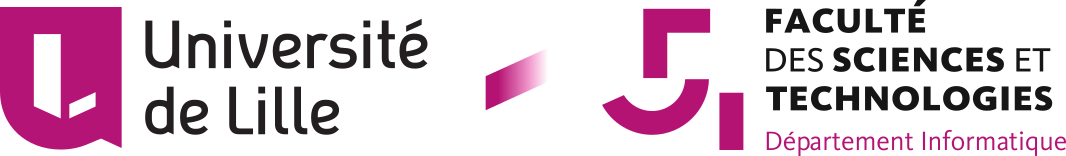
*Université de Lille, Cité Scientifique, 59650 Villeneuve-d'Ascq*



**Maintenance et ajout de fonctionnalités d’une application interne**

**CONFIDENTIEL**

Durée : 8 avril - 2 août 2019

Soutenu le 13 juin 2019

Par : Dylan Quentin

**Licence 3 mention informatique, parcours MIAGE**

*Encadrant entreprise : Thomas Shakeshaft*

*Encadrant universitaire : Jean-Claude Tarby*



# Remerciements

Pour commencer, je souhaiterais remercier toutes les personnes qui ont participé à la bonne réalisation de mon stage et m’ont ainsi permis de me fixer sur mon parcours professionnel.

Dans un premier temps, je remercie la société Norsys de m’avoir accueilli une seconde fois et plus particulièrement Sylvain BREUZARD, PDG du groupe Norsys, et Denis CASSORET, directeur d'agence, pour leur accueil et la confiance qu’ils m’ont accordée dès mon arrivée dans l’entreprise.

Je remercie également Mélanie WILFART, responsable des ressources humaines, pour m’avoir offert l'opportunité de refaire un stage à Norsys.

De plus je remercie Céline DE WILDE, chef de projet pour m’avoir accueilli, intégré dans l’équipe et m’avoir transmis ses connaissances dans la gestion de projets, ainsi que Thomas SHAKESHAFT, directeur des opérations qui est aussi mon tuteur de stage, pour m’avoir suivi durant ces derniers mois.

Je remercie également Thomas BUISINE, Nathan BERGAMINI, Hamid ABDELGHANI LIDRISSI et Thomas DEBLOCK, membres de mon équipe, pour les connaissances transmises et leur gentillesse lors de mon intégration.

Enfin, je souhaite remercier mon encadrant universitaire, Jean-Claude TARBY, professeur à l’Université de Lille, ainsi que les différents professeurs de communication pour leurs conseils dans la rédaction d’un rapport de stage et dans la communication au sein d’une entreprise ainsi que les différents professeurs de la licence 3 MIAGE pour m’avoir formé afin de réaliser ce stage.

# Résumé

Ce rapport intitulé “Maintenance et ajout de fonctionnalités d’une application interne” a été rédigé par Dylan Quentin. Le but de celui-ci est d’expliquer l’essentiel du travail réalisé en tant que développeur stagiaire à Norsys Nord, localisée à Ennevelin. Il s’agit d’une ESN ayant pour conviction de créer un équilibre entre les valeurs économique, humaines, sociétales et environnementales.

Ma mission principale, durant ces trois mois, fut la réalisation de différentes évolutions sur l’application interne de Boulanger nommée “B4CO”. En plus de cela, j’ai travaillé sur la maintenance des fonctionnalités déjà existante.

Ces différentes missions ont pour but, à terme, de faire vivre l’application tout en l’améliorant en fonction des différents besoins du client.

L’évolution principale sur laquelle j’ai travaillé concerne la possibilité, pour certains utilisateurs, de poser des heures de délégations sur leur temps de présence.

Ce travail a été réalisé au sein d’une équipe de six personnes composée de, Thomas BUISINE, Nathan BERGAMINI, Hamid ABDELGHANI LIDRISSI, Thomas DEBLOCK, moi-même et enfin Céline DE WILDE qui a piloté le projet.

Tout au long de ce projet, différents échanges ont été réalisés avec le client afin de détailler certaines de ses demandes, et permettre à celui-ci de suivre l’avancement du projet. À ce jour l’évolution principale sur laquelle j’ai travaillé a été finalisée et sera mise en production prochainement. La partie maintenance est, quant à elle,toujours alimentée et de nouvelles évolutions sont en préparations.

# **Table des matières**

[**Introduction**](#_2idopvoq3jyu) **5**

[**1. Le contexte**](#_92vey9evrwcm) **6**

[1.1. Présentation de l’entreprise](#_r46sdutrl5de) 6

[1.2. La stratégie de Norsys](#_g0zxqjnku5qe) 7

[1.3. Mes missions](#_9arw592vx0nc) 8

[**2. Ma contribution dans le projet**](#_a29jjyqidj42) **10**

[2.1 La TMA de B4CO](#_qff18vyahle9) 10

[2.1.1 Déroulement de la TMA](#_9s8vciun02sp) 10

[2.1.2. Agilité et TMA](#_z6wkprjwk1qt) 11

[2.1.3 L’importance de GIT](#_6zwleaa0zvsp) 12

[2.1.4 Résolution et validation](#_o8aj9pbuic9q) 14

[2.2. Evolution sur la création de délégation](#_ihqnaidm6ezn) 15

[2.2.1. Objectif de la mission](#_52stg6v6zxc1) 15

[2.2.2. Gestion du projet](#_q3bxgqsznzcr) 16

[2.2.3. Solutions techniques](#_gubu83k34z40) 17

[2.3. Résultats et difficultés](#_dlmtvun7xt6x) 21

[**Conclusion**](#_z95159nlr1s5) **23**

[**Bilan**](#_sn40j5nvxl1b) **24**

[**Bibliographie**](#_7vc32dv5bnbx) **25**

[**Annexes**](#_3nttrntem6b7) **26**

[**Glossaire**](#_hx7zl61khix3) **35**

[**Table des figures**](#_acfv6becgfcb) **37**

# 

# Introduction

Au terme de mon année de licence 3 parcours informatique mention MIAGE, j’ai réalisé un stage d’une durée de quatre mois. Ce stage a pour but de mettre en application les connaissances acquises cette année et tout au long des années précédentes. C’était également pour moi l’occasion d’approfondir mes connaissances à la fois techniquement et sur le monde de l’entreprise. Ce n’est pas ma première expérience dans ce domaine car j’ai réalisé l’année dernière un stage dans la même entreprise, Norsys Nord à Ennevelin. La différence avec l’année précédente c’est que cette fois j’ai travaillé sur un projet client et non sur un projet interne. J’ai décidé de retourner dans cette entreprise car l’ambiance de travail m’avait vraiment plu et je savais que la mission correspondait à ce que je cherchais dans ce deuxième stage.

La mission qui m’a été confiée était de développer les nouvelles fonctionnalités que voulait mettre en place Boulanger, sur leur application interne appelée “B4CO”. La partie sur laquelle j’ai le plus travaillé est la création d’une nouvelle interface permettant à certains utilisateurs de poser des heures de délégations. Ces employés sont des élus qui peuvent, pendant leur temps de travail, réaliser des tâches qui sont propres à leurs fonctions. Le client avait rédigé un cahier des charges dans lequel il avait indiqué les différentes exigences pour cette fonctionnalité. Pour réussir à répondre aux différentes attentes du client, nous avons dû communiquer régulièrement avec le celui-ci, et adapter nos développements en fonction des technologies déjà présentes sur le projet.

En plus des évolutions, il y avait une partie de maintenance de l’application. Pour cette partie, le but était de résoudre les différents problèmes que les utilisateurs rencontrent. Pour ce faire nous avions un outil de ticketing, Mantis, afin de lister les différents problèmes et de montrer leur évolution au client. Je n’ai pas beaucoup fait de maintenance mais cela m’a quand même permis de me familiariser avec l’application.

