests en vue de valider les fonctionnalités développées ainsi que leurs limites. Un tel test permet également de faire ressortir les potentielles évolutions de l'application.

1. Partie UML

Une étude *UML* est une étape clef dans la phase de conception d'une application de type orientée objets : en effet, il s'agit là de la première partie (ie partie d'analyse) de la phase de conception de ladite application. Composée d'une étude en langage courant et de documents formalisés, cette phase d'analyse permet notamment d'identifier les acteurs du système et l'ensemble des cas d'utilisations.

Il est important de noter que cette étude *UML* se base sur l'identification des acteurs et de leurs besoins. En effet, ce sont les besoins des utilisateurs qui vont permettre de faire ressortir les cas d'utilisation et donc déterminer les réelles fonctionnalités du système.

Cette étude consiste donc en une modélisation de la future application de gestion de congés. En adoptant une démarche orientée par les besoins des utilisateurs du système, différents diagrammes seront utilisés pour faire ressortir les fonctionnalités du système, visualiser le cycle de vie du système ou encore représenter l'architecture conceptuelle du système.

1.1. Analyse des différents acteurs et besoins utilisateurs

Pour nous orienter sur le type d'application attendue, la société cliente nous a fourni un résumé de sa vision de l'application et de ses attentes. A partir de cet énoncé, nous avons conjointement pris un temps de réflexion afin de faire ressortir les potentiels acteurs de l'application et les besoins des utilisateurs. Ainsi, nous avons identifié les acteurs suivants :

- Tout d'abord, nous avons identifié quatre acteurs principaux : l'employé classique, le chef d'équipe, l'employé du service RH et le responsable RH. Ces acteurs sont en effet les principaux utilisateurs de l'application et chacun d'entre eux dispose de besoins communs mais aussi spécifiques à leur poste.
- Puis, nous avons identifié la fiche de gestion des effectifs comme étant un acteur secondaire. Cette fiche permet d'ajouter (ou supprimer) des employés dans le système.

Une fois les acteurs identifiés, nous avons donc réfléchi aux différents besoins des utilisateurs. Nous avons obtenu une liste (plutôt exhaustive) des besoins de besoins des utilisateurs. Cependant, nous nous sommes aperçus que nous pouvions factoriser certains d'entre eux. L'objectif de cette factorisation est d'identifier des cas d'utilisation spécifiques : un cas d'utilisation (ie "*use case*") représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier.

Par conséquent, nous avons regroupé nos besoins utilisateurs en cinq cas d'utilisation. Les cas d'utilisation retenus sont donc les suivants :

- Gérer une demande de congés
- Déposer une demande de congés
- Administrer une demande de congés par un membre du service RH
- Afficher les informations relatives aux demandes de congés
- Gérer une fiche employé

Chacun de ces cas d'utilisation correspond à une spécification d'un comportement attendu du système (ie de l'application). L'identification des acteurs du système et des besoins utilisateurs fut la première étape de notre étude *UML*. Cette étape est non négligeable si nous souhaitons développer une application qui répond au mieux aux besoins de la société cliente.

Les cas d'utilisation que nous avons précédemment identifiés présentent des relations entre eux. La résultante de ces relations consiste en un diagramme de cas d'utilisation : un tel diagramme décrit les comportements attendus par le système d'un point de vue des utilisateurs/clients. Nous avons ainsi une description de ce que fait le système et non de comment celui-ci le fait. Il permet également de fixer le périmètre entre le système et son environnement, tout en visualisant les services qu'il rend.



Figure 1. Diagramme de cas d'utilisations établi en début d'étude UML : ce diagramme permet de voir l'interaction entre les différents cas d'utilisations.

La figure précédente vous présente le diagramme de cas d'utilisations auquel nous avons abouti en ce début d'étude *UML*. Nous avons fait apparaître des relations

spécifiques entre certains cas d'utilisations. Nous allons vous présenter les trois types de relations présentes pour des relations spécifiques (le principe restera le même pour les autres relations présentes dans le diagramme).

Commençons par la relation d'association entre un cas d'utilisation et un acteur. Cette relation est représentée par un trait plein. Par exemple, cette relation est utilisée pour traduire qu'un employé participe au cas d'utilisation *Gérer une demande de congés*.

Nous avons également plusieurs relations d'héritage. Elle peut être utilisée pour traduire une relation entre deux acteurs ou entre deux cas d'utilisation. Dans notre étude, nous retrouvons les deux utilisations : dans les deux cas, cette relation spécifie que l'un des acteurs/cas d'utilisation spécifie ou généralise l'autre. Par exemple, nous avons utilisé l'héritage pour traduire qu'un chef d'équipe est un employé classique avec des fonctions supplémentaires. Ainsi, la notion de chef d'équipe est une spécialisation de la notion d'employé.

Pour finir, nous retrouvons une relation d'extension : cette dernière traduit le fait que le cas d'utilisation de base nécessite le cas d'extension (mais ceci n'est pas obligatoire). Dans notre cas, le cas d'utilisation *Gérer une demande de congés* peut nécessiter le dépôt d'un commentaire (ie cas d'utilisation *Laisser un commentaire sur une demande*) mais le dépôt d'un commentaire reste facultatif.

A partir de ces cas d'utilisation, nous avons pu apporter une réflexion supplémentaire à chacun d'entre eux et ainsi obtenir différents diagrammes pour préciser les fonctionnalités du système.

1.2. Gérer une demande de congés

Lors de sa connexion à *DaysOffManager*, l'employé est redirigé vers sa page d'accueil dans laquelle toutes ses demandes de congés sont affichées. C'est à partir de cette page que deux scénarios sont possibles pour sa gestion de congés. Le premier est la modification d'une demande et le second est la suppression de cette demande. Il y a tout de même un prérequis pour cette gestion de congés. Les demandes effectuées ne doivent pas déjà être validées par un employé des ressources humaines. Dans le cas contraire, il est impossible de les modifier ou supprimer.

Dans le cas de la modification, un formulaire est appelé dans lequel l'employé doit remplir les informations à modifier sur sa demande. Les informations entrées sont ensuite vérifiées. Si elles ne sont pas valides (pas assez de jours disponibles par exemple), l'employé est de nouveau invité à ressaisir les informations jusqu'à ce que la modification soit valide. À tout moment, l'employé peut revenir à sa page d'accueil pour annuler sa modification.

Dans le cas de la suppression, un bouton *supprimer* est disponible pour chaque demande effectuée à la page d'accueil de l'employé. L'employé doit cliquer sur celui-ci pour supprimer sa demande.

L'ensemble de ces scénarios se retrouvent sur le diagramme d'activité du cas d'utilisation, diagramme présenté ci-après.

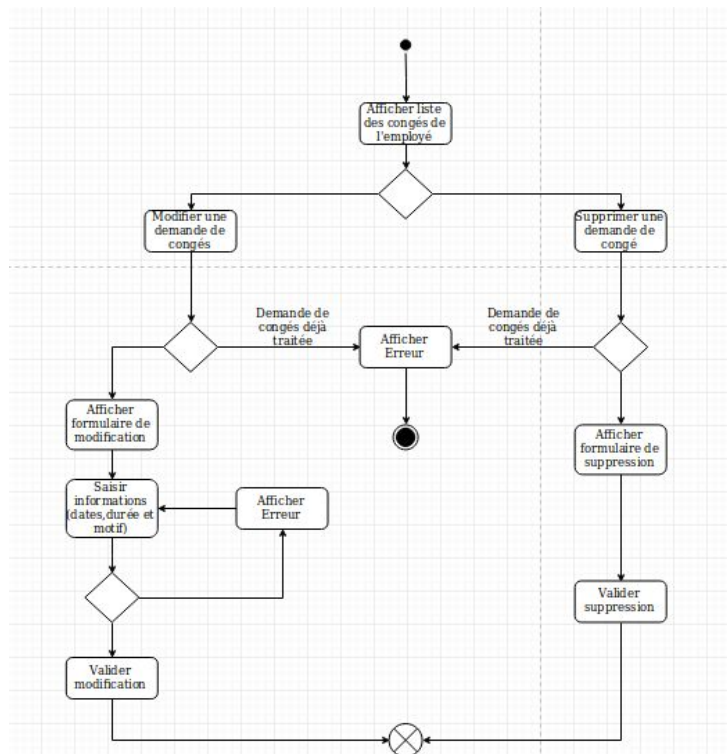


Figure 2. Diagramme d'activités pour la gestion d'une demande de congés: ce diagramme permet de présenter graphiquement le déroulement du cas d'utilisation gérer une demande de congés.

1.3. Déposer une demande de congés

Le dépôt d'une demande de congés est l'un des cas d'utilisation primordiaux du système. Il s'agit du dépôt d'une demande de congés auprès du logiciel de congés. Par conséquent, l'employé commence par s'authentifier auprès du logiciel de gestion de congés. Suite à cela, le logiciel affiche le récapitulatif de ses demandes de congés et son solde de congés. L'employé choisit d'effectuer une demande de congés. Puis, l'employé saisit successivement la date de début de sa période de congés, la date de fin de sa période de congés, la durée de période de congés (durée en jours), le motif de la demande. Enfin, l'employé valide sa demande de congés et le logiciel enregistre la demande faite par l'employé.

Pour traduire cet enchaînement d'actions, le diagramme d'activités présenté ci-après fut établi.

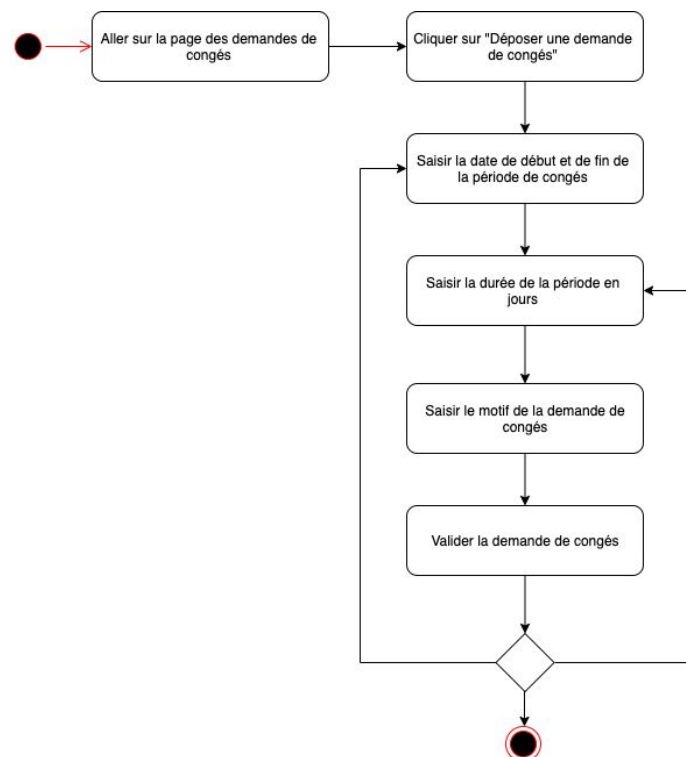


Figure 3. Diagramme d'activités associé au cas d'utilisation Déposer une demande de congés : ce diagramme permet de présenter graphiquement le déroulement du cas d'utilisation Déposer une demande de congés.

Nous remarquons que des scénarios alternatifs peuvent être identifiés : nous en avons identifié deux. Le premier est que le solde de congés dont l'employé dispose n'est pas suffisant, ainsi l'employé doit de nouveau saisir les dates de sa période de congés afin qu'elles concordent avec son solde de congés disponible. Le second est l'annulation de la demande de la part de l'employé.

Lors du développement de l'application nous avons fait évoluer le premier scénario alternatif. En effet, ce n'est pas le système qui fait la vérification mais le membre du service RH qui s'occupera de vérifier la concordance avec le nombre de jours disponibles pour sa période de congés. De même, lors de sa connexion à l'application, le solde disponible pour l'employé n'est pas affiché sur sa page d'accueil mais plutôt dans son profil.

1.4. Administrer une demande de congés par un membre du service RH

L'administration d'une demande de congés par un membre du service RH est un cas d'utilisation central du système. Il s'agit là de consulter les demandes, de valider la demande, de refuser la demande et de laisser un commentaire. Ce cas

d'utilisation est associé à un acteur spécifique : un employé du service RH. En effet, seul un membre du service RH peut traiter la demande d'un employé.

Tout d'abord, le système affiche une page des demandes en attentes. Ensuite, l'employé RH clique sur une demande, vérifie la période de congés et le motif puis valide la demande. Ensuite, le système demande à l'employé s'il souhaite laisser un commentaire. L'employé RH peut donc laisser un commentaire et le système enregistre la validation. Pour finir, le système notifie l'employé avec le commentaire laissé par l'employé du service RH et retire la demande des demandes en attentes.

Pour traduire cet enchaînement d'actions, le diagramme d'activités présenté ci-après fut établi.

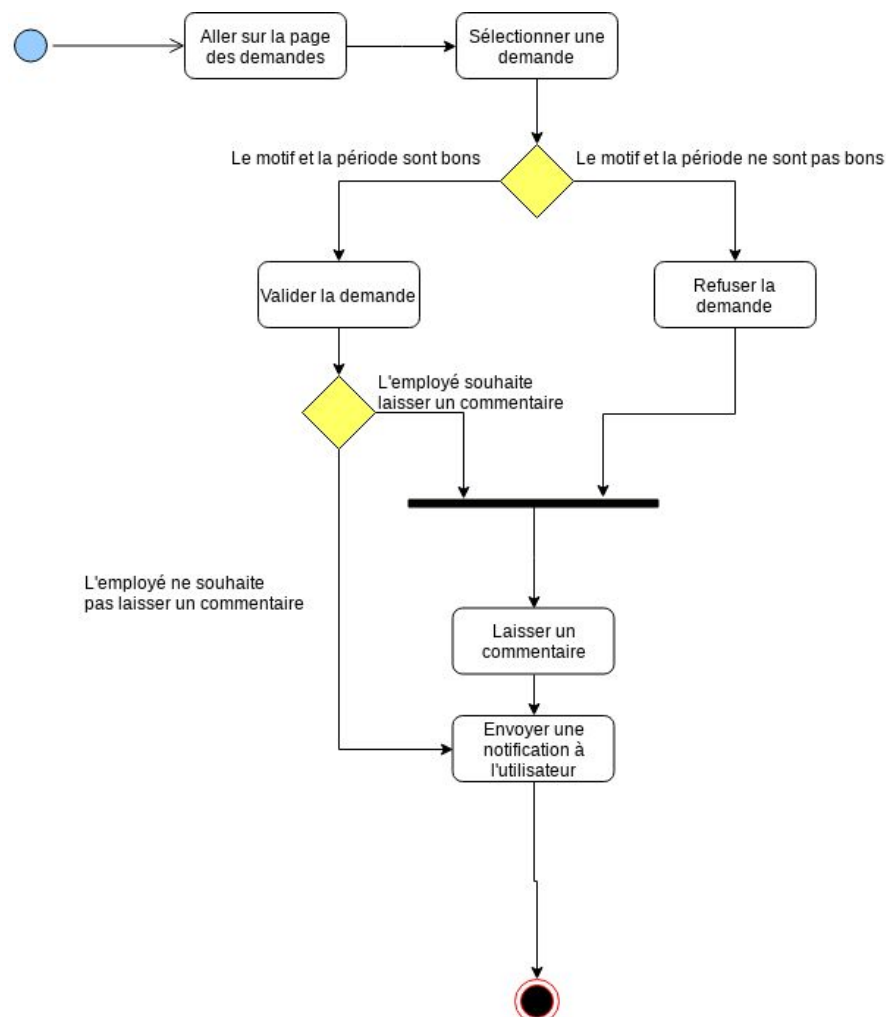


Figure 4. Diagramme d'activités associé au cas d'utilisation Administrer une demande de congés par un membre du service RH

A tout moment, l'administration de la demande peut être annulée par l'administration et se retrouve donc comme non modifiée. Si la demande vient d'un employé du service RH, seul le responsable RH peut l'accepter ou la refuser. De plus, si l'employé du service RH ne veut pas laisser de commentaire, le système enregistre la demande sans commentaire. Si l'employé du service RH refuse la demande, le

système oblige l'employé à laisser un commentaire et enregistrer le refus après la saisie du commentaire par l'employé du service RH.

Lors de l'implémentation de la fonctionnalité dans l'application, nous avons fait quelques modifications. En effet, l'employé du service RH consulte les demandes traitées et non traitées sur une seule et même page. Il peut également traiter les demandes sur cette page. De plus, l'employé du service RH peut filtrer les demandes pour n'afficher que celles qui l'intéressent.

1.5. Gérer une fiche employé

La gestion d'une fiche employé a été distinguée en trois scénarios : un nominal et deux alternatifs. Pour commencer, le scénario nominal décrit la gestion d'une fiche employé pour une personne embauchée pour la première fois dans l'entreprise.

Dans ce cas, l'employé n'est pas encore enregistré dans le système de gestion de congés. Une personne du service des Ressources Humaines crée donc une nouvelle fiche employé et saisit l'ensemble des informations administratives de la personne concernée. Par la suite, une validation est effectuée par les deux personnes et le logiciel finit par envoyer un mail de confirmation à l'employé.

Dans le cas où l'employé a déjà été embauché au moins une fois dans une entreprise, il possède déjà une fiche employé. Dans ce cas-là, deux actions sont possibles : soit la personne n'a pas de modifications à effectuer et donc elle reçoit juste un mail de confirmation lui rappelant le contenu de sa fiche, soit la personne doit réaliser des modifications sur sa fiche. Dans le cas échéant, l'employé vient à éditer, à l'aide d'une personne des Ressources Humaines, le contenu de sa fiche. Après modification, une validation est réalisée par les deux personnes et un mail de confirmation est envoyé à l'employé.

Pour illustrer ces différents scénarios, nous avons réalisé différents diagrammes dont celui d'activité présenté dans la figure ci-dessous.

