Table des matières

Introduction générale

L'une de clés de réussite d'une entreprise est la gestion efficace de ses ressources, et l'une des ressources les plus importantes dont ils disposent est le personnel (staff).

Une entreprise doit disposer d'un personnel adéquat pour l'exécution des activités de ses projets. Avoir juste le nombre requis de membres du personnel n'aidera pas à exécuter avec succès ces activités. Ces membres du personnel doivent avoir les compétences, la motivation et la disponibilité nécessaires pour mener à bien un projet donné.

Afin de gérer efficacement ses personnels et ses projets, Prosodie Capgemini Maroc a mise en place une application web nommée « COPWEB » qui remplace l'ancien système caractérisé par :

- La complexité de la tâche (utilisation des fichiers Excel).
- La perte de temps liée à la saisie multiple des données chaque fois.
- Des problèmes de sécurité et de fiabilité des données.
- Le système résiste le changement (n'est pas extensible).

Malgré l'adoption des dernières technologies pour la gestion du personnel et des projets, les demandes de staffing qui sont toujours gérées à l'aide des fichiers Excel et envois et réceptions des mails remontent beaucoup de problèmes qu'il faut résoudre.

C'est dans ce contexte s'intègre mon projet de fin d'études, qui a pour objectif la mise en place d'une solution permettant de remplacer le système actuel basé sur des fichiers Excel par une application web qui rend la gestion des demandes de staffing facile et efficace.

Pour mener à bien cette application, nous avons tout d'abord effectué une phase d'étude de l'existant afin de définir ses problèmes. Après nous avons spécifié les besoins fonctionnels et les besoins non fonctionnels. Ensuite, nous avons procédé à sa conception ainsi qu'aux choix technologiques pour sa réalisation pour enfin la mettre en œuvre.

D'autre part, la mise en place d'un projet d'une telle ampleur nécessite une bonne organisation et une méthodologie. Nous avons donc décidé de suivre la méthode Scrum.

Ce rapport s'articulera donc, autour de cinq chapitres. Le premier chapitre présentera l'organisme d'accueil, exposera l'étude de l'existant, mettra l'accent sur les critiques de l'existant et introduira finalement l'application de demandes de staffing.

Le second chapitre définira le périmètre du projet, ensuite spécifiera les besoins fonctionnels, non fonctionnel et finalement les besoins techniques.

Le troisième chapitre sera consacré à la description des diagrammes de classes, des diagrammes de séquence des différents cas d'utilisation et de workflow des demandes de staffing à l'aide de diagramme d'état.

Le chapitre suivant portera sur la réalisation de notre solution, il exposera l'environnement de développement, détaillera les différentes étapes de mise en œuvre de projet et finalement présentera les différentes interfaces de l'application.

Ce rapport sera clôturé par une conclusion générale résumant le travail réalisé et proposant des perspectives dans le but d'améliorer notre travail.

Chapitre 1 : Présentation du cadre général du projet

Introduction

Dans cette première partie, nous commençons tout d'abord par une présentation de l'entreprise d'accueil. Ensuite, nous faisons une étude et description de ce qui existait. Puis, nous mettons l'accent sur les grands choix techniques adoptés. Enfin, une introduction de module de gestion de demandes de staffing.

1. Présentation de l'organisme d'accueil

1.1. Le groupe Capgemini

Créé en 1967 à Grenoble, en France, par Serge Kampf, **Capgemini** est devenu un leader mondial du conseil, des services informatiques et de l'infogérance.

Le groupe **Capgemini** développe ses activités dans une quarantaine de pays. Un groupe international et résolument multiculturel, **Capgemini** est fort de 200 000 collaborateurs qui font vivre au quotidien les sept valeurs fondatrices du Groupe : l'honnêteté, l'audace, la confiance, la liberté, la solidarité, la simplicité et le plaisir.

Capgemini offre une large palette de prestations organisées autour de quatre grands secteurs :

- **Le conseil en management (CS)** : aider les entreprises et organisations à identifier, structurer et mettre en œuvre les transformations qui amélioreront durablement leurs résultats et leur compétitivité.
- **L'intégration de systèmes (TS)** : concevoir, développer et mettre en œuvre tous types de projets informatiques comportant l'intégration de systèmes complexes et le développement d'applications informatiques.
- **L'infogérance (OS)** : Prendre en charge totalement ou partiellement, pour plusieurs années, le système d'information d'un client ou d'un regroupement de plusieurs clients tant en matière d'applications que d'infrastructures, et les activités métiers s'y rattachant.
- **Les services informatiques de proximité (SOGETI)** : fournir des services informatiques répondant aux besoins de proximité en matière d'infrastructures, d'applications, d'ingénierie, de tests et d'exploitation.

1.2. Fiche technique de Capgemini

Capgemini est la première entreprise de services du numérique (ESN) dans le secteur des services informatiques en France et est la neuvième mondial d'après le Global Outsourcing 100 en 2014. Elle a été créée par Serge Kampf le 1er octobre 1967 à Grenoble(France) sous le nom de SOGETI (Société pour la Gestion de l'Entreprise et Traitement de l'Information).

Création	1967

Fondateur	Serge Kampf
Forme juridique	Société anonyme à conseil d'administration
Slogan	*People matter, results count*. (l'homme est vital, le résultat capital.)
Siège social Maroc	1100, bd El Qods Casanearshore, shore 8, Imm A
Direction	Paul Hermelin, président-directeur général
Activité	Entreprise de services du numérique (ESN)
Produits	Conseil en management, Intégration de systèmes, Infogérance, Services informatiques de proximité
Filiales	SOGETI, Capgemini Consulting, SASP Biarritz olympique
Effectif	200 000 (2017)
Site web	http://www.capgemini.com

1.5. Présence internationale

Le groupe Capgemini est présent dans une quarantaine de pays, ceci est représenté dans la figure ci-dessous :

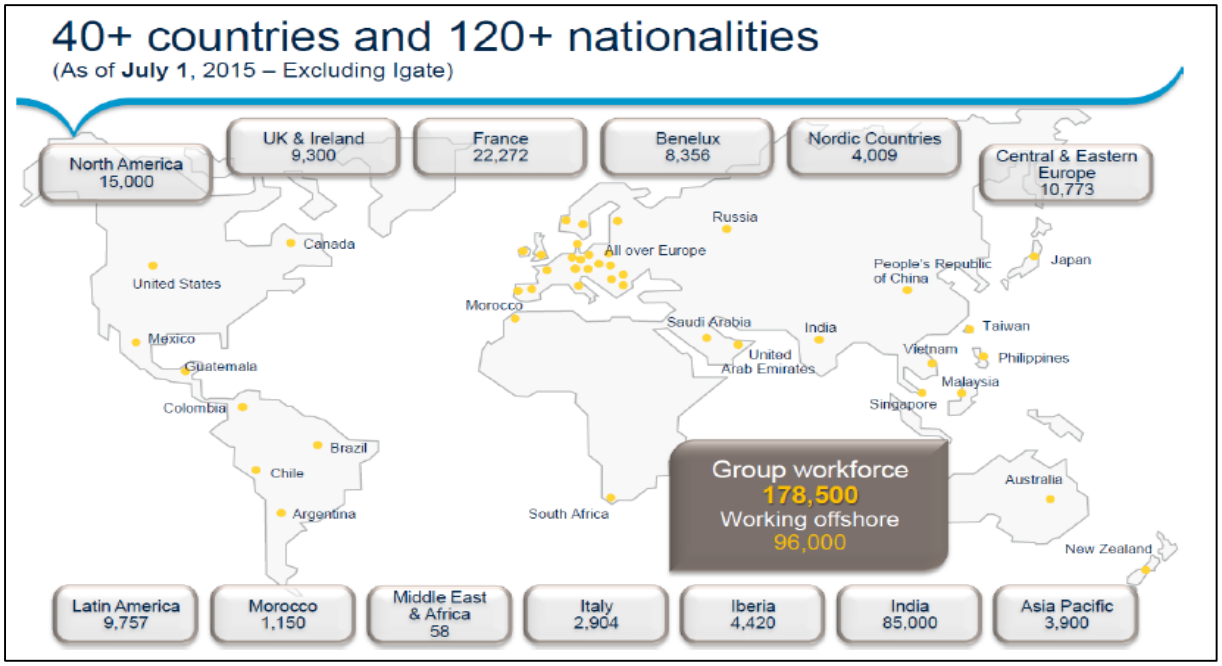

Figure 1 - Présence internationale de Capgemini

1.6. Capgemini Technology Services Maroc

L'année 2007 a été marquée par la création de **Capgemini Technology Services** Maroc avec une première agence à Casablanca. En janvier 2010, Capgemini TS Maroc a déménagé vers les nouveaux locaux de Casa Nearshore et a lancé en juillet son activité Infrastructure Management (activité SOGETI). Capgemini compte aujourd'hui plus de 1500 collaborateurs au Maroc. Le développement de cette filiale marocaine s'inscrit dans le cadre du système mondial de production de Capgemini Rightshore.

1.6.1. Métiers de Capgemini Technology Services Maroc

Application Services AS : Capgemini Technology Services Maroc réalise des prestations liées au développement et la maintenance des systèmes d'information autour de solutions techniques mettant en œuvre les nouvelles technologies (Java/J2ee, .Net, SharePoint, etc.), la solution de gestion documentaire, le Cobol/Mainframe, etc.

Ces prestations incluent les activités de conception technique, réalisation et tests des projets de construction de solutions logicielles, support, corrections et évolutions pour les projets de maintenance de solutions logicielles, conception et exécution de scénarios de tests de projets de tierce maintenance applicative (TMA).

Pour garantir une qualité optimale, la gouvernance de Capgemini Maroc est renforcée par :

- **Le Directeur des Opérations** : pilote les activités de production et avant-ventes.
- À ce titre, il coordonne le recrutement, la formation, le staffing, dans un souci de tenu du délai, de productivité et de satisfaction client.
- **Le Ressource Manager** : fluidifie les staffing des projets en fonction des creux et des montées en charge ou des besoins d'expertise. Il pilote la matrice des compétences du centre et en déduit le plan de recrutement.
- **Le Skill Group Manager** : pilote les activités et les ressources pour le compte des clients. Il est le contact privilégié pour : les questions RH (carrière, parcours, formation, etc.), les questions administratives (congés, déplacement, etc.).

Infrastructure Services (IS) : l'infrastructure Services est une activité qui a débuté chez Capgemini Technology Services Maroc le 1er octobre 2010.

Les activités de l'IS sont axées autour des infrastructures techniques permettant aux utilisateurs de ses clients de pouvoir travailler sur leurs applications métier. Nous trouverons donc, dans le périmètre de l'IS les activités destinées à la maintenance opérationnelle des postes de travail des utilisateurs, des plateformes hébergeant les applications métiers clients, du réseau ainsi que des outils utilisés.

Ainsi nous trouverons au sein du métier IS les activités suivantes :

- Le Service desk.

- La Supervision et le Pilotage d'exploitation.
- L'Ingénierie de production.
- L'Administration des serveurs, des bases de données, etc.

En plus des divisions opérationnelles (Strategic Business Units (SBU)) : AS et IS, trois fonctions supports sont parmi les métiers de Capgemini Technology Services Maroc :

- **Ressources Humaines** : le département Ressources Humaines est le premier point de contact d'un nouveau collaborateur avec CAPGEMINI depuis le début du processus de recrutement, intégration, gestion de la carrière et le développement au sein de Capgemini.
- **Administration et Finance** : est le contact privilégié pour toute demande d'ordre administratif : contrat de travail, attestations, bulletins de paie, assurance maladie, retraite, fournitures de bureau, dossier visa, réservation, hébergement, dotation de voyage et remboursement des frais de déplacement, etc.
- **Informatique et Système d'information IT** : gère tout le parc informatique AS et IS.

La figure suivante présente l'organigramme des unités opérationnelles du groupe Capgemini.

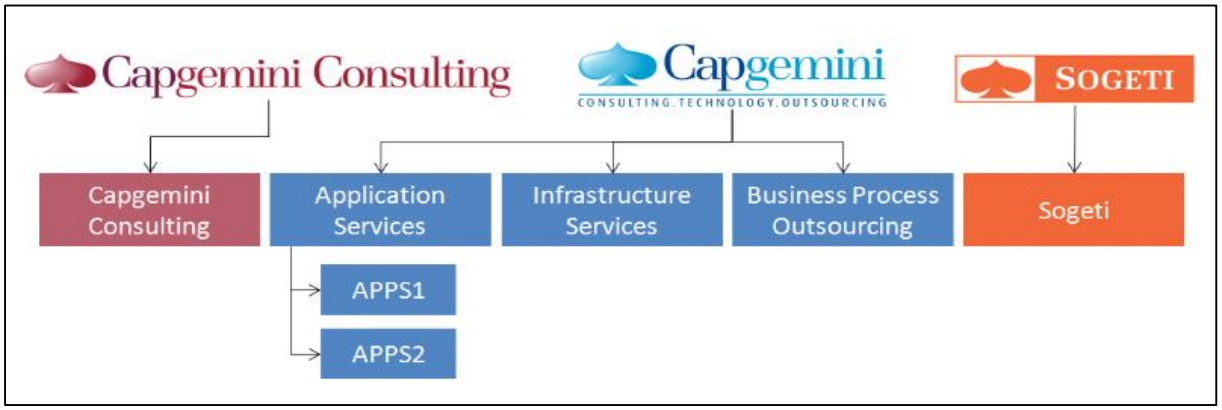

Figure 2 - Organigramme des unités opérationnelles du groupe Capgemini

Et la figure ci-dessous montre la représentation schématique des liens fonctionnels, organisationnels et hiérarchiques de Capgemini Maroc.

Figure 3 - Représentation schématique des liens fonctionnels et hiérarchiques de Capgemini Maroc

1.7. Prosodie

Prosodie est une entreprise française du secteur technologies de l'information et de la communication, spécialiste de l'informatique et des télécommunications acquise par Capgemini en 2011.

Afin d'étendre ses activités, Prosodie a décidé de créer un centre de service multi-pôles au sein de Capgemini Rabat en 2013, ce centre de service est constitué de trois pôles qui sont :

- **Pôle web** : ce pôle est responsable du développement des projets web multi-technologie au sein de prosodie.
- **Pôle TRA** : En mars 2013, Prosodie a lancé la mise en place d'une équipe de Tierce Recette Applicatif (TRA) en s'appuyant notamment sur des ressources Capgemini au Maroc. La Tierce Recette Applicatif est une usine de tests de bout en bout (multi-clients, multi-projets, multi-technologies) notamment les applications SVI et applications web.

- **Pôle SVI** : Ce pôle est responsable du développement de solutions vocales interactives permettant de comprendre le langage humain et acheminer les appels selon les besoins clients, il s'agit d'un projet de développement et maintenance corrective et évolutive des applications vocales et frontales pour plusieurs grands comptes en France. Ce pôle a été mis en place au Maroc en mai 2014.

En effet mon stage s'est déroulé au sein de **Prosodie Maroc** dans le cadre de pôle web.

1.8. Moyens humains

Le suivi de l'équipe et l'organisation du projet sont illustrés sur le tableau suivant :

Tableau 1 - Suivi de l'équipe de développement

	Nom et Prénom	Rôle
Comité de suivi	Pr Arsalane ZARGHILI Pr Jamal KHARROUBI	Encadrant au sein de FST Fès. Encadrant au sein de FST Fès.
	M. Otmane BENKIRANE Mme. Sandy RODRIGUEZ	Delivery Manager, encadrant au sein de Prosodie Capgemini. Operation Delivery Manager, Prosodie France.
Maîtrise d'ouvrage	M. Rodolphe MARAVAL	Directeur de Groupe Capgemini, Ressouces Manager, Directeur de projet.
Maîtrise d'œuvre	Yamin MECHQI Youssef EL FALLAQY	Ingénieurs d'études et développement. Stagiaire ingénieurs d'études et développement.

2. Étude de l'existant

Prosodie Capgemini se sert pour la gestion de projets des fichiers Excel avec des Macros qui font appel à la base de données. Chaque collaborateur a un *Capacity Operational Planning* (COP). Son COP contient les différents projets qui lui sont affectés, leurs timings avec la charge que le collaborateur doit passer dans le projet entre 0.1 et 5.0, avec 0.1 présente une heure et 1.0 une journée, chaque collaborateur travaille 5 jours/7 et l'affectation de la charge se fait par semaine.

2.1. Modules existants dans le fichier Excel

La version Excel de COPWEB se compose de plusieurs modules, chaque module est représenté par une feuille du fichier Excel. Ces modules permettent la gestion des collaborateurs et des projets à l'aide des Macros.

2.1.1. BDD Utilisateur

Le premier module permet l'accès aux différentes informations des collaborateurs telles que :

- Nom et Prénom : le nom et le prénom de collaborateur.
- Fonction : la fonction de collaborateur (exemple : Ingénieur Développement, Chef de
- Projet, Administrateur Technique...).
- Direction : La direction de collaborateur (Skill Technical Ressources, Client Delivery, etc.).
- Groupe : Le groupe de collaborateur (Cloud, etc.).
- Centre : Le centre où le collaborateur travaille (Rabat, Casablanca, Lyon, etc.).

De plus, ce module permet la gestion des collaborateurs, l'accès à leurs compétences et l'exécution des filtres sur leurs informations.

2.1.2. BDD Projets

Ce module permet d'accéder aux informations des projets Prosodie, ainsi que d'effectuer des recherches avec des filtres sur les projets avec des différents paramètres tels que la direction, le chef de projet, etc.

2.1.3. COP Individuel

Ce module contient la charge que le collaborateur doit effectuer sur chacun des projets dont il est affecté. Il contient le planning hebdomadaire des collaborateurs pour les mois prochains.

La figure suivante montre un exemple du COP individuel d'un collaborateur (format Excel).

PREVISIONNEL

(Planning hebdo) (Calendrier : France)

Prénom Nom: Jean-François GIMENEZ

Date: 02/05/2017

Catégorie	Projet	Détail	OT	Planifié	2017 mai 01 S18 04	08 S19 4	15 S20 5	22 S21 4	29 S22 5	juin 05 S23 4	12 S24 5	19 S25 5	26 S26 5	juil 03 S27 5	10 S28 4	17 S29 5	24 S30 5	31 S31 5	août 07 S32 5	14 S33 4	21 S34 5	28 S35 5	sept 04 S36 5	11 S37 5	18 S38 5	25 S39 5
Congés / Absences	Prosodie / Congés & RTT	-	4-DDS	24 j	1	2	1	1	1	1	1	1	1						5	4	5					
Total Congés / Absences				24 j	1	2	1	1	1	1	1	1	1						5	4	5					
DIGITAL RUN	Rexel / Energeasy Connect - RUN prestations (MCO)	-	51218-RE	4 j						1	1	1	1													
	Rexel / Energeasy Connect - RUN prestations (MCO)	-	51218-RE	2 j		2																				
Total DIGITAL RUN				6 j		2				1	1	1	1													
PLATEFORME BUILD	DIGICARE / PF Digicare - Prototype	-	41362-01	23,5 j	2,5			3	3	3	3	3	3	3												
Total PLATEFORME BUILD				23,5 j	2,5			3	3	3	3	3	3	3												
PLATEFORME RUN	GIE Sintia / Récurrent INF - Sintia	-	51166-RE	-																						
Total PLATEFORME RUN				-																						
Probable BUILD	GIE Sintia / Evolutions 2017	-	[?]	5,5 j	0,5			0,5	0,5		0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5								
Total Probable BUILD				5,5 j	0,5			0,5	0,5		0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5								
Projets internes	Prosodie / Industrialisation des outils	COP Web	86402-01	36 j	2			2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2		
Total Projets internes				36 j	2			2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2		
SOLUTIONS RUN	Prosodie / Offres & Comp techniques - CENTREX	-	[?]	-																						
Total SOLUTIONS RUN				-																						
Total général				95 j	6	4	1	6,5	6,5	7	7,5	7,5	7,5	5,5	2,5	2,5	2,5	2,5	7	6	7	2	2	2	-	-

Figure 4 - Exemple du COP individuel d'un collaborateur (format Excel)

2.1.4. Prévis Projet

C'est un module contenant la liste des collaborateurs affectés à un projet donné, une description de responsabilité pour chaque collaborateur ainsi que la somme des charges a effectué.

Ce module permet aux responsables de projet d'avoir une idée sur les charges de projet pour être en mesure de savoir si le projet a besoin de plus ou moins de charge et si besoin de faire une demande de staffing.

2.2. Processus de l'utilisation de l'outil

À l'arrivé d'un nouveau projet, le chef de projet envoie un mail de demande de staffing à un ressources manager, suite à ce mail le manager effectue une recherche au niveau des ressources disponibles en concordance avec le planning de projet et compétences demandées pour l'affectation des charges dans les COP des collaborateurs sélectionnés pour le projet.

2.3. Critique de l'existant

Au fil des années, l'outil Excel s'est étendu aux entreprises à tous les niveaux. Quel que soit le projet mis en place, la question de l'interface avec Excel se pose. Il est trop souvent utilisé pour ce qu'il n'est pas capable de fournir. Adopté en masse, car facile à appréhender et contrôlé par de nombreux utilisateurs, l'outil Excel montre également de plus en plus ses limites lorsqu'il est utilisé de façon excessive. Voici quelque raison pour lesquels, le système actuel de gestion des projets et des ressources doit changer :

1. **La difficulté de mettre à jour les COP**

- Les COP se situent dans un serveur de Prosodie au niveau de l'OnShore (France dans ce cas), l'ouverture des fichiers prend beaucoup de temps.
- Les COP sont des fichiers Excel et peuvent bloquer de temps à autre.
- Après chaque mise à jour l'utilisateur est amené à enregistrer son COP à qui invoque des MACROS pour faire des appels et mises à jour dans la base de données ce qui peut entraîner des problématiques de synchronisation et non-cohérence de données.