Le problème qui se pose est : Comment répondre aux attentes du client tout en s’adaptant au choix technologique qui avait été fait sur le projet et continuer de maintenir l’application ? Pour répondre à cela je vais dans un premier temps définir le contexte de mon stage avec une présentation plus détaillée de l’entreprise et de la mission. Ensuite je vais montrer ma contribution au sein du projet, d’abord sur la maintenance de l’application et pour finir sur l’évolution concernant les heures de délégations.

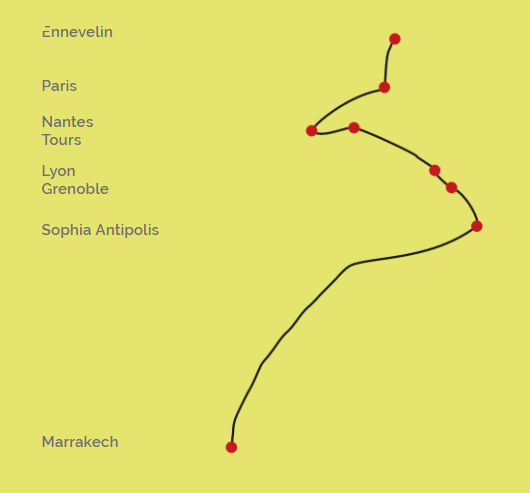
# 1. Le contexte

## 1.1. Présentation de l’entreprise

Norsys est une entreprise de service du numérique (ESN) anciennement appelée société de services et d’ingénierie en informatique (SS2I). Elle a été fondée en 1994 par un groupe de trois personnes composé à la fois d’ingénieurs et de dirigeants possédant beaucoup de compétences dans le domaine des systèmes d’information. Sylvain BREUZARD le PDG de Norsys et président du conseil d'administration de GREENPEACE a pour objectif de faire de l’entreprise “un leader technologique dans le domaine des Systèmes d'Informations”. L’entreprise a remarqué que depuis quelques années le niveau moyen des informaticiens était en baisse. Refusant de s’engouffrer dans un “business trop facile” l’entreprise a décidé depuis 2015 de tendre vers le haut de gamme.

Pour Norsys le haut de gamme se prouve à tous les niveaux de l’entreprise peu importe la place et le niveau des collaborateurs. C’est pour cela que pour faire du haut de gamme il faut être minutieux à tous les détails même les plus petits. C’est ainsi que des moyens et des dynamiques humaines sont mises en place dans l’entreprise. C’est tous ces principes que j’ai appris pendant mon stage.

Norsys s’est adaptée, au long des années, aux évolutions des différentes technologies : client/serveur de première et de deuxième génération, Internet/Intranet, E-Commerce, J2EE et serveurs d'applications. Aujourd’hui l’entreprise s’est spécialisée dans la prise en charge de grands projets intégrant les nouvelles technologies, avec une forte proximité du système d’information tout en donnant des conseils et des expertises techniques. Aujourd’hui Norsys se décompose en plusieurs secteurs d’activités, les plus importants sont dans les domaines de : la santé, le commerce, l’assurance, les services et également en télécom et médias.

L’entreprise est divisée en 8 agences, qui sont listées sur la figure 1, et l’agence d’Ennevelin est le siège social de l’entreprise. 

### 

Fig 1: Liste des agences

Comme nous l’avons déjà dit Norsys travaille avec beaucoup de clients dans différents secteurs d’activité, nous pouvons voir certains de ses clients sur la figure 2. Beaucoup de missions sont réalisées directement chez le client comme pour Leroy Merlin. Mais l’ambition de l’entreprise est d’avoir un taux de mission beaucoup plus important en agence. Pour mon cas j’ai travaillé sur un projet pour Boulanger et ce projet a été réalisé en agence.



Fig 2: Logos de quelques clients (anciens ou actuels) de Norsys

## 1.2. La stratégie de Norsys

La stratégie mise en place par le PDG Sylvain BREUZARD s’appelle la “Performance globale”. A cette idée sont associés trois challenges pour faire de Norsys un exemple de modèle économique, environnemental et sociétal. Cette stratégie n’a pas été développée seule mais par les collaborateurs. Afin d’y arriver plusieurs actions ont été menées comme des démarches d’écoute et de veille, un séminaire de créativité et la formation de plusieurs groupes de travail dont l’objectif est de trouver des nouvelles idées et d’évaluer la réalisation de celle-ci.

Trois challenges ressortent de cette stratégie :

1. “ Devenir la plus grande des sociétés de conseil et d'ingénierie à taille humaine “
2. “ Être la société de conseil et d'ingénierie la mieux notée par ses clients “
3. “ Être la société de conseil et d'ingénierie où l'on est le mieux pour travailler “

Les noeuds borroméens sont le symbole de Norsys, il s’agit de l’union de tous les challenges qui forment un tout pour l’entreprise comme nous pouvons le voir sur la figure Fig3.

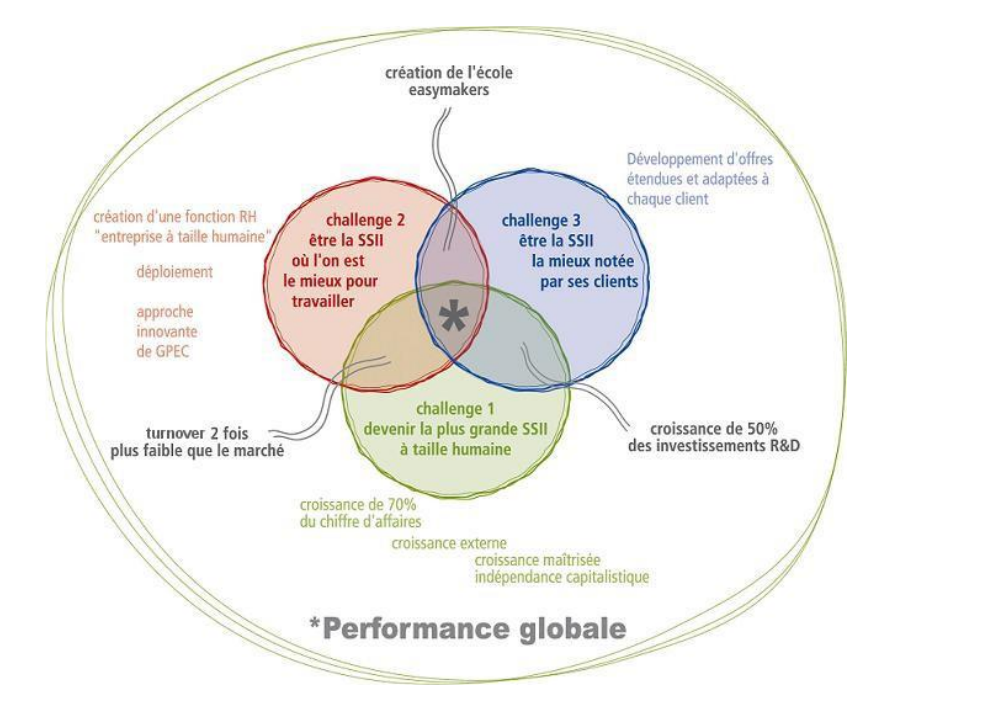


Fig 3: Schéma de la performance globale

## 1.3. Mes missions

Dès le début de mon stage j’ai été intégré à l’équipe qui travaille sur “B4CO” qui est la nouvelle application interne de Boulanger. Cette équipe était composée de Thomas BUISINE, Nathan BERGAMINI, Hamid ABDELGHANI LIDRISSI et de Céline DE WILDE qui est pilote de ce projet. Au bout de quelques jours l’équipe a été renforcée avec Thomas DEBLOCK.