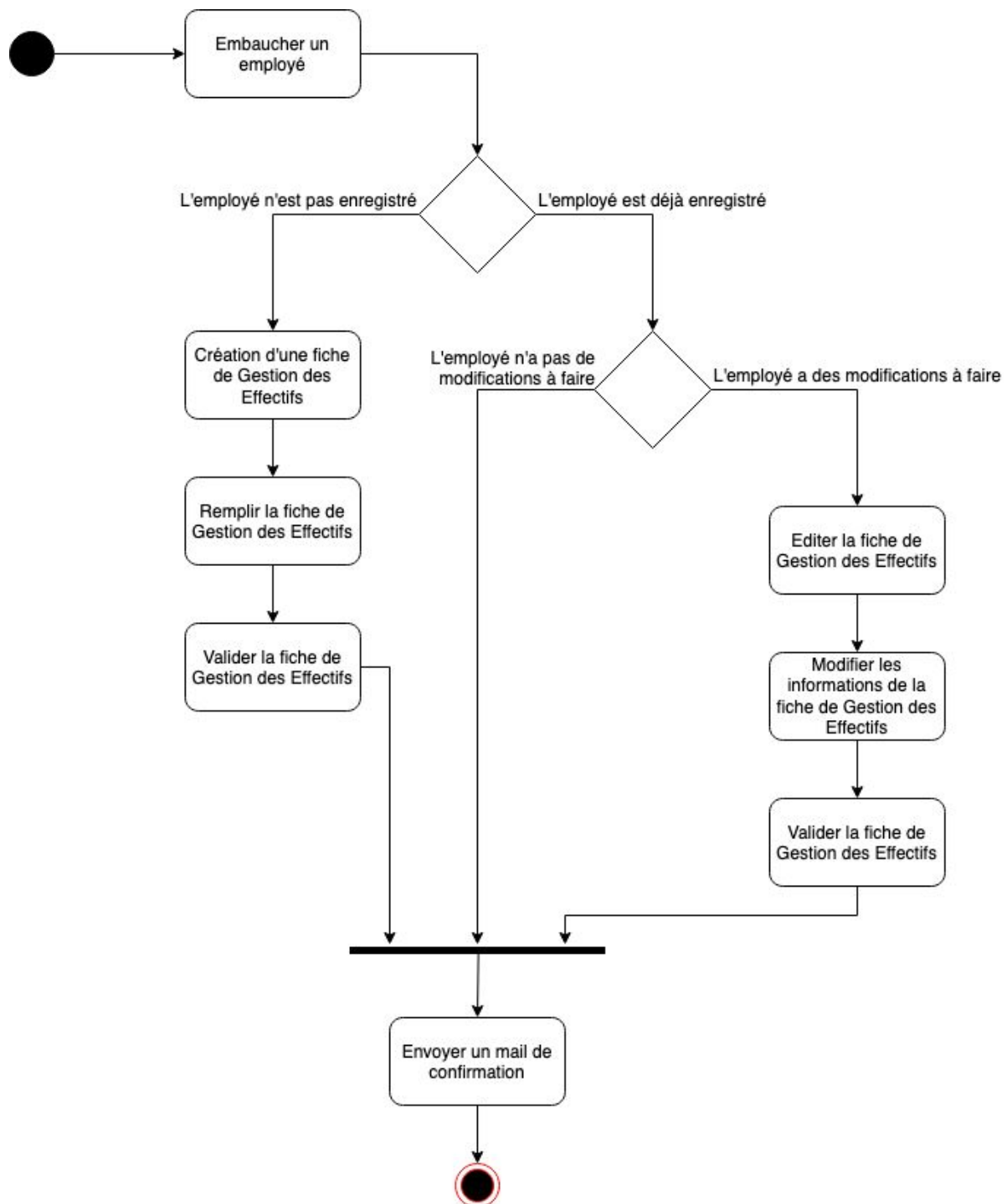


Figure 5. Diagramme d'activité relatif à la gestion d'une fiche employé

Dans un premier temps, la première activité que l'on peut noter est l'embauche d'un employé. Ensuite, si un employé n'a jamais été enregistré dans le système, alors il crée sa fiche de gestion des effectifs (à l'aide d'une RH) et la valide.

Dans un second temps, si une personne a déjà été enregistrée dans le système (ce qui veut dire que cette personne a déjà été embauchée au moins une fois) alors la personne a le choix de modifier ou non ses informations personnelles.

Enfin, dans tous les cas, un mail de confirmation est envoyé à l'employé, lui rappelant les informations rentrées dans le système.

1.6. Afficher les informations relatives aux demandes de congés

Nous avons identifié quatre scénarios possibles pour l'affichage des informations relatives aux demandes de congés. Ils dépendent principalement de l'employé qui s'est connecté à l'application.

Dans le cas d'un responsable RH, le système de gestion des congés doit afficher toutes les demandes de congés. Ce responsable peut sélectionner n'importe quelle demande. Ainsi, le système affichera toutes les informations relatives à la demande sélectionnée.

Pour un employé RH, le système ne doit afficher que les demandes de congés non-relatives à des demandes faites par l'équipe RH. Pour un chef d'équipe, le système n'autorise que l'affichage des demandes de congés des employés appartenant à son équipe et les siennes. Pour un employé, le système n'affiche que ses demandes de congés.

Plusieurs types d'affichages sont également disponibles. Si l'utilisateur souhaite afficher les demandes refusées, acceptées ou en cours, le système affichera ces demandes en fonction de ce statut. Bien évidemment, l'affichage va dépendre du métier de l'utilisateurs (présentés ci-dessus). Pour un chef d'équipe qui ne souhaite que voir ses demandes, celui-ci peut cliquer sur *sélectionner mes demandes*. Ainsi, le système n'affiche que les demandes du chef d'équipe.

1.7. Diagramme de classes global de l'application

Le diagramme final d'origine faisait apparaître quatre classes : la classe Demande, la classe Employé, les classes RH et TeamLeader héritant d'employé, ainsi qu'une énumération potentielle du motif.

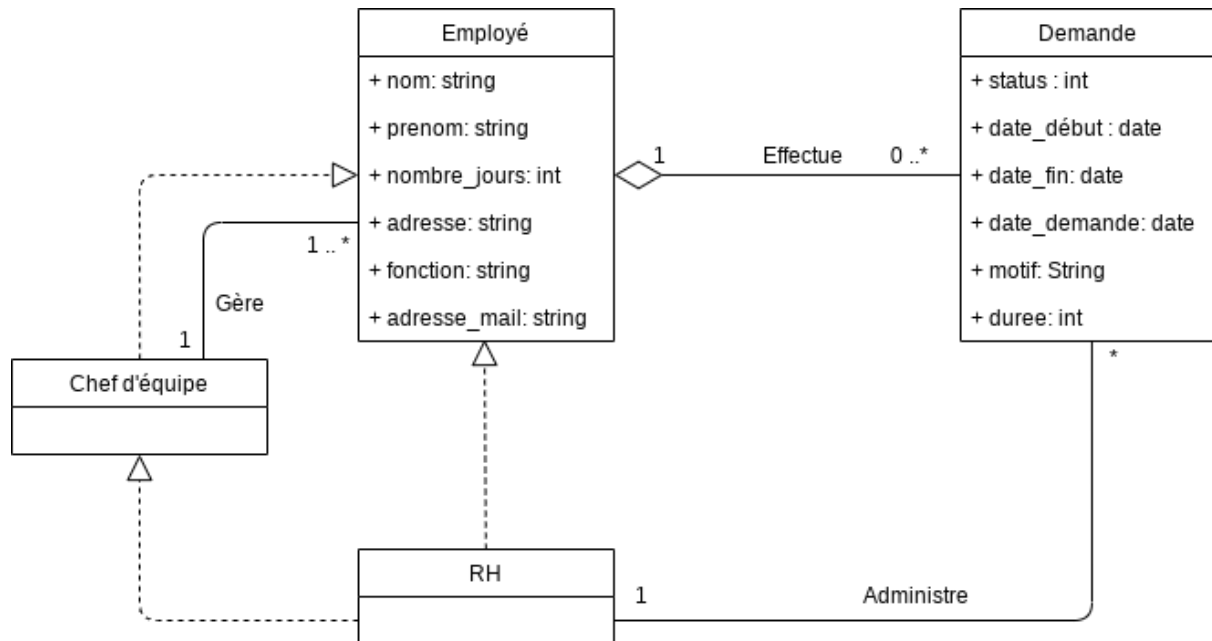


Figure 6. Diagramme de classe global de l'application : ce diagramme de classe est le diagramme de classe initialement établi en fin d'étude UML.

Suite l'implémentation de la base de données et aux développements relatifs aux fonctionnalités de l'application, nous nous retrouvons avec seulement trois classes :

- La classe **Employé**, qui contient désormais deux booléens indiquant si l'employé est RH ou/et si il est chef d'une équipe.
- La classe **Team**, qui contient les informations relatives à une équipe, à savoir son nom, son id, son leader et sa description
- La classe **Demande** à laquelle un **String** permettant de mettre un commentaire a été ajouté.

Chacune de ces classes permet de faire le lien avec la base de données et d'implémenter au mieux l'application.

Nous avons choisi de supprimer l'héritage des classes car ça n'était pas utile dans l'application. Le fait d'ajouter des classes n'apportait pas de factorisation supplémentaire de méthode et rendait plus compliqué la création et la vérification du rôle de l'employé.

L'Enum **Motif** est renvoyé directement sous forme de **String** depuis la base de données, sa description n'étant pas utilisée dans l'application.

Cette première partie sur l'étude UML a donc été une étape clé dans la phase de conception de notre application.

Nous avons pu définir les différents acteurs principaux et secondaires qui vont apparaître dans notre application ainsi que leur rôles.

Enfin, cette partie nous a permis de détailler et de mettre en valeur, à l'aide de diagrammes, les différentes fonctionnalités à implémenter par la suite dans la seconde partie du projet.

2. Partie Java Edition Entreprise

Suite à l'étude UML qui vous a été présentée précédemment, nous avons pu passer au développement de l'application. Pour ce faire, nous avons été chargés de mettre en place un certain environnement de travail pour le bon fonctionnement de l'application : création d'une base de données avec *MySQL*, usage d'un serveur *Apache Tomcat 9.0* via l'éditeur *Eclipse* ou encore un ensemble de *JavaServer Pages (JSP)* ou *Servlet* (ie une classe Java qui permet de créer dynamiquement des données au sein d'un serveur HTTP).

Une fois notre environnement de travail opérationnel, nous avons donc pu implémenter notre application de gestion de congés (ie ses fonctionnalités, ses modes d'utilisation spécifiques).

2.1. Mise en place d'une base de données via *MySQL*

Le développement d'une application telle que celle demandée nécessite la mise en place d'une base de données. Via *MySQL*, nous avons pu établir une base de données pour stocker les données brutes relatives à l'application de gestion de congés. Pour établir cette base, nous sommes partis des différents diagrammes de classes définis dans la partie *UML* du projet.

Après plusieurs temps de réflexion, nous avons pu établir la base de données présentées ci-après : il s'agit là de la base de données obtenue en fin de projet (celle-ci fut modifiée lorsque nous constatons un manque d'informations dans celle-ci ou des contraintes manquantes entre certaines tables).

Dans le but de vous présenter la base de données implémentée de manière claire et précise, nous utiliserons un dictionnaire de données : ce dernier est normalement associé au schéma Associations-Entités (dans notre cas, il est traduit le diagramme de classe précédemment présenté, outre quelques modifications apportées en cours de route).

Table employe Caractérise un employé-utilisateur de l'application		
Nom de l'attribut	Type	Commentaires
mail	Varchar	Adresse mail de l'employé
firstName	Varchar	Prénom de l'employé
surname	Varchar	Nom de l'employé
birthDate	Varchar	Date de naissance de l'employé
address	Varchar	Adresse postale de l'employé
nbDays	Integer	Solde de congés disponible affecté à l'employé
fonction	String	Poste de l'employé
pwd	Varchar	Mot de passe de l'employé pour se connecter à l'application (mot de passe non nul)
team	Integer	Numéro de l'équipe dans laquelle l'employé se trouve (par défaut 0 : l'employé n'appartient à aucune équipe par défaut)

Figure 7 : Table employe de la base de données : la clé primaire de la table est l'attribut mail seulement car chaque employé dispose d'une adresse mail unique. De plus, l'attribut team doit être un numéro d'équipe existant dans la table team.

Table team Caractérise les différentes équipes qui peuvent relier les utilisateurs		
Nom de l'attribut	Type	Commentaires
noTeam	Integer	Numéro de l'équipe (non nul car 0 définit le fait que les employés ne se trouvent dans aucune équipe)
name	Varchar	Nom de l'équipe
leader	Varchar	Adresse mail du chef d'équipe
description	Varchar	Description du rôle de l'équipe

Figure 8 : Table team de la base de données : la clé primaire de la table est l'attribut noTeam seulement car chaque équipe dispose d'un numéro unique. De plus, l'attribut leader doit correspondre à une des adresses mail présentes dans la table employe.

Table demand Caractérise une demande de congés déposée par un employé		
Nom de l'attribut	Integer	Commentaires
id	Integer	Identifiant de la demande : cet attribut est défini en série (ie un identifiant est créé en même temps que la demande est ajoutée)
employe	Varchar	Mail de l'employé qui dépose la demande
status	String	Statut de la demande déposée
beginDate	Date	Date de début de la période de congés
endDate	date	Date de fin de la période de congés
demandDate	dateTime	Date de dépôt de la demande de congés
reason	Varchar	Motif de la demande
duration	Integer	Durée de la période de congés
comment	Varchar	Commentaires

Figure 9 : Table demand de la base de données : la clé primaire de la table est l'attribut id seulement car chaque demande dispose d'un numéro unique. De plus, l'attribut employe doit correspondre à une des adresses mail présentes dans la table employe. De plus, l'attribut reason doit être un des motifs de la table reason.

Table reason Caractérise un motif de congés		
Nom de l'attribut	Integer	Commentaires
name	Varchar	Nom du motif
description	Varchar	Description associée au motif

Figure 10 : Table reason de la base de données : la clé primaire de la table est l'attribut name seulement car il ne peut y avoir deux motifs avec le même nom.

En plus de tables présentées via le dictionnaire de données, nous avons également utilisé des *triggers*. Ces derniers sont des déclencheurs, c'est-à-dire des objets de la base de données qui permettent d'exécuter une instruction spécifique ou plusieurs instructions lorsqu'il y a un ajout, une modification ou une suppression dans la table à laquelle ils ont été rattachés.

Nous avons donc utilisé les trois déclencheurs suivants :

- `decrementeNbDays` : ce premier déclencheur permet de mettre à jour le solde de congés d'un employé. Lorsque le statut de la demande passe de en cours d'approbation à approuvé, le solde est décrémenté du nombre de jours indiqué dans la demande de congés. Le solde se voit incrémenter du nombre de jours indiqué dans la demande de congés si la demande est supprimée. Ce déclencheur a lieu avant de mettre à jour la table demand.
- `insertDemand` : ce déclencheur ajoute le statut par défaut (ie le statut "en cours d'approbation") à chaque nouvelle demande créée. Ce déclencheur se fait avant l'insertion de la nouvelle demande dans la table demand.
- `insertTeam` : il permet de mettre à jour le poste d'un employé qui vient de créer une équipe et ce avant d'insérer la nouvelle équipe dans la table team.

Pour finir, nous avons ajouté un événement dans la base de données permettant de valider automatiquement les demandes après 48h. Ainsi, chaque demande non traitée par un membre du service RH se verra validée au bout de 48h, évitant donc que certaines demandes ne soient jamais validées.

2.2. Les différentes pages de l'application

2.2.1. Différents rôles de l'application de gestion de congés

Nous avons défini quatre rôles. Ils sont respectivement employé (employé classique), employeRH, teamLeader (chef d'équipe) et RespoRH (responsable RH). Les fonctionnalités offertes par l'application *DaysOffManager* dépendent d'eux.

Un employé peut afficher, gérer sa liste de demandes de congés, modifier ses informations de profils (adresse, mot de passe...) et consulter le calendrier.

Un chef d'équipe va pouvoir afficher sa liste de demandes ainsi que celle de son équipe. De plus, il possède les mêmes fonctionnalités qu'un employé.

Un employé RH, possède plus de fonctionnalités qu'un employé. En plus de gérer ses congés, il peut afficher, valider ou refuser les demandes de congés des employés et chef d'équipes (responsable RH et employés RH exclus).

Un responsable RH possède toutes les fonctionnalités de l'application. Il peut afficher et gérer toutes les demandes de congés de n'importe quel employé même celles pour les employés RH.

2.2.2. Description des différentes pages

Le concept de notre application est simple. Nous avons séparé en deux espaces définis la partie gestion de congés pour l'employé et la partie ressources humaines. Nous avons fait ce choix afin de donner une meilleure expérience à l'utilisateur en lui permettant de séparer un peu plus son travail de sa vie personnelle.

Pour ce faire nous avons implémenté un booléen mode "RH". L'utilisateur a juste à cliquer sur le bouton pour changer de mode et avoir les pages dédiées.

Un template a été mis en place via la page home et un String, currentUser, prend la valeur de la page que l'on veut afficher.

Comme expliqué plus haut, il y a quatre rôles dans notre application et chacun des rôles à une interface différente. En effet, la barre de navigation offre de nouveaux onglets selon les besoins. La page Équipe apparaît uniquement pour les Leaders et le bouton RH uniquement pour les employés en ressources humaines et leur responsable.

Les différentes pages disponibles sont donc :

- Pour tous :

La page calendrier qui permet de répertorier tous les congés de l'employé sous deux formes, un calendrier et un tableau. Le tableau permet à l'employé de modifier les demandes en attente : les dates, le nombre de jours, et le motif. Cette page permet aussi de poser un congé si les critères de dates et de nombre de jours. Tous les employés ont aussi accès à une page Profil, leur permettant de changer de mot de passe ou d'adresse postale.

- Pour les chefs d'équipe :

Les chefs d'équipe ont accès à une page team similaire à la page calendrier répertoriant les congés de leurs employés. Cette page possède donc un calendrier et un tableau récapitulatif.

- Pour les employés RH et leur responsable :

Un bouton mode RH apparaît. Celui-ci amène sur l'interface RH qui comprend deux onglets : la page gestion des demandes et celle de gestion des employés. La page de gestion de demande offre à l'employé RH un tableau récapitulatif de toutes les demandes et des filtres permettant de trier les demandes à afficher. Elle permet aussi d'accepter, de refuser et de laisser un commentaire sur une demande. Enfin, elle offre la possibilité à l'employé RH de modifier le motif de la demande. Sur cette page, seul le responsable RH peut voir les demandes des employés RH. La page de

gestion des employés permet de créer un employé, de le modifier, de le supprimer mais aussi de créer une équipe.

L'application possède également une page d'aide permettant aux utilisateurs de comprendre leur rôle sur l'application. Pour finir, l'application dispose d'une page de connexion qui n'est pas incluse dans le template original. Elle permet à l'utilisateur de se connecter à l'aide d'un email et d'un mot de passe.

2.3. Gestion du CSS

Afin de rendre la partie visuelle de l'application plus simple à mettre en place , nous avons décidé d'utiliser bootstrap. Un seul fichier de style contient le css utilisé dans le projet. Le reste est directement intégré dans les composants via la balise style.

Dans le but d'unifier les images et icônes, nous avons ajouté *font-awesome* au projet. Cette bibliothèque d'icônes permet, en ajoutant une balise `<i>` de mettre l'image de notre choix et d'ajuster sa police et son style selon le besoin.

2.4. Mise en place du calendrier des congés

L'objectif de cette partie a été de mettre en place un calendrier capable d'afficher soit les différentes demandes d'un employé, soit les demandes d'une équipe. Pour cela, il a fallu dans un premier temps réfléchir à comment modéliser ce calendrier. Le but ici n'étant pas de le développer, nous avons décidé de nous appuyer sur le plugin *jQuery Fullcalendar*. Ce plugin consiste à créer le visuel du calendrier et à charger des événements (comme les demandes d'un employé). Pour utiliser ce plugin, nous avons créé une page JSP nommée **calendar.jsp** qui joue le rôle de vue et qui est composée notamment du script JavaScript permettant le rendu du calendrier.

Afin de pouvoir charger les événements dans le calendrier nous avons utilisé la librairie java JSON.simple afin d'être capable de construire un JSONArray composé de JSONObject. Ces objets sont eux mêmes définis de la manière suivante.

```
{  
  "color": "",  
  "end": "",  
  "start": "",  
  "textColor": "",  
  "title": "",  
  "description": ""  
}
```

Figure 11 : structure d'un objet JSON utilisée par le plugin jQuery Fullcalendar afin d'afficher un événement

Comme le montre l'exemple ci-dessus, un événement est caractérisé par différents éléments :

- **color** est un paramètre permettant de modifier la couleur de fond d'une case d'un événement,
- **end** est un paramètre représentant la date de fin d'un événement (sous la forme yyyy-MM-dd),
- **start** est un paramètre représentant la date de départ d'un événement (sous la forme yyyy-MM-dd),
- **textColor** est un paramètre permettant de modifier la couleur du texte d'un événement,
- **title** est un paramètre permettant d'afficher du texte dans la case d'un événement,
- **description** est un paramètre optionnel permettant d'afficher un commentaire un passant la souris sur un événement (à l'aide de Bootstrap tooltips).