2. **Manque de communication des changements de COP**

En effet, si l'administrateur oublie d'envoyer un mail notifiant les changements de COPs, et comme les collaborateurs n'ont pas accès, ils ne sauront pas le changement.

3. **Il est difficile d'affecter les charges aux collaborateurs**

Pour que l'administrateur arrive à affecter les charges aux collaborateurs :

- Il faut consulter tous les COPs de l'équipe.
- Chercher leurs disponibilités en concordance avec le besoin du projet.

4. **Raisons techniques**

L'usage des fichiers Excel génère beaucoup de problèmes comme :

- Des enregistrements qui ne sont pas pris en considération lorsque l'outil plante.
- Lorsque l'outil plante parfois, il faut refaire tout le processus.
- L'inexistence d'un système de gestion de données à part les MACROS qui manquent de flexibilité.

En plus, le fait de mettre tout dans un fichier Excel rend la maintenance très lourde. Après chaque modification, correction ou l'ajout d'une nouvelle fonctionnalité, il faut s'assurer que toute personne reçoit la dernière version de COP.

Une application web est le choix idéal pour remédier à ses problèmes. D'où la spécification fonctionnelle de la solution.

2.5. Introduction du module « Demandes de Staffing »

L'objectif du projet COPWEB, c'est de simplifier l'affectation des collaborateurs Prosodie dans des tâches ou projets selon leurs compétences, ainsi que le suivi des projets. Comme décrit avant [cf. Paragraphe 2.2] lorsqu'un nouveau projet ou un besoin arrive, le chef de projet doit effectuer une demande de staffing auprès d'un RDS.

Nous allons décrire brièvement dans cette section le processus d'une demande de staffing et introduire le module de gestion des demandes de staffing.

2.5.1. Envoi d'une demande de staffing

À l'arrivé d'un nouveau projet, le chef de projet doit faire une demande de staffing en envoyant un mail à un RDS. Le format de mail était sous forme d'une Template que le responsable de projet doit remplir.

Prosodie Capgemini

Demande de Staffing

* : les champs portant cette mention sont obligatoires

Nom du client	INTRUM ◉ Embarqué ○ Prospect Client International ☑ Langue: FR Onshore ☑
Contribution attendue	Dispositif AVVT: ☐ Démo Personnalisée ☐ Soutenance ☐ POC Customisé ☐ POC Innovation Dispositif AVVT Projet: ☐ Etude de faisabilité ☐ Chiffrage fonctionnelle ☐ Chiffrage Technique ☐ Extension de périmètre Dispositif Projet: ☐ New Deal ☐ BCE ☐ Transfert de compte ☐ Montée de version ☐ Migration Etat de la demande: Signé Nature: Run Secteur: Delivery 2 Pilote du projet : Client Delivery
Planning	Date souhaitée : 08/05/2017 ou Période Souhaitée: par ex : S20
Compte	N° d'opportunité : BCS-15510-V01 Code affaire principal INTRUM01
Contexte et enjeux	assurer le run du client
Besoins client	Assurer le run du client Intrum à 0,5J par mois par Mathieu Pasteur (qui est le seul à connaitre le SVI call&pay spécifique de ce client)
Périmètre	22 applis
Documents clés	Plan de charge (Build et Run) ou le COP NAF sont un pré requis INTRUM - Plan de Charge -20170405.xlsx Feuille de calcul Microsoft Excel 25,4 Ko

Figure 5 - Demande du staffing (format Mail)

2.5.2. Réception des demandes de staffing

Lorsque le Resource Manager reçoit un mail, il l'insère dans un fichier Excel manuellement, et il choisit un RDS pour qu'il s'occupe du traitement de la demande et archive les demandes annulées. Enfin, le RDS traite la demande, informe le demandeur et le Resource Manager et finalement effectue les modifications nécessaires pour mettre à jour les COP individuels des membres staffés ou le faire directement à partir du prévis projet.

Parfois le traitement d'une demande nécessite une communication avec le demandeur, et création d'un nouveau projet si le projet de la demande n'existe pas dans BDD projets.

2.5.3. Version Web de « Demandes de staffing »

Notre mission, c'est l'intégration d'un nouveau module dans le COPWEB afin de simplifier d'une part, la création et envoi des demandes du staffing, et d'une autre part la gestion des demandes de staffing.

Conclusion

Dans ce premier chapitre, nous avons pu situer le cadre général du projet, à savoir la l'entreprise d'accueil. De plus nous avons fait une étude approfondie de ce qui existait avant la version web de COPWEB, et de l'existant et à partir laquelle nous avons décrit les différents modules et leurs fonctionnalités. Puis une introduction de module de gestion des demandes de staffing qui fera l'objet de reste du rapport dans le cadre du projet COPWEB. Cette étude a permis ainsi d'introduire les besoins demandés au cours de ce module.

Dans le chapitre suivant, nous allons entamer la phase d'analyse et spécifications des besoins.

Chapitre 2 : Analyse et spécification des besoins

Introduction

Ce chapitre consiste d'abord à déterminer les besoins fonctionnels et les objectifs visés dans le cahier des charges tout en respectant certaines contraintes correspondant aux besoins non fonctionnels de notre projet. Ensuite nous passons aux besoins techniques dans lesquelles nous allons définir l'architecture du projet ainsi que les design patterns et les frameworks utilisés.

1. Périmètre du projet

1.1. Objectifs du projet

L'objectif du projet est d'introduire un nouveau module dans le projet COPWEB qui facilite aux chefs de projets la création et la modification des demandes de staffing ainsi que le suivi de leurs anciennes demandes. Le module doit permettre aussi aux RDS de traiter les demandes de staffing. Il permettra aussi d'offrir un flux de travail d'une demande et le suivi de ce flux ainsi que la gestion des droits sur les demandes.

1.2. Méthodologie de développement

Capgemini aujourd'hui est consciente de l'importance de l'implication de son client dans le processus de développement pour aboutir à un produit finalement utilisable et fonctionnel pour son client, l'entreprise n'a plus peur d'impliquer le client dans son processus de développement de la solution, au contraire, les entreprises aujourd'hui cherchent à appliquer une méthodologie agile, et même la combinaison de plusieurs, par exemple Extreme Programming (XP) et SCRUM, d'une manière très naturelle, il apparaît que les deux sont vivement sollicitées dans le monde professionnel alors qu'une méthodologie comme cascade elle est soit appliqué au sein d'un Sprint soit elle n'est pas du tout utilisée à cause de son manque d'agilité qui ne convient pas aux objectifs stratégiques d'une entreprise.

Malgré que ce soit un projet interne, son importance et utilité implique que notre choix se porte sur la méthode de gestion de projet SCRUM avec intégration continue.

1.2.1 La méthode agile SCRUM

SCRUM [1] est une méthode agile dédiée à la gestion de projets. Cette méthode de gestion a pour objectif d'améliorer la productivité de l'équipe de développement.

La méthode SCRUM (Jeff Sutherland & Ken Schwaber, 2001) s'inscrit dans un mouvement plus vaste : celui de l'agilité. Pour qu'une méthode soit validée comme étant agile, elle doit respecter le manifeste Agile :

- L'importance est donnée :
    - Aux individus et aux interactions plus qu'aux processus et aux outils.

  - À des logiciels immédiatement disponibles plus qu'à une documentation exhaustive.
  - À la collaboration avec le client plus qu'à la négociation contractuelle.
  - À la réactivité face au changement plus qu'au respect d'un plan figé.
- SCRUM repose sur les piliers suivants :
  - Une équipe responsable et en auto-organisation : l'équipe de développement se doit être autonome pour pallier les éventuels changements d'un client trop indécis.
  - Un avancement du produit par une série de « sprints » : les itérations sont courtes, 2 à 4 semaines au plus, pour permettre des interventions rapides en cas de problèmes.
  - Des exigences définies : ce sont les différents éléments à mettre en œuvre. Ces éléments sont regroupés sous l'appellation de « Backlog du produit ».

La figure suivante montre le schéma de SCRUM :

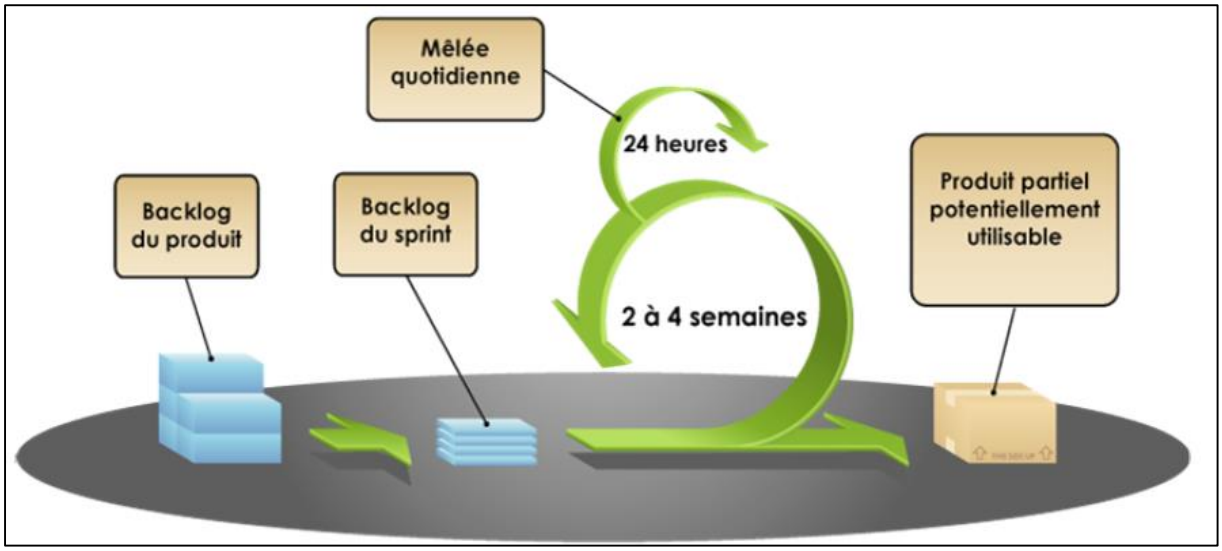

Figure 6 - Schéma du fonctionnement de SCRUM

1.2.2. L'intégration continue

L'intégration continue [2] est un ensemble de pratiques utilisées en génie logiciel consistant à vérifier à chaque modification de code source que le résultat des modifications ne produit pas de régression dans l'application développée. Le concept a pour la première fois été mentionné par Grady Booch et se réfère généralement à la pratique de l'Extreme Programming. Le principal but de cette pratique est de détecter les problèmes d'intégration au plus tôt lors du développement.

De plus, elle permet d'automatiser l'exécution des suites de tests et de voir l'évolution du développement du logiciel.

L'intégration continue repose souvent sur la mise en place d'une brique logicielle permettant l'automatisation de tâches : compilation, tests unitaires et fonctionnels, validation produit,

tests de performance à chaque changement du code, cette brique logicielle va exécuter un ensemble de tâches et produire un ensemble de résultats, que le développeur peut par la suite consulter.

Cette intégration permet ainsi de ne pas oublier des éléments lors de la mise en production et donc améliorer la qualité du produit.

Pour appliquer cette technique, il faut d'abord que :

- Le code source soit partagé (en utilisant des logiciels de gestion de versions tels que Concurrent Versions System (CVS), Subversion (SVN), GIT, Mercurial, etc.).
- Les développeurs intègrent (commit) quotidiennement (au moins) leurs modifications.
- Des tests d'intégration soient développés pour valider l'application.

Les principaux avantages d'une telle technique de développement sont :

- Le test immédiat des modifications.
- La notification rapide en cas de code incompatible ou manquant.
- Les problèmes d'intégration sont détectés et réparés de façon continue, évitant les problèmes de dernière minute.
- Une version est toujours disponible pour un test, une démonstration ou une distribution.

Dans ce projet, nous avons utilisé **Jenkins** comme serveur d'intégration continue pour Java, et **SonarQube** comme logiciel de supervision de la qualité du code.

2. Besoins fonctionnels

2.1. Identifications des acteurs

Dans le cadre de notre analyse, les acteurs que nous avons pu identifier sont :

- **Client Delivery (CD)** : un département regroupant les équipes de management de projets, organisés par secteurs, qui aura pour missions principales d'assurer le bon pilotage de l'ensemble des projets, français et internationaux.
- **Skill and Technical Ressources (STR)** : ce département est focalisé sur la gestion et le bon développement des compétences fonctionnelles et techniques nécessaires au delivery des projets clients. Le RDS appartient à ce département.
- Et un **administrateur**.

2.2. Identifications des cas d'utilisation

L'accès à l'ensemble des fonctionnalités du module requiert que l'utilisateur soit connecté à l'aide de ses identifiants Prosodie.

Les fonctionnalités offertes par le nouveau module sont décrites par les cas d'utilisation suivants :

Tableau 2 - Les cas d'utilisations, leurs descriptions et leurs acteurs

Cas d'utilisation	Description	Acteur
Consulter la liste des demandes	La liste des demandes est un tableau avec quelques informations importantes telles que l'état, demandeur, projet, dates importantes (création, modification, etc.), etc.	Tous les utilisateurs
Filtrer les demandes	Les utilisateurs peuvent effectuer des recherches sur l'ensemble des demandes selon plusieurs informations : - Le RDS - Le Projet - Etc.	Tous les utilisateurs
Créer une demande	L'utilisateur peut choisir de sauvegarder la demande en état brouillon ou de sauvegarder et envoyer la demande pour être traité dans l'état suivant.	Tous les utilisateurs
Consulter une demande	À partir de la liste des demandes, l'utilisateur peut accéder à l'ensemble des détails de la demande. La consultation de la demande inclut l'accès à l'ensemble des commentaires et historique d'actions et modifications effectuées sur la demande.	Tous les utilisateurs
Modifier une demande	À partir de ce cas d'utilisation, l'utilisateur peut soit modifier la demande ou effectuer une	Tous les utilisateurs

	action en fonction de l'état en cours et les actions possibles.	
Ajouter / Supprimer un commentaire	Tout utilisateur peut ajouter un commentaire sur une demande ainsi que supprimer les commentaires dont il est propriétaire. Les commentaires sont de la forme d'une discutions entre les différentes parties qui interviennent.	Tous les utilisateurs
Valider les demandes « En cours »	Le RDS peut valider les demandes avec l'état « En cours ».	Le RDS (département STR)
Change l'état d'une demande	Un Administrateur peut changer l'état de n'importe quelle demande.	Administrateur

2.3. Diagramme des cas d'utilisation

La figure suivante représente le diagramme des cas d'utilisation souhaité :

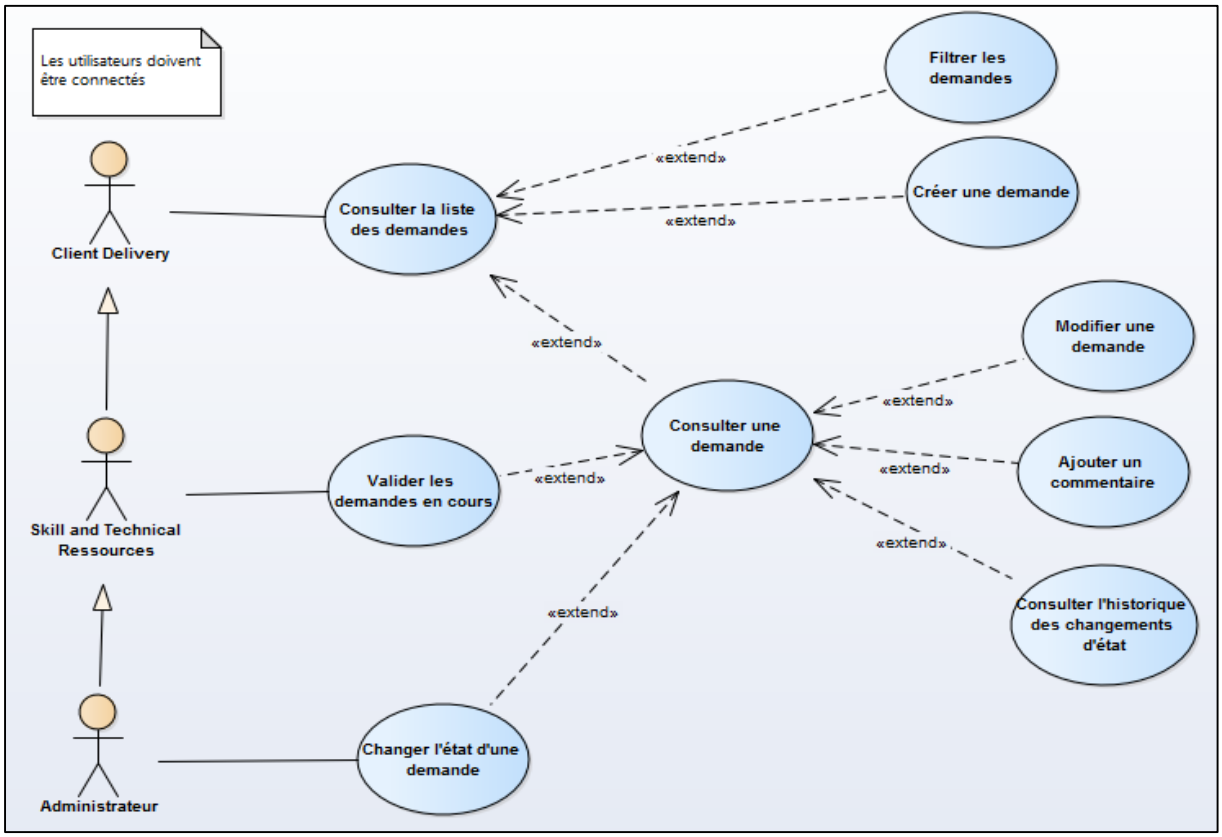

Figure 7 - Diagramme des cas d'utilisation

Pour mieux comprendre les cas d'utilisation exprimés dans le diagramme des cas d'utilisation, nous allons donner la description textuelle des principaux cas d'utilisation sous forme de tableau qu'identifie le scénario de chaque cas d'utilisation en question.

Le tableau ci-dessous nous suggère la description textuelle du cas d'utilisation « Créer une demande.

Titre	**Créer une demande**
Acteur	Tous les utilisateurs
Scénario	**1.** Sur la liste des demandes de staffing cliquer sur « Créer une demande ». **2.** Le modal de la création d'une demande s'ouvre. **3.** Saisir les informations de la demande. **4.** Cliquer sur « Save ».

Tableau 3 - Scénario de création d'une demande

À travers le tableau suivant, nous dégageons la description textuelle du cas d'utilisation « Modifier une demande ».

Titre	**Modifier une demande**
Acteur	Tous les utilisateurs
Scénario	**1.** Chercher la demande à l'aide des filtres. **2.** Double-cliquer sur la demande ou clique sur l'icône de modification. **3.** La vue de la consultation s'ouvre. **4.** Effectuer des modifications et envoyer. **5.** Une alerte de confirmation de succès ou échec de modification s'affiche sous forme d'une notification. **6.** Les données dans le tableau de la liste sont rafraîchies.

Tableau 4 - Scénario de modification d'une demande

À travers le tableau suivant, nous illustrons la description textuelle du cas d'utilisation « Ajouter un commentaire ».

Titre	**Ajouter un commentaire**
Acteur	Tous les utilisateurs
Scénario	**1.** Chercher la demande à l'aide des filtres.

	2. Double-cliquer sur la demande ou cliquer sur bouton détails. **3.** La vue de la consultation s'ouvre. **4.** Cliquer sur l'onglet « Suivi de la demande ». **5.** La liste des commentaires s'affiche avec la possibilité de supprimer ses propres commentaires. **6.** Écrire un commentaire. **7.** Cliquer « Envoyer » pour enregistrer le commentaire dans la liste des commentaires à sauvegarder dans la base de données. **8.** Cliquer « Save » pour sauvegarder la liste des commentaires dans la base de données.

Tableau 5 - Scénario d'ajout d'un commentaire

Le tableau ci-dessous nous propose la description textuelle du cas d'utilisation « Supprimer un commentaire ».

Titre	**Supprimer un commentaire**
Acteur	Tous les utilisateurs
Scénario	**9.** Chercher la demande à l'aide des filtres. **10.** Double-cliquer sur la demande ou cliquer sur bouton détails. **11.** La vue de la consultation s'ouvre. **12.** Cliquer sur l'onglet « Suivi de la demande ». **13.** La liste des commentaires s'affiche avec la possibilité de supprimer ses propres commentaires. **14.** Cliquer sur l'icône de suppression. **15.** Cliquer « Save » pour enregistrer les modifications.

Tableau 6 - Scénario de suppression d'un commentaire

La description textuelle du cas d'utilisation « Valider les demandes « En cours » » est présentée par le tableau suivant.

Titre	**Valider les demandes « En cours »**
Acteur	Le RDS
Scénario	**16.** Chercher la demande à l'aide des filtres.

	17. Double-cliquer sur la demande ou clique sur l'icône de modification. **18.** Effectuer une action pour transiter à un autre état.