C’est Thomas B, qui est la personne de l’équipe à avoir le plus d’expérience sur le projet, qui m’a expliqué l’existant et le but de ce projet. Celui ci est de maintenir l’application, donc de faire de la TMA, partie que nous allons appeler le “run”, et également de faire des évolutions de l’application, partie que nous allons appeler le “build”. En annexe il y a l’écran d’accueil de l’application.

Cette application vient remplacer celle qui existait précédemment “BForYou”. Elle a été développée principalement par le prestataire Pictime mais la TMA a été confiée à Norsys. C’est une grosse application qui s’occupe par exemple de la génération des plannings, avec un moteur sous contrainte nommé OptaPlanner, du suivi de présence d’un employé, de l’édition des objectifs mensuels et pleins d’autres choses encore. Elle a été mise en production en mars de cette année dans les magasins Boulanger. Cette mise en production et le fait que l’application offre beaucoup de fonctionnalités, font que les retours des utilisateurs sont assez nombreux et pas mal d’anomalies sont détectées. J’ai eu l’occasion de faire quelques correctifs, j’ai du par exemple supprimer un message d’erreur qui n’avait pas lieu d’être quand l’utilisateur avait une session expirée. Grâce à ces correctifs j’ai pu découvrir comment était structurée l’application et mieux me repérer dans le projet. Le problème de ce projet c’est qu’il n’y a pas de documentation, c’est donc parfois compliqué de s’y retrouver.

Lors de ce stage j’ai beaucoup travaillé sur la partie build, l’évolution principale était la mise en place de la possibilité pour les élus de poser leurs heures de délégations sur l’application. Afin de bien répondre aux attentes du client nous avions un cahier des charges qu’on retrouve en annexe 4. J’ai réalisé le développement avec Hamid mais le reste de l’équipe était quand même présent pour nous assister lorsque nous avions des questions ou des blocages.

# 

# 2. Ma contribution dans le projet

Comme dit précédemment B4CO est un projet récent, il manque encore des fonctionnalités pour qu’il soit complet. De plus la mise en production récente de l’application a engendré beaucoup de retour de la part des utilisateurs. En vue de la montée de charge que Boulanger ont reçue, ils ont décidé de confier la maintenance à un prestataire externe, Norsys, mais ont également décidé de confier des évolutions à réaliser.

Je vais détailler dans cette partie les deux missions qui m’ont été confiées, dans un premier temps je vais parler de la TMA puis finalement je parlerai de l’évolution sur laquelle j’ai travaillé.

## 2.1 La TMA de B4CO

### 2.1.1 Déroulement de la TMA

Afin de mieux comprendre comment se déroule la procédure de la TMA entre Boulanger et Norsys, j’ai réalisé un schéma, figure 4 ci-dessous, qui montre tous les états par lesquels passe une anomalie.

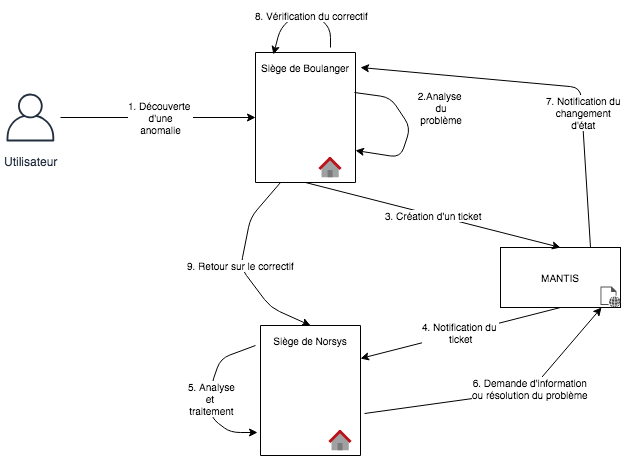


Fig 4: Schéma de la procédure de la TMA

Lorsqu’un utilisateur, peu importe son rôle dans l’entreprise, rencontre une anomalie sur l’application, il va remonter l’information jusqu’au siège de Boulanger. Ensuite les personnes en charge de l’application vont vérifier que l’anomalie en est bien une et si c’est le cas elle sera transmise sur l’application **MANTIS**. Il s’agit d’une application de ticketing, c’est-à-dire que les utilisateurs peuvent créer des tickets afin de remonter une anomalie au service technique (ici Norsys). Ensuite celui-ci pourra mettre à jour l'avancement de sa résolution jusqu'à sa clôture. En annexe se trouve la liste des tickets présents sur l’application le mercredi 29 mai 2019 (annexe 1) et également le détail d’un ticket (annexe 2).

Une fois le ticket créé, notre équipe est informée de l'existence de celui-ci. Le ticket va d'abord être analysé afin d’être chiffré en **jour-homme**. Ensuite en fonction de son niveau d’importance nous allons le traiter plus ou moins rapidement. Si lors de l’analyse nous remarquons qu’il manque des informations pour bien comprendre le problème, nous changeons l’état du ticket en “attente d’information” et en ajoutant une note pour détailler les informations manquantes. Une fois l’anomalie réglée, nous devons mettre à jour l’avancement du ticket sur MANTIS afin d’informer Boulanger que le problème a été réglé. L’anomalie est alors ensuite testée de leur côté pour valider ou non le correctif.

En plus des échanges sur MANTIS, la communication avec le client est très importante à la fois par mail et par téléphone. C’est grâce à ces échanges que l’avancement des correctifs est plus efficace.

### 2.1.2. Agilité et TMA

Afin de mieux expliquer comment nous traitons les anomalies je vais prendre l’exemple d’un ticket que j’ai corrigé. Lors de l’utilisation de l’application il y a plusieurs observations qui ont été réalisées par Boulanger sur la connexion. Déjà il n’y avait pas de lien de redirection vers la page “mot de passe oublié” et de plus lorsqu’il y avait une session expirée, l’application nous montrait deux messages d’erreurs. Ils ont alors créé un ticket nommé “mire de connexion” sur Mantis.

Comme nous travaillons en agilité, ce ticket a été ajouté dans la colonne “à faire” du tableau d’avancement, tableau que nous pouvons trouver en annexe 3. Un ticket sur le tableau d’avancement (voir figure 5) possède un code couleur, vert pour le run et bleu pour le build, un numéro qui indique l’identifiant MANTIS auquel correspond ce ticket et les initiales de la personne qui est en train de travailler dessus.

# 

# 

# Fig 5: Ticket sur le tableau d’avancement

Le tableau nous permet de suivre l’avancement d’un ticket d’une manière plus claire et plus pratique que MANTIS. De gauche à droite, par suite logique, nous trouvons :

* A faire : c’est la liste des tâches que nous devons réaliser.
* En cours : ici il faut mettre la tâche sur laquelle nous sommes en train de travailler. Il ne faut pas mettre plus d’un ticket par personne sinon cela n’a aucun sens.
* Attente d’info : le ticket est mis ici lorsque nous avons besoin d’information de la part du client.
* A valider : c’est lorsque notre tâche doit être validée par un membre de l’équipe.
* Dev : notre correctif se trouve sur l’environnement de dev et doit être testé par le client.
* Qua : le correctif a été validé, mais des tests peuvent encore être réalisés dessus avant la mise en production. Il s’agit de l’environnement de qualité.

### 2.1.3 L’importance de GIT

Maintenant que la présentation du tableau d’avancement a été faite nous pouvons continuer de suivre l’évolution du ticket “mire de connexion”.

Une fois ajouté au tableau d’avancement, je me suis occupé de cette tâche. C’est une décision qui a été prise en équipe car je venais d’arriver sur le projet et d’après eux, c’est le genre de tâche qui allait me faire comprendre la structure du projet.