Une fois ce JSONArray construit, il suffit de le passer au code JavaScript qui s'occupe d'effectuer le rendu du calendrier. Le travail à réaliser dans cette partie est donc de récupérer les informations dont nous avons besoin en base de données pour construire un JSONArray.

2.4.1. Calendrier des employés

Dans le cadre d'un employé, nous voulons pouvoir récupérer ses demandes présentes en base de données (une demande est caractérisée par les attributs **id**, **employe**, **status**, **beginDate**, **endDate**, **demandDate**, **reason**, **duration** et **comment**). Pour cela, nous avons développé un DAO **DemandDAOImpl.java** contenant notamment la méthode suivante.

```
public List<Demand> findAllDemandeFromEmploye(String mail) {
    return findBy("select * from demand where
employee='"+mail+"'");
}
```

Figure 12 : Méthode provenant du DAO DemandDAOImpl.java permettant de retourner une liste de demandes d'un employé

Cette méthode prend en paramètre l'adresse mail d'un employé (généralement celui actuellement connecté) et retourne une liste de demandes. Le type **Demand** est une classe de notre modèle permettant de pouvoir passer des données SQL en données Java . La méthode **findBy** est une méthode commune au DAO utilisée pour interroger la base de données.

Après cela, nous avons créé une Servlet **CalendarServlet.java** qui joue le rôle de contrôleur. C'est donc dans ce fichier que l'on crée les différents objets JSON à passer au JavaScript. Une particularité lors de la construction de ces objets est l'état de la demande. En effet, chaque état est différencié sur le calendrier par une couleur et un titre différent. De plus, une subtilité du plugin Fullcalendar est que le jour de fin choisis est considéré comme travaillé. Pour résoudre le problème, nous avons donc dû allonger de un jour la date de fin de chaque demande.

Une fois ce travail réalisé, il ne reste plus qu'à passer l'ensemble de ces objets (sous le format d'un JSONArray) au JavaScript qui s'occupe du rendu du calendrier. Voici le résultat pour l'employé Maëlle Heyrendt.

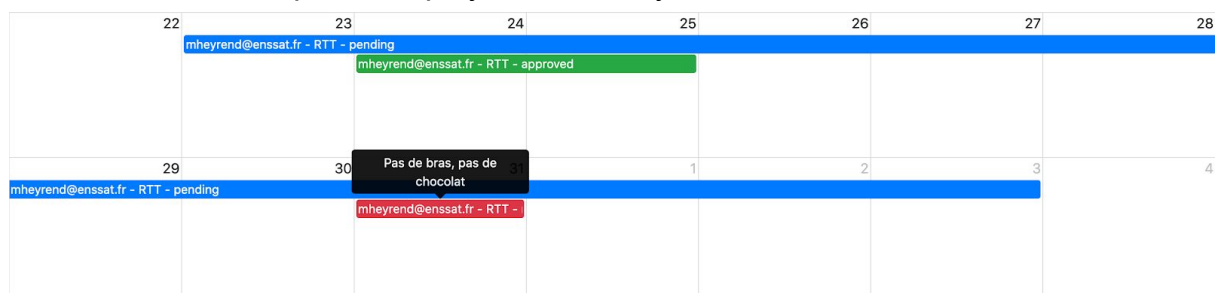


Figure 13 : Exemple de calendrier de congés d'un employé

2.4.2. Calendrier des chefs d'équipe

Dans le cadre d'un chef d'équipe, nous voulons pouvoir récupérer l'ensemble des demandes des membres de son équipe. Pour cela, nous utilisons le même DAO que celui pour les demandes d'un employé mais avec une méthode différentes.

```
public List<Demand> findAllDemandeFromTeam(String mail) {
```

```
return findBy("select * from demand where employe
in(select mail from employe where team=(select noTeam from
team where leader='"+ mail +"'"));");
}
```

Figure 14 : Méthode provenant du DAO DemandDAOImpl.java permettant de retourner une liste de demandes d'une équipe

Cette méthode prend en paramètre l'adresse mail d'un chef d'équipe (généralement celui actuellement connecté) et retourne la liste de toutes les demandes faites par les membres de son équipe. Par la suite, la partie construction et utilisation des objets JSON est la même que celui pour la partie précédente sur les demandes d'un employé. Voici le résultat de l'équipe responsable des Ressources Humaines.

26	27	28	29	30	31	1
						b@b.b - RTT - approved b@b.b - RTT - refused c@c.c - RTT - refused
2	3	4	5	6	7	8
b@b.b - RTT - approved b@b.b - RTT - refused						

Figure 15 : Exemple de calendrier de congés de l'équipe responsable des Ressources Humaines

2.5. Gestion des statistiques des équipes

L'une des fonctionnalités principales pour un chef d'équipe ou encore un membre du service des Ressources Humaines est la visualisation des statistiques de leurs équipes respectives. Une telle fonctionnalité peut leur permettre d'anticiper la prise de congés des équipes, solliciter les équipes afin d'avoir une nouvelle répartition des congés ou encore analyser les motifs récurrents des congés déposés.

Afin de rendre cette fonctionnalité disponible pour les utilisateurs concernés, il nous a fallu identifier les statistiques que nous jugions pertinentes pour les utilisateurs. Mais alors, comment définir la pertinence des différentes statistiques ?

Pour une telle tâche, nous nous sommes mis dans la peau d'un membre du service RH et nous nous sommes interrogés sur les statistiques dont il pourrait avoir besoin dans son quotidien. Par conséquent, nous avons retenu les quatre statistiques suivantes :

- Le nombre de congés déposés par mois (et ce toutes équipes confondues)

- Le nombre de congés déposés par équipe
- Le nombre de congés déposés en fonction du motif de la demande (et ce toutes équipes confondues)
- Le nombre de congés par poste (et ce toutes équipes confondues)

Pour garantir la pérennité de ces statistiques, nous avons choisi de prendre en compte l'état des demandes de congés. Ainsi, ces statistiques concernent uniquement les demandes qui ont été validées par un membre du service des Ressources Humaines. Faute de temps, nous nous sommes limités à ces quatre statistiques : le nombre de demandes refusées par mois et les motifs des demandes refusées par mois auraient pu se voir ajoutés à nos statistiques.

Nos statistiques étant fixées, nous avons donc pu écrire des requêtes SQL afin de récupérer les données appropriées dans la base de données (précédemment décrite dans ce rapport). Pour gérer ces requêtes, nous les créons une fonction par requêtes (ie par statistique) dans le fichier DemandeDAOImpl.java.

Nous avons choisis d'intégrer les statistiques avec les demandes : une telle implémentation nous permet de facilement récupérer nos demandes et de prendre état de leur statut. Ainsi, ceci facilite l'affichage des demandes et des statistiques (en cas d'absence de demandes, aucune statistique ne se verra afficher). Ceci est en adéquation avec les deux modes présents dans l'application ("Mode RH" et "Mode Employé").

Ci-après, la figure vous présente les quatre fonctions permettant la récupération dans la base de données.

```
// Requête SQL pour récupérer le nombre de congés déposés par
équipe
public ArrayList<List<String>> getDaysOffPerTeam() {
    return this.findNbDemandPerX("SELECT team.name AS team,
SUM(demand.duration)                                AS nbDays FROM
(demand JOIN employe ON demand.employe = employe.mail) LEFT
JOIN team ON employe.team WHERE demand.status = 'approved'
GROUP BY employe.team;", "nbDays", "team");
}

// Requête SQL pour récupérer le nombre de congés déposés par
mois (toutes équipes confondues)
public ArrayList<List<String>> getDaysOffPerMonth() {
```



```
        return this.findNbDemandPerX("SELECT MONTH(beginDate) AS  
month, SUM(duration) AS nbDays FROM demand WHERE  
demand.status = 'approved' GROUP BY  
MONTH(beginDate);", "nbDays", "month");  
    }  
  
    // Requête SQL pour récupérer le nombre de congés déposés par  
    motif (toutes équipes confondues)  
    public ArrayList<List<String>> getDaysOffPerReason() {  
        return this.findNbDemandPerX("SELECT reason, SUM(duration)  
AS nbDays FROM demand WHERE demand.status = 'approved' GROUP  
BY reason;", "nbDays", "reason");  
    }  
  
    // Requête SQL pour récupérer le nombre de congés déposés par  
    poste (toutes équipes confondues)  
    public ArrayList<List<String>> getDaysOffPerJob() {  
        return this.findNbDemandPerX("SELECT fonction,  
SUM(duration) AS nbDays FROM demand JOIN employe ON  
demand.employe = employe.mail WHERE demand.status =  
'approved' GROUP BY fonction;", "nbDays", "fonction");  
    }
```

Figure 16 : Requêtes SQL permettant de récupérer les données requises pour la constitution des statistiques fournies aux chefs d'équipes et aux membres du service RH : pour chacune des requêtes nous vérifions bien que la demande a été approuvée par un membre du service RH. Un affichage dans l'éditeur MySQL nous permet de vérifier le bon résultat des requêtes au préalable.

Pour la récupération du nombre de congés par motif et du nombre de congés par mois, nous avons des requêtes SQL simples (ie sans jointure requise entre deux tables). En effet, nous avons juste besoin de sélectionner les attributs utiles dans la table demand : respectivement l'attribut reason et l'attribut beginDate (dont nous récupérerons seulement le mois).

Pour les deux autres fonctions, nous remarquons que les requêtes sont plus complexes puisqu'elles nécessitent des jointures. En effet, nous utilisons la jointure pour récupérer le nombre de congés par équipe car il existe une table team qui nous impose de faire une jointure à gauche pour faire correspondre les employés à leurs équipes respectives. De même, pour le nombre de congés par poste, seule la table employe tient compte du poste donc une jointure entre demand et employe viendra donner le nombre de congés souhaité.

Une fois les données récupérées depuis la base de données, il nous faut les afficher sur la page dédiée aux statistiques du Mode RH de l'application. Par faute de temps, nous avons affiché la page des statistiques uniquement pour un membre du service RH. Cependant, les requêtes détaillées à la figure ci-dessus sont adaptables pour afficher les statistiques à fournir à un chef d'équipe. Également, une page similaire à celle du Mode RH pourrait être mise en place.

Pour ce faire, nous avons utilisé Google Chart API, un service Web interactif qui crée des graphiques à partir de données fournies par l'utilisateur. Un tel choix fut fait afin de garantir un affichage homogène pour l'ensemble des statistiques et aussi pour le large choix des diagrammes choisis (et le panel de couleurs correspondant à nos attentes). La figure ci-après vous présente le code permettant l'affichage du nombre de congés par équipe. Le code pour les autres statistiques est similaire.

```
<script                                     type="text/javascript"
src="https://www.gstatic.com/charts/loader.js">
</script>
<script type="text/javascript">
google.charts.load("current", {packages:["corechart"]});
google.charts.setOnLoadCallback(drawChartPerTeam);
var dataDaysOffPerTeam = [['Team', 'NbDays']];

function drawChartPerTeam() {
    var data = google.visualization
        .arrayToDataTable(getDataPerTeam(dataDaysOffPerTeam));
    var options = {
        title: 'Nombre de congés par équipe (*)',
        width: 600,
```

```
        height: 400,  
        is3D: true,  
        legend: { position: 'bottom' },  
    };  
    var chart = new google.visualization  
        .PieChart(document.getElementById('chartPerTeam'));  
    chart.draw(data, options);  
}  
  
function getDataPerTeam(dataPerTeam) {  
    <% for(int i=0;i<daysOffPerTeam.size();i++){%>  
        dataPerTeam.push(["<%=daysOffPerTeam.get(i).get(0) !=  
null ?    daysOffPerTeam.get(i).get(0).toString()    :    "Sans  
team"%>", <%=daysOffPerTeam.get(i).get(1)%>]);  
    <%}%>  
    return dataPerTeam;  
}  
</script>
```

Figure 17 : Code sollicitant l'API Google Chart dans le but d'afficher un diagramme de type circulaire : ce diagramme représente le nombre de congés par équipe.

Nous commençons par charger une librairie afin de charger l'ensemble des paquets utiles pour l'affichage des différents types de graphiques. La deuxième permet de récupérer la dernière version de la librairie et d'afficher tous les graphiques dits de base (barre, colonne, ligne, zone, zone étagée, bulle, tarte, beigne, combo, bougeoir, histogramme, dispersion).

Cet extrait du code suppose également que nous avons une fonction JavaScript nommée `drawChartPerTeam` définie dans la page Web : elle est unique et elle définit comme affichant un diagramme circulaire. Cette fonction prend en paramètres un "tableau de tableau" : les données récupérées sont placées dans un tableau (fonctionnant sur le principe de clef-valeur) qui sera placé dans un tableau général. Ce tableau général comprend autant d'éléments (ie tableaux) qu'il existe d'équipe.

Comme il peut s'écouler un certain temps avant que le chargement de la librairie ne soit terminé, nous devons enregistrer une fonction de rappel. Pour cela,

nous appelons `setOnLoadCallback` en passant une fonction comme argument. Une fois le chargement des paquets terminé, la fonction de rappel se verra appelée et nous obtenons le résultat présenté ci-dessous.



(*) Ces nombres de congés ont été calculés pour tous postes confondus.

Figure 18 : Nombre de congés déposés par équipe

Afin d'obtenir plusieurs graphiques sur une même page, nous créons une fonction de rappel par fonction nécessaire à l'affichage des statistiques. Pour les trois autres statistiques, seuls les types du "tableau de tableau" et le type du diagramme varient.

Dans cette seconde partie dédiée au développement JEE, nous avons mis en place l'environnement de travail demandé pour le bon fonctionnement de l'application. De plus, nous avons réussi à implémenter, à partir de notre travail sur la modélisation UML, notre application de gestion de congés (comprenant l'ensemble des fonctionnalités et acteurs principaux et secondaires que nous avons déterminé).

3. Jeux de tests

Cette dernière partie du rapport est consacrée à la mise en place d'un plan de tests. Les objectifs d'un tel plan de tests sont de valider le bon fonctionnement de l'application (ie le bon fonctionnement de l'ensemble des fonctionnalités implémentées) ainsi que de faire ressortir les limites de l'application. En effet, nous avons dû faire des choix de conception suite à des contraintes de temps ou encore par soucis d'homogénéisation vis-à-vis d'autres fonctionnalités.

Ainsi, des limites ont pu voir le jour dans l'usage de l'application et donnent également naissance à de potentielles évolutions pour DaysOffManager.

3.1 Connexion et affichage de la page d'accueil

3.1.1 La vue Employé

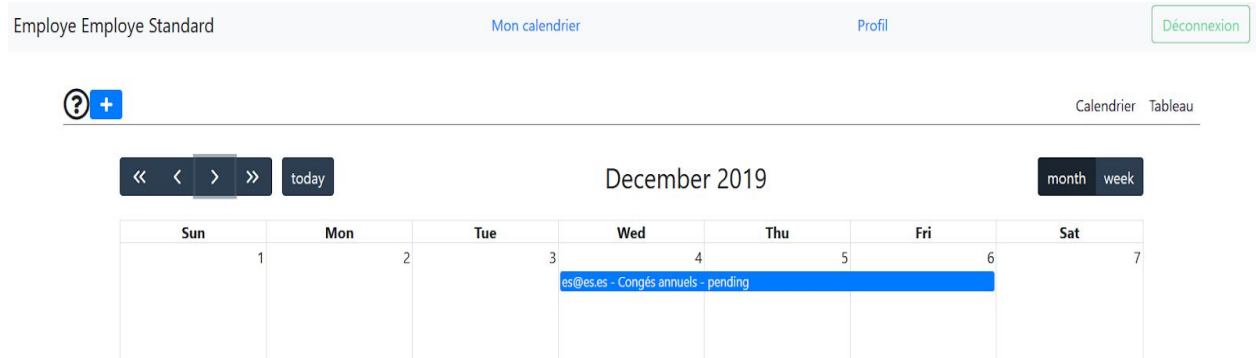


Figure 19 : page d'accueil d'un employé

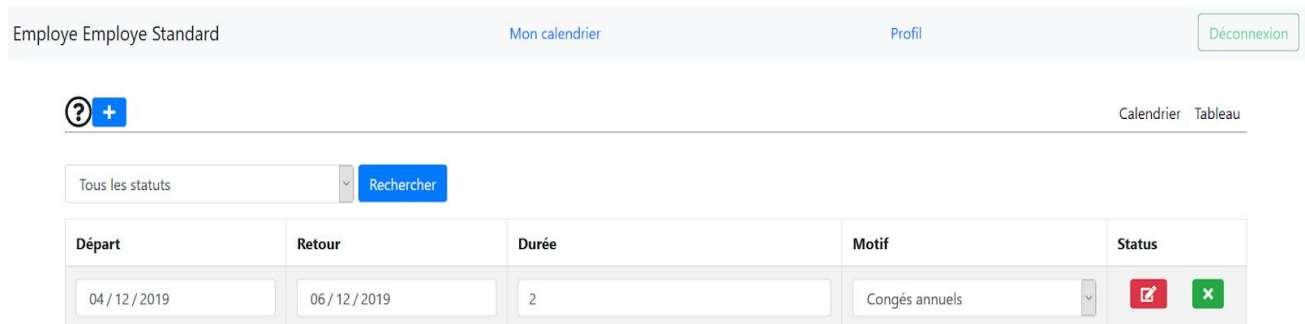


Figure 20 : page d'accueil alternative d'un employé

Nous pouvons constater lorsqu'un employé standard se connecte, celui-ci peut apercevoir son calendrier avec ses demandes de congés affichées (figure x). S'il choisit d'observer ses congés en forme de liste en cliquant sur "Tableau", il peut directement gérer ses congés qui n'ont pas encore été traités. Nous avons donc bien implémenté la page d'accueil pour un employé classique.