Tableau 7 - Scénario de validation d'une demande « En cours »

3. Besoins non fonctionnels

La nature du notre projet exige certaines règles à respecter. Pour cela l'ensemble des réalisations doivent respecter les besoins suivants :

- **Ergonomie de l'interface** : Notre application doit offrir une interface ergonomique, conviviale, compatible avec tous les navigateurs et s'adapter à tous les écrans avec des résolutions différentes.
- **Disponibilité** : Notre application doit être disponible à tout instant pour être utilisée par n'importe quel utilisateur, et doit être facilement accessible via n'importe quel ordinateur.
- **Extensibilité** : Notre application doit permettre l'ajout ou la modification de nouvelles fonctionnalités en respectant les bonnes pratiques de codage.
- **Fiabilité** : Notre application doit rendre des résultats corrects, quelles que soient les conditions d'exploitation.

4. Besoins techniques

Durant cette phase, nous présentons l'architecture logicielle adoptée, citer les Design patterns utilisés, et finalement, les frameworks utilisés.

4.1. Architecture

L'architecture en couches est la conséquence inéluctable d'une approche qui s'appuie sur la réalisation de composants réutilisables.

Dans le but de réaliser un système puissant, évolutif et modulaire, nous avons adopté une architecture en couches basée sur le modèle Model View Controller (MVC) qui nous garantit le maximum de découplage entre les couches logicielles mises en oeuvre.

Notre application est séparée en trois couches logiques distinctes :

- **La couche présentation** : représente la partie présentation de l'application, son rôle est d'afficher le résultat des requêtes et de présenter des formulaires de soumission. La génération des vues se fait avec Thymeleaf.
- **La couche métier** : est gérée par des contrôleurs et des web services. De ce fait, cette couche cache toute la complexité du traitement d'une requête utilisateur.
- **La couche persistance** : représente la partie qui contient le code d'accès aux données de l'application. L'accès aux données peut se faire grâce aux interfaces de la Java Persistence API (JPA).

Web Services

Les Web Services sont des services informatiques de la famille des technologies web permettant la communication entre des applications hétérogènes dans des environnements distribués.

Ils ont été proposés à la base comme solution d'intégrations de différents logiciels développés par des entreprises leur permettant de communiquer entre eux.

L'interaction la entre partie Back-end et la partie Front-end de notre application se fait grâce à l'API qui est implémenté avec des web services REST [3].

4.2. Architecture REST

Representational State Transfer (REST) est l'un de ces acronymes qui représente une non-technologie comme peuvent l'être AJAX, Web 2.0 et autres.

REST est un style d'architecture qui repose sur le protocole HTTP : l'accès à une ressource (par son Uniform Resource Identifier (URI) unique) permet de procéder à diverses opérations (GET lecture / POST écriture / PUT modification / DELETE suppression, etc.).

Le choix d'implémenter une API dite RESTful (implémenté par des web service REST) vient du fait que les web services REST sont très robustes lorsqu'ils sont utilisés dans une application orientée données (DOA) [4].

Les objets avec lesquels les différents composants de l'architecture REST communiquent sont des objets JavaScript Object Notation (JSON). Le Back-end bien sûr communique avec la base de données.

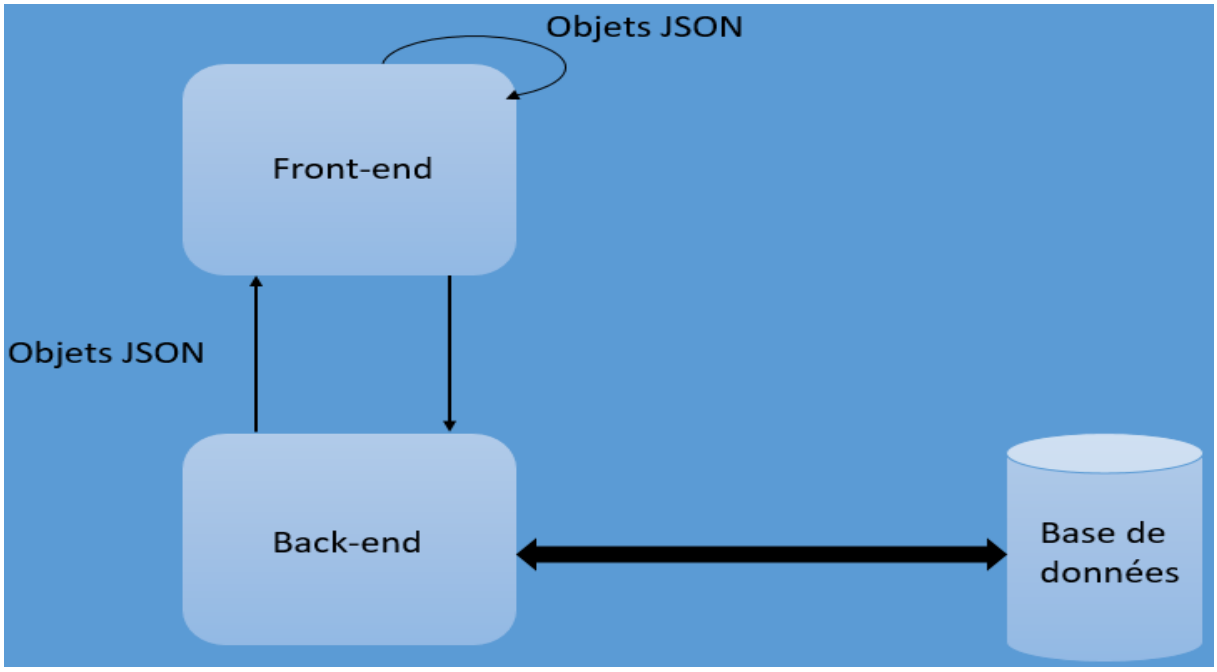

Figure 8 - Architecture haute niveau de l'application Back-end, Front-end

4.3. Design Patterns

Le modèle MVC est un désigne pattern dont l'objectif est de séparer les différentes parties d'une application. Cette séparation a l'avantage de répartir les groupes de travail sur le projet.

La modification d'une partie n'affectant pas les autres.

- **Model (modèle)** : le rôle de cette couche est de structurer les données et, dans une moindre mesure, de proposer une interface d'accès. Le modèle peut avoir un rôle plus ou moins important suivant les besoins.
- **View (vue)** : cette couche concerne toute la partie présentation à l'utilisateur final, c'est-à-dire l'aspect extérieur. La plupart du temps, la vue est accompagnée d'un moteur de Template.
- **Controller (contrôleur)** : ce composant est le plus complexe à saisir théoriquement. Il s'agit de la logique de contrôle qui va agir comme une glue entre les parties modèle et vue. Il est responsable de la collecte des données externes, du traitement du modèle, de la fusion avec la vue et de l'envoi de la réponse au client.

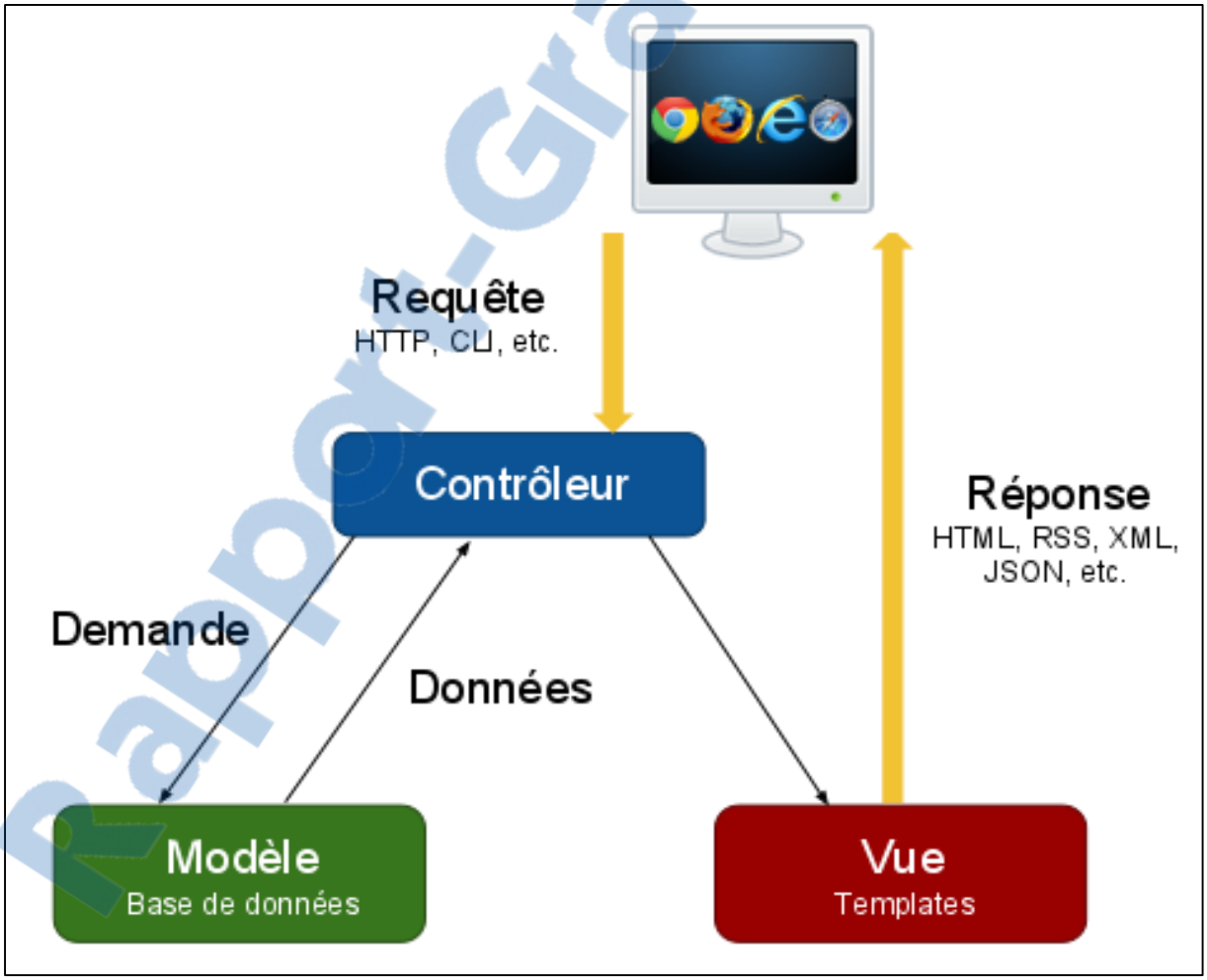

Figure 9 - Fonctionnement du modèle MVC

State Design Pattern

Utilisé en génie logiciel, ce patron de conception (Design Pattern) est utilisé lorsqu'il est souhaité pouvoir changer le comportement de l'état d'un objet sans pour autant en changer l'instance.

La classe censée changée d'état à un lien vers une classe abstraite « State ». Cette classe abstraite définit les différentes méthodes qui seront à redéfinir dans les implémentations. Dans chaque classe dérivée, l'appel à la méthode X pourra avoir un comportement différent.

La classe pouvant changer d'état appellera les services de sa classe d'état dont l'instance change quand le comportement de notre classe change. De plus, l'instance de la classe pouvant changer d'état peut être passée en paramètre à la méthode X de sa classe d'état. Ceci permet de changer l'état de la classe pendant l'exécution de la méthode X par instanciation d'un nouvel état.

Ce pattern permet donc à la classe de passer d'un état à l'autre de telle façon que cette dernière apparaît changer de type dynamiquement (sans changer d'instance).

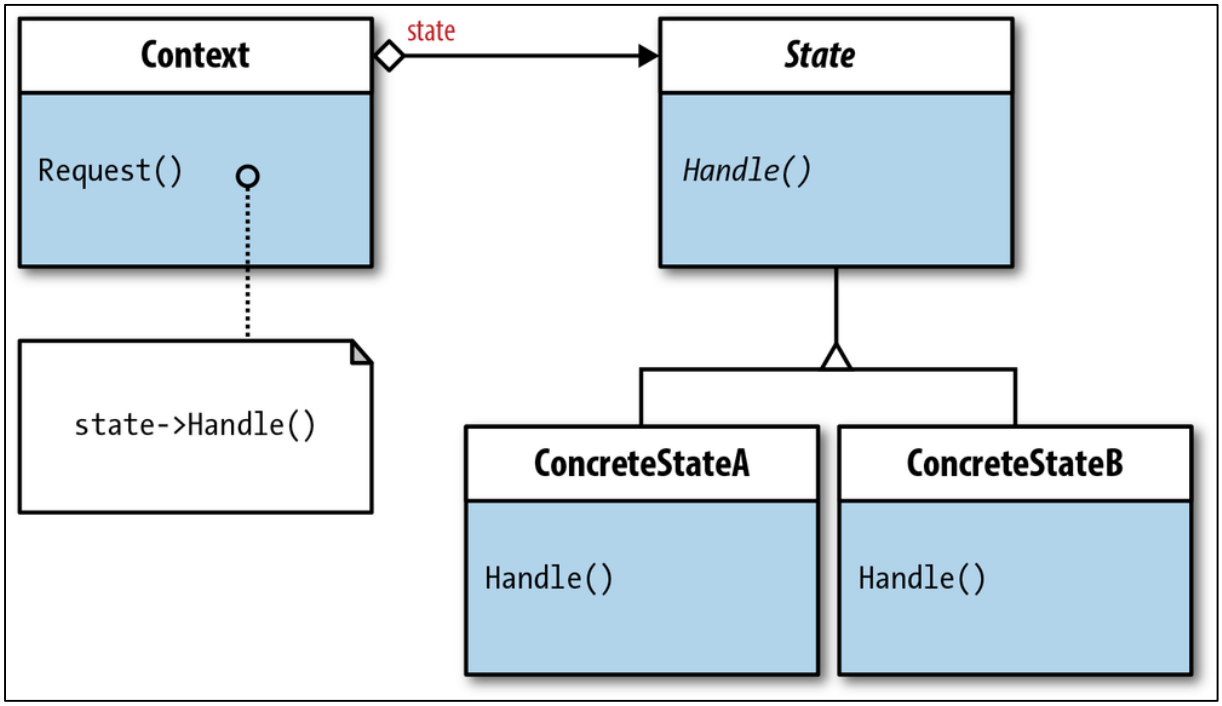

Figure 10 - Structure de State Pattern

Nous avons implémenté ce pattern pour prendre en charge les différentes actions que nous pouvons faire sur une demande de staffing, ainsi que les transitions entre états selon l'action et l'état en cours.

4.4. Frameworks

Afin de se bénéficier des fonctionnalités déjà implémentées par la communauté de développement Web, nous avons utilisé un ensemble de frameworks tels que :

- **Spring** [5]: est un framework libre pour construire et définir l'infrastructure d'une application java, dont il facilite le développement et les tests.

  Le framework Spring fournit un modèle complet de programmation et de configuration pour les applications d'entreprise modernes basées sur Java - sur n'importe quelle plate-forme de déploiement. Un élément clé de Spring est un support d'infrastructure au niveau de l'application :

  Spring se concentre sur la « plumbing (plomberie) » des applications d'entreprise afin que les équipes puissent se concentrer sur la logique commerciale au niveau de l'application, sans liens inutiles avec des environnements de déploiement spécifiques.

  Spring est caractérisé par :
  - Injection automatique des dépendances entre les classes.
  - Programmation orientée vers l'aspect (séparer tout ce qui est technique comme la gestion des exceptions, de tout ce qui est logique métier).
  - Application Web Spring MVC et les service web RESTful.
  - Etc.
- **Spring MVC :** Le framework Spring MVC fournit une architecture Model View Controller (MVC) et des composants prêts à utiliser pour développer des applications web flexible et peu couplé. Le modèle MVC permet de séparer les différents aspects de l'application tout en fournissant un faible couplage entre ces éléments.

  Le modèle (Model) encapsule les données de l'application et, en général, elles sont constituées de POJO.

  La vue (View) est responsable du rendu des données du modèle et, en général, il génère une sortie HTML que le navigateur du client peut interpréter.

  Le contrôleur (Controller) est responsable du traitement des demandes des utilisateurs et de la construction d'un modèle approprié et le transmet à la vue pour le rendu.

  La structure Spring MVC est conçue autour d'un DispatcherServlet qui gère toutes les requêtes et réponses HTTP. Le processus de traitement des requêtes est illustré dans le diagramme suivant :

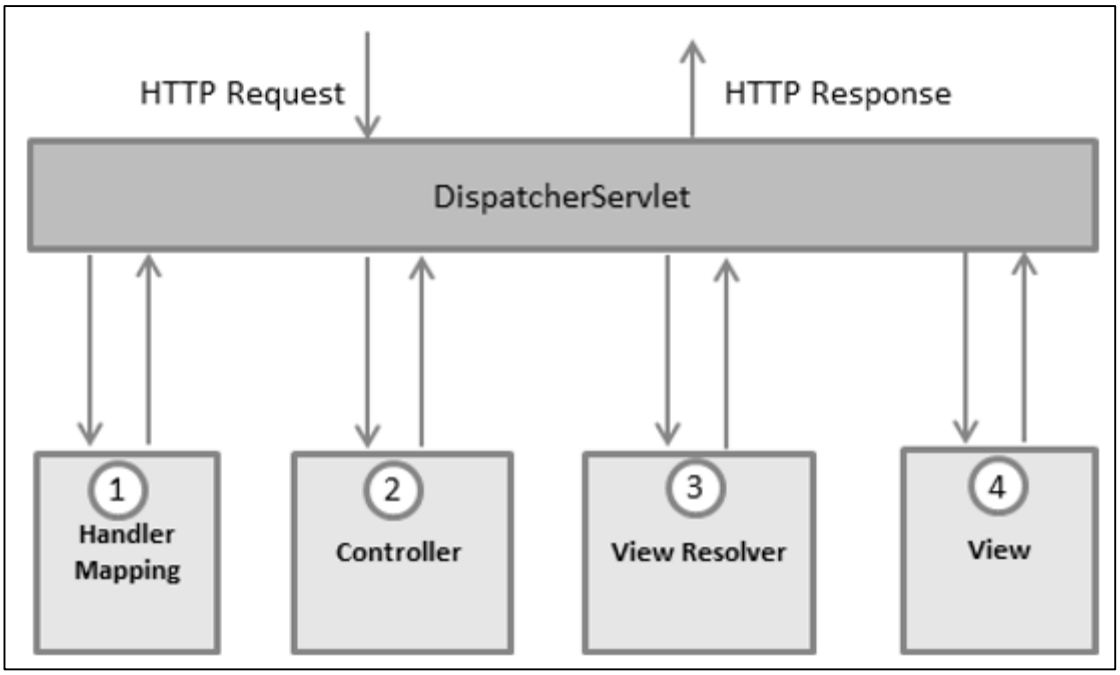

Figure 11 - Traitement des requêtes selon Spring MVC

Après avoir reçu une requête HTTP, DispatcherServlet consulte HandlerMapping pour appeler le contrôleur approprié.

Le contrôleur reçoit la demande et fait appel aux méthodes de service appropriées en fonction de la méthode HTTP GET, POST, PUT ou DELETE. La méthode de service définit les données du modèle en fonction de la logique métier définie et renvoie le nom de la vue au DispatcherServlet.

Le DispatcherServlet prend l'aide de ViewResolver pour sélectionner la vue définie pour la demande.

Une fois la vue déterminée, le DispatcherServlet passe les données du modèle à la vue qui est finalement rendue sur le navigateur.

- **Spring Data** [6]**:** Spring Data a pour mission de fournir un modèle de programmation familier et cohérent basé sur Spring pour l'accès aux données tout en conservant les traits spéciaux du magasin de données sous-jacent.

Il facilite l'utilisation des technologies d'accès aux données, des bases de données relationnelles et non relationnelles, des cadres de réduction de la carte et des services de données basés sur le cloud. Il s'agit d'un projet de parapluie qui contient de nombreux sous-projets spécifiques à une base de données donnée. Les projets sont développés en travaillant en collaboration avec de nombreuses entreprises et développeurs derrière ces technologies passionnantes.

Spring Data est caractérisé par :

- Référentiel puissant et abstractions de mappage d'objets personnalisés.
- Dérive de requête dynamique à partir des noms de méthodes de dépôt.
- Possibilité d'intégrer un code de dépôt personnalisé.
- Intégration avancée avec les contrôleurs Spring MVC.
- Classes de base de domaine de mise en œuvre fournissant des propriétés de base.

- **Spring Data JPA :** Spring Data JPA, qui fait partie de la plus grande famille Spring Data, facilite la mise en œuvre facile des référentiels basés sur JPA. Ce module traite d'un support amélioré pour les couches d'accès aux données basées sur JPA. Il est plus facile de construire des applications Spring qui utilisent des technologies d'accès aux données.

  La mise en œuvre d'une couche d'accès aux données d'une application a été lourde pendant un certain temps. Il faut écrire trop de code d'écriture pour exécuter des requêtes simples, ainsi que pour effectuer une pagination et une vérification. Spring Data JPA vise à améliorer considérablement la mise en œuvre des couches d'accès aux données en réduisant l'effort au montant nécessaire. En tant que développeurs, nous écrivons nos interfaces de dépôt, y compris les méthodes de recherche personnalisées, et Spring fournira la mise en œuvre automatiquement.

- **Hibernate** [7]: Hibernate est une couche résidant dans la JVM permettant d'assurer le mapping des objets JAVA cachés dans la JVM au modèle relationnel ou modèle de données. Hibernate assure aussi le transfert des classes Java dans les entités de données et donc des données des objets dans les entités et les tables.