Le plus compliqué pour moi était de comprendre dans un premier temps si le problème venait de la partie **front** du projet ou de la partie **back**. Car en effet le projet est divisé en plusieurs projet **GIT** : une partie front-end qui s’occupe de l’affichage des données, une partie back-end qui elle, s’occupe du traitement des données, une partie configurations qui va regrouper tous les fichiers de configurations du projet et également la partie **OptaPlanner** qui va elle générer les plannings. Nous retrouvons ces projets sur la figure 6, sauf OptaPlanner car nous ne touchons pas à cette partie du projet.

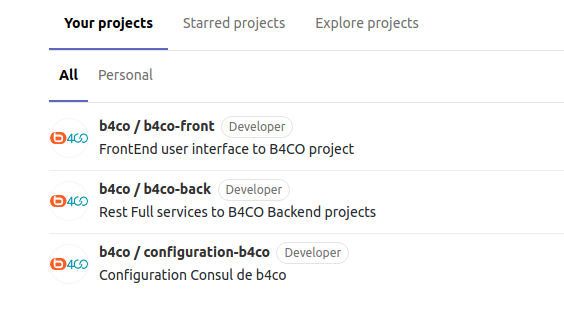


Fig 6: Projets sur le GIT

GIT est un outil de contrôle de version, il nous permet de stocker nos projets et d’en contrôler les versions. Si j’ajoute des fonctionnalités sur mon projet je peux les ajouter dans le GIT afin de le faire évoluer, mais si je change d’avis je pourrais revenir à la version précédente.

Grâce à GIT nous avons la possibilité de créer une **branche** dans le projet, une branche est la copie d’une autre afin de pouvoir travailler sur des versions de code qui divergent.

Afin de mieux expliquer le fonctionnement des branches sur ce projet, il faut savoir qu’il y a différents environnements de l’application :

* L’environnement de dev, qui permet de tester les différents développements que nous faisons.
* La qua (ou la qualité), qui est l’environnement de pré-production, il y a des données différentes de celle de la dev et les jeux de tests sont plus complets, il s’agit d’une sécurité avant la mise en production.
* La production, qui est l’environnement final donc celui sur lequel l’application tourne avec des vraies données et des vrais utilisateurs.

Sur GIT l’environnement de dev et de qua correspondent à la branche develop et l’environnement de prod à la branche master. Lorsque nous faisons un correctif nous devons créer une branche à partir de develop (qui représente l’environnement de dev). Cela permet ainsi aux développeurs de travailler sur des tâches différentes et de faire évoluer l’application plus rapidement.

Pour notre cas nous avons créé la branche “feat/43542” avec 43542 l’identifiant MANTIS de notre ticket. Une fois que le développement est fait je vais réaliser une **merge request** de ma branche vers develop. Il s’agit d’une demande faite à une autre personne de l’équipe qui devra regarder ces changements et vérifier que rien n’est incohérent. Si c’est le cas, il va l’accepter et mes changements se retrouveront sur la branche develop. Ensuite lors de la mise en production il se retrouvera sur la branche master. C’est tout cet enchaînement qui est représenté sur la figure 7.

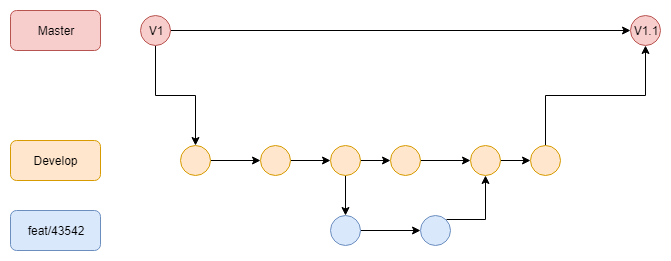


Fig 7: Gestion des branches GIT

### 2.1.4 Résolution et validation

Après avoir créé ma branche, j’ai cherché à l’aide des outils de développements disponibles sous chrome, comme la console, le network (qui permet de voir les requêtes du front vers le back) et l’inspecteur d’élément, l’origine du deuxième message d’erreur. L’erreur venait bien de la partie front. Pour le problème de redirection il fallait juste ajouter une ligne dans le message d’erreur de la connexion, ce message se trouve dans un fichier **JSON**. L’**IDE** que j’utilise est Intellij et les outils disponibles permettent de naviguer plus simplement dans le projet. Une fois mon correctif apporté j’ai passé ma tâche sur le tableau d’avancement dans la colonne “à valider”, le ticket sur MANTIS à l’état “résolu” et j’ai affecté une merge request à un membre de l’équipe.

Maintenant que le ticket a été résolu il reste à le faire valider par le client. Pour qu’il soit testé il faut le mettre sur l’environnement de dev. Pour cela on utilise **Jenkins**, il s’agit d’un outil d’intégration continue, son fonctionnement est simple, lorsque nous lançons un “build”, l’outil va récupérer le code présent sur la branche develop et va vérifier que ces changements ne créent aucun problème et si c’est le cas le build va échouer. Lors de cette procédure il va par exemple vérifier que tous les tests passent. Si le build fonctionne, notre correctif sera présent sur l’environnement de dev et pourra être testé par notre contact direct avec Boulanger, il s’agit d’Alban DUPONCHEL, chef de projet. Il va ensuite décider des correctifs qui seront présents lors de la prochaine mise en qual afin d’être **recetté** par Boulanger. Cela signifie qu’un ensemble de tests va être réalisé afin de vérifier que les changements correspondent bien aux attentes. Si la recette est correcte alors les changements seront mis en production lors de la prochaine mise en prod. C’est tout ce processus qui est résumé dans la figure 8.

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

Fig 8: Procédure de validation

## 2.2. Evolution sur la création de délégation

### 2.2.1. Objectif de la mission

Cette évolution est découpée en deux parties, une partie employée et une partie manager. Les employés doivent pouvoir poser des heures de présences de délégations. Ces heures ne concernent que le personnel élu, comme par exemple les représentants syndicaux. Il s’agit d’heures durant lesquels les employés réalisent un travail propre à leur fonction d’élu. Les employés doivent avoir la possibilité de poser des heures et les managers eux de poser des heures aux personnes de leur équipe.

Avant notre évolution la procédure était la suivante :

* Le titulaire d’un mandat donne un bon de délégation à son manager avec le motif des

heures et le nombre d’heures prises.

* Le manager transmet le bon de délégation à son gestionnaire de paie.
* Le gestionnaire de paie saisie dans le Back end les infos : heures de délégation, plage couverte par la délégation.
* Impacts sur l’annexe du bulletin de paie.

Cette procédure étant assez longue, ils ont souhaité donner la possibilité aux élus de poser leurs heures directement sur B4CO.