3.1.2 La vue Employé RH

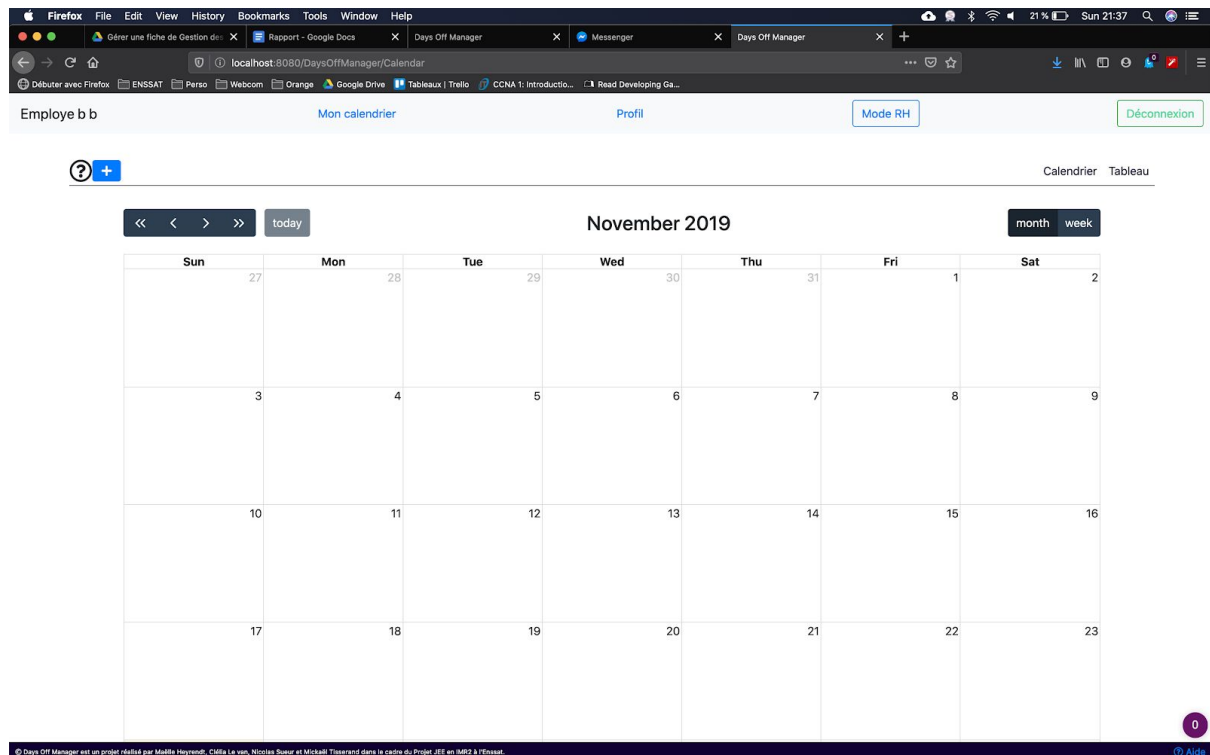


Figure 21 : page d'accueil d'un employé RH

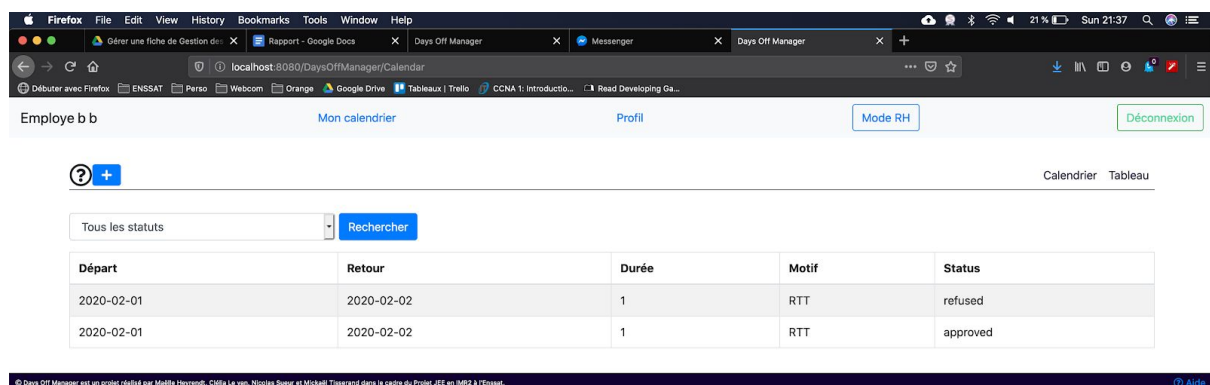


Figure 22 : page d'accueil alternative d'un employé RH

La page d'accueil d'un employé RH est semblable à celle d'un employé normal. La seule différence se trouve au niveau de la barre de navigation où l'on retrouve un bouton "Mode RH" pour passer en mode RH (gérer des demandes de congés et des employés).

3.1.3 La vue Chef d'équipe

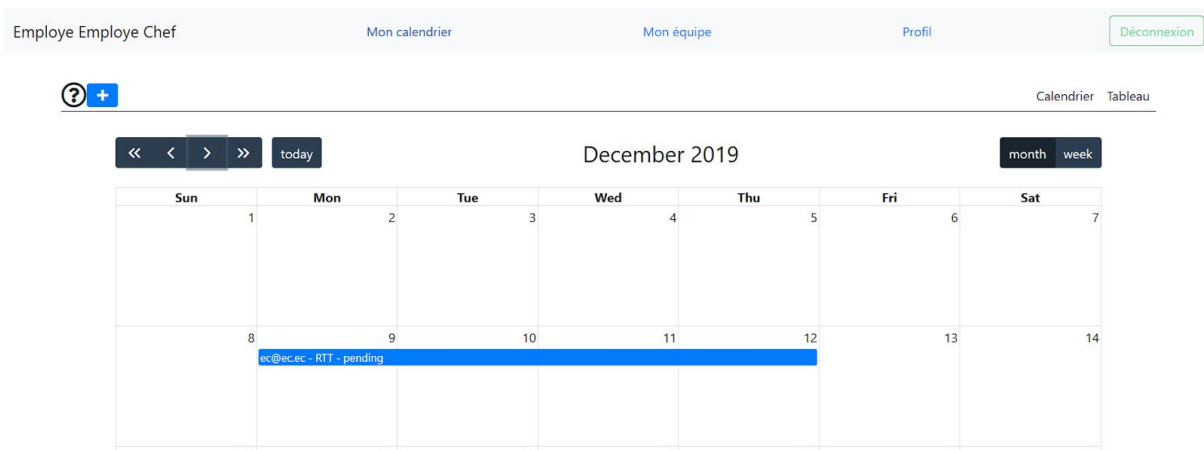


Figure 22 : page d'accueil d'un chef d'équipe

Le chef d'équipe quant à lui possède également la même page d'accueil qu'un employé standard. Il possède une option supplémentaire qu'est le bouton "Mon équipe".

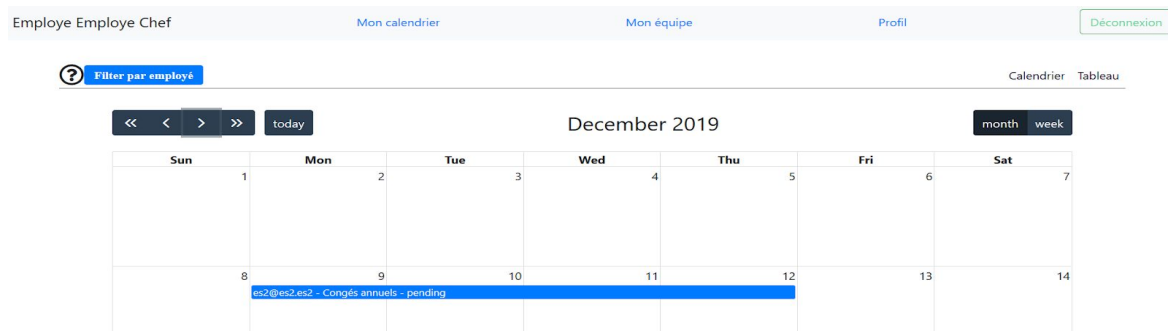


Figure 23 : page d'accueil alternative d'un chef d'équipe

Si le chef d'équipe décide d'appuyer sur ce bouton, il peut apercevoir la liste des demandes de congés des membres de son équipe sous deux formes. On peut distinguer la première qu'est sous forme de calendrier avec la figure x ci-dessus.

Employe
Employe Chef
Mon calendrier
Mon équipe
Profil
Déconnexion

? +
Calendrier
Tableau

Tous les statuts
Rechercher



Départ	Retour	Durée	Motif	Status
09 / 12 / 2019	12 / 12 / 2019	3	RTT	 

Figure 24 : liste des demandes de congés du chef d'équipe

Si le chef d'équipe souhaite uniquement voir ses propres demandes de congés, il suffit de faire comme un employé standard et d'appuyer sur le bouton "Tableau". Le résultat est affiché via la figure x ci dessus.

Employe
Employe Chef
Mon calendrier
Mon équipe
Profil
Déconnexion

Calendrier
Tableau

Nom Prénom	Départ	Retour	Durée	Motif	Status
Standard2 Employe	2019-12-09	2019-12-12	3	Congés annuels	pending

Figure 25 : liste alternative avec les demandes de congés des membres de l'équipe

La seconde forme est une liste de demandes des congés dans laquelle le chef d'équipe peut apercevoir l'ensemble des congés demandés par les membres de son équipe. Comme on peut le constater, il ne peut pas modifier ces congés. Nous avons donc répondu aux contraintes à respecter.

3.1.4 La vue Responsable RH

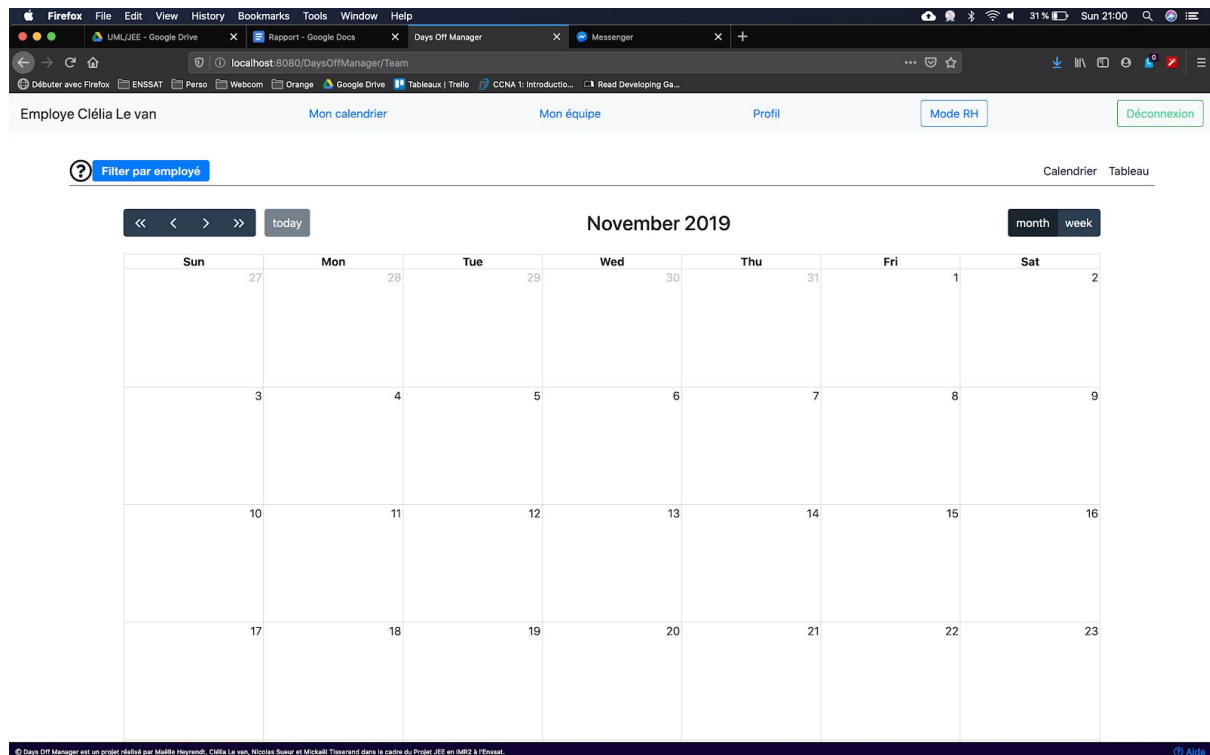


Figure 26 : page d'accueil d'un employé RH

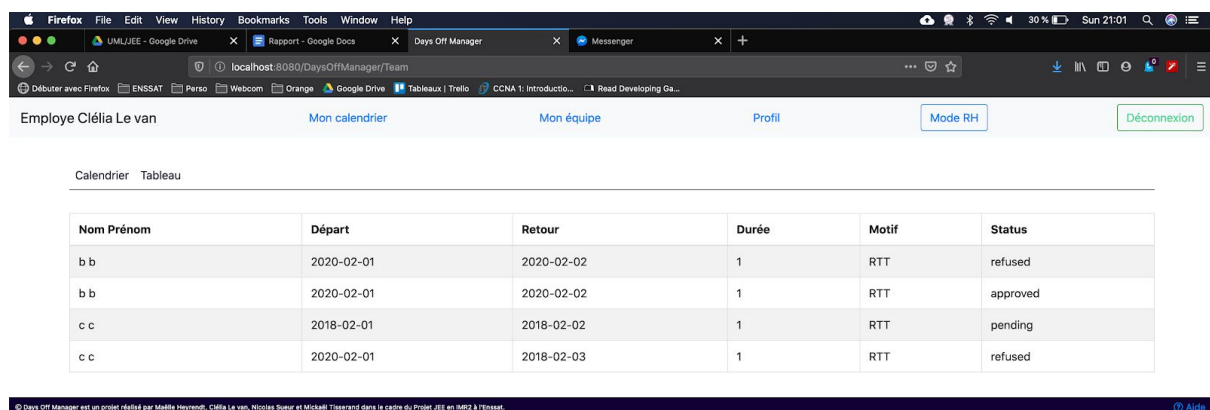


Figure 27 : page d'accueil alternative d'un employé RH

Un responsable RH cumule les droits d'un employé RH et d'un chef d'équipe. On peut retrouver les deux types de pages d'accueil que tout employé possède. On remarque cependant que la barre de navigation comporte à la fois l'onglet "Mon équipe" permettant d'afficher les demandes d'une équipe et le bouton "Mode RH" pour passer en mode d'administration RH.

3.2 Gestion des employés

3.2.1 Création d'un employé

Figure 28 : interface de création d'un employé

Sueur	Nicolas	nsueur@enssat.fr	1	Employe		
-------	---------	------------------	---	---------	--	--

Figure 29 : extrait du tableau de gestion des employés

En tant que RH, il est possible de créer un nouvel employé à partir du mode RH. Pour l'exemple ci-dessus, on ajoute un nouvel employé Nicolas Sueur avec ses informations administratives. En appuyant sur "Ajouter" on crée le nouvel employé et on le retrouve dans l'interface de gestion des employés.

3.2.2 Suppression d'un employé

Figure 30 : extrait du tableau de gestion des employés

Nous avons un employé Wall eeee qui a démissionné, nous voulons donc le supprimer de la base de données puisqu'il ne travaille plus pour l'entreprise. Pour cela, l'employé RH va cliquer sur l'icône rouge "supprimer".

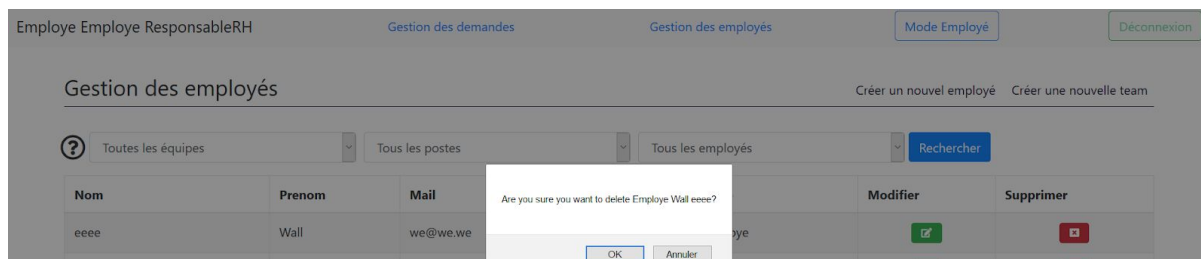


Figure 31 : extrait du tableau de gestion des employés

Pour confirmer la suppression et qu'il ne s'agit pas d'une erreur, une fenêtre pop-up s'active. L'employé RH doit cliquer sur "ok" pour confirmer la suppression.

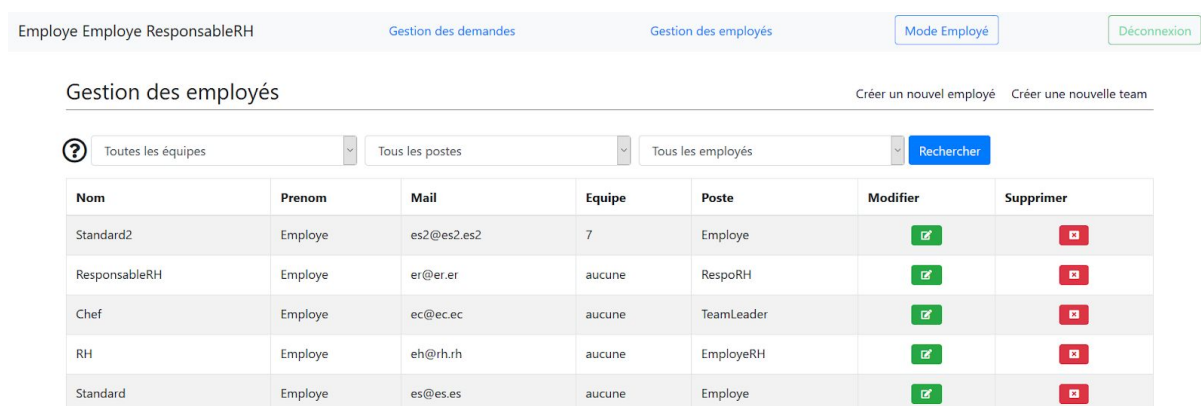


Figure 32 : extrait du tableau de gestion des employés avec Wall eeee manquant

On remarque via la figure x que l'employe Wall eeee est manquant. La suppression s'est donc bien effectuée et nous avons montré que cette fonctionnalités est opérationnelle.

3.2.3 Modification d'un employé par un employé RH

Sueur	Nicolas	nsueur@enssat.fr	1	Employe		
-------	---------	------------------	---	---------	--	--

Figure 33 : extrait de l'interface de gestion des employés avant modification

Figure 34 : interface de modification d'un employé

Sueur	Nicolas	nsueur@enssat.fr	1	TeamLeader		
-------	---------	------------------	---	------------	--	--

Figure 35 : extrait de l'interface de gestion des employés après modification

Pour ce test, un employé Nicolas Sueur demande à devenir chef d'équipe suite à la création d'une nouvelle équipe. En appuyant sur le bouton de modification on arrive sur une fenêtre semblable à celle de création d'un employé. On saisit les nouvelles informations (dans ce cas on met le paramètre "chef d'équipe" à "oui") et on appuie sur le bouton "modifier". On remarque que l'employé est maintenant un TeamLeader.

3.2.4 Modification d'un employé via son profil

Figure 36 : interface de modification du profil

Ce test à pour but de valider le fonctionnement pour la modification du profil d'un employé via sa page de profil. Nous avons l'employé nommé Employe Standard qui souhaite changer son adresse postale. L'employé renseigne sa nouvelle adresse qu'est "4 rue de la paix".

Employé dans l'équipe 0

Mail : es@es.es

Né(e) le : 01-04-199

Adresse postale :

4 rue de la paix

Envoyer

Figure 37 : changement d'adresse de l'utilisateur

Lorsque l'on recharge la page de profil de l'employé, celui-ci à maintenant sa nouvelle adresse renseignée dans le champ "Adresse postale". Nous avons donc bien validé le fonctionnement de la modification de l'employé via sa page de profil.