  Hibernate présente aussi un langage de manipulation des objets mappés connus sous le nom Hibernate Query Language (HQL). Il s'agit de faire des select, Update et Delete avec des opérations de somme, comptage (count), de calcul de moyenne, etc. HQL présente aussi des limites telles que l'utilisation des opérations d'Union.
  Dans sa couche la plus proche de la base de données, Hibernate utilise JDBC (JDBC Template) pour dialoguer avec la base de données.

  L'avantage d'hibernate, c'est clairement de masquer la logique relationnelle aux développeurs objets et/ou d'interconnecter « facilement » des objets avec une base de données « business » existante.
  La partie relationnelle étant laissée aux experts en base de données (écriture des fichiers de mapping notamment).

  Caractéristiques :
  - Pas de SQL dans le code.
  - Un modèle purement objet pour la manipulation de données, ce qui est bien pratique.
  - HQL qui permet des requêtes objet, similaire à SQL.

  - Un système de session garantissant unicité et cohérence des objets, accompagné d'un système transactionnel.
  - Toujours la main (mais c'est déconseillé) pour faire du SQL pur et dur.
  - Le cache de données.
  - Les requêtes optimisées pour la plupart des cas business.
- **Thymeleaf** [8]: est un moteur de Template, sous licence Apache 2.0, écrit en Java pouvant générer du XML/XHTML/HTML5. Thymeleaf peut être utilisé dans un environnement web (utilisant l'API Servlet) ou non-web. Son but principal est d'être utilisé dans un environnement web pour la génération de vue pour les applications web basées sur le modèle MVC.

  Caractéristiques :
  - C'est un moteur de template écrit en Java traitant les fichiers XML, XHTML et HTML5.
  - Thymeleaf permet de traiter à la fois les fichiers appartenant à un site web ou non. Il n'y a pas de dépendance vis-à-vis de l'API Servlet.
  - Thymeleaf est composé de plusieurs modules appelés dialecte :
    - Les caractéristiques d'un dialecte (par exemple : évaluation, itération, etc.) s'appliquent à travers les balises et/ou les attributs des templates.
    - Deux dialectes sont disponibles : Standard et le SpringStandard (pour les applications Spring MVC mais en utilisant la même syntaxe que le dialecte Standard).
    - Les développeurs peuvent étendre les fonctionnalités des dialectes proposés ou bien créer leur propre dialecte.
  - Plusieurs modes de Template sont disponibles :
    - XML
    - XHTML 1.0 and 1.1
    - HTML5
    - Le support de l'internationalisation des textes.
- **Typescript** [9] [10]: est un langage OpenSource édité par Microsoft. Il consiste à apporter un sur-ensemble au langage JavaScript (JS) permettant ainsi, de bénéficier de fonctionnalités complémentaires à celui-ci. Par exemple, le typage statique et générique, les classes abstraites ou bien les énumérations.

  Le navigateur web continue à n'interpréter que le JavaScript (JS). C'est pourquoi, lorsqu'on développe en Typescript, il est nécessaire de réaliser une phase de transpilation qui va littéralement transcrire vos codes Typescript en code JS simple, interprétable par votre navigateur.

  Cette étape est réalisée en amont et n'impacte donc aucunement les performances. De plus, le développeur peut choisir pendant cette phase de transpilation la version de JS de sortie (ECMA [11]) ce qui permet d'optimiser son code JS sur les navigateurs récents.

  Caractéristiques :
  - **La lisibilité**

- Un code propre et performant est une chose, mais c'est encore mieux s'il est agréable à lire.
- La clarté du code est due à la présence de classes et modules, qui permettent de faire une vraie POO (Programmation Orientée Objet).

- **Typage**
  - Le Typage, d'où il tire son nom, permet une meilleure gestion des erreurs.
- **Code/Performance**
  - Il y a bien évidement d'autres avantages, comme la vitesse d'exécution par rapport aux concurrents.
  - Une gestion des arguments qui permet de leur assigner une valeur par défaut qui est impossible en JS natif.
- **Partenariats**
  - Angular, actuellement le plus gros framework Front (développé par Google) a adopté le TypeScript depuis version majeure 2.0.
- **Communautaire**
  - A noter que le TypeScript est open source malgré le fait qu'il soit développé chez Microsoft. Ce qui est également assez intéressant, car si l'engouement pour celui-ci est important, nous pourrions nous attendre à de belles fonctionnalités et une forte contribution communautaire.
  - C'est donc une raison supplémentaire de s'intéresser à ce langage qui risque de grossir encore un peu plus.

Conclusion

Dans ce chapitre, nous avons commencé par la définition du périmètre du projet, après nous avons énoncé les spécifications fonctionnelles et techniques de notre système au cours desquelles les besoins fonctionnels ont été traduits en diagramme des cas d'utilisation pour différents acteurs, les besoins techniques par l'architecture 3-tiers de l'application ainsi que les frameworks et les design patterns utilisés.

Dans le chapitre suivant, nous allons entamer la conception du projet.

Chapitre 3 : Conception

Introduction

Après avoir achevé l'étape d'analyse et spécification des besoins, nous allons entamer dans ce chapitre la partie conception de notre solution, nous allons présenter les différents diagrammes réalisés lors de cette phase.

1. Diagramme de classes

Le diagramme de classe est un diagramme pivot de l'ensemble de la modélisation d'un système, chaque classe de ce diagramme intègre une partie dédiée aux données et une autre partie dédiée aux traitements.

1.1. Demande de staffing

Le résultat des analyses effectué dans la phase précédente a abouti au diagramme de classes de demande de staffing suivant :

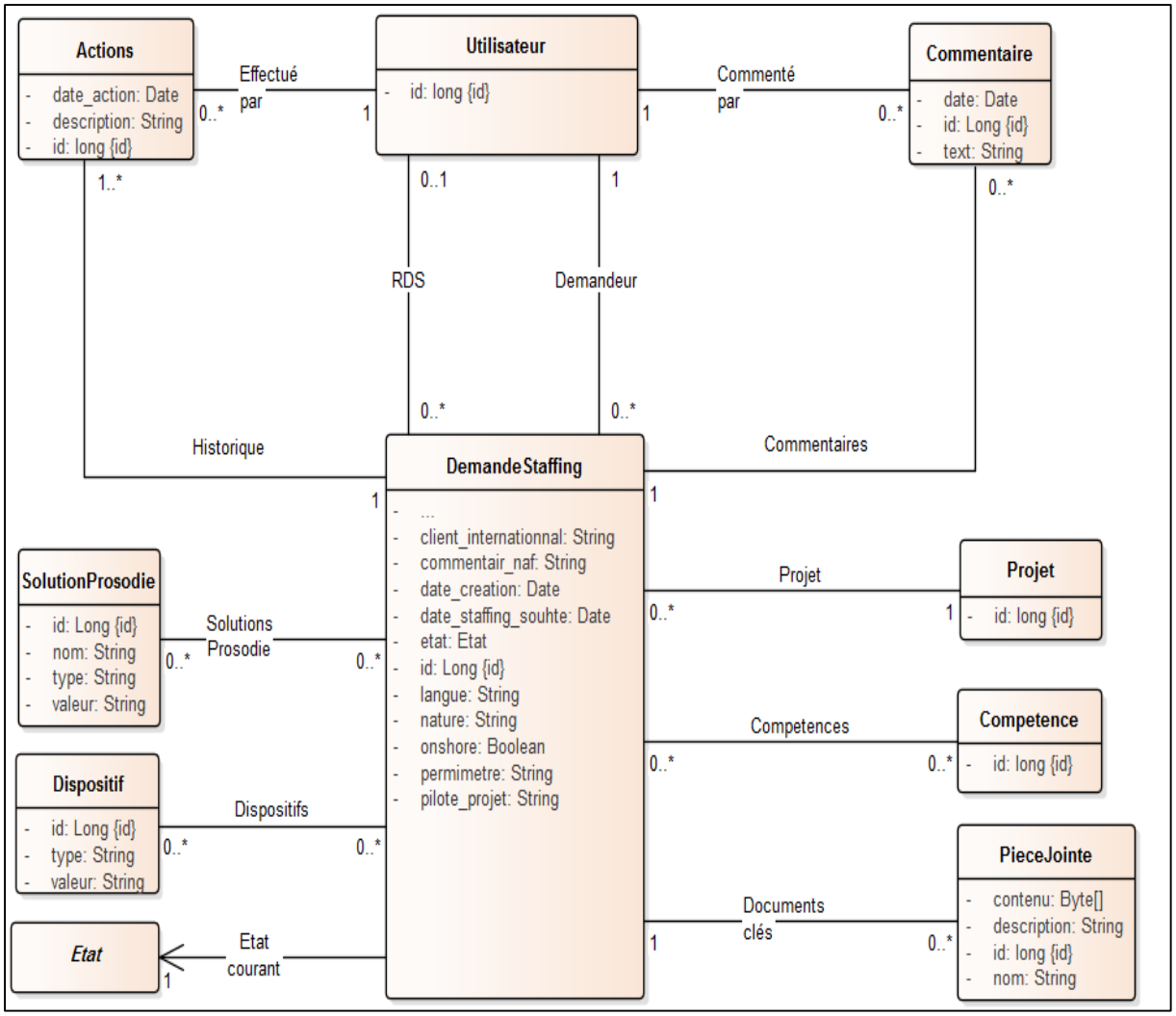

Figure 12 - Diagramme de classe de demande de staffing

Ce diagramme contient :

- Une classe « **DemandeStaffing** » a comme objectif la représentation d'une demande de staffing. C'est la classe principale qui contient toutes les informations nécessaires d'une demande de staffing (date de création, client, projet, etc.).
- La classe « **Utilisateur** » qui représente un collaborateur Prosodie, cette classe est utilisée dans le COPWEB pour la modélisation des COPs individuels et pour l'authentification.
- Une classe « **Projet** » désignant un projet auquel nous voulons effectuer une demande de staffing. Cette classe aussi est utilisée dans le COPWEB.
- La classe « **Dispositif** » qui modélise un dispositif qui représente un ensemble de mesures prises, de moyens mis en œuvre pour une intervention précise sur un projet. Par exemple : Migration, Gestion de problème, etc.
- Une classe « **Competence** » représentant les compétences nécessaires pour travailler sur un projet.
- Une classe « **SolutionProsodie** » modélisant les solutions Prosodie.
- La classe « **Commentaire** », pour la modélisation d'un commentaire ajouté par un utilisateur.
- La classe « **PieceJointe** », pour représenter un document clé.
- Une classe « **Etat** » a comme objectif la représentation d'état de la demande, selon le design pattern « *State* ».
- La classe « **Actions** », pour garder un historique des actions effectuées sur une demande de staffing.

Maintenant que nous avons défini toutes les classes du diagramme, nous allons décrire les relations entre eux :

- **Relation « RDS »** : cette relation existe une fois l'utilisateur qui représente le RDS traite la demande de staffing.
- **Relation « Demandeur »** : représente l'utilisateur qui effectue la demande de staffing. Un utilisateur peut avoir plusieurs demandes, dans l'autre côté une demande n'appartient qu'à un seul utilisateur.
- **Relation « projet »** : cette relation existe entre une demande de staffing et le projet pour lequel nous voulons effectuer cette demande. Si le projet n'existe pas dans la base de données, le type de client doit être mis à « Prospet » qui veut dire un client éventuel.
- **Relation « Dispositifs »** : entre une demande de staffing et la liste des dispositifs de la demande. Cette relation nécessite la création d'une troisième table associative, car c'est une relation plusieurs-plusieurs.
- **Relation « Competences »** : cette relation existe entre une demande de staffing et la liste des compétences requises par la demande. Cette relation nécessite la création d'une troisième table associative, car c'est une relation plusieurs-plusieurs.
- **Relation « Solutions Prosodie »** : entre une demande de staffing et la liste des Solutions Prosodie de la demande. Cette relation nécessite la création d'une troisième table associative, car c'est une relation plusieurs-plusieurs.
- **Relation « Commenté par »** : désigne l'utilisateur qui a ajouté le commentaire.

- **Relation « Commentaires »** : indique la liste des commentaires d'une demande. Une demande de staffing peut avoir plusieurs commentaires et de l'autre côté un commentaire appartient à une seule demande.
- **Relation « Documents clés »** : représente la liste des documents clés contenant des informations sur la demande nécessaires pour le traitement de la demande.
- **Relation « Etat courant »** : représente l'état en cours de la demande. Cette relation sera modélisée grâce à un attribut de la class « DemandeStaffing » de type « Etat » et qui peut prendre plusieurs formes selon l'état en cours et le « State Pattern ». Les valeurs qui seront insérées dans la base de données sont sous la forme des énumérations que nous allons voir dans le diagramme de « State Pattern ».
- **Relation « Historique »** : représente la liste des actions effectuées sur une demande de staffing.
- **Relation « Effectué par »** : existe entre un utilisateur et la liste des actions, elle indique l'utilisateur qui a effectué l'action sur une demande.

1.2. State Pattern

La figure ci-dessous montre le diagramme de classe de « State Pattern » utilisé pour représenter les différentes états d'une demande de staffing.

Chaque état de la demande est modélisé par une classe de « State Pattern ».

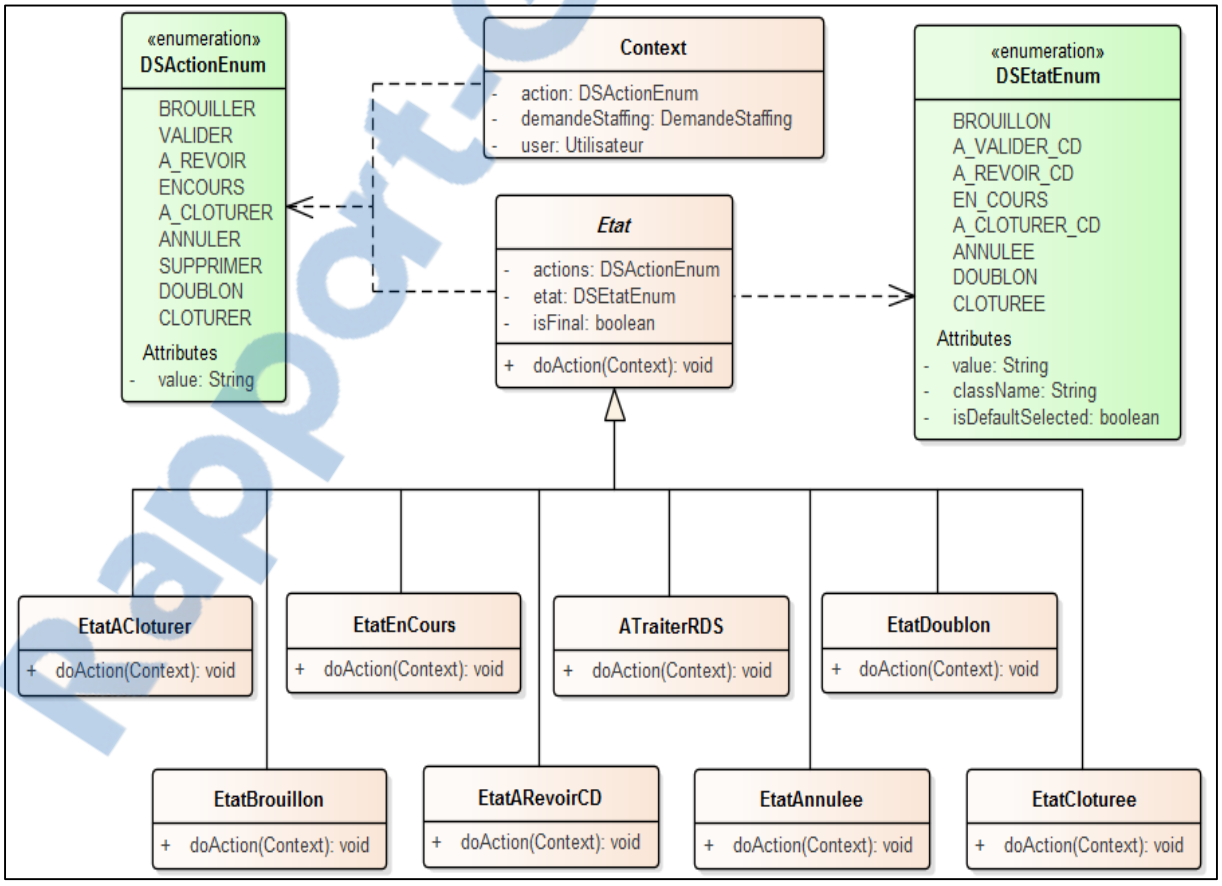

Figure 13 - Diagramme de classe « State Pattern »

Pour bien comprendre ce diagramme, nous allons définir l'ensemble des classes qu'il contient.

1.2.1. Classe « DSEtatEnum »

C'est une classe qui contient la liste des énumérations. Chaque énumération contient les attributs suivants :

- ***Value*** : le nom de l'état. C'est une chaîne de caractères qui représente la valeur à afficher pour cet état.
- ***className*** : le nom de la classe représenté par l'état. L'instanciation de l'état se fait dynamiquement grâce à la classe qui représente l'état et qui étend « Etat ». C'est cet attribut qui indique la classe.
- ***isDefaultSelected*** : indique l'état par défaut lors de la création de la demande.

1.2.2. Classe « DSActionEnum »

Cette classe contient la liste des énumérations des différentes actions possibles. Chaque énumération contient une valeur qui indique le nom de l'action.

L'objectif de cette classe des énumérations est d'injecter automatiquement la liste des actions possibles dans la vue, et les communiquer au web service responsable de la gestion de workflow et actions sur une demande.

1.2.3. Classe « Contexte »

Une classe contenant toutes les informations nécessaires pour le traitement d'une action. En gros, il faut passer une instance de cette classe avec toutes les informations demandés pour le traitement d'une action (appel de « *doAction ()* »).

Cette classe contient :

- ***demandeStaffing*** : la demande de staffing qui doit être modifié.
- ***user*** : l'utilisateur qui à effecté l'action sur la demande. Ceci est nécessaire pour la gestion des permissions.
- ***action*** : l'action effectuée par l'utilisateur.

1.2.4. Classe « Etat »

Une abstraction d'un état de la demande. Chaque implémentation d'état doit étendre de cette classe.

Chaque état doit contenir les attributs suivants :

- ***etat*** : l'énumération de l'état pour savoir la valeur à afficher dans les vues comme état courant.
- ***actions*** : la liste des actions possibles dans cet état. Cette liste permet d'afficher la liste des actions possibles sur la demande. La liste facilite aussi l'ajout d'autres actions possibles sans avoir besoin de modifier les vues aussi.
- ***isFinal*** : Un booléen qui indique si l'état est final.

Cette classe contient une méthode abstraite « *doAction ()* ». Cette méthode doit être implémentée par toute classe héritant de la classe « Etat » et doit contenir le traitement qu'il faut faire dans le contexte donné en paramètre. En générale, cette méthode doit permettre de vérifier que l'utilisateur a le droit d'effectuer l'action sur la demande pendant l'état courant, changer l'état de la demande au cas de succès ou générer une exception au cas d'échec ou au cas où l'utilisateur n'a pas le droit d'effectuer l'action, et finalement sauvegarder le changement dans l'historique des actions.

1.2.5. Classes héritant de la classe « Etat »

Ce sont des classes qui implémentent les états d'une demande. Chaque classe doit implémenter la méthode « *doAction ()* » comme décrite dans le paragraphe ci-dessus.

Si un nouveau état est ajouté, il suffit de créer une nouvelle classe héritant de « Etat », et implémenter la méthode « *doAction ()* » par le traitement à faire sans oublier d'ajouter l'énumération de l'état ajouté.

2. Diagramme d'état

Le diagramme d'état c'est un diagramme qui représente l'enchainement de tous les états caractéristiques d'un objet, la figure suivante montre le diagramme d'état de la demande de staffing :

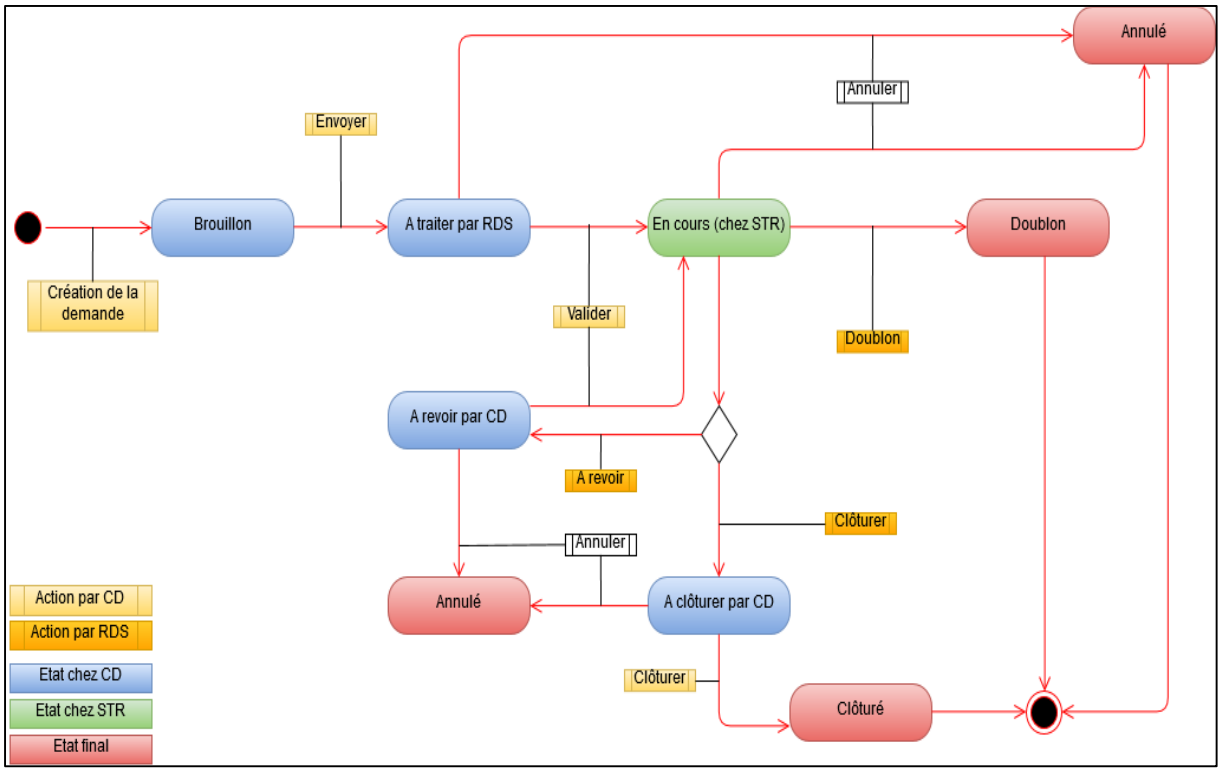

Figure 14 - Diagramme d'état d'une demande de staffing

2.1. État « Brouillon »

Après la création d'une demande, elle passe automatiquement à l'état « Brouillon ».