### 2.2.2. Gestion du projet

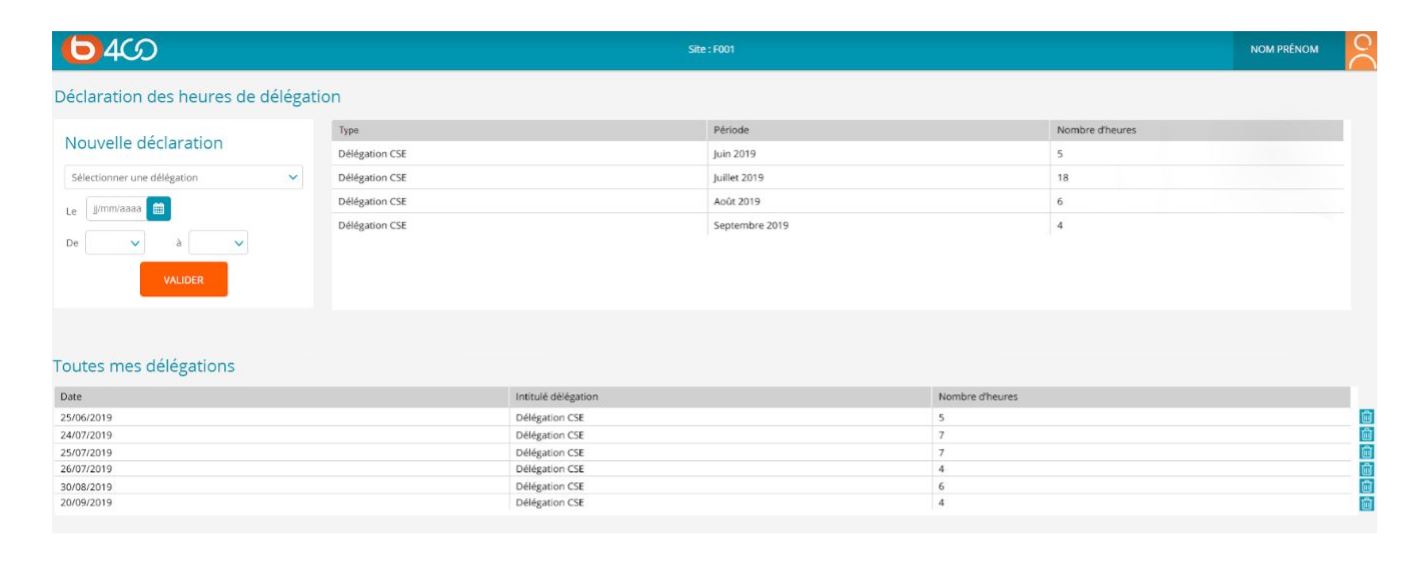
Afin d’exprimer leurs besoins, Boulanger nous ont fourni un cahier des charges dans lequel est spécifié ce que fait chaque partie de l’évolution et les différentes règles de gestion qu’il faut suivre. Un extrait du cahier des charges est disponible en annexe 4.

Suite à ce cahier des charges, il y a eu un atelier de découpage et de chiffrage des tâches, je n’ai pas été présent à cet atelier, il a été fait avant mon arrivée dans l’équipe, mais cela m’a permis de voir les différentes choses qu’il fallait faire pour cette évolution. Comme évoqué précédemment, il y a deux parties : la partie employée et la partie manager.

Dans la partie employée il y avait trois grosses parties :

* La création d’une délégation, avec choix du type, de la date et de la plage horaire.
* L’affichage des contingents de délégations, on y retrouve le nombre d’heures posées par mois et par type de délégation.
* L’affichage des délégations qui correspond à la liste de toutes les délégations qui ont été posées par l’employé.

Chaque affichage doit être filtré sur les dates du mois précédent à deux mois après. On retrouve ces trois parties sur la maquette disponible dans le cahier des charges, figure 9 ci-dessous.

Fig 9: Maquette de la page employée

Pour la partie manager, il faut ajouter la possibilité de poser des heures pour les employés de leur équipe. Sur l’application ils peuvent déjà poser des heures d'absences et de présences mais il faut en plus rajouter les heures de délégations. En effet lorsqu’ils accèdent

au planning de leur équipe, ils choisissent un jour puis ils sélectionnent une plage horaire et ensuite ils doivent choisir un type d'événement grâce à une popin. Suite à chaque sélection dans cette popin il faut remplir différentes informations, là pour notre cas il faut juste rajouter la plage horaire comme nous pouvons le voir sur la figure 10. De plus dans ces deux parties il faut traiter les différentes erreurs possibles comme une saisie de date incorrecte. 

Fig 10: Maquette de la popin manager

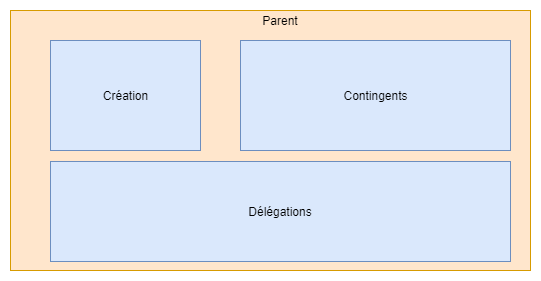
Lors de cette évolution nous avons repris beaucoup de principes de l’agilité. Comme dit dans la partie sur la TMA, nous avons un tableau d’avancement sur lequel nous faisons suivre l’avancement de chacune des tâches. Nous avons rajouté sur le tableau toutes les tâches qui ont été identifiées lors de l’atelier de découpage. Pour rappel, afin de les différencier des charges sur la TMA nous avons utilisé un code couleur, vert pour la TMA et bleu pour les évolutions. Nous avons réalisé un seul **sprint** pour cette évolution. Afin de le démarrer nous avons eu une réunion dans les locaux de Boulanger le 16 avril pour rappeler les enjeux et l’organisation du sprint. Il avait été convenu une date de démo, qui signifiait sa fin, mais lors du sprint elle a été décalée, je reviendrais là-dessus dans la partie 2.3. Résultats et difficultés.

Ensuite nous l’avons démarré et nous avons échangé régulièrement avec le client lorsque nous avions des questions. Dans l’équipe nous faisons des **daily** tous les matins, c’est un rituel pendant lequel chaque membre de l’équipe explique ce qu’il a fait la veille et dit ce qu’il compte faire dans la journée. Cela a plusieurs buts, déjà nous pouvons nous rendre compte de l’avancement des tâches mais aussi de remarquer si l’un des membres de l’équipe est en difficulté.

Suite au développement, c’est au tour de la démo dans laquelle nous présentons les évolutions au client puis finalement vient la **rétro**. Il s’agit d’une petite réunion effectuée à la fin de chaque sprint dans le but de faire le point sur ce qui était bien lors de cette itération et sur ce qui peut être amélioré.

### 2.2.3. Solutions techniques

Sur cette évolution je n’ai travaillé que sur la partie employée, je vais expliquer ici les solutions qui ont été apportées pour cette partie.

Il y avait trois **composants** à réaliser. Il s’agit d’une partie de l’interface, on peut y trouver juste de l’affichage de liste, un formulaire etc… Ici nous distinguons bien que sur notre écran, voir le schéma de la figure 11, qu’il y a un composant pour la création d’une délégation, un pour l’affichage des contingents et un pour l’affichage des délégations. Nous avons créé un composant principal qui représente Fig 11 : Gestion des composants 

l’ensemble de la page puis de ce composant nous en avons créé les trois autres cités précédemment. Chaque composant est composé de quatre fichiers : la page HTML qui est la vue, le contrôleur qui va appeler le bon service et traiter les données, le SCSS qui va donner du style à notre page et finalement un fichier de test.

En vue de comprendre comment chaque composant récupère et traite les données je vais prendre l’exemple de la liste des contingents, on retrouve le schéma de ce fonctionnement sur la figure 12. On notera que cette architecture reprend les principes de l’architecture **REST**.

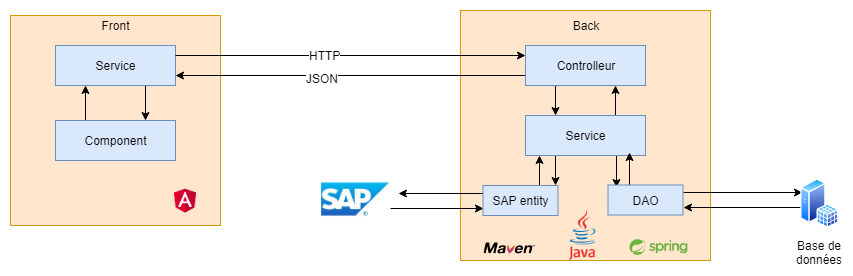
Lors du chargement de la page le contrôleur du composant mère va appeler le service qui récupère la liste des contingents, celui-ci va faire une requête **HTTP** au back dans notre cas il va s’agir d’un **GET** car nous voulons récupérer des données et ce get se fera avec une **URI** spécifique. Dans le back le contrôleur possédant l’URI appelé par le front va faire un appel vers le service avec les paramètres fournis et va parfois rajouter des paramètres comme la gestion des dates. Le service va retourner les informations de **SAP**.