3.3 Gestion relatives aux demandes de congés

3.3.1 Création d'une demande de congés

The screenshot shows the 'Days Off Manager' web application in a Firefox browser. The user is 'Employe Maëlle Heyrendt'. The interface includes a navigation bar with links for 'Mon calendrier', 'Mon équipe', 'Profil', and 'Déconnexion'. Below the navigation bar, there are input fields for 'Du : 11/25/2019', 'au : 11/29/2019', 'Motif : Congés annuels', and 'Durée : 5'. A 'Faire la demande' button is visible. A calendar for November 2019 is displayed, with a date picker showing the selected dates. The calendar grid shows days from Sunday to Saturday, with the selected dates highlighted in blue.

Figure 38 : interface de demande de congés

Nous allons montrer que la création d'une demande de congés est fonctionnelle. Pour cela, nous créer cette demande avec l'utilisateur Maëlle Heyrendt du 25 novembre 2019 au 29 novembre 2019. Une fois les champs renseignés (voir figure x ci-dessus), il faut cliquer sur "Faire la demande".

24	25	26	27	28	29	30
	mheyrendt@enssat.fr - Congés - pending					

Figure 39 : extrait du calendrier des congés d'un employé

Lorsque cela est fait, on affiche la demande de congés via son calendrier. On remarque que la demande est affichée avec la valeur "pending" (en attente). Nous avons donc validé de fonctionnement de la création de congés.

3.3.2 Modification d'une demande côté employé

Figure 40 : interface de modifications de congés avec congés en cours de modification

Nous allons tester le fonctionnement pour la modification d'une demande de congés. Nous allons choisir les nouvelles dates d'une demande déjà effectuée au préalable. Ici, on modifie la date de retour. Le retour s'effectue le 09/11/2019 au lieu du 06/11/2019. Lorsque le changement est fait, on appuie sur le bouton rouge qui correspond à la modification.

Figure 41 : interface de demande de congés avec congés modifié

On peut apercevoir le message “Demande modifiée” avec les nouvelles dates pour la demande de congés modifiée précédemment. Nous avons donc validé cette fonctionnalité.

3.3.3 Modification d'une demande côté rh

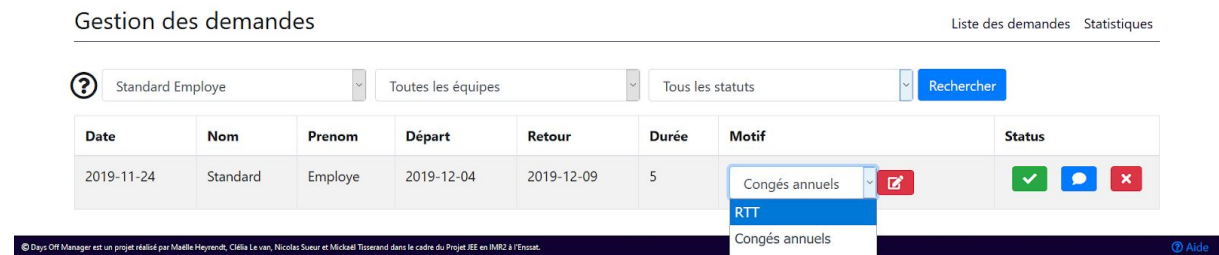


Figure 42 : interface de modifications de congés côté RH avec congés en cours de modification

Nous allons montrer que la modification d'une demande de congés d'un employé par un employé RH fonctionne. Pour cela, l'employé RH va modifier le motif de la demande de congés. Le motif va passer du statut “Congés annuels” à “RTT”.

En cliquant sur le bouton modifier (bouton rouge) et en effectuant de nouveau la recherche des demandes, on obtient le résultat illustré via la figure x ci-dessous :

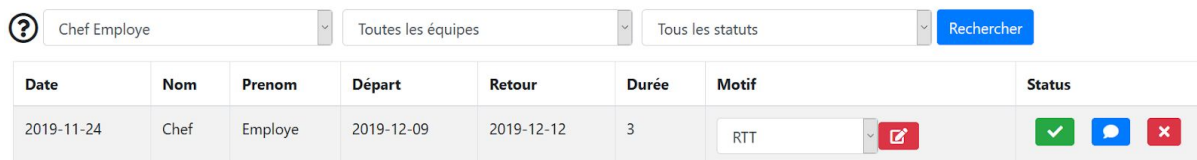


Figure 43 : interface de modifications de congés côté RH avec congés modifié

On peut remarquer que le motif est devenu “RTT”, on peut donc s'assurer que la modification des demandes de congés du côté d'un employé RH fonctionne.

3.3.4. Affichage des statistiques de demandes de congés

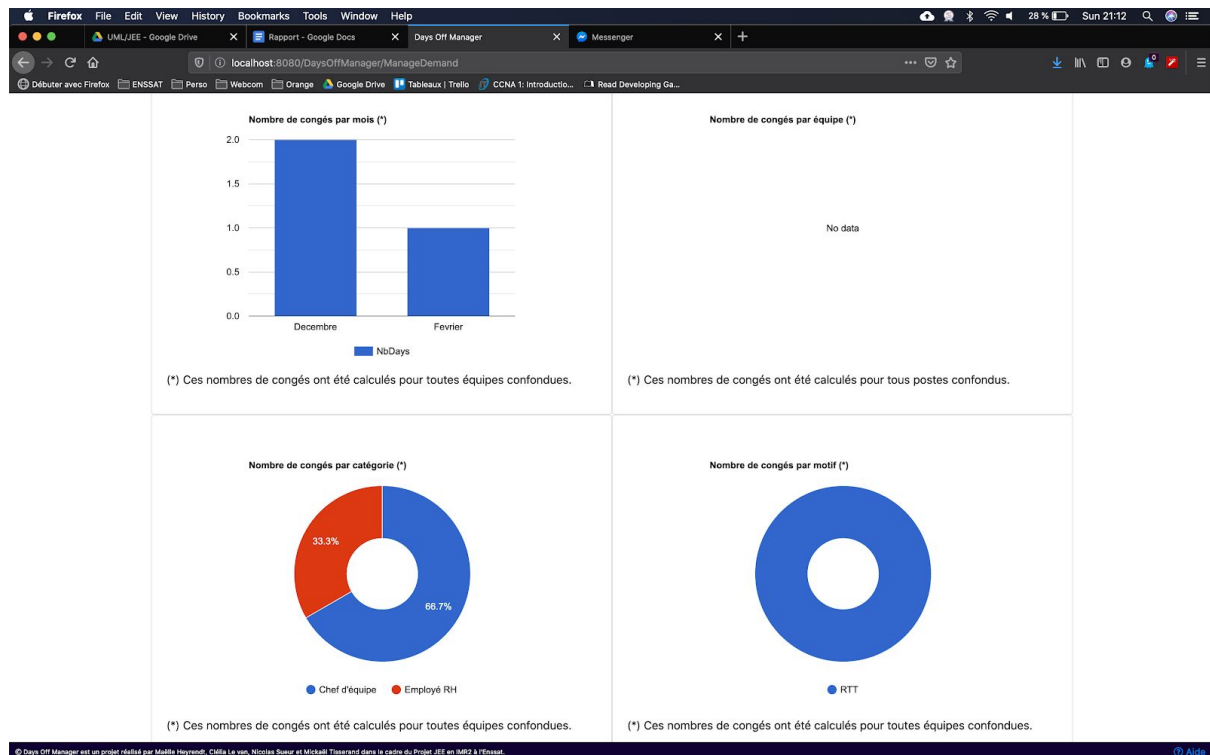


Figure 44 : interface des statiques de demandes de congés

Sur cette interface, on peut retrouver différentes informations relatives aux demandes de congés des employés d'une entreprise. Cette interface est utile aux RH.

3.4 Gestion des teams

3.4.1 Création d'une team

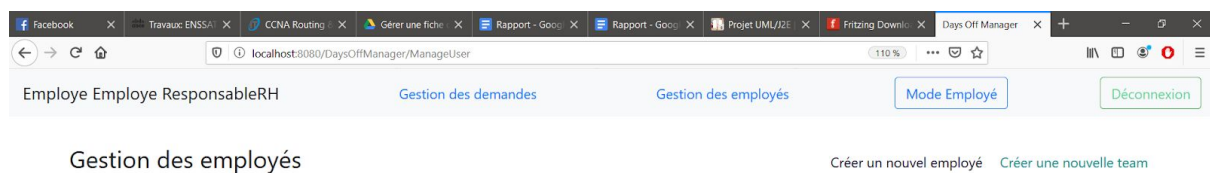


Figure 45 : interface de gestion des employés

Nous allons maintenant tester la création d'une nouvelle équipe. Pour cela, un employé RH va cliquer sur le bouton "Créer une nouvelle team". Il va être redirigé vers un formulaire illustré via la figure ci-dessous:

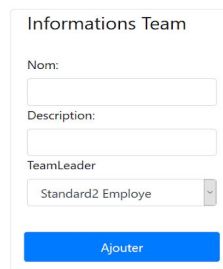


Figure 46 : interface de création de team

Dans l'exemple ci-dessus, on remarque qu'en allant dans le sous menu "Créer une nouvelle team" il est possible d'afficher la page de création d'une team. On y rentre différentes informations comme le nom, sa description ainsi que le chef de cette équipe (parmis un liste d'employés définis). On va par exemple créer l'équipe nommée "Team1" avec une description et un chef d'équipe quelconque. En cliquant sur le bouton "Ajouter", on est redirigé vers la page de gestion des employés. En affichant la liste des équipes, on peut retrouver l'équipe que nous avons créé. Le résultat est affiché ci-dessous:

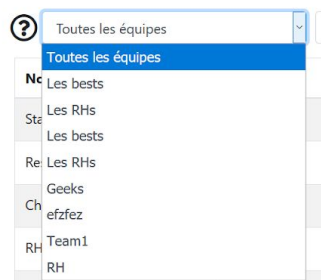


Figure 47 : Affichage de la liste des équipes

Team1 est bien affichée dans la liste des équipes. Notre création d'équipe fonctionne correctement.

Avec ces tests, nous avons montré plusieurs aspects de notre application. Tout d'abord, l'affichage de la page d'accueil et des congés sont différentes en fonction du rôle de chaque employé (s'il est RH, chef d'équipe, employé, ou responsable RH). Un employé classique peut déposer, supprimer et modifier des congés qui n'ont pas encore été traités. Il peut également changer ses informations personnelles. Ensuite, nous avons montré qu'un chef d'équipe peut afficher la liste des congés de son équipe. Du point de vue des employés RH, ces derniers peuvent créer, modifier et supprimer des employés et de gérer des demandes de congés (appartenant à des équipes non RH). Ils sont également capables de créer une équipe avec un chef. Seul le responsable RH est capable de gérer les congés RH d'après nos tests. Nous avons donc répondu aux critères de nos diagrammes UML.

Conclusion

Nous avons réalisé, lors de cette période académique, un projet mêlant modélisation graphique UML et développement JEE.

Lors de ce projet, nous avons dans un premier temps réalisé une étude UML qui est une étape clef dans la phase de conception d'une application orientée objets. Notamment, nous avons détaillé à travers différents diagrammes les fonctionnalités de notre application et avons pu en faire ressortir les différents acteurs.

Ensuite, avant de développer l'application de gestion de congés, nous avons conceptualisé une base de données regroupant les différentes informations nécessaires au bon fonctionnement du logiciel.

Enfin, à l'aide de l'étude UML réalisé au préalable, nous avons développé les différentes fonctionnalités imaginées à l'aide de Java EE qui est une plateforme Java destinée aux applications d'entreprises.

Annexes

Table des matières des annexes

Annexe 1 : Gérer une demande de congés	43
Annexe 2 : Déposer une demande de congés	48
Annexe 3 : Administrer une demande de congés par un membre du service RH	50
Annexe 4 : Gérer une fiche employé	53
Annexe 5 : Afficher les informations relatives aux demandes de congés	56

Annexe 1 : Gérer une demande de congés

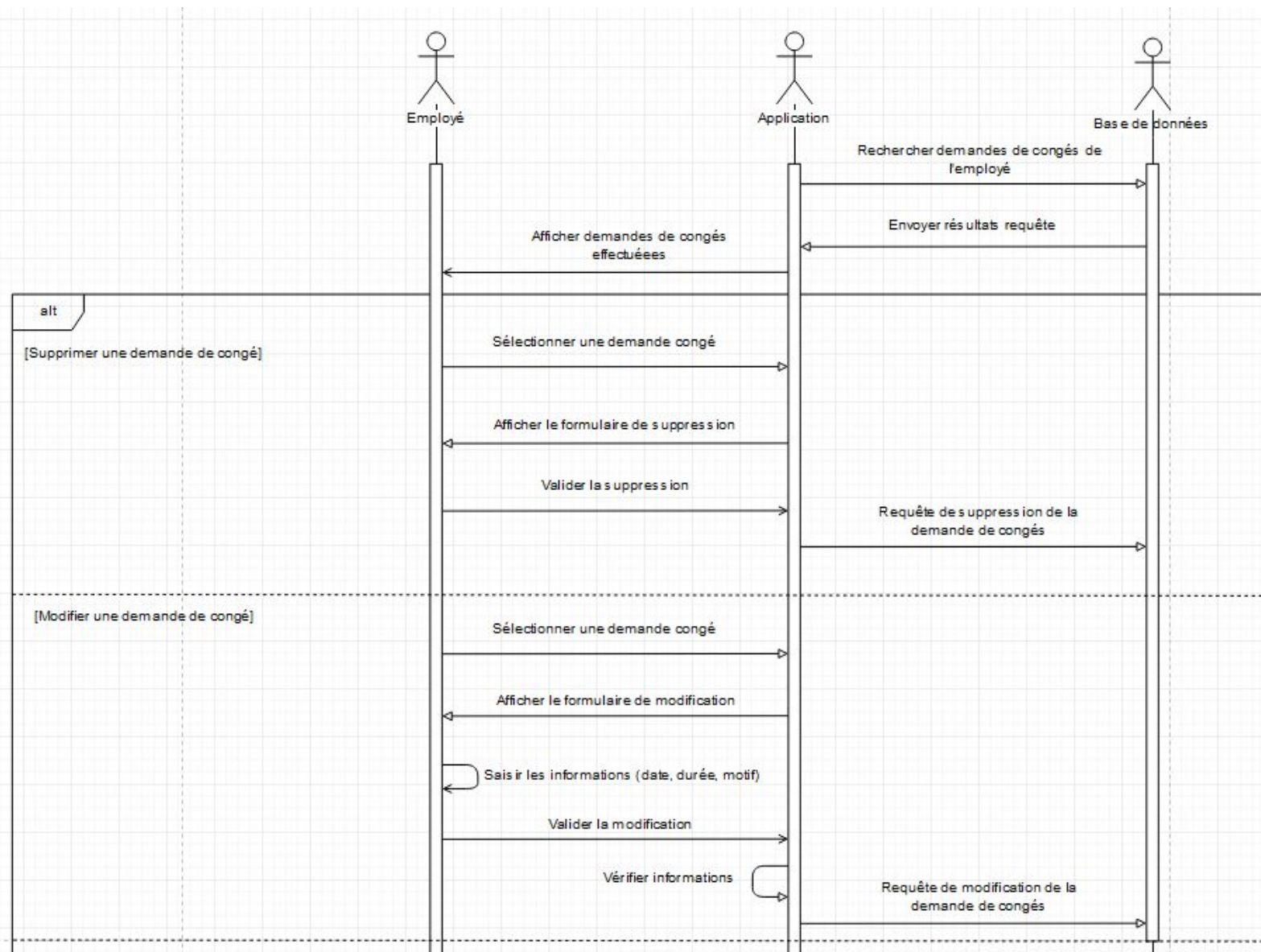


Figure 48 : Première partie du diagramme de séquence relatif à la gestion d'une demande de congés côté employé

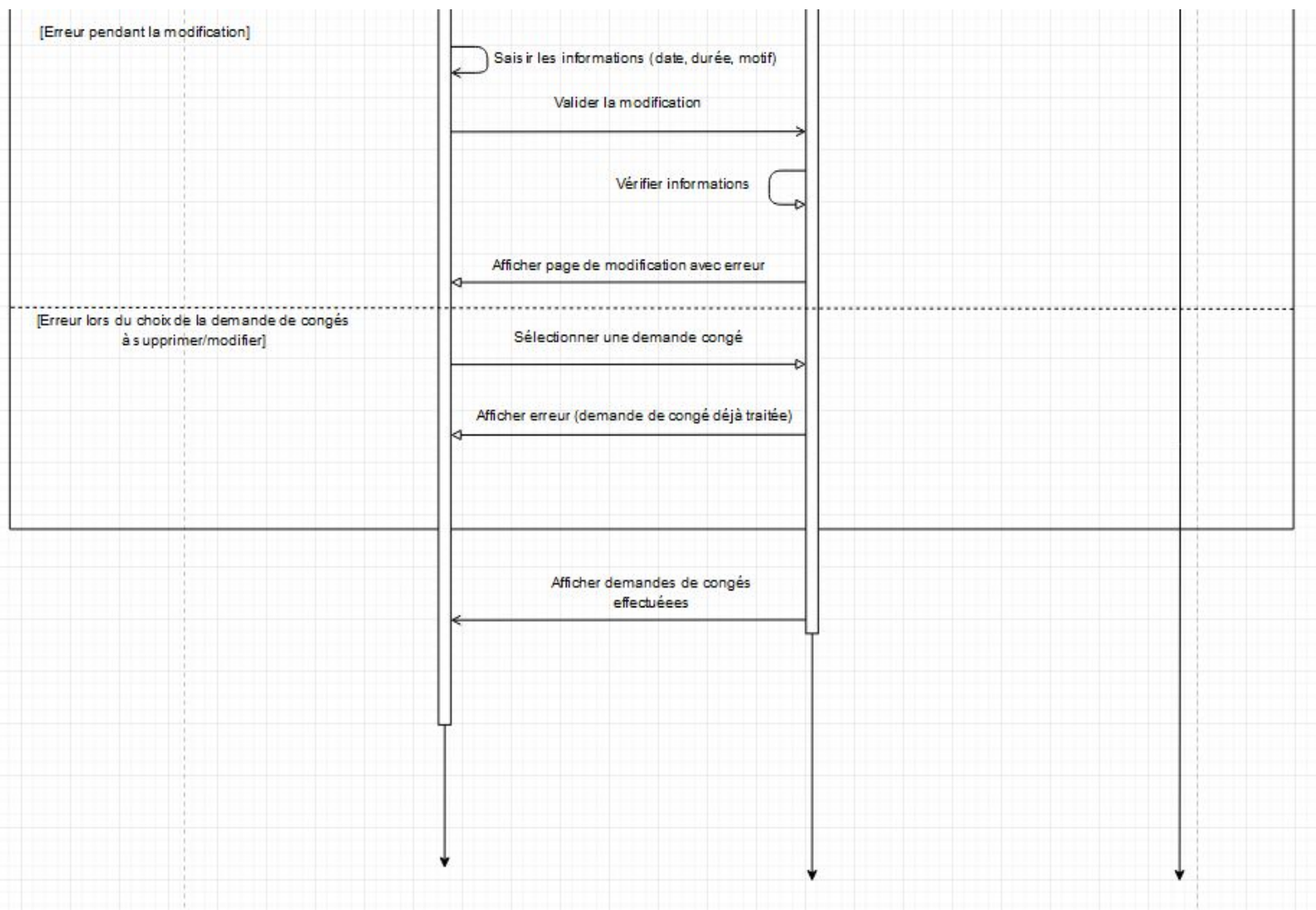


Figure 49 : Seconde partie du diagramme de séquence relatif à la gestion d'une demande de congés côté employé