L'action possible dans cet état est la suivante :

- **Envoyer** : la demande passe à l'état suivant.

2.2. État « À traiter par RDS »

Lorsque le demandeur envoie la demande à partir de l'état Brouillon, la demande passe à l'état à traiter par RDS.

Les Actions possibles dans cet état sont les suivants :

- **Annuler** : marquer la demande comme étant annulée.
- **Valider** : validation de la demande par le demandeur ou toute personne de même département que lui ou éventuellement un administrateur. La demande alors passe à l'état suivant pour être traitée.

Nous pouvons transiter vers cet état directement après la création de la demande si le demandeur choisit d'envoyer directement au lieu d'enregistrer.

2.3. État « En cours (chez STR) »

C'est l'état le plus important de la demande. Une demande dans cet état signifie qu'elle est en train d'être traitée par un RDS. D'éventuelles communications entre le demandeur et des RDSs ou même d'autres personnes peuvent avoir lieu à l'aide des commentaires.

Il n'est plus possible de modifier cette demande que par un RDS ou administrateur.

Les Actions possibles dans cet état sont les suivants :

- **Annuler** : marquer la demande comme étant annulée.
- **Doublon** : marquer la demande comme étant Doublon d'une autre demande.
- **Clôturer** : clôturer la demande et l'envoyer au demandeur pour qu'il la clôture aussi.
- **A Revoir par CD** : noter que la demande a été traitée mais nécessite d'être revue par le demandeur.

2.4. État « À revoir par CD »

Si des informations sont manquantes pour le traitement de la demande, et le RDS trouve de mal à traiter la demande, ce dernier marque la demande comme étant besoin d'être revue par le demandeur.

Les Actions possibles dans cet état sont les suivants :

- **Annuler** : marquer la demande comme étant annulée.

- **Valider** : validation de la demande par le demandeur ou toute personne de même département que le demandeur.

2.5. État « À clôturer par CD »

Lorsque le RDS clôture la demande, celle-ci devrait être clôturée ou, autrement dit confirmée par le demandeur.

Les Actions possibles dans cet état sont les suivants :

- **Annuler** : marquer la demande comme étant annulée.
- **Clôturer** : clôturer et mettre fin à la demande.

2.6. État « Clôturé »

Une demande avec l'état « Clôturé » signifie que la demande a été traitée avec succès par toutes les parties et le staffing a eu lieu.

La demande est alors archivée et aucune action n'est plus possible sur la demande, mais elle est toujours consultable.

2.7. État « Doublon »

Si un RDS remarque que la demande est doublon d'une autre demande (c'est la même demande qui est déjà traiter ou en cours de traitement), la demande sera marquée comme « Doublon ».

La demande est alors archivée et aucune action n'est plus possible sur la demande, mais elle est toujours consultable.

2.8. État « Annulé »

L'annulation d'une demande peut se faire pendant n'importe quel état à part l'état brouillon, et les états finaux où la demande est considérée archivée.

Dans cet état sauf l'administrateur peut modifier l'état de la demande vers l'état précédent.

3. La base de données de l'application

La figure ci-dessous représente les tables utilisées dans notre base des données. Ces tables représentent l'unité de persistance de notre application.

La relation plusieurs à plusieurs entre les tables est représentée par une troisième table contenant les clés primaires des deux tables en plus d'autres informations.

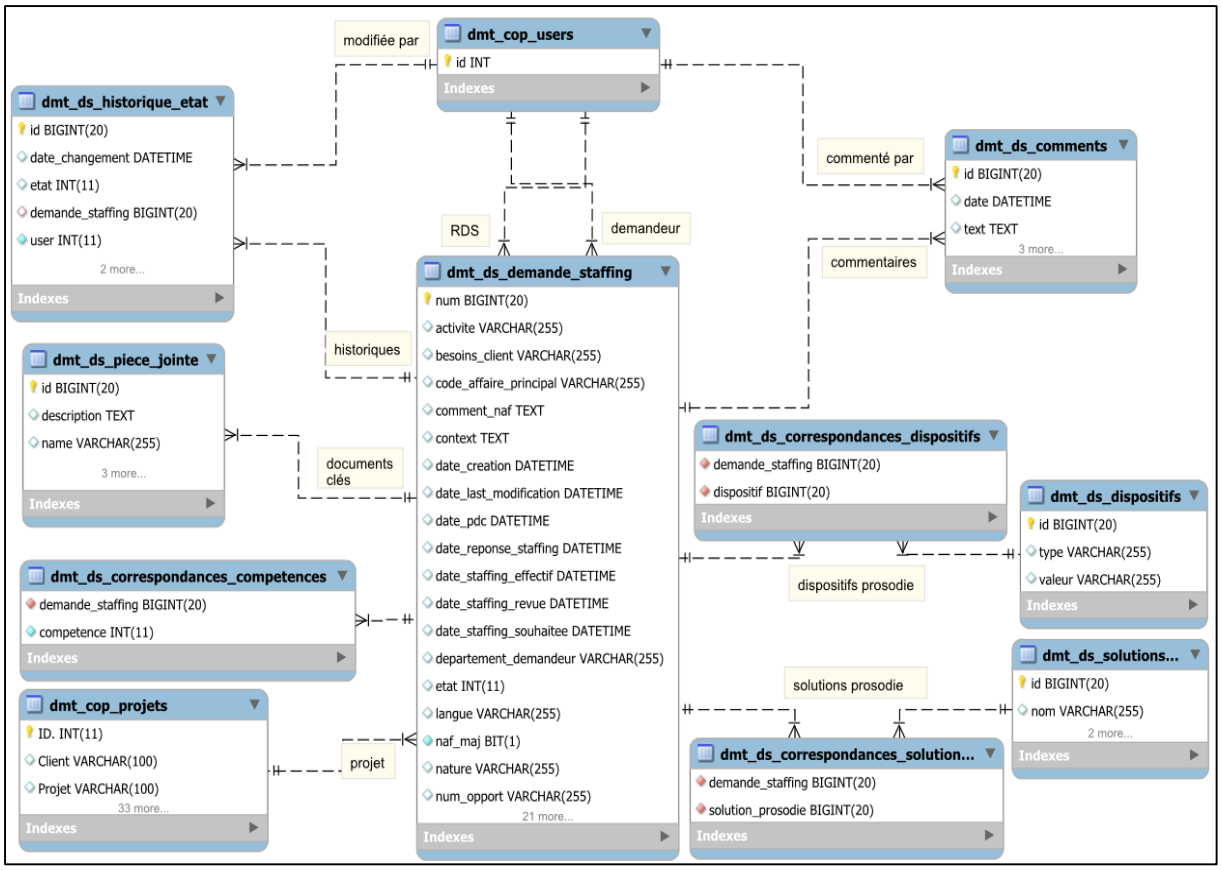

Figure 15 - Schéma de la base de données de l'application

4. Diagramme de séquence

Les diagrammes de séquences servent à illustrer les cas d'utilisation, ils permettent de représenter les interactions dans le temps entre les objets du système.

3.1. Diagramme de séquences du cas d'utilisation « Créer une demande »

Une fois authentifié, l'utilisateur peut créer une demande de staffing en accédant à l'interface de création à partir de l'interface de consultation de la liste des demandes.

L'utilisateur doit remplir tous les champs nécessaires à la création de la demande de staffing pour pouvoir la sauvegarder. Sinon un message d'erreur s'affiche indiquant les champs vides qui doivent être remplis.

Le scénario de création d'une demande est décrit dans le diagramme de séquence montré dans la figure ci-dessous, la figure [cf. Figure 17] et la figure [cf. Figure 18]:

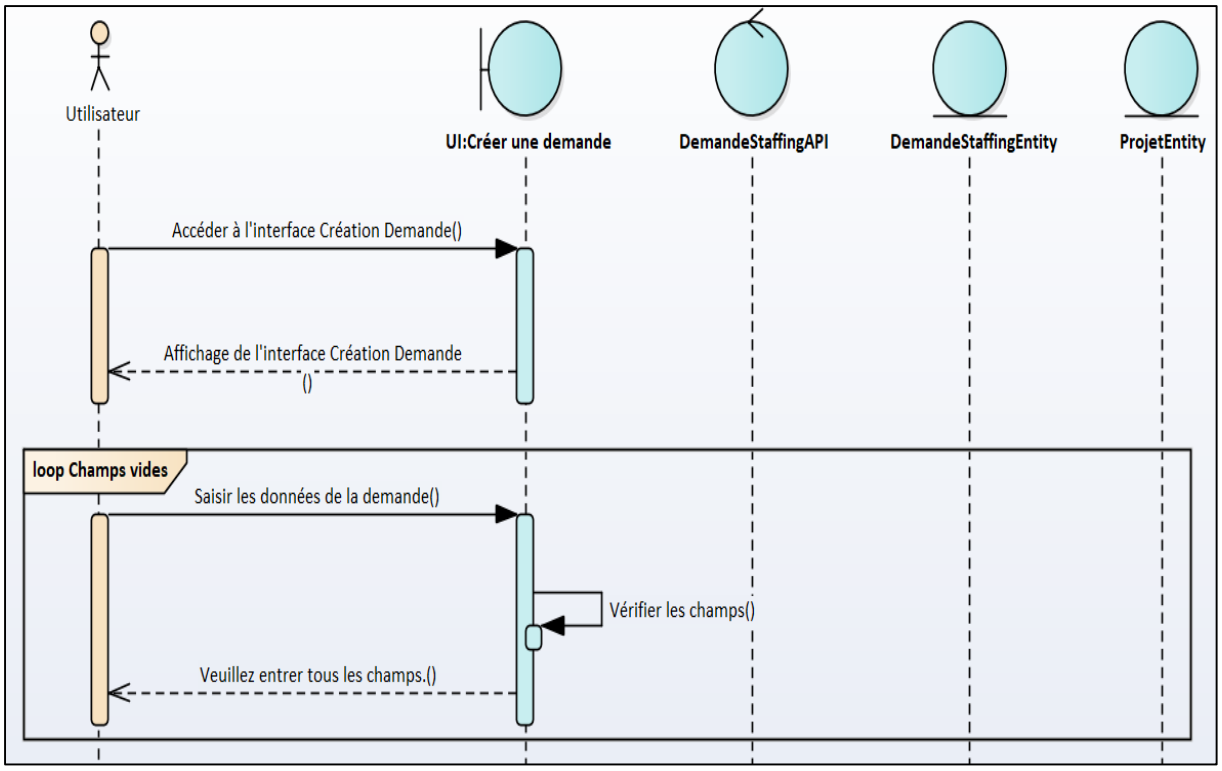

Figure 16 - Diagramme de séquence du cas d'utilisation « Créer une demande »

Si le projet n'existe pas dans la base de données, l'utilisateur peut l'ajouter à travers l'interface de création des projets. Une fois le projet est ajouté, l'utilisateur peut le sélectionner à partir de la liste des projets.

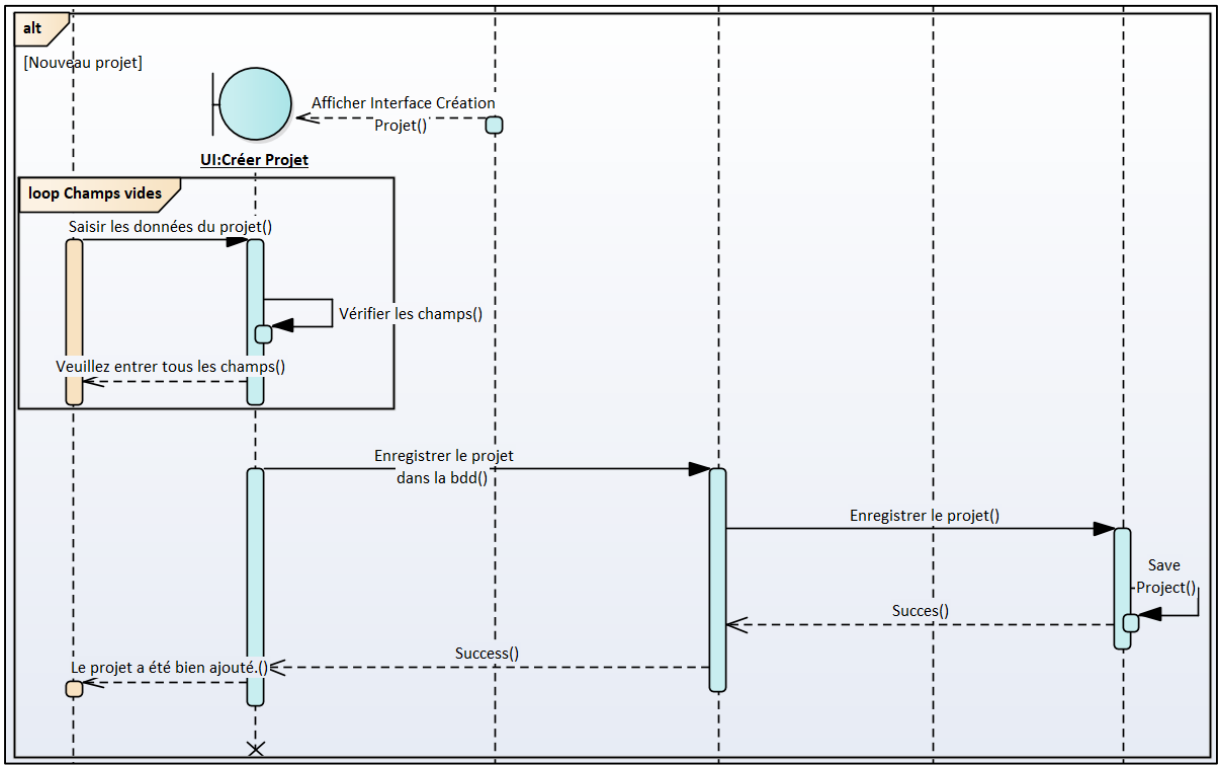

Figure 17 - Diagramme de séquence du cas d'utilisation « Créer une demande » (nouveau projet)

Si l'utilisateur choisit de sauvegarder la demande seulement, alors cette dernière est créée en état « Brouillon », sinon s'il choisit de l'envoyer aussi, alors elle est créée dans l'état « À traiter par RDS ».

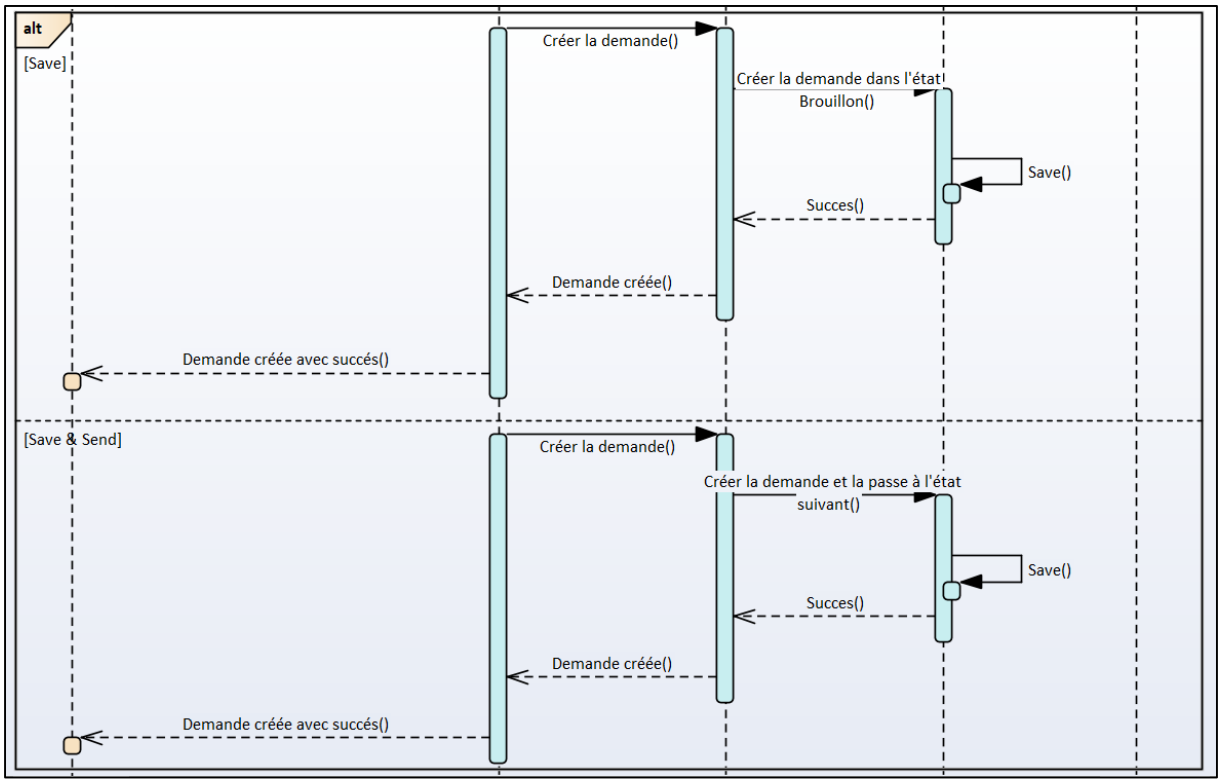

Figure 18 - Diagramme de séquence du cas d'utilisation « Créer une demande » (enregistrer)

3.2. Diagramme de séquences du cas d'utilisation « Modifier une demande »

Une fois authentifié, l'utilisateur peut modifier une demande de staffing à partir de l'interface de consultation d'une demande.

Le scénario souhaité est décrit dans le diagramme de séquence montré dans la figure ci-dessous, la figure [cf. Figure 20], la figure [cf. Figure 21] et la figure [cf. Figure 22] :

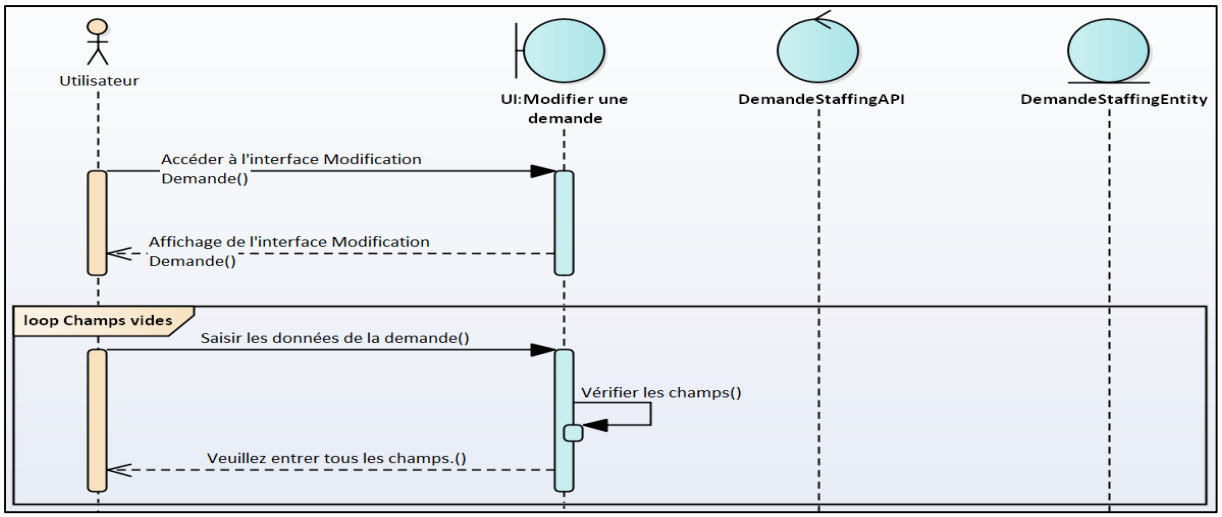

Figure 19 - Diagramme de séquence du cas d'utilisation « Modifier une demande »

L'utilisateur peut choisir de changer l'état d'une demande de staffing, la figure suivante montre le cas où l'état d'une demande peut être changé vers un état précédent.