Il s’agit d’un progiciel de gestion intégré qui permet de gérer les processus d’une entreprise. Dans notre cas SAP va s’occuper de toutes les contraintes liées aux délégations, il va pouvoir nous fournir toutes les données dont on a besoin et lors de la création d’une délégation il va nous retourner une erreur si une contrainte n’est pas respectée (par exemple s’il y a un chevauchement avec un autre évènement).

Afin de récupérer les données disponibles sur SAP nous avons eu les requêtes qu’il fallait faire au serveur. Ces requêtes sont appelées dans le service avec un objet SAP entity dans lequel nous devons mettre les champs correspondant aux données. Le serveur va nous retourner une liste de cet objet que nous pourrons alors traiter dans le contrôleur. Avant d’être renvoyé au front le contrôleur va **mapper**, c'est-à-dire qu’il va établir une correspondance, l’objet SAP entity en objet JSON.

Le service du front va récupérer le JSON et le mapper en objet angular. Le contrôleur du front va lui récupérer la liste puis la page HTML va afficher les données des objets de la liste.

De l’annexe 5 à 9 se trouve des extraits du code sur la récupération des contingents du front vers le back. On retrouve le composant mère en annexe 5, le service front en 6, le contrôleur back en 7, le service qui va faire l’appel SAP en 8 et finalement l’objet SAP entity en 9.

Fig 12: Echange de données dans l’application

Pour le composant de la liste des délégations c’est le même principe que pour la liste des contingents, pareil pour la création et la suppression sauf que là on va faire des **POST** et l’objet retour sera une réponse HTTP.

En plus de la récupération et le traitement des données il fallait faire un contrôle sur la saisie de l’utilisateur. Ces vérifications devaient être gérées dans la partie front.

Sur le formulaire de création, il fallait afficher un message d’erreur si l’utilisateur essayé de saisir dans le passé. Pour ce faire dans le code HTML, j’ai rajouté des **directives** angular. Pour cette erreur un ngIf qui permet d’afficher un élément seulement si la condition est vraie. À chaque changement de date une fonction dans le contrôleur est appelée, cette fonction va vérifier la validité de la date saisie, si la saisie est correcte il n’y aura aucune erreur par contre s’il y a une erreur de format la condition pour l’affichage de l’erreur de format sera changé en vrai et s’il y a une date dans le passé c’est la condition pour l’affichage de l’erreur de date qui sera à vrai. Les messages d’erreurs se trouvent dans un fichier JSON ce qui permet de les centraliser et de les modifier facilement.

Un autre contrôle devait être réalisé sur la plage horaire saisie. Tout d’abord l’heure de début ne pouvait pas être supérieure à l’heure de fin. Il y avait également une condition particulière lorsque l’employé est cadre, si c’est le cas il ne peut saisir que des plages de quatre heures sinon un message d’erreur doit être affiché. Le plus compliqué avec ce traitement était de récupérer le statut d’un employé. Il y avait bien des services back qui permettent de récupérer l’information mais aucun n’avait été encore utilisé dans ce but. J’ai dû créer un contrôleur pour récupérer l’information qui nous intéressait. Ensuite le traitement de l’erreur était similaire à celui de la date. On peut voir en figure 13 deux erreurs de saisies qui sont affichées.



## 

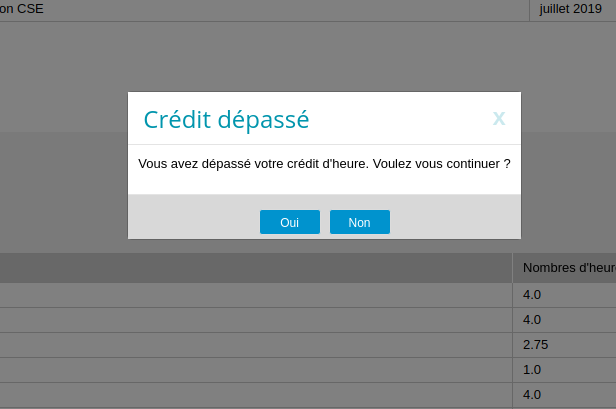
## 

Fig 13: Messages d’erreurs lors de la création

Les délégations sont soumises à une limite. Il y a un compteur qui varie selon chaque employé et selon le mois. Lors de la création il faut vérifier que ce compteur n’est pas dépassé. Le client voulait qu’il y ait une vérification et que s’il est dépassé, l’utilisateur puisse quand même avoir la possibilité de poser ses heures mais d’afficher un message préventif.

Le compteur est récupéré en même temps que les contingents donc le contrôle n’est pas difficile. Ce qui était plus compliqué, c’était la création d’une popin car nous sommes sur une ancienne version d’angular et dans le projet l’affichage de popin est géré par la librairie EXT. Mon équipe et moi-même avons décidé de ne pas utiliser cette librairie, car elle n’est pas assez récente et elle est compliquée à manier. Ces choix sont aussi faits car dans les mois à venir il devrait y avoir une montée de version du projet et nous ne voulons pas ajouter du code obsolète.

Nous avons utilisé une modal qui est quelque chose de spécifique à angular et qui permet d’afficher des boîtes de dialogue. Elle s’ouvre quand le crédit est dépassé et demande à l’utilisateur s'il souhaite tout de même continuer. On la retrouve ci-dessous sur la figure 14.



## 

## 

## 

Fig 14: Popin de confirmation

Afin d’améliorer la qualité du projet nous avons mis en place des tests, qui étaient déjà présents dans le back mais pas dans le front. L'important ici est de vérifier que chaque partie du code se comporte comme elle devrait et que les règles du cahier des charges sont respectées. Une fois la configuration faite nous avons pu tester les différentes exigences de Boulanger comme la liste du type de délégation qui doit être triée par ordre alphabétique.

Côté back nous avons fait des tests à la fois sur le contrôleur et le service. Nous pouvons prendre en exemple le test réalisé sur le contrôleur qui récupère le statut d’un employé, le code est visible ci-dessous en figure 15. Le “when” présent dans le test permet de mocker le service appelé. Un **mock** est un objet simulé qui reproduit le comportement d’un autre objet. Ici on fait en sorte que le service nous retourne bien l’objet qui nous intéresse. Cela permet que le test soit bien unitaire, on traite seulement le comportement du contrôleur et non celui du service.

Nous vérifions que le contrôleur retourne bien un fichier JSON au bon format avec les bonnes données et qu'aucune erreur n’est détectée. Le verify à la fin permet de vérifier que l’appel au service a bien eu lieu.