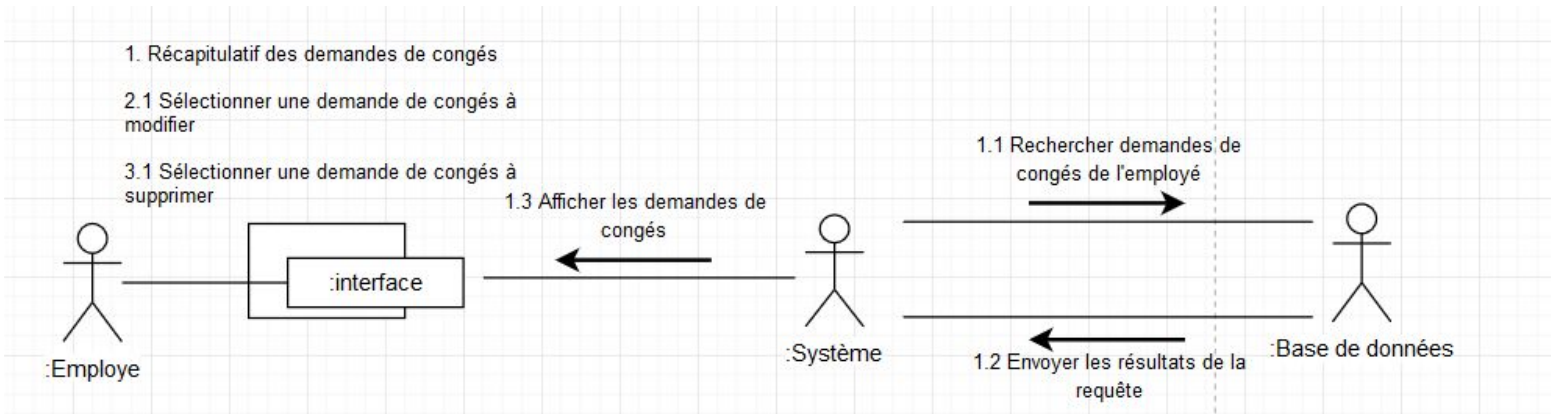


Figure 50 : Première partie du diagramme de communication relatif à la gestion d'une demande de congés côté employé

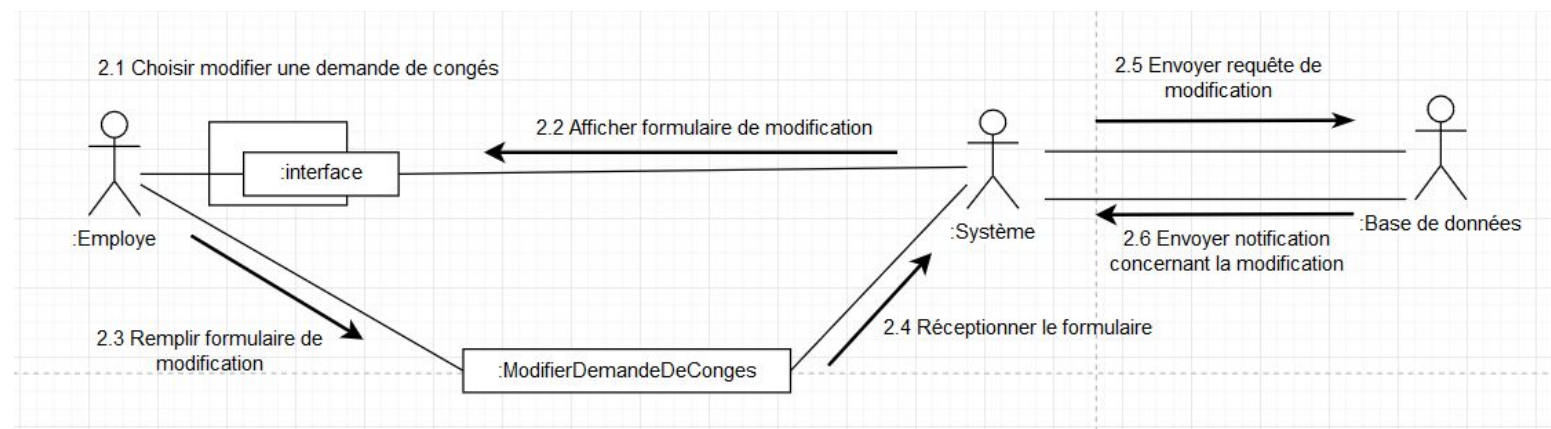


Figure 51 : Deuxième partie du diagramme de communication relatif à la gestion d'une demande de congés côté employé

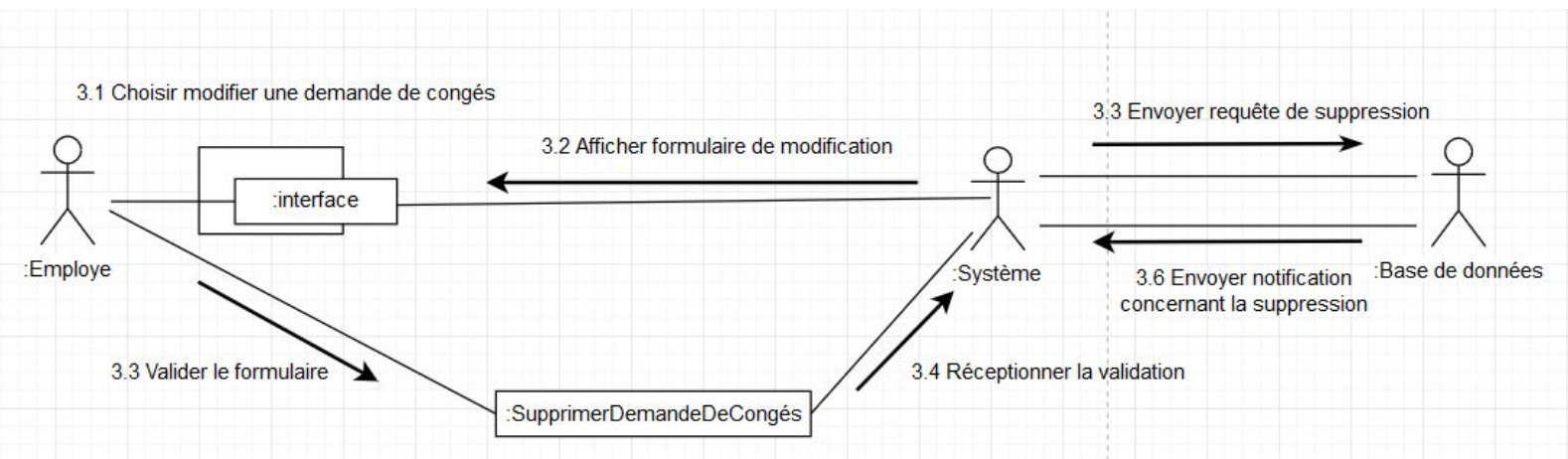


Figure 52 : Troisième partie du diagramme de communication relatif à la gestion d'une demande de congés côté employé

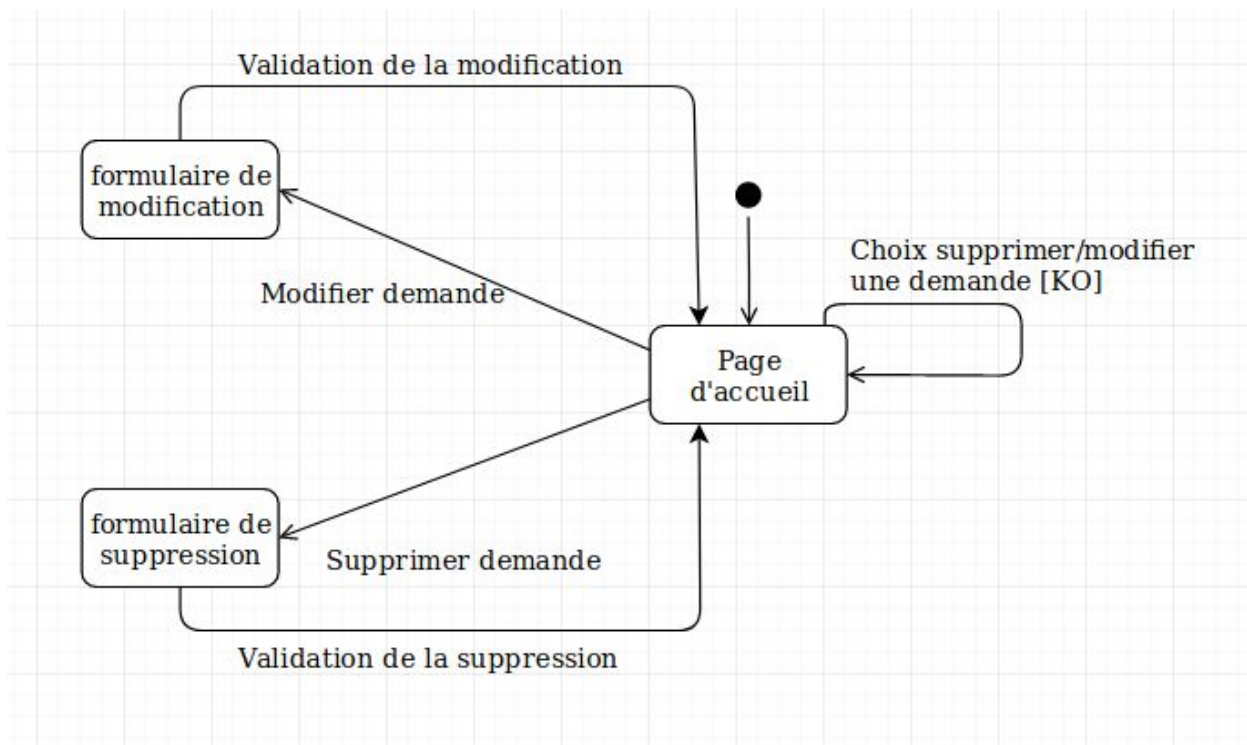


Figure 53 : Diagramme de Transition-Etat relatif à la gestion d'une demande de congés côté employé

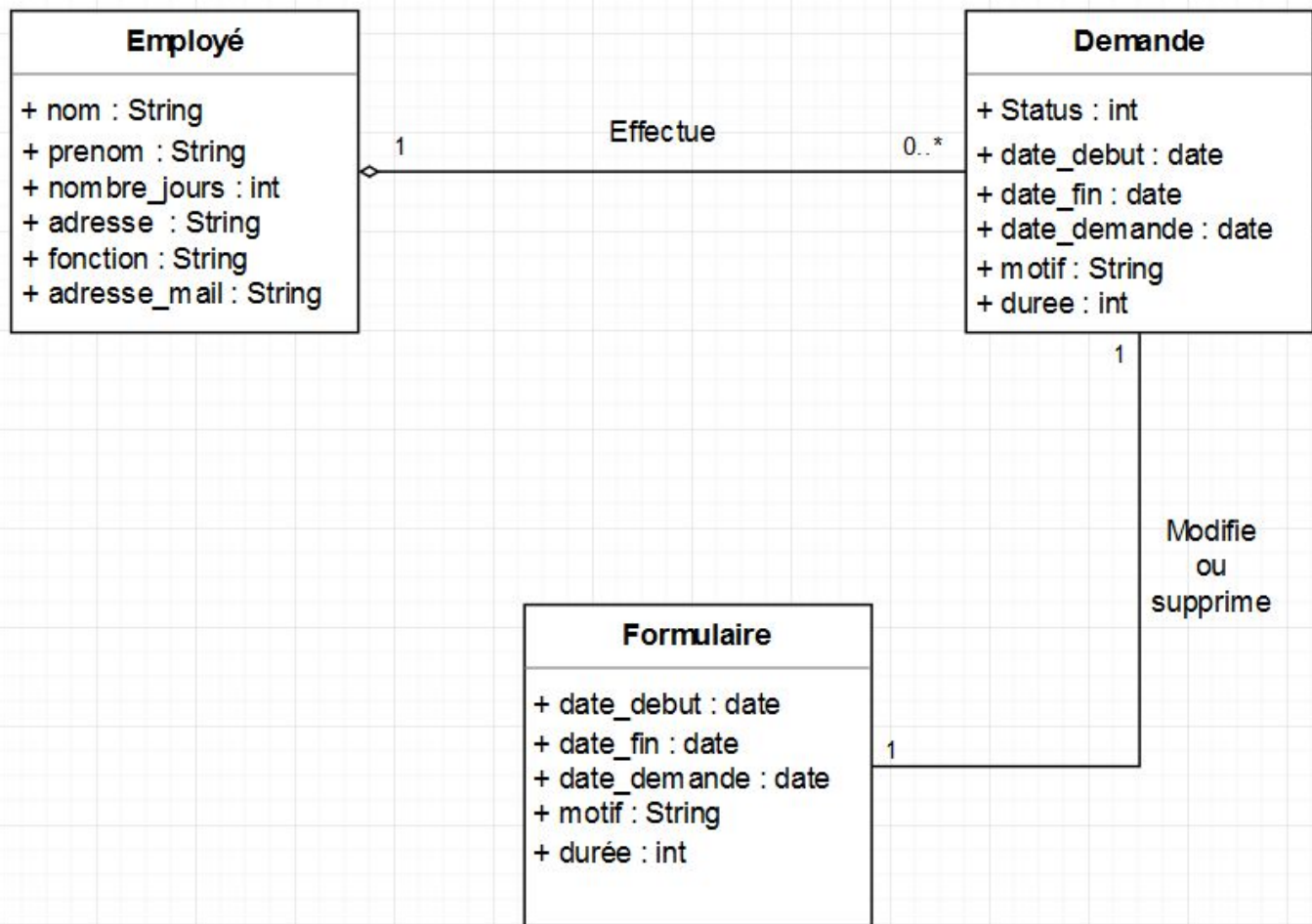


Figure 54 : Diagramme de Classe relatif à la gestion d'une demande de congés côté employé