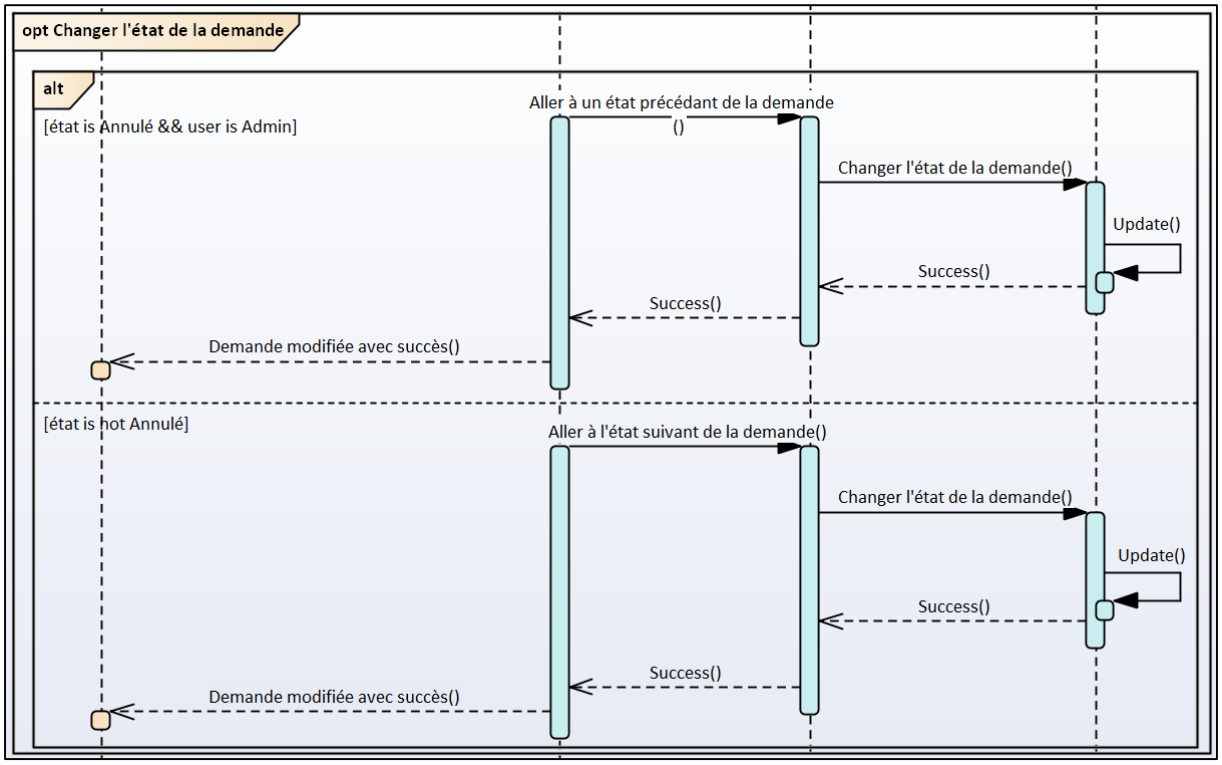

Figure 20 - Diagramme de séquence du cas d'utilisation « Modifier une demande » (changer l'état)

L'utilisateur peut choisir de sauvegarder ses modifications et quitter l'interface de modification, ou de sauvegarder ses modifications et passer à la demande suivante.

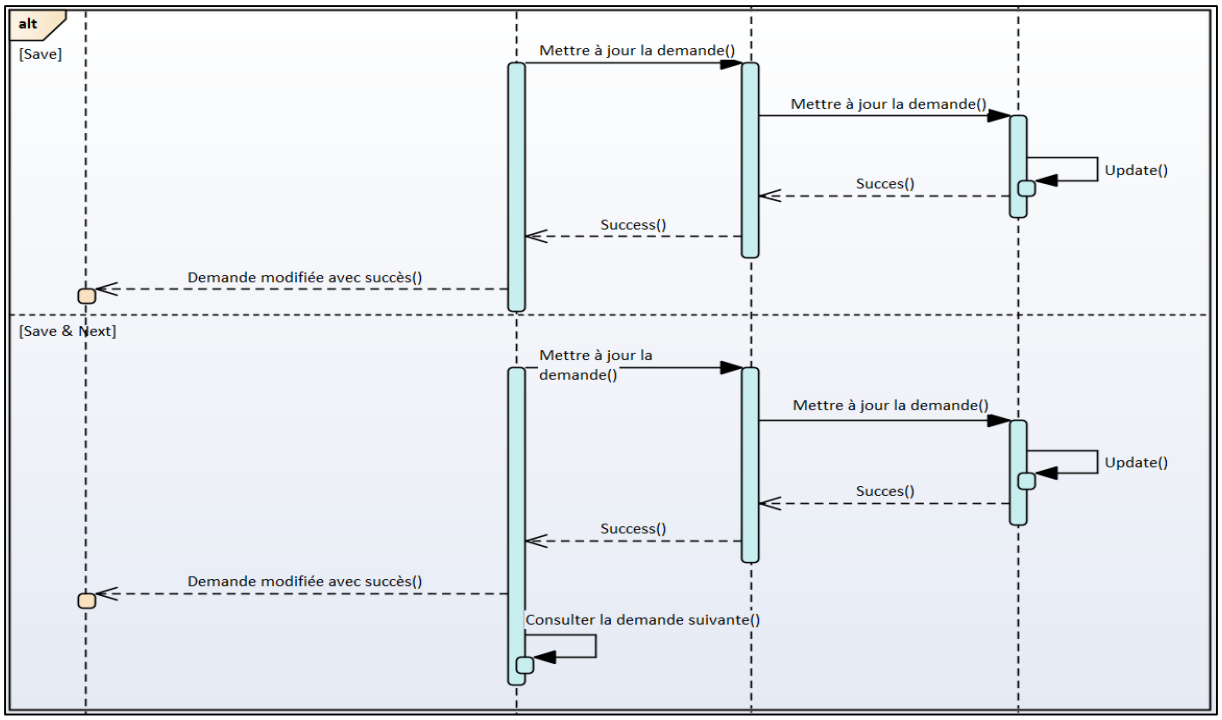

Figure 21 - Diagramme de séquence du cas d'utilisation « Modifier une demande » (enregistrer)

Si l'utilisateur a envoyé des commentaires ou a ajouté des pièces jointes, une boîte de dialogue de confirmation sera affichée pour le prévenir qu'ils ont être perdues s'il choisit de quitter l'interface.

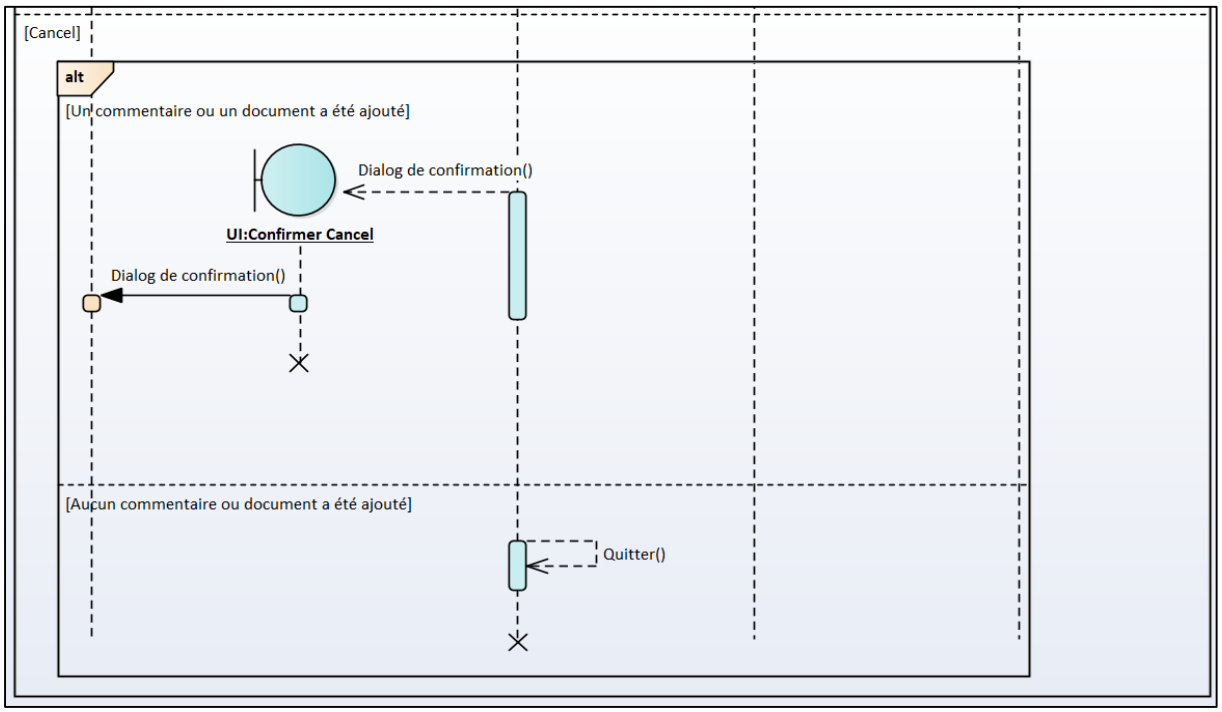

Figure 22 - Diagramme de séquence du cas d'utilisation « Modifier une demande » (annuler)

Conclusion

Au cours de ce chapitre, nous avons détaillé le diagramme de classes, le diagramme d'état et quelques diagrammes de séquences de notre application.

La phase de conception est terminée, nous sommes prêts donc à transformer ces diagrammes en une application fonctionnelle, cette implémentation sera détaillée dans le prochain chapitre.

Chapitre 4 : Réalisation

Introduction

Ce chapitre constitue le dernier volet de ce rapport, il traite la phase qui a pour objectif l'implémentation de notre application. Ainsi, nous allons passer en revue l'environnement technique du projet. Ensuite, nous décrivons les différentes étapes de mise en œuvre de notre application en deux phases, back-end et front-end. Finalement nous donnons un aperçu sur le travail réalisé.

1. Environnement du projet

1.1. Netbeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, JavaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java).

1.2. GIT

GIT [12], c'est un gestionnaire de versions décentralisé, aussi connu sous le terme de Distributed Version Control System (DVCS) pour les amateurs.

Le terme distribué est très important, car c'est ce qui le différencie d'autres gestionnaires historiquement plus connus comme Subversion (SVN) ou le vénérable Concurrent Versions System (CVS).

Ça veut dire que chaque développeur possède sa propre copie du dépôt, chez lui, localement, contrairement à un gestionnaire centralisé ou tout est centralisé sur un même serveur.

GIT est principalement utilisable en ligne de commande, mais il existe des interfaces graphiques sous les principaux OS du marché (OS X, Linux, Windows).

Voici, quelques notions de base de GIT :

- **Dépôt** : Un répertoire ou de l'espace de stockage où vos projets peuvent vivre. Parfois les utilisateurs GitHub [13] raccourcissent ça en « repository ». Il peut être local sur un répertoire de votre ordinateur, ou ce peut être un espace de stockage sur GitHub ou un autre hébergeur en ligne. À l'intérieur d'un dépôt, il est possible de conserver des fichiers de code, des fichiers texte, des images.

- **Contrôle de Version** : fondamentalement, l'objectif pour lequel GIT a été conçu. Quand vous avez un fichier Microsoft Word, vous l'écrasez à chaque fois que vous faites une nouvelle sauvegarde, ou vous sauvegardez plusieurs versions. Avec GIT, vous n'êtes plus obligé de faire ça. GIT conserve des "instantané" de chaque point dans l'historique d'un projet, de sorte que vous ne pouvez jamais le perdre ou l'écraser.
- **Commit** : c'est la commande qui donne à GIT toute sa puissance. Quand vous « committez », vous prenez un « instantané », une « photo » (snapshot) de votre dépôt à ce stade, vous donnant un point de contrôle que vous puissiez ensuite réévaluer ou restaurer votre projet à un état précédent.
- **Branche** : plusieurs personnes travaillent sur un projet en même temps sans que GIT ne s'embrouille, car habituellement, elles se « débranchent » du projet principal avec leurs propres versions complètes des modifications qu'elles ont chacune produite de leur côté. Après avoir terminé, il est temps de « fusionner » (merge) cette branche pour la ramener vers la branche « master », le répertoire principal du projet.

1.3 Jenkins

Jenkins est un outil open source d'intégration continue, fork de l'outil Hudson après les différends entre son auteur, Kohsuke Kawaguchi, et Oracle. Écrit en Java, Jenkins fonctionne dans un conteneur de servlets tel qu'Apache Tomcat, ou en mode autonome avec son propre serveur Web embarqué.

Jenkins s'interface avec des systèmes de gestion de versions tels que CVS, GIT et Subversion, et exécute des projets basés sur Apache Ant et Apache Maven aussi bien que des scripts arbitraires en shell Unix ou batch Windows.

Les générations de projets peuvent être amorcées par différents moyens, tels que des mécanismes de planification similaires au CRON, des systèmes de dépendances entre générations, ou par des requêtes sur certaines URL spécifiques.

L'utilisateur de Jenkins dans ce projet permet de garder toujours un œil sur le projet. Si un problème est survenu lors de build de projet un émail est envoyé automatiquement à l'équipe de développement, l'erreur doit être alors corrigée le plus rapide possible.

De plus Jenkins peut être utilisé pour faire un déploiement automatique de l'application dans les environnements spécifiés.

1.4 SonarQube

SonarQube [14] (précédemment Sonar) est un logiciel libre permettant de mesurer la qualité du code source en continu.

Les importantes fonctionnalités de SonarQube sont les suivants :

- Support de plus de vingt-cinq langages (Java, C, C++, Objective-C, C#, PHP, Flex, Groovy, JavaScript, Python, PL/SQL, COBOL, etc.), dont certains sont sous licence commerciale.
- Reporting sur :
    - Identification des duplications de code.
    - Mesure du niveau de documentation.
    - Respect des règles de programmation.
    - Détection des bugs potentiels.
    - Évolution de la couverture de code par les tests unitaires.
    - Analyse de la répartition de la complexité.
    - Analyse du design et de l'architecture d'une application.
- Évolution dans le temps et vues différentielles.
- Intégration avec l'environnement de développement Eclipse.

2. Réalisation

La réalisation du projet se décompose en deux parties (Front-end et Back-end). Dans cette section nous allons détailler les deux parties de système, les différentes classes et composants utilisés.

2.1. Partie Back-end

La partie Back-end se décompose aussi en plusieurs parties. Comme mentionné avant [cf. Chapitre 2, Paragraphe 4.1], le projet adopte l'architecture en couches basée sur le modèle MVC qui nous garantit le maximum de découplage entre les couches logicielles mises en œuvre.

La couche métier est réalisée par des contrôleurs qui s'occupent d'envoyer la vue demandée. La couche persistance s'occupe de toute interaction avec la base de données. Et toute communication entre Front-end et Back-end se fait à l'aide des web services basés sur l'architecture REST. Et finalement la gestion de flux de travail et transitions entre les états d'une demande de staffing se fait grâce au design pattern « State Pattern ».

2.1.1. Couche persistance

La couche persistance est réalisé grâce au « Spring Data JPA et Hibernate », l'ensemble des entités sont regroupées dans un package « entities » qui regroupent l'ensemble des classes qui définissent les entités, leurs champs et relations entre les tables.

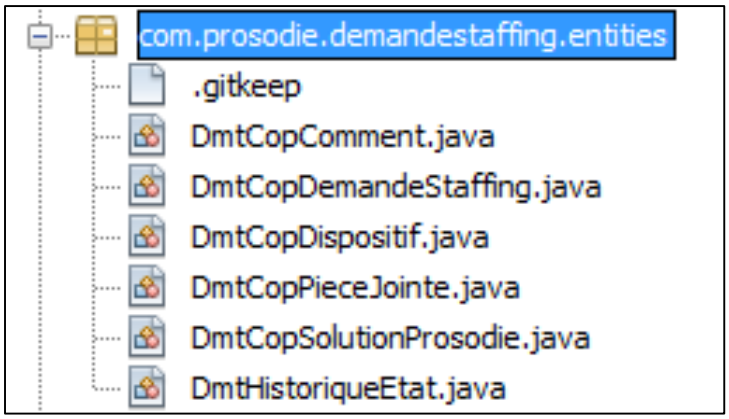

Figure 23 - Les entités du module « Demande Staffing »

Quelques entités existaient déjà et utilisé dans les modules existants de COPWEB. Ces entités sont regroupées dans un sous-package « entities » du package « COPWEB ».

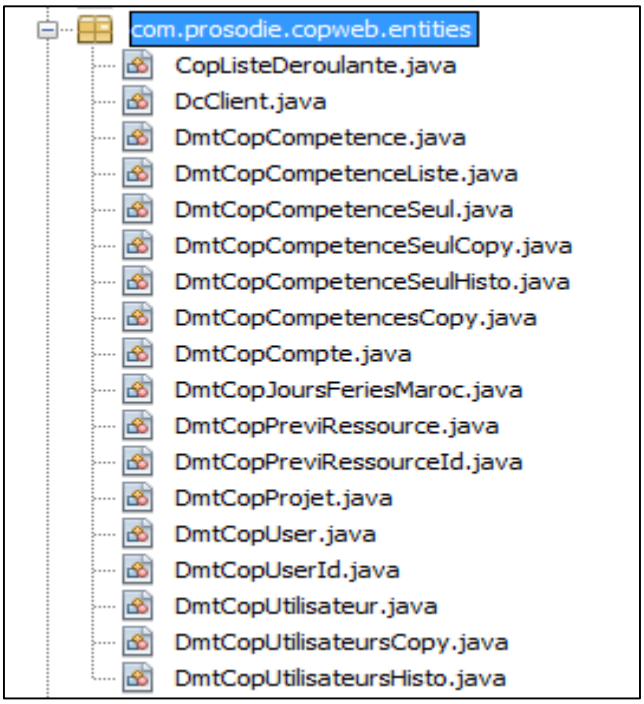

Figure 24 - Les entités du COPWEB

Le rôle principal des entités, c'est la création des tables dans la base de données. Nous pouvons à l'aide de ses classes d'ajouter, supprimer ou modifier dans la structure de la base de données. Ils sont généralement utilisés pour créer la table pour la première fois.

La déclaration des entités, leurs champs et leurs relations se font à l'aide des annotations Spring.

Spring s'occupe de mapping entre les attributs déclarés dans les entités et les attributs correspondants dans le Système de Gestion de Base de Données (SGBD), ainsi que les relations. Voici un exemple d'une entité :

```java
@Entity
@Table(name = "dmt_ds_comments")
public class DmtCopComment implements Serializable {

    /**
     *
     */
    private static final long serialVersionUID = -5185348675880236439L;

    @Id
    @GeneratedValue
    @Column(name = "id", unique = true)
    private long id;

    @Column(name = "date")
    private Date date;
```

Figure 25 - L'entité « Comment »

Voici une description de quelques annotations Spring :

- **@Entity** : pour indiquer que c'est une entité.
- **@Table** : pour indiquer des informations sur la table (nom de la table, etc.).
- **@Id** : pour indiquer que le champ représente la clé primaire de la table.
- **@DefaultGeneratedValue** : pour indiquer que le champ est générer automatique (auto-incrément).
- **@Column** : pour indiquer que c'est un champ de la table.
- **@ManyToOne** : pour configurer la relation (plusieurs à un).
- **@JoinColumn** : pour configurer le champ de jointure pour une relation.

En plus des entités, des modèles ont été créés. La manipulation des données se fait à l'aide de ses modèles. Toute entité sera transférée en classe métier (modèle) correspondante pour toute manipulation d'informations, et les objets métiers seront de même convertis vers leurs classes entité pour insertion ou sauvegarde des changements dans la base de données.

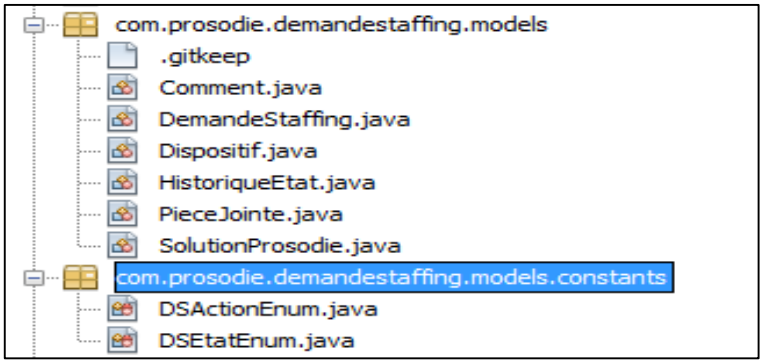

Figure 26 - Les modèles du module « Demande Staffing »

Les conversions entre entité et modèle (dit Data Access Object (DAO)) se font par les deux méthodes « toDao() » et « toEntity() ».

2.1.2. Couche d'accès aux données

La couche d'accès aux données est implémentée à l'aide de Spring Data JPA. Les interfaces d'accès aux données sont regroupées dans un package « repositories ». Chaque interface de ce package représente une interface d'accès aux données pour une entité. Toutes ses interfaces étendent de l'interface « CrudRepository » [15] proposée par Spring Data JPA qui prend deux types génériques (l'entité, et le type de l'id de la table) et qui contient un ensemble de méthodes utiles tel que :

- count() : renvoie le nombre des entités.
- save(entity) : sauvegarde l'entité donnée.
- delete(entity) : supprime l'entité donnée. (On peut donner l'id comme paramètre au lieu de l'entité elle-même.
- findAll() : renvoie toutes les instances.
- findOne(ID) : renvoie l'instance avec l'id donné.