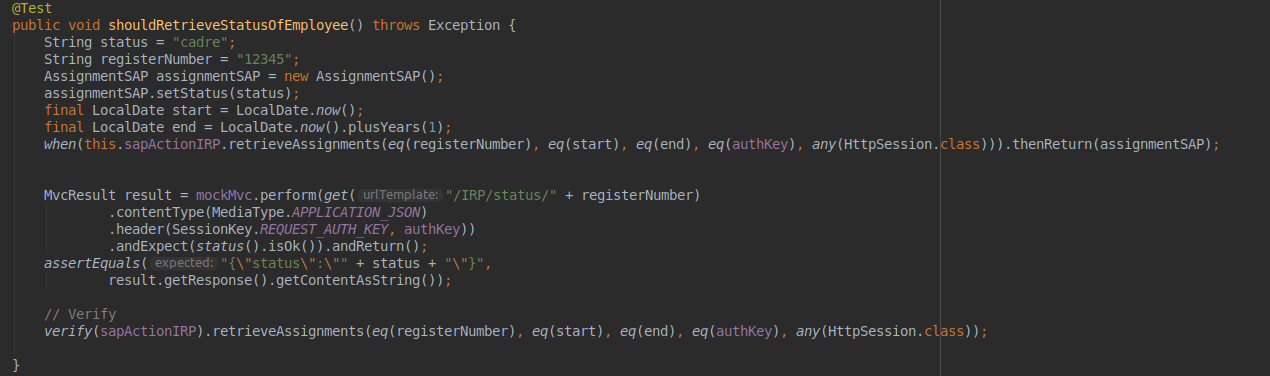


Fig 15: Test du contrôleur de récupération de statut

## 2.3. Résultats et difficultés

Dès mon arrivée sur le projet j’ai eu peur de ne pas m’en sortir car je n’avais jamais fait d’angular et je n’avais jamais travaillé sur un projet de cette taille. Malgré mes doutes j’ai vite réussi à m’y mettre, la TMA m’a permis de vite comprendre la structure du projet car il fallait beaucoup naviguer à l’intérieur de celui-ci. J’ai commencé à bien comprendre le fonctionnement d’angular une fois que j’ai travaillé sur la partie build. Nous avons commencé l’évolution en **pair programming** avec Hamid et cela m’a beaucoup aidé.

D’autres difficultées ont été rencontrées lors du sprint mais ce n’était pas des problèmes techniques mais plutôt des problèmes de compréhension. En effet le cahier des charges que nous avait fourni le client n’était pas assez précis et nous avons dû vite demander des compléments d’informations. De plus les jeux de tests étaient inexistants et parfois nous ne savions pas si ce que nous avions développé répondait vraiment aux attentes du client.

Nous avons dû décaler la démo, ce qui d’habitude ne se fait pas surtout en agilité, car nous avons reçu les appels SAP seulement 3 jours avant la démo initiale. Lorsque nous les avons reçu nous avons dû modifier quelques éléments à nos développements et cela nous a fait perdre beaucoup de temps.

Mais malgré toutes ces complications nous avons réussi à aller au bout du sprint et nous avons présenté les évolutions. Avant d’aller au siège de Boulanger, qu’on peut voir sur la figure 16, nous avons bien préparé la démo. Mon rôle à moi était de présenter la partie employée, que je maitrise le mieux. Lors de cette réunion était présent toute l’équipe Norsys, Alban qui est le chef de projet avec qui nous échangeons et trois autres chefs de projet responsables de B4CO.

# 

# 

# 

Fig 16: Siège social de Boulanger à Lesquin

Ils ont été très satisfaits de la présentation de la démo et de ce que nous avons produit. Cependant quelques retours ont été fait :

* Modifier le message lorsque heure de début < heure de fin : message incohérent.
* Pas de suppression possible dans le passé pour le collaborateur.

Après cette démo nous avons réalisé une rétro à Norsys afin de savoir ce qu’on devait garder sur ce sprint et savoir ce qui pouvait être amélioré. Ce que nous avons retenu :

* à garder : la communication régulière avec le client, l’ambiance d’équipe et la préparation de démo.
* à commencer : des revues de code afin d’avoir un code d’encore meilleure qualité, être plus exigeant sur les spécifications fonctionnelles et ne pas commencer un sprint sans avoir toutes les informations.

C’est grâce à ces rituels que l’équipe s’améliore et devient de plus en plus performante.

# Conclusion

Si je devais choisir une image du jeu Dixit pour imager mon stage, ça serait celle d’un garçon dans les airs, sur un mont de livre, et un autre personnage en bas sur un livre ouvert. Le personnage en bas représente mon arrivée à Norsys, avec un certain niveau de compétence et avec mes connaissances du monde de l’entreprise. Le personnage sur le mont de livre, correspond à ce que m’a apporté le stage, donc à moi aujourd’hui. Comme nous avons pu le voir dans l’ensemble de ce rapport, la diversité des outils utilisés et les différentes techniques d’agilité m’ont apporté beaucoup en connaissance technique et en gestion de projet. Ce fut également pour moi l’occasion de voir comment travaille une équipe qui est sur un projet client sans être chez le client, c’est quelque chose de très technique à mettre en place, mais l’expérience de travail en est meilleure. Le fait que le garçon soit dans les nuages, montre la liberté que j’ai ressentie dans cette entreprise, une liberté que je ne pense pas pouvoir avoir dans toutes les entreprises. 

Le résultat que j’ai produit sur B4CO correspond bien aux attentes du client. La page employé répond bien au cahier des charges, et toutes les règles de gestion ont été mise en place. Aujourd’hui, l’application est toujours en cours de maintenance, il y a toujours beaucoup d’anomalies à corriger, et des nouvelles évolutions vont être réalisées. Je ressens une grande fierté quant à la solution proposée, car le travail que j’ai apporté va être mis en production d’ici peu et va être utilisé par beaucoup d’utilisateurs. Ce que j’ai réalisé n’est peut être qu’une toute petite partie de B4CO, mais cela reste quelque chose de très grand pour moi.

Pour conclure, ce stage m’a rassuré sur mon projet professionnel. Avant celui-ci, j’avais décidé de continuer mes études en MIAGE car j’aime le développement informatique mais également le pilotage de projet. Ces trois mois m’ont permis de confirmer que j’étais dans la bonne voie. Mais en plus de faire mon master en MIAGE, je souhaite le faire en alternance et j’aimerais pouvoir continuer de travailler sur cette application afin de continuer d’évoluer et également de voir les prochaines évolutions à venir.

# 

# Bilan

Techniquement ce stage m’a apporté beaucoup de choses et m’a rappelé beaucoup de notions. Tout d’abord j’ai revu les normes REST que j’avais déjà eu l’occasion de voir l’an dernier en stage. Mais cette année contrairement à l’année dernière, j’ai appris à utiliser angular et je ne fais que progresser sur cette technologie.

Avant de parler du travail en équipe, je vais parler du travail en autonomie. Il y a en effet un véritable travail d’autonomie à faire afin de comprendre les technos, de se documenter et également de se rendre compte lorsqu’on bloque sur une tâche et qu’il nous faut de l’aide.

L’avantage du travail en équipe est que le partage de connaissances est beaucoup plus simple, c’est bien grâce à Hamid et Thomas que mes compétences en angular ont vite progressé.

Mais les relations humaines ne s’arrêtent pas au travail. Afin de mieux nous intégrer et également dans le but d’animer l’agence beaucoup d’événements ont eu lieu dans l’entreprise. Comme par exemple le campus qui est une soirée qui a pour but de rassembler tous les collaborateurs de Norsys le temps de deux semaines.

Après avoir vécu deux mois en stage chez Norsys je peux affirmer que mon choix d’étude est le bon et que je veux continuer en MIAGE mais de préférence en alternance afin d’avoir des compétences que je ne pourrais pas forcément avoir à l’université.

# 

# 

# 

# 

# Bibliographie

Des rappels sur REST

<https://blog.nicolashachet.com/niveaux/confirme/larchitecture-rest-expliquee-en-5-regles/>

Pour la standardisation d’une URI et sa définition

<http://www.standard-du-web.com/uniform_resource_identifier.php>

Afin de bien différencier tous les types de requêtes HTTP

<https://openclassrooms.com/fr/courses/1118811-les-requetes-http>

Sur l’agilité

<https://www.agiliste.fr/introduction-methodes-agiles/>

Sur l’effet tunnel

<https://www.innovaxion.net/2015/05/11/l-effet-tunnel-en-gestion-de-projet/>

Référentiel agile

<http://referentiel.institut-agile.fr/tdd.html>

Complément d’information

<https://www.wikipedia.org/>

Information sur l’entreprise

<https://www.norsys.fr/>

# 

# 

# 

# Annexes

Annexe 1 : Liste des tickets sur MANTIS

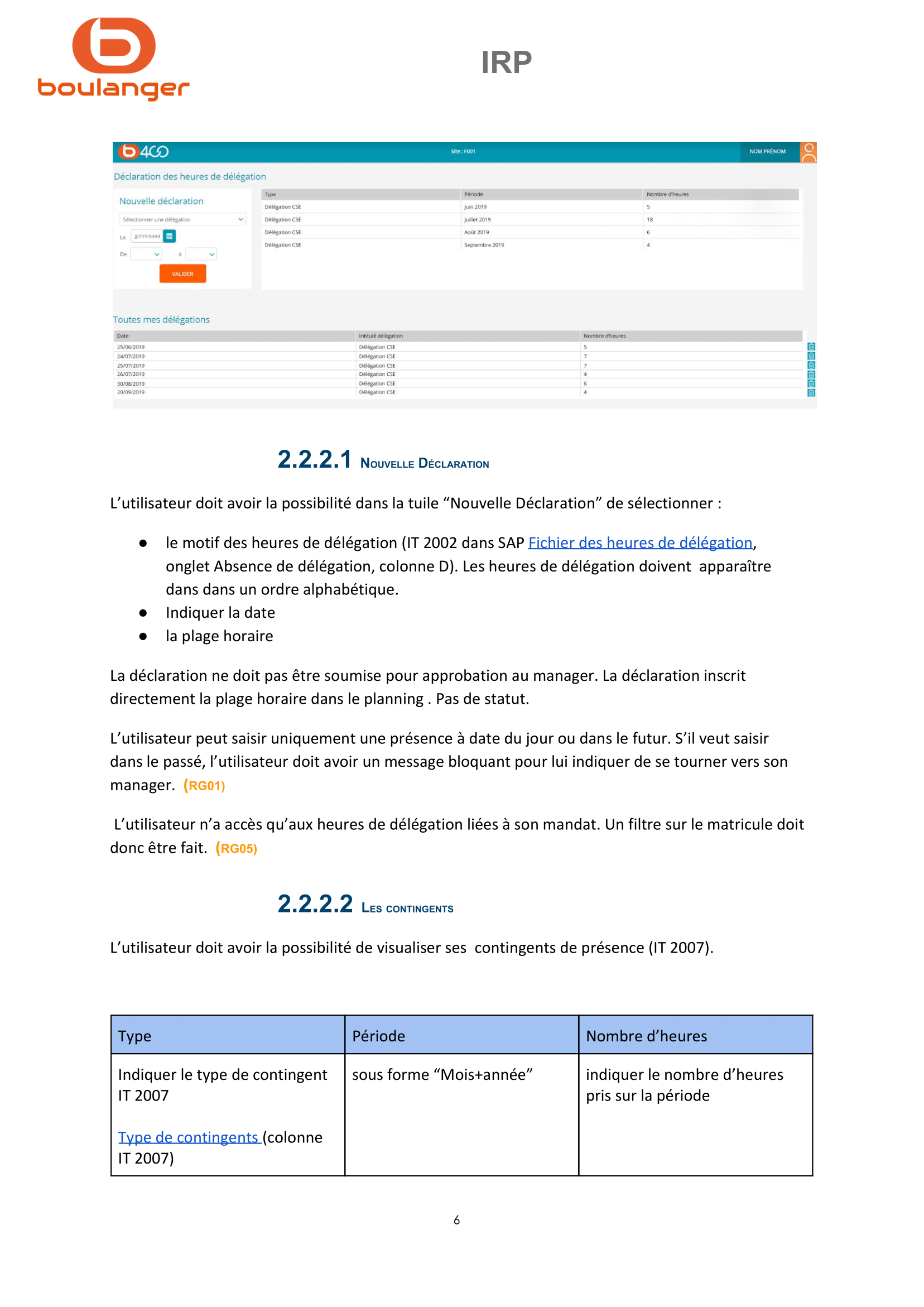


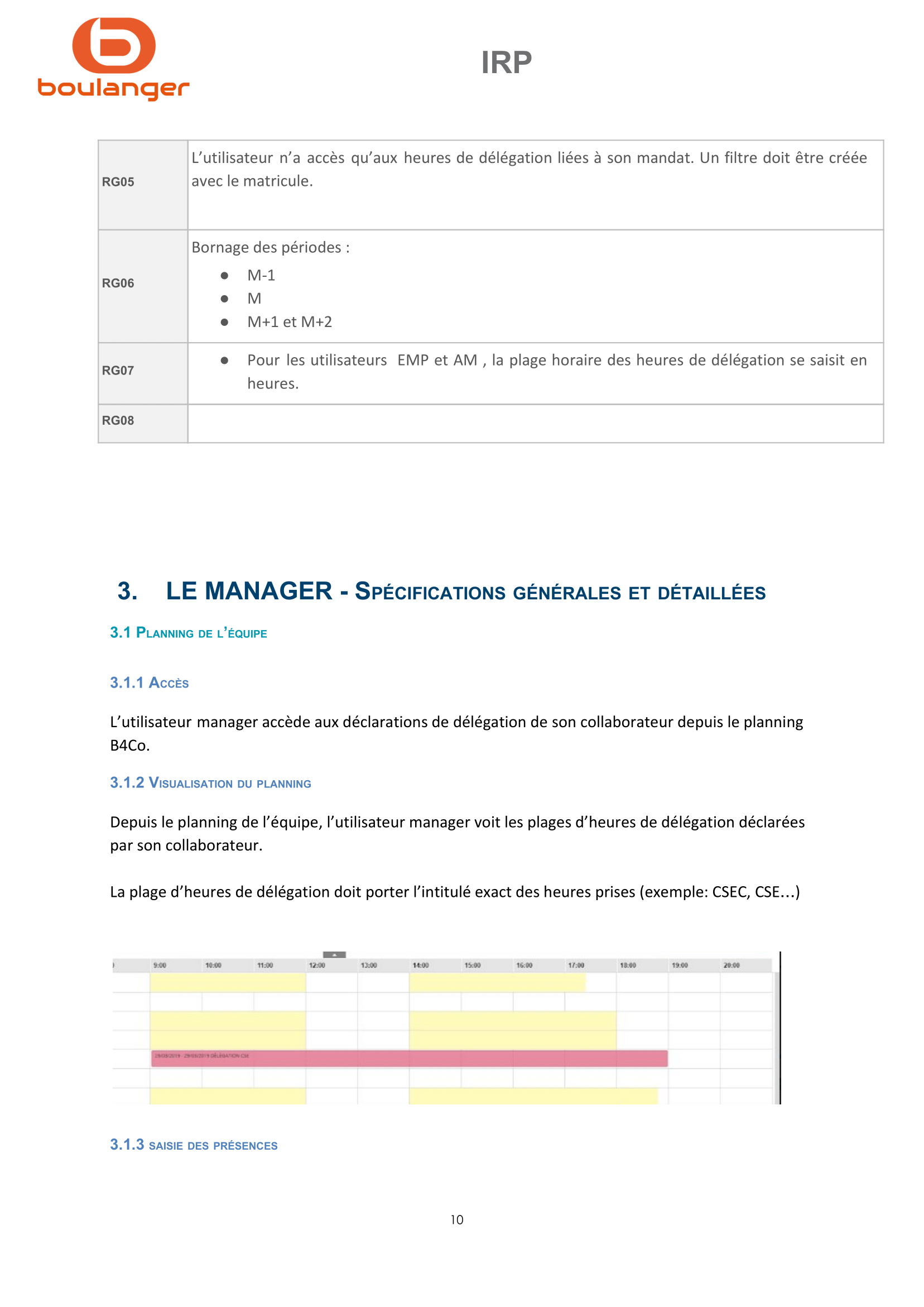