Annexe 2 : Déposer une demande de congés

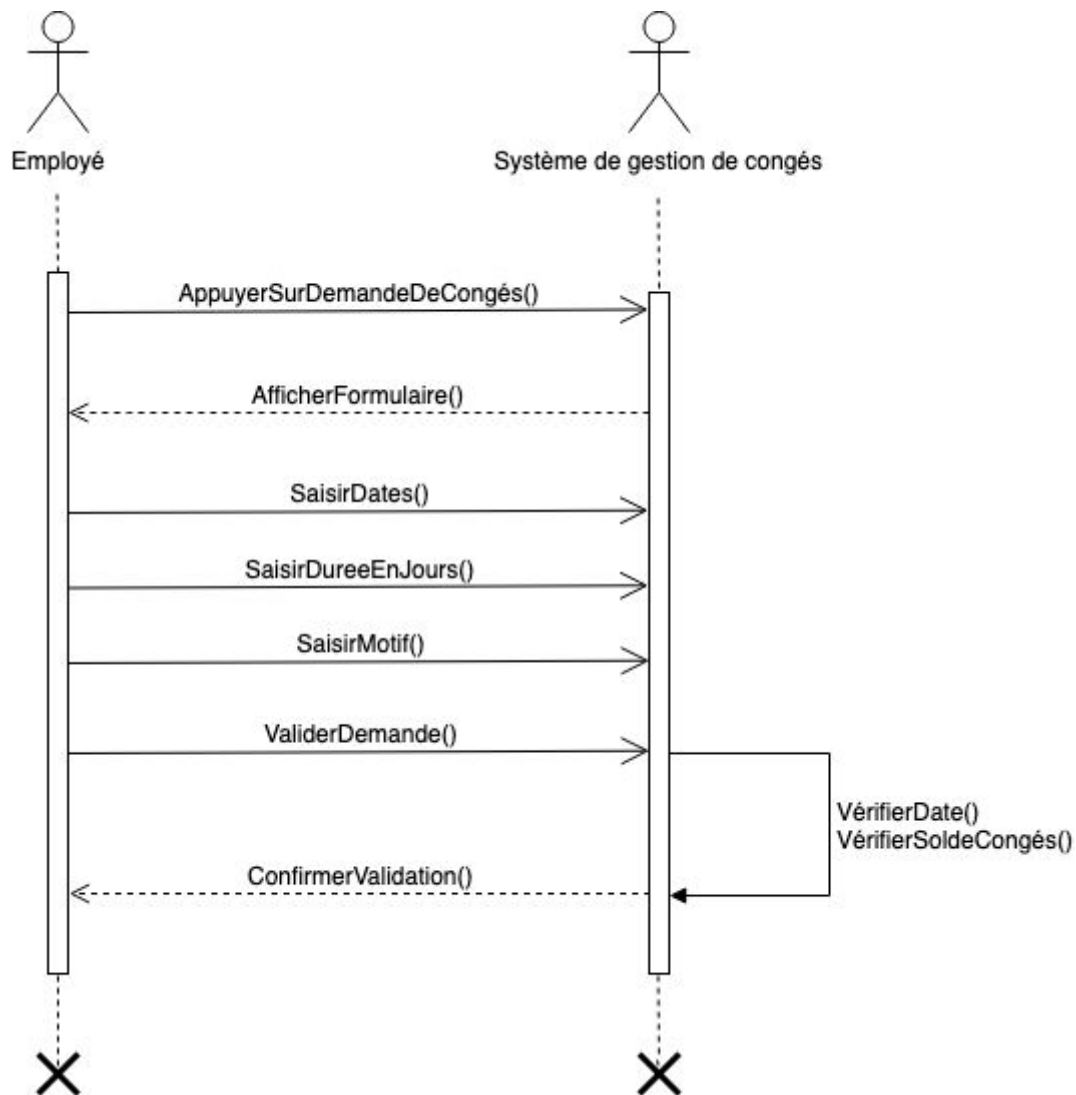


Figure 55 : diagramme de séquence relatif au dépôt d'une demande de congés

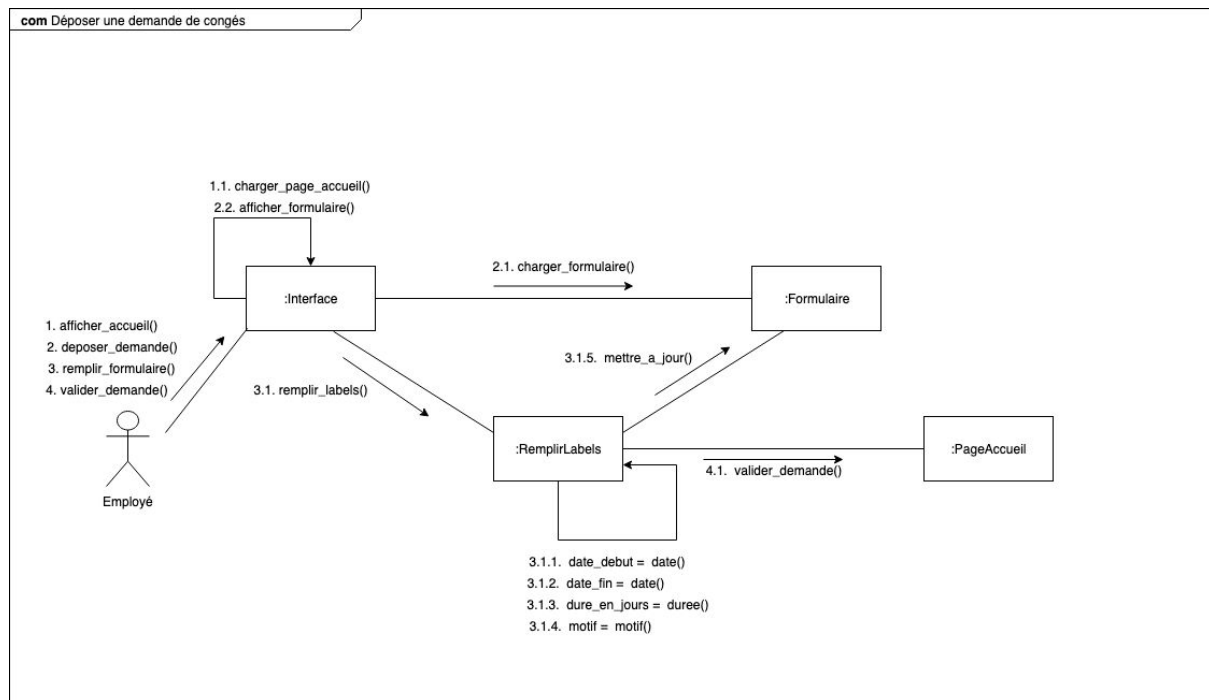


Figure 56 : diagramme de communication relatif au dépôt d'une demande de congés

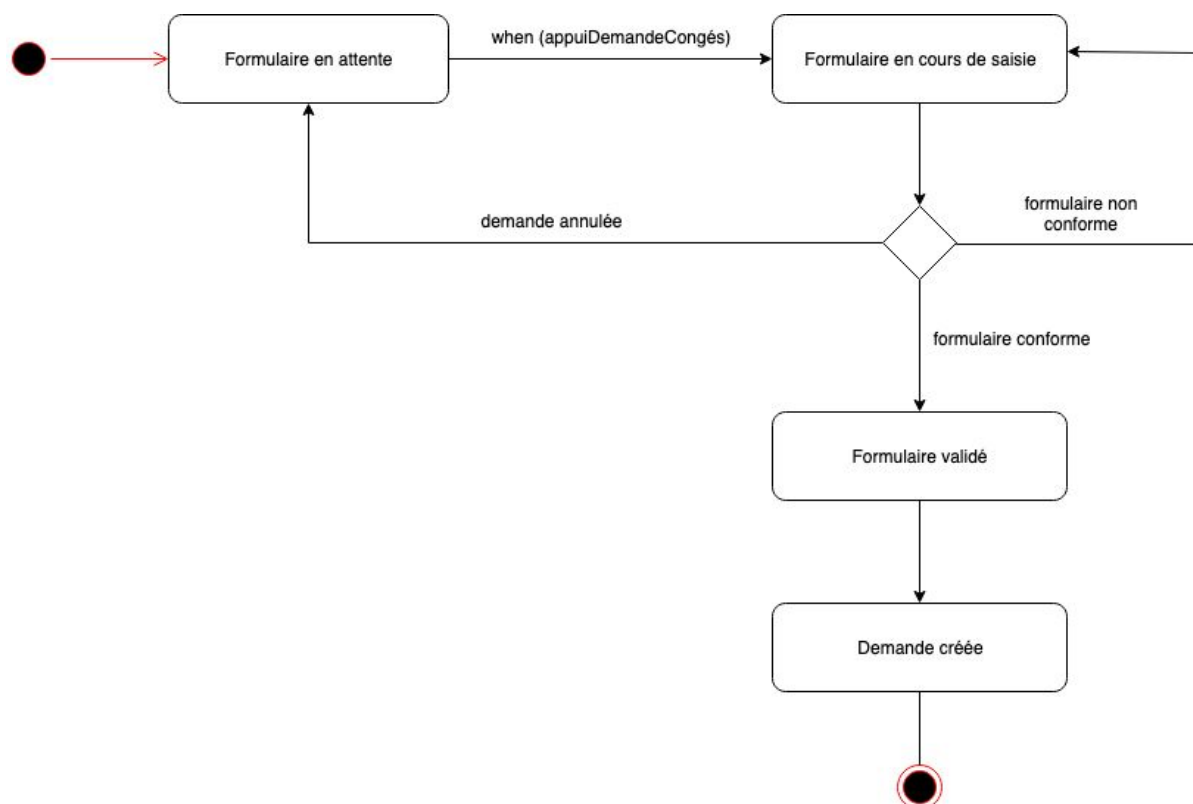


Figure 57 : diagramme d'état transition relatif au dépôt d'une demande de congés

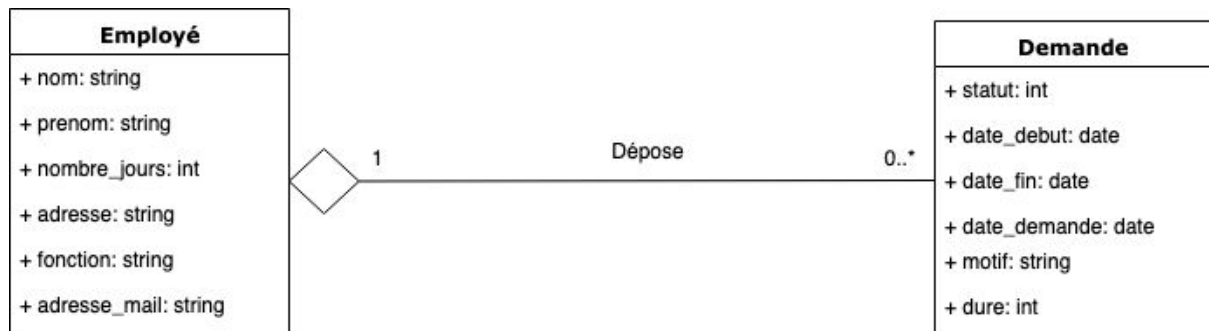


Figure 58 : diagramme de classe relatif au dépôt d'une demande de congés

Annexe 3 : Administrer une demande de congés par un membre du service RH

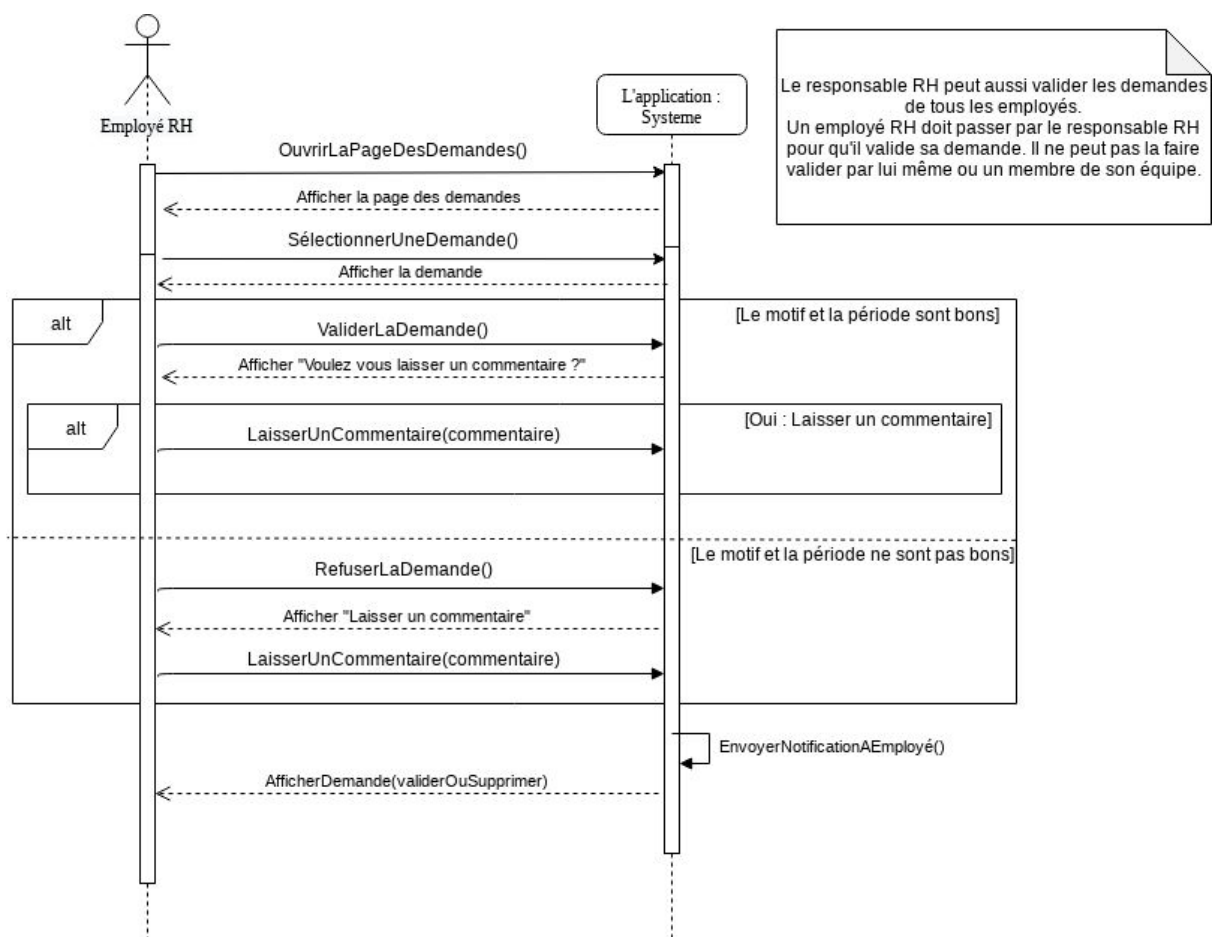


Figure 59 : diagramme de séquence relatif à l'administration d'une demande de congés par un membre du service RH

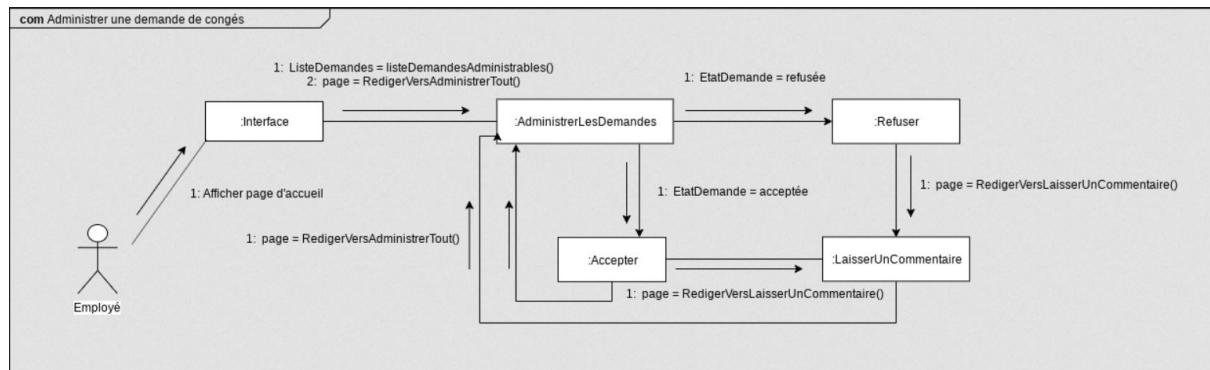


Figure 60 : diagramme de communication relatif à l'administration d'une demande de congés par un membre du service RH

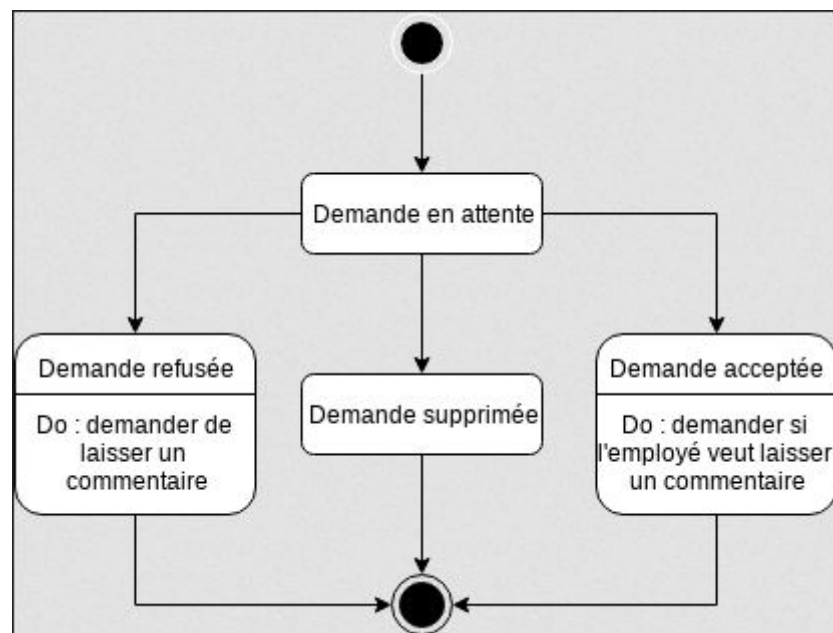


Figure 61 : diagramme d'état transition relatif à l'administration d'une demande de congés par un membre du service RH

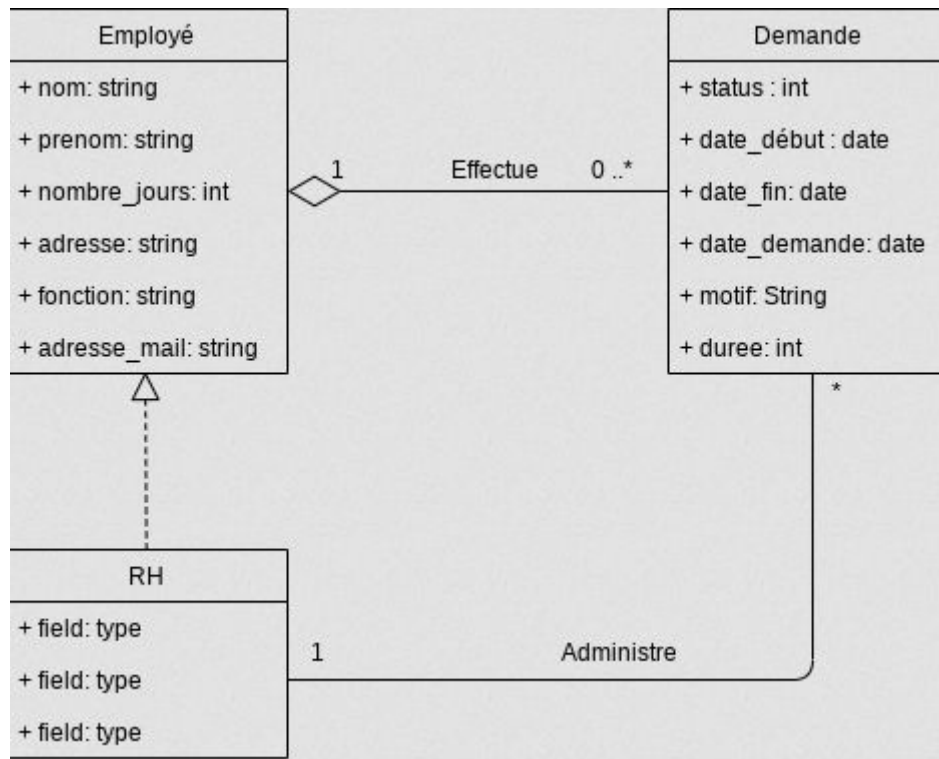


Figure 62 : diagramme de classe relatif à l'administration d'une demande de congés par un membre du service RH

Annexe 4 : Gérer une fiche employé

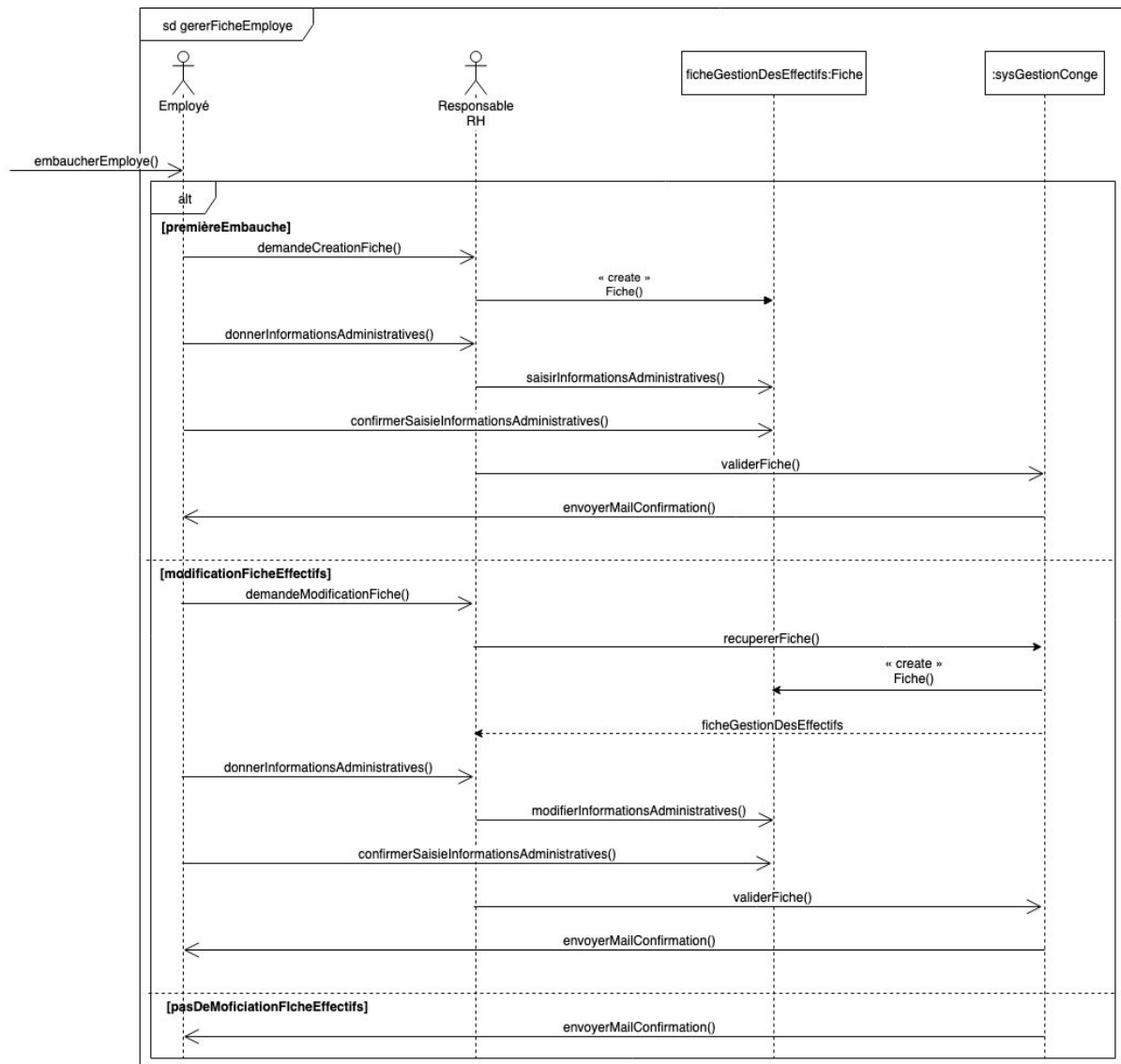


Figure 63 : diagramme de séquence relatif à la gestion d'une fiche employé

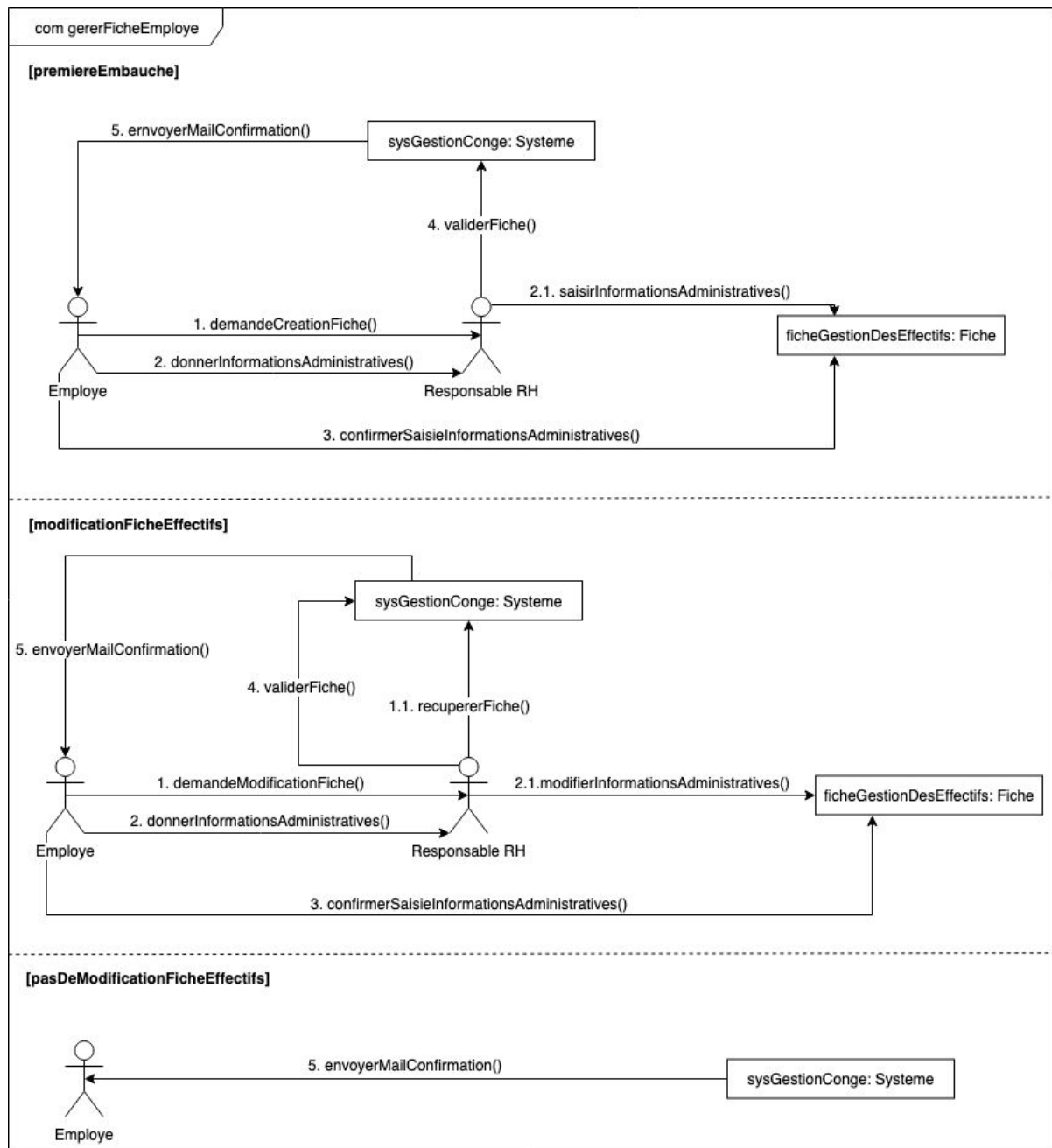


Figure 64 : diagramme de communication relatif à la gestion d'une fiche employé

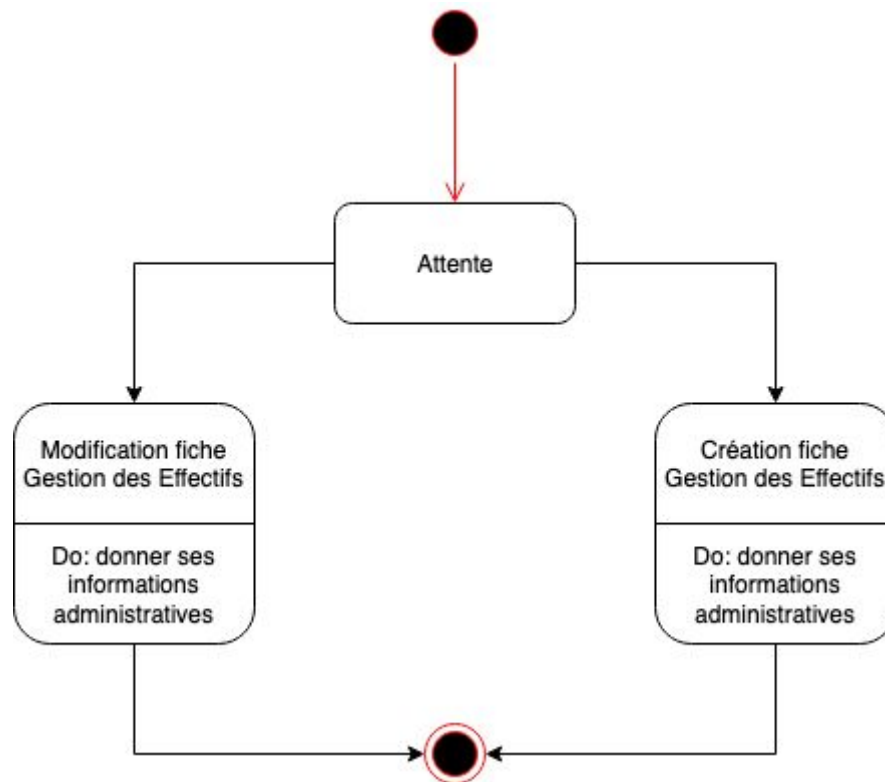


Figure 65 : diagramme d'état transition relatif à la gestion d'une fiche employé

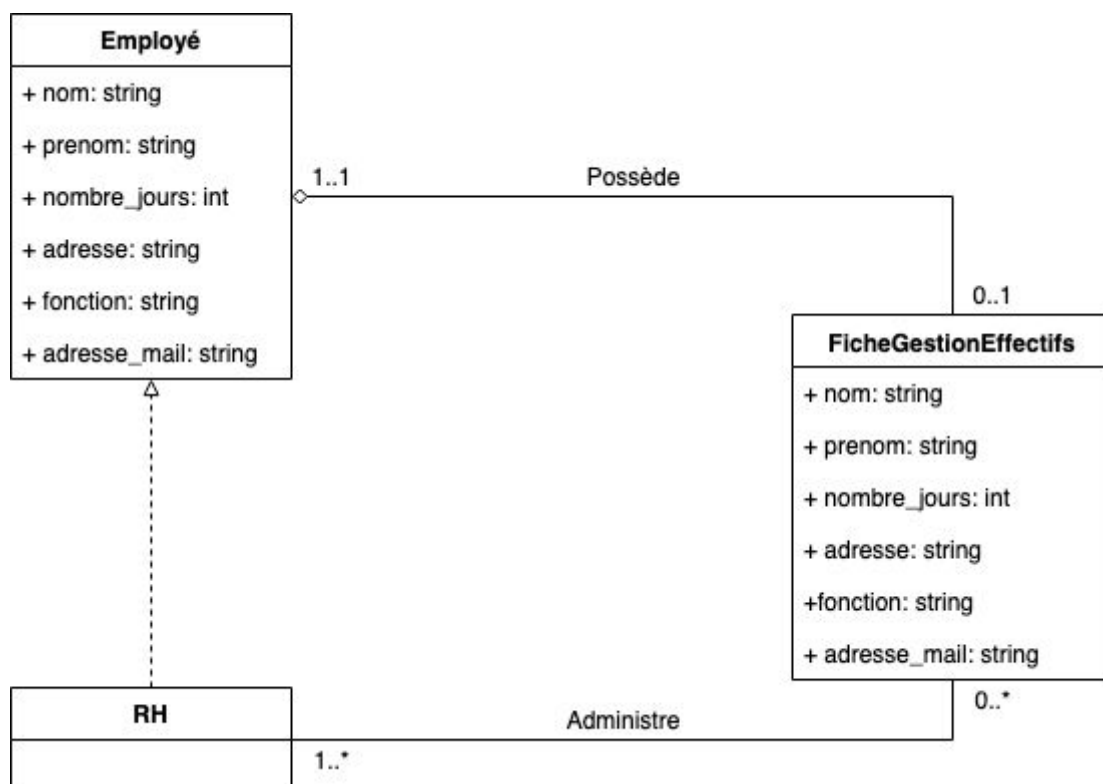


Figure 66 : diagramme de classe relatif à la gestion d'une fiche employé

Annexe 5 : Afficher les informations relatives aux demandes de congés

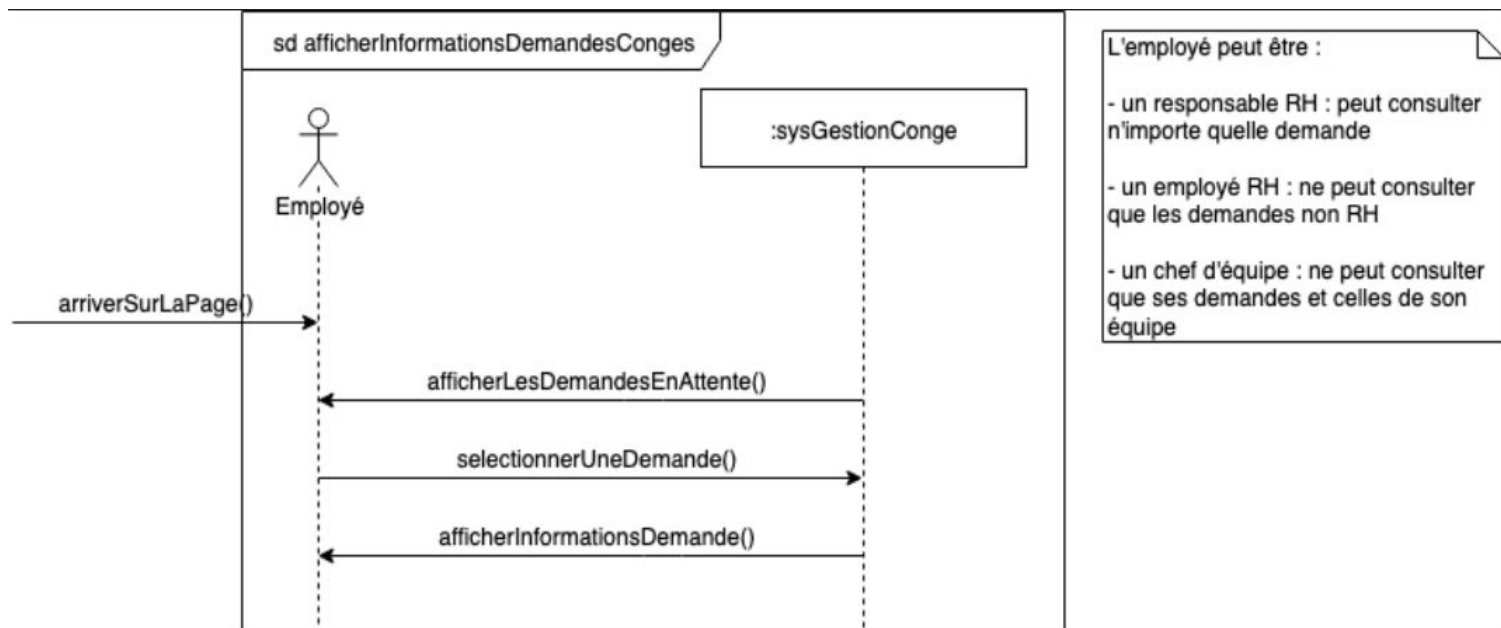


Figure 67 : diagramme de séquence relatif aux demandes de congés

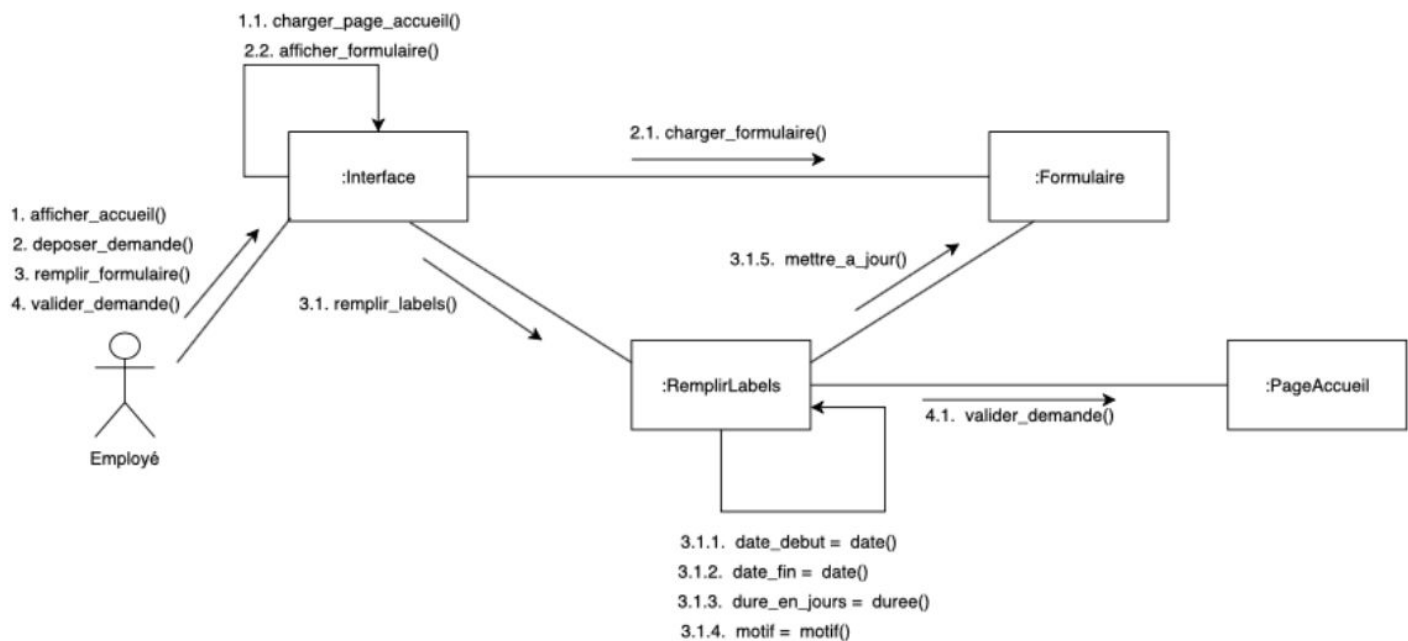


Figure 68 : diagramme de communication relatif aux demandes de congés

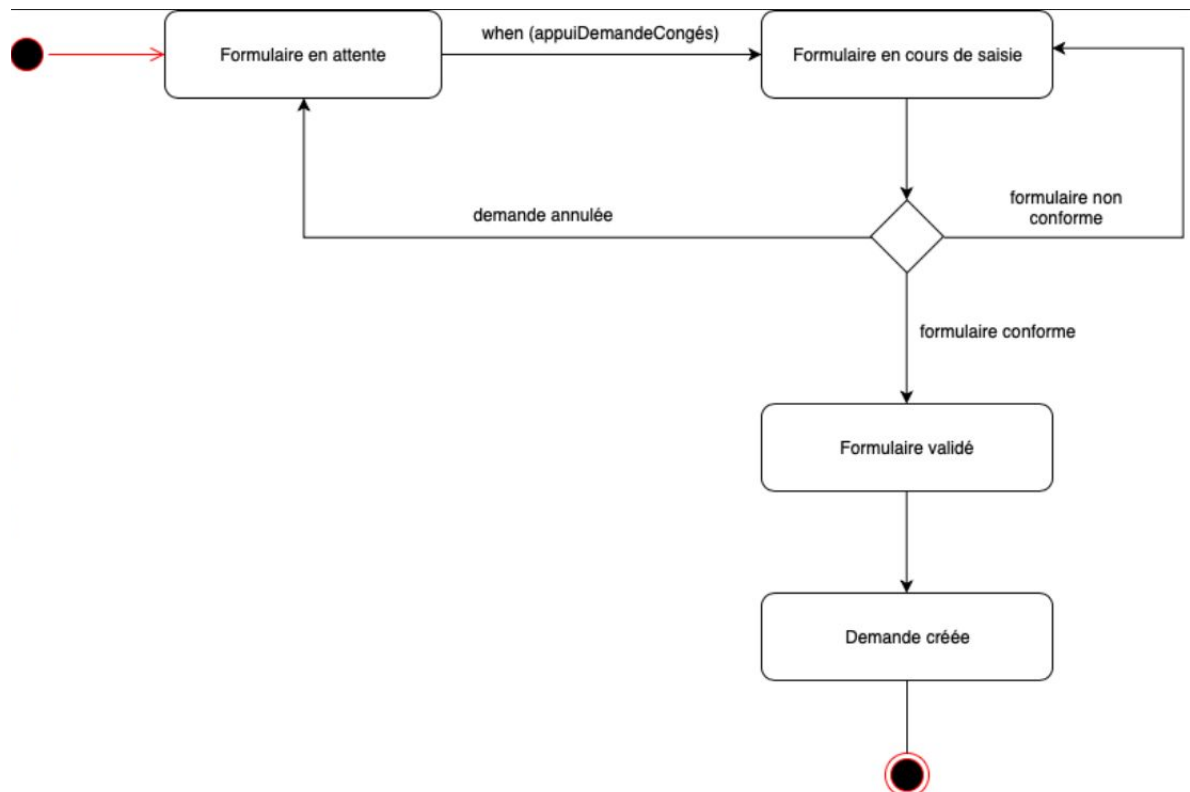


Figure 69 : diagramme de transition-état relatif aux demandes de congés

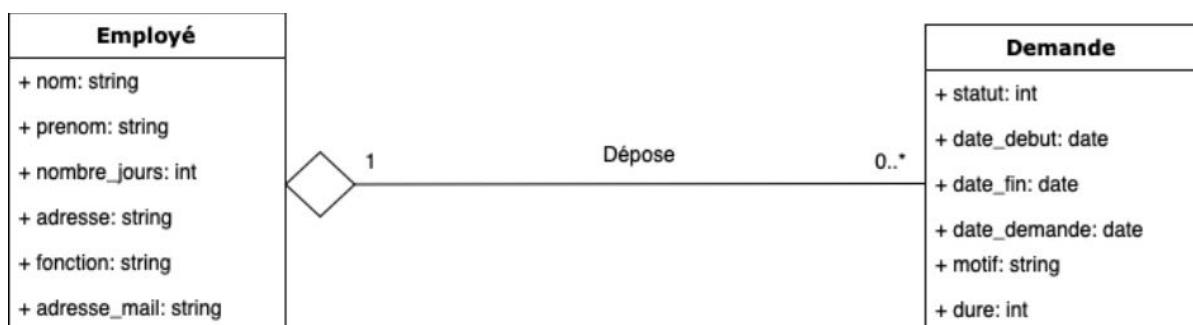


Figure 70 : diagramme de classe relatif aux demandes de congés