Il est possible aussi d'ajouter nos propres méthodes en ajoutant une nouvelle méthode (abstraite) et spécifier la requête à exécuter en HQL à l'aide de l'annotation « @Query ».

Voici un exemple d'interface d'accès aux données (Repository). Ce repository contient une nouvelle méthode qui permet de filtrer les demandes de staffing à l'aide des filtres. Il n'est plus besoin d'implémenter le traitement ou ajouter la requête en SQL chaque fois qu'on veut filtrer les demandes, mais on peut appeler directement cette méthode à l'aide de repository :

```
public interface DmtCopDemandeStaffingRepository extends CrudRepository<DmtCopDemandeStaffing, Long> {

    @Query("SELECT d FROM DmtCopDemandeStaffing d WHERE (contact_str =:contactSTR or :contactSTR < 0) AND "
            + "(:demandeur < 0 OR demandeur =:demandeur) AND "
            + "(departement_demandeur IN (:departement) or '' IN (:departement)) AND " + "(d.etat IN (:etat)) AND "
            + "(:projet < 0 OR projet =:projet)")
    List<DmtCopDemandeStaffing> findByFilters(@Param("departement") List<String> departement,
            @Param("etat") List<DSEtatEnum> etat, @Param("contactSTR") int contactSTR,
            @Param("demandeur") int demandeur, @Param("projet") int projet);

}
```

Figure 27 - Interface d'accès aux données « DemandeStaffingRespository »

Les classes « Repository » sont regroupées dans un package « repositories ».

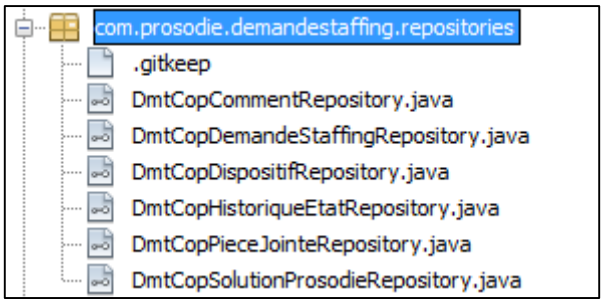

Figure 28 - Les repositories du module « Demande Staffing »

2.1.3. Couche métier

La couche métier est représentée par des contrôleurs. Le module de staffing contient un seul contrôleur MVC, c'est le contrôleur qui permet de faire la correspondance entre la vue de la demande de staffing et l'URL d'accès au module. De plus ce contrôleur renvoie quelques informations de base à afficher dans la vue. Parmi ses informations, on trouve :

- La liste des compétences par leurs groupes.
- La liste des groupes (départements).
- La liste des langues.
- La liste des états.

2.1.4. Web services REST

Comme déjà mentionné [cf. Chapitre 2, Paragraphe 4.1] la communication entre Front-end et Back-end se fait à l'aide d'un API REST et des appels AJAX. L'ensemble de contrôleurs REST est regroupé dans le sous-package « API » du package « controllers ».

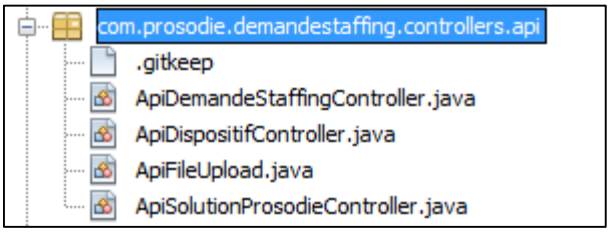

Figure 29 - Les contrôleurs REST du module « Demande Staffing »

Chacun des contrôleurs regroupe l'ensemble des méthodes pour Suppression, Consultation, Modification et Création. En plus de ses quatre méthodes nécessaires, d'autres ont été ajoutées pour effectuer un traitement spécifique tel que l'application des actions pour le changement d'état, etc.

2.1.5. State pattern

L'interface « State » et l'ensemble de ses implémentations sont regroupés dans le package « states ».

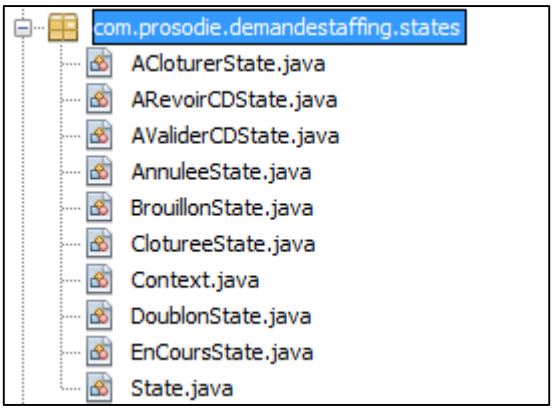

Figure 30 - Les différentes classes du State Pattern

La gestion des droits et erreurs dans le design pattern des états se fait à l'aide des exceptions. Suite aux messages de « Sonar » qui demandent d'implémenter nos propres exceptions au lieu de renvoyer une instance de classe « Exception », nous avons implémenté un ensemble d'exceptions.

Le diagramme des classes suivant décrit l'ensemble des classes des exceptions, et leurs liens de parenté.

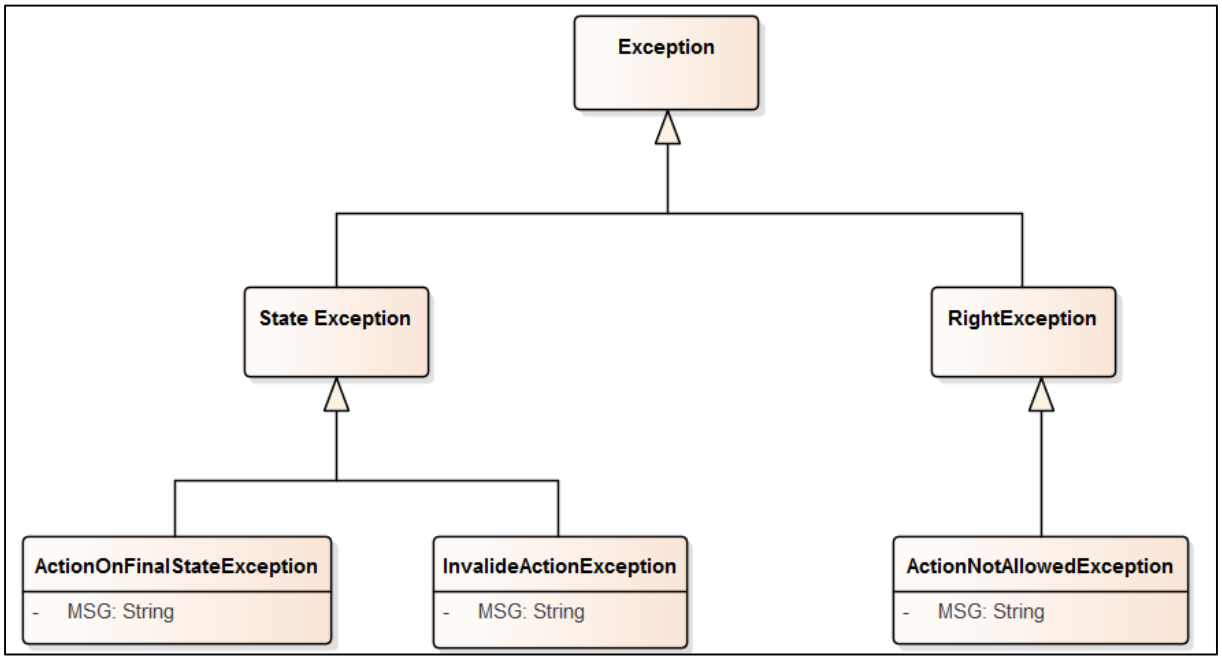

Figure 31 - Diagramme de classe des exceptions

- **ActionOnFinalStateException** : renvoyée lorsqu'une action est appelée dans un état final.
- **InvalidActionException** : renvoyée lorsque l'action ne peut pas être effectuée dans l'état courant de la demande.
- **ActionNotAlloweException** : renvoyée lorsque l'utilisateur n'a pas le droit d'effectuer l'action demandée.

Chacune de ces trois classes contient un attribut statique appelé « MSG » qui contient le message par défaut a renvoyer par l'exception. Il est toujours possible de renvoyer un message personnalisé au lieu du message par défaut.

La méthode « doAction() » des états renvoie par conséquent soit **StateException** ou **RightException**. Ce message est ensuite envoyé à l'utilisateur sous forme d'une notification.

2.2. Partie Front-end

La partie front-end est séparée de même en deux parties. La première partie consiste de différentes vues sous la forme des templates en fichiers HTML. La deuxième partie est de la forme des fichiers « TypeScript » qui s'occupe de tout ce qui est éventuellement dans la vue, ainsi qu'interaction avec le web service à l'aide des appels AJAX.

La gestion des demandes de staffing se fait à l'aide d'une vue qui contient la liste des demandes et à partir de là que nous pouvons aussi filtrer les demandes, créer une nouvelle demande et modifier une demande. De ce fait, la vue est séparée en trois sous vues :

- La Vue « **demande-staffing** » : c'est la vue qui contient le tableau avec la liste, et qui inclut aussi les deux autres vues afin de les afficher dans la même page. Cette vue inclut aussi un formulaire pour pouvoir filtrer les demandes à l'aide d'un ensemble de critères.
- La Vue « **add-demandestaffing** » : cette vue est de la forme d'un modal qui s'affiche en dessus de la liste des demandes et qui est de la forme d'un formulaire à remplir par les différentes informations d'une demande.
- La Vue « **update-demandestaffing** » : cette vue est de la forme d'un modal qui s'affiche en dessus de la liste des demandes et qui est de la forme d'un formulaire qui contient les informations de la demande avec possibilité de les modifier si l'utilisateur a le droit de le faire. Cette vue contient aussi deux onglets de plus :
    - Onglet « **Suivi de la demande** » : un onglet qui affiche la liste des commentaires et donne à l'utilisateur le pouvoir de supprimer ses commentaires ainsi que saisir un nouveau.
    - Onglet « **Historiques** » : un onglet qui affiche l'ensemble des actions et modifications sur la demande, l'utilisateur qui a effectué la modification et la date de cette modification.

2.2.1. Typescript

L'utilisation de TypeScript est un avantage très important et qui a aidé énormément lors de création des vues. En fait ceci est grâce à la possibilité de la réutilisation des modules génériques déjà développés au niveau de TypeScript.

Dans le COPWEB les pages sont de la forme des classes TypeScript qui étendent de la classe « ***page.PageBase*** » qui est une classe générique qui prend comme type une autre classe qui est de la forme d'une configuration et qui étend de la classe « ***page.PageConfigurationBase*** ».

Pour pouvoir créer une page, il faut tout d'abord créer une classe qui étend « ***page.PageConfigurationBase*** ». Cette classe doit contenir l'ensemble des composants JQuery et qui sont les composants, dont l'utilisateur interagit avec. Ensuite, il faut créer une classe qui étend « ***page.PageBase*** » sans oublier de donner comme type générique la classe de configuration de la page à créer.

Pour étendre de la classe « ***page.PageBase*** », il faut redéfinir deux méthodes :

- **initComponents*()*** : cette méthode doit contenir l'initialisation de tous les composants de la page. Par exemple initialiser le formule de mise à jour par les informations d'une demande.
- **initEvents*()*** : cette méthode doit initialiser les événements sur l'ensemble des composants de la page tel que les boutons, etc.

Après avoir créé la page, il faut indiquer quand la page doit être initialisée. Ceci se fait à l'aide de « ***PageFactory*** ». On indique à ce denier l'identifiant ou la classe HTML de composant à initialiser avec une fonction qui crée une instance de page et lui passe en paramètre une instance de configuration de la page. « ***PageFactory*** » fait appel à la méthode « init() » de la page qui de même fait appel au « initComponents() » puis « initEvents() ».

2.2.2. Composants Typescript

Nous avons utilisé plusieurs composants réutilisables pendant la création des différentes vues.

Voici la liste des composants que nous avons utilisés :

- **DataTables** [16] : c'est un plug-in pour la bibliothèque JQuery de JavaScript. C'est un outil très flexible, basé sur les fondements de l'amélioration progressive, et ajoutera des contrôles d'interaction avancés à n'importe quelle table HTML tel que :
  - Pagination, recherche instantanée et commande multi colonnes.
  - Prend en charge presque n'importe quelle source de données : Document Object Model (DOM), JS, AJAX et traitement côté serveur.
  - Des options étendues et une belle et expressive API.
  - Logiciel open source gratuit.

  À l'aide de ce plug-in tout traitement lié à la liste des demandes par exemple le rafraîchissement de la liste selon les filtres et l'affichage des détails d'une demande, peut être regroupé dans un module TypeScript largement maintenable et réutilisable. De plus il y'a plus besoin d'envoyer des nouvelles requêtes au serveur pour afficher les détails d'une demande, car il est possible d'accéder à la liste à partir de DataTable.
- **Autocomplete** : c'est un plug-in utilisé pour mettre en place des objets réutilisables pour créer des zones de texte avec la fonctionnalité d'auto complétion. Voici les implémentations que nous avons réalisées :
  - « **clientautocomplete** » : permet de mettre en place une zone de texte pour sélectionner un client parmi ceux existant dans la base de données en auto complétion.

- « **projectautocomplete** » : permet de mettre en place une zone de texte pour sélectionner un projet parmi ceux existant dans la base de données en auto complétion.
- « **userautocomplete** » : permet de mettre en place une zone de texte pour sélectionner un collaborateur en auto complétion.

- **Select2** [17] : Select2 offre une boîte de sélection personnalisable avec prise en charge de la recherche, du marquage, des jeux de données à distance, du défilement infini et de nombreuses autres options très utilisées.
  - **Les langues** : Select2 est livré avec un support pour les environnements RTL, la recherche avec des diacritiques et plus de 40 langues intégrées.
  - **Support de données à distance** : en utilisant AJAX, il est possible de rechercher efficacement de grandes listes d'articles.
  - **Entièrement extensible** : le système de plugin permet de personnaliser facilement Select2 pour fonctionner exactement comme nous le souhaitons.
  - **Création d'éléments dynamiques** : permet aux utilisateurs de taper une nouvelle option et de l'ajouter à la volée.
  - **Support du navigateur complet** : le support pour les navigateurs modernes et anciens est intégré, y compris Internet Explorer 8.

3. Présentation des interfaces

Après les phases d'étude de l'existant, la modélisation fonctionnelle et la conception, nous avons développé les interfaces de notre application.

L'interface principale est l'interface de consultation des demandes. La liste des demandes est sous forme d'un tableau permettant d'effectuer des filtres sur la liste. Le contenu du tableau peut être changé selon les états des demandes.

À partir de cette interface, nous pouvons accéder à l'interface de création ou de modification d'une demande.

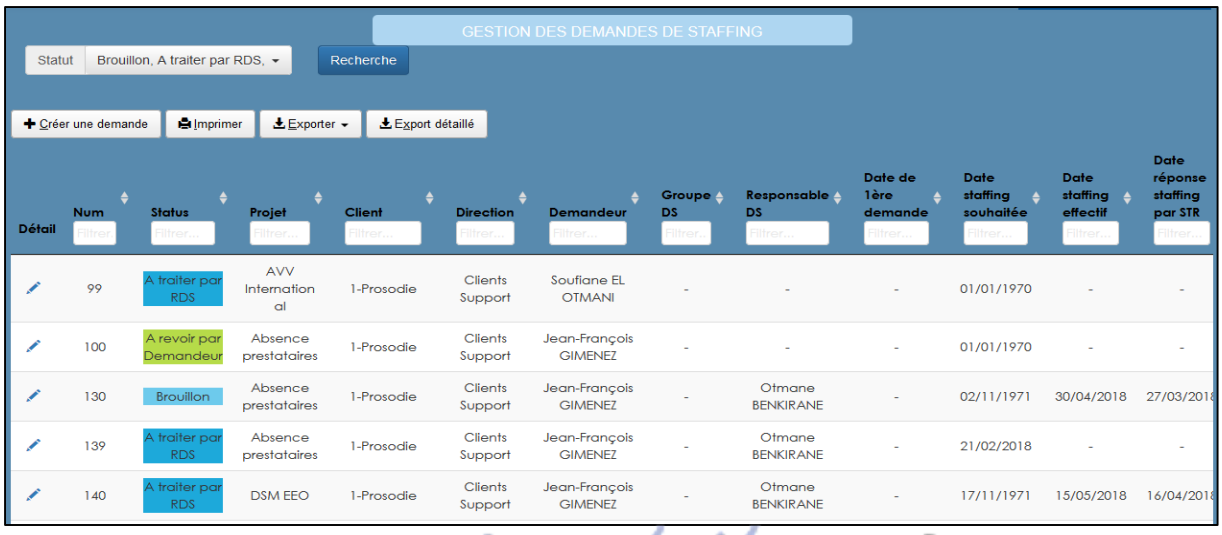

Figure 32 - Interface de consultation des demandes

Il y'a aussi la possibilité d'imprimer ou d'exporter le contenu de la table selon des filtres sous forme d'un fichier PDF, Excel ou CSV.

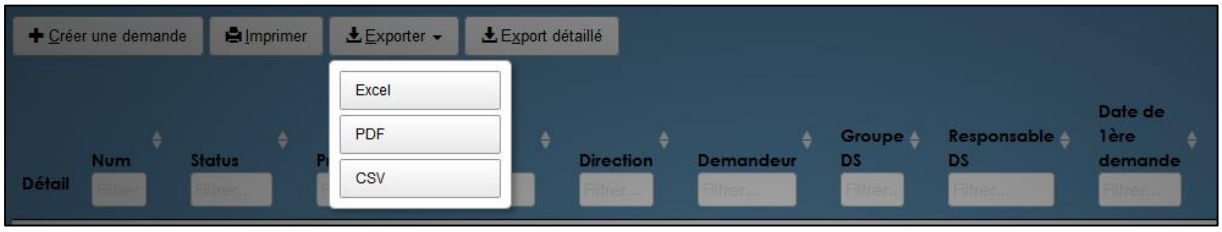

Figure 33 - Les différents formats d'un fichier exporté

La figure ci-dessous montre un exemple des demandes exportées sous forme d'un fichier PDF.

COP WEB - Demandes Staffing | ListeCOP Web - Accueil

Détail	Num	Status	Projet	Client	Direction	Demandeur	Groupe DS	Responsable DS	Date de 1ère demande	Date staffing souhaitée	Date staffing effectif	Date réponse staffing par STR
	130	Brouillon	Absence prestataires	1-Prosodie	Clients Support	Jean-François GIMENEZ	-	Otmane BENKIRANE	-	02/11/1971	30/04/2018	27/03/2018
	139	A traiter par RDS	Absence prestataires	1-Prosodie	Clients Support	Jean-François GIMENEZ	-	Otmane BENKIRANE	-	21/02/2018	-	-
	140	A traiter par RDS	DSM EEO	1-Prosodie	Clients Support	Jean-François GIMENEZ	-	Otmane BENKIRANE	-	17/11/1971	15/05/2018	16/04/2018
	142	A traiter par RDS	DSM EEO	1-Prosodie	Clients Support	Rodolphe MARAVAL	-	Otmane BENKIRANE	-	01/06/2018	30/04/2018	16/04/2018
	143	Brouillon	Absence prestataires	1-Prosodie	Clients Support	Jean-François GIMENEZ	-	Otmane BENKIRANE	-	-	-	-
	147	A traiter par RDS	Touche BIS	PLACOPLATRE	Clients Support	Rodolphe MARAVAL	-	Otmane BENKIRANE	-	29/10/1971	08/05/2018	20/03/2018
	99	A traiter par RDS	AVV International	1-Prosodie	Clients Support	Soufiane EL OTMANI	-	-	-	01/01/1970	-	-
	100	A revoir par Demandeur	Absence prestataires	1-Prosodie	Clients Support	Jean-François	-	-	-	01/01/1970	-	-

Figure 34 - Liste des demandes exportée en PDF

La figure suivante présente l'interface d'ajout d'une demande de staffing.

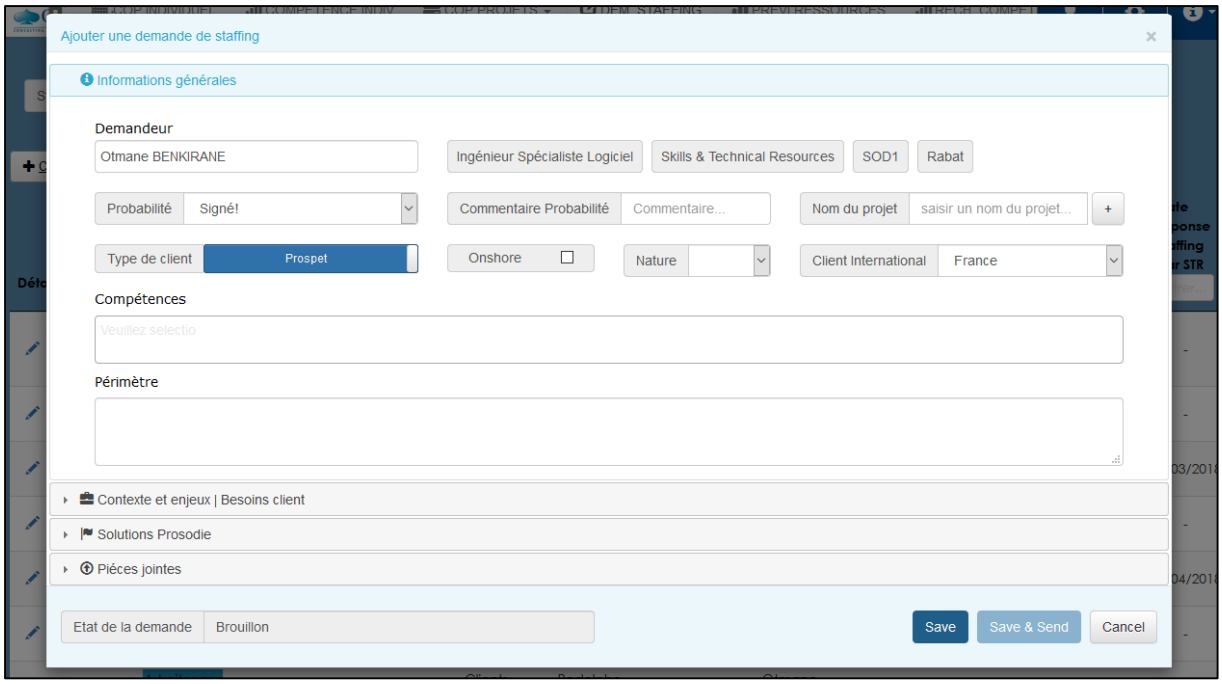

Figure 35 - Interface d'ajout d'une demande

La figure ci-dessous expose l'interface de modification d'une demande, il comprend trois ongles. Le premier contient les informations sur le demandeur, les compétences, etc. Le deuxième contient les commentaires sur une demande avec les informations du RDS. Le troisième comprend l'historique des actions effectuées sur une demande.

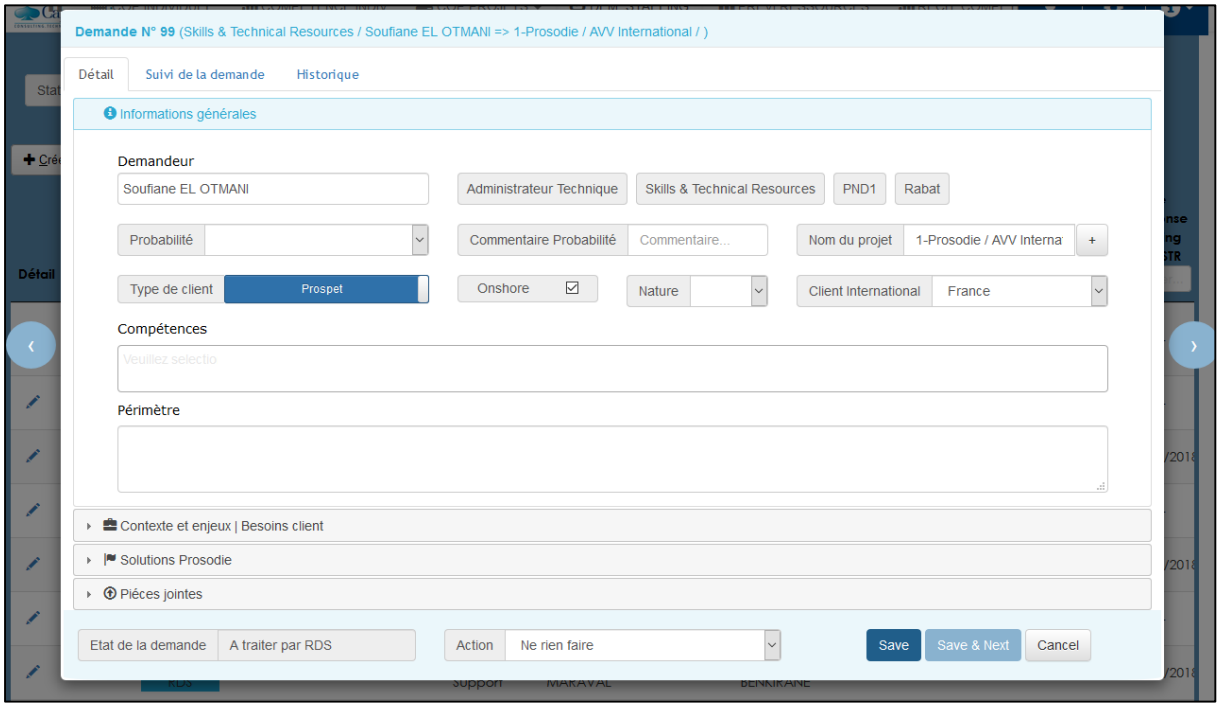

Figure 36 - Interface de modification d'une demande

Le demandeur peut aussi ajouter des documents clés contenant des informations sur la demande ou nécessaires pour le traitement de la demande.

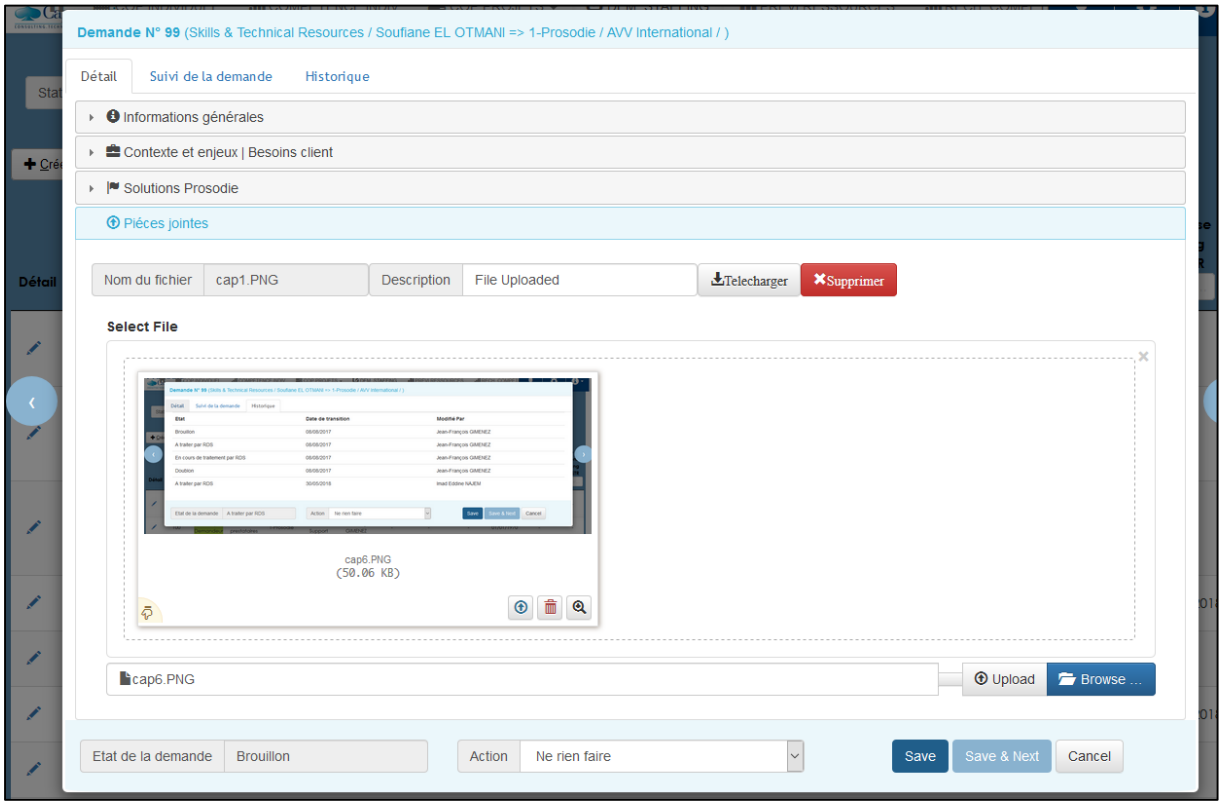

Figure 37 - Ajout des pièces jointes

Cet onglet contient les informations sur le RDS, les dates du staffing et les commentaires ajoutés par les différentes parties qui participent au traitement de la demande.

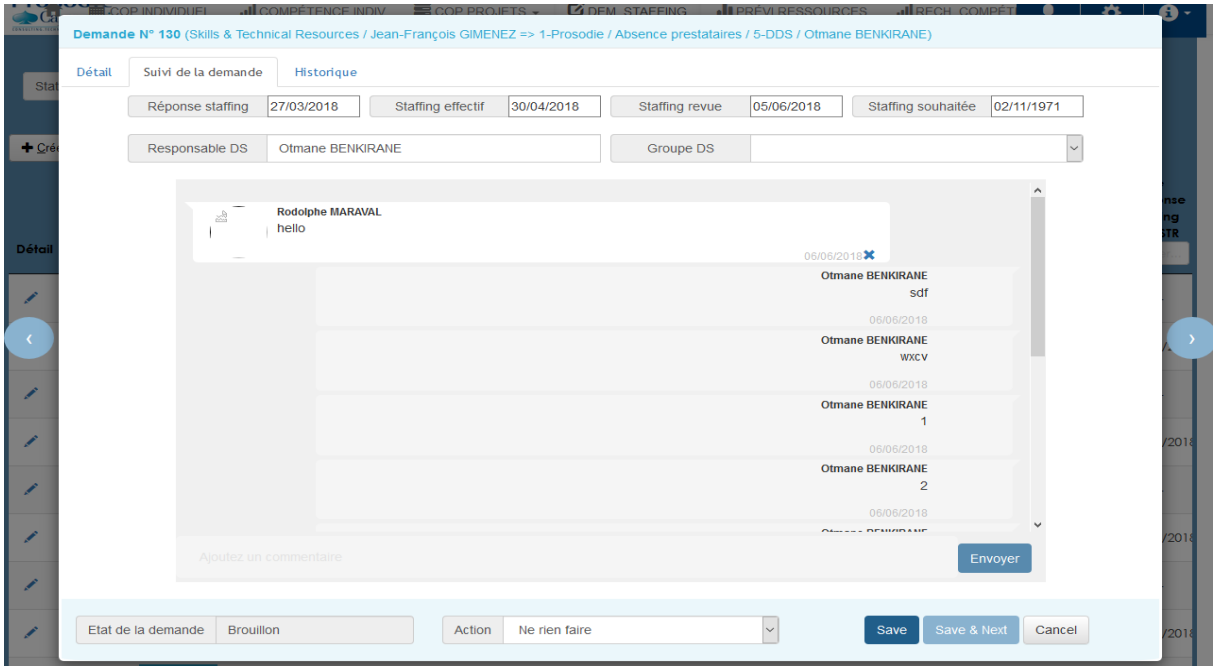

Figure 38 - Onglet « Suivi de la demande »

Le dernier onglet affiche l'historique des actions effectuées sur la demande.

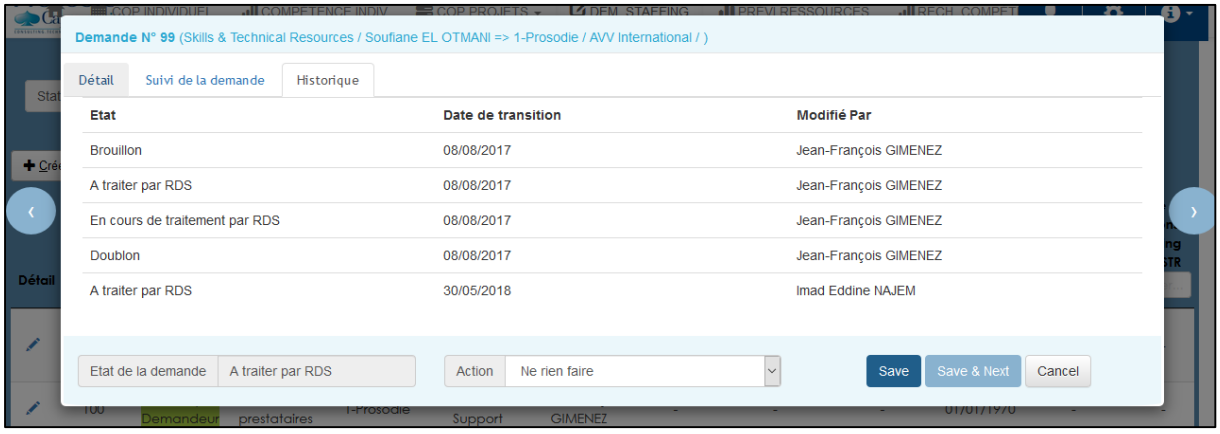

Figure 39 - Onglet « Historique »

La figure suivante montre le message d'erreur affiché lorsqu'un utilisateur essaye d'effectuer une action sur un état final d'une demande.

Seul l'administrateur peut effectuer une action sur une demande à un état final.

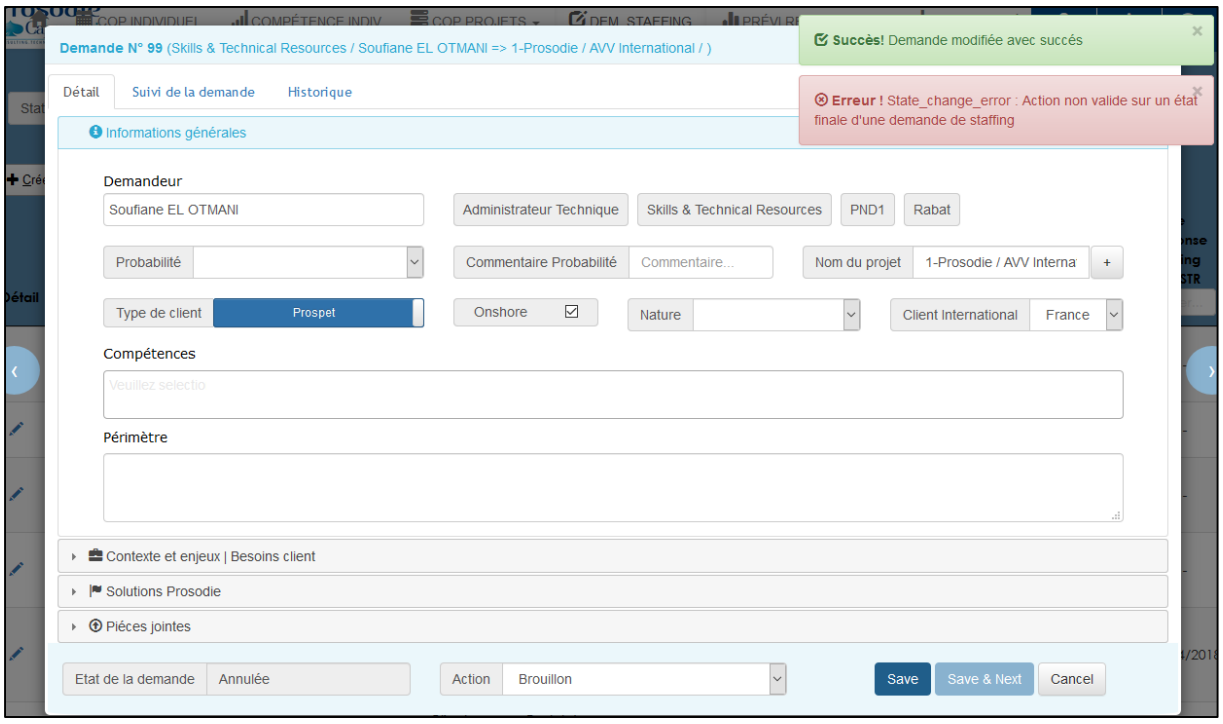

Figure 40 - Erreur d'une action non valide sur un état final

Conclusion

Tout au long de ce chapitre, nous avons abordé l'environnement de travail. Ensuite, nous avons détaillé des deux parties de système, les différents composants et classes utilisés, afin de présenter finalement les différentes principales interfaces de notre application réalisée.

Conclusion générale

Les entreprises font face à de multiples défis dans l'économie mondiale. Pour rester compétitives sur ce marché exigeant, elles adoptent constamment les dernières technologies afin d'améliorer les services et d'augmenter leurs résultats. Cette dynamique technologique a créé une forte demande de personnel hautement qualifié pour la mise en œuvre de projets critiques. Cela nécessite les bons moyens pour gérer efficacement le staffing pour ces projets.

Afin de gérer les demandes de staffing pour ses projets, Prosodie s'en sert des fichiers Excel et des Macros pour la communication avec la base de données. Ce système de gestion remonte plusieurs problèmes tels que :

- La difficulté d'utilisation.
- La perte de temps liée à la saisie multiple des données.
- Des problèmes de sécurité et de fiabilité des données.
- Le système résiste le changement.

Afin de pallier ces problèmes, nous avons choisi de remplacer le système actuel par une application web pour la gestion des demandes de staffing.

Pour arriver au résultat attendu, nous avons commencé par une étude du système existant afin d'identifier les différentes exigences du futur système. Ensuite, nous avons entamé une étude conceptuelle des demandes de staffing afin de mettre en œuvre une application permettant la gestion et le suivi des demandes de staffing.

Ce travail est encore d'actualité et ne s'arrête pas à ce niveau. En effet, l'application peut être accessible de façon plus rapide qui rend l'expérience utilisateur beaucoup plus agréable, nous proposons d'utiliser le fameux framework de Javascript « Angular » afin de séparer totalement tout ce qui Back-end de tout ce qui est Front-end, de ce fait nous pouvons même développer des fonctionnalités avec des langages différents et y accéder à l'aide des web services.

Finalement, vu l'accomplissement de projet, nous souhaitons très fortement qu'il soit le fruit du progrès, de l'évolution et qu'il reste à la hauteur des exigences de l'entreprise Prosodie.

Webographie

[1] http://igm.univ-mlv.fr/~dr/XPOSE2008/SCRUM/presentation.php Description de la méthode Agile Scrum pour la gestion des projets, consulté en Mars 2018

[2] https://www.commentcamarche.com/faq/24318-concept-de-l-integration-continue Concept de l'Intégration Continue, consulté en Mars 2018

[3] http://www.croes.org/gerald/blog/qu-est-ce-que-rest/447/ Concept de l'architecture REST, consulté en Avril 2018

[4] http://www.bdpedia.fr/applications/ Le concept des applications orientées données, consulté en Mars 2018

[5] http://www.opentuto.com/quest-ce-que-le-framework-spring-2/ Présentation du framework spring, consulté en Mars 2018

[6] https://projects.spring.io/spring-data/ Présentation du framework Spring Data, consulté en Avril 2018

[7] http://www.commentcamarche.net/faq/19368-hibernate-partie-1-presentation Présentation du framework Hibernate, consulté en Mars 2018

[8] https://www.thymeleaf.org/documentation.html Documentation du Template Engine Thymeleaf, consulté en Mars 2018

[9] http://www.supinfo.com/articles/single/445-presentation-typescript Présentation de Typescript, consulté en Mars 2018

[10] https://www.typescriptlang.org/docs/home.html Documentation de Typescript, consulté en Mars 2018

[11] https://apprendre-a-coder.com/es6/ Introduction à ECMA, consulté en Avril 2018

[12] https://www.atlassian.com/git/tutorials/what-is-git Présentation de système de contrôle de version GIT, consulté en Mars 2018

[13] https://openclassrooms.com/courses/gerer-son-code-avec-git-et-github/github-qu-est-ce-que-c-est Introduction à Github, consulté en Mars 2018

[14] https://linsolas.developpez.com/articles/java/qualite/sonar/?page=page_1 Contrôle de la qualité de ses projets avec Sonar, consulté en Mars 2018

[15] https://docs.spring.io/spring-data/commons/docs/current/api/org /springframework/ data/repository/CrudRepository.html Documentation de l'interface CRUDRepository, consulté en Avril 2018

[16] https://datatables.net/ Le site officiel du plugin Datatable de JQuery, consulté en Mai 2018

[17] https://select2.org/ Le site officiel du plugin Select2 de JQuery, consulté en Mai 2018

Conception et réalisation d'une application de gestion des demandes de staffing

Résumé

Pour améliorer sa performance, l'entreprise d'aujourd'hui vise à remplacer le système traditionnel de gestion interne de ses activités basé sur des fichiers Excels par les dernières technologies web. D'ailleurs c'est le cas de Prosodie Capgemini Maroc qui souhaite développer une application web de gestion des demandes de staffing qui va remplacer l'ancien système utilisant des fichiers Excels et des Macros pour interagir avec la base de données. L'objectif de cette application est de fournir à Prosodie une alternative au système traditionnel ainsi que des nouvelles fonctionnalités qui vont aider à améliorer les circonstances de déroulement des projets et assigner le bon personnel au bon projet. Pour mener à bien cette mission, nous avons commencé par une étude approfondie du fonctionnement du système traditionnel pour l'analyse et la définition des besoins, afin de mettre en œuvre une nouvelle application permettant la gestion et le suivi des demandes de staffing.

Mots clés: demande de staffing, Spring, REST, Typescript, MySQL

Conception and realization of an application to manage staffing requests

Abstract

To enhance its performance, today's enterprise seeks to replace the traditional activities management system based on Excel files with the latest web technologies. To ensure that its projects has sufficient resources with the right skills based on project schedule for successful completion, Prosodie Capgemini wants to develop a web application to manage staffing demands that will replace the old system based on Excel files and Macros to interact with the database. The application will provide new features in addition to those offered by the old system, they will help improve the circumstances of projects and assign the right staff to the right project. To complete this mission, we began with a thorough study of the operation of the old system to analyze and to define the needs of the new application, in order to implement a new application allowing the management and the follow-up of staffing demands.

Keywords: staffing request, Spring, REST, Typescript, MySQL

Master Systèmes Intelligents & Réseaux
Département d'Informatique
Faculté des Sciences et Techniques de Fès
A.U. 2017 - 2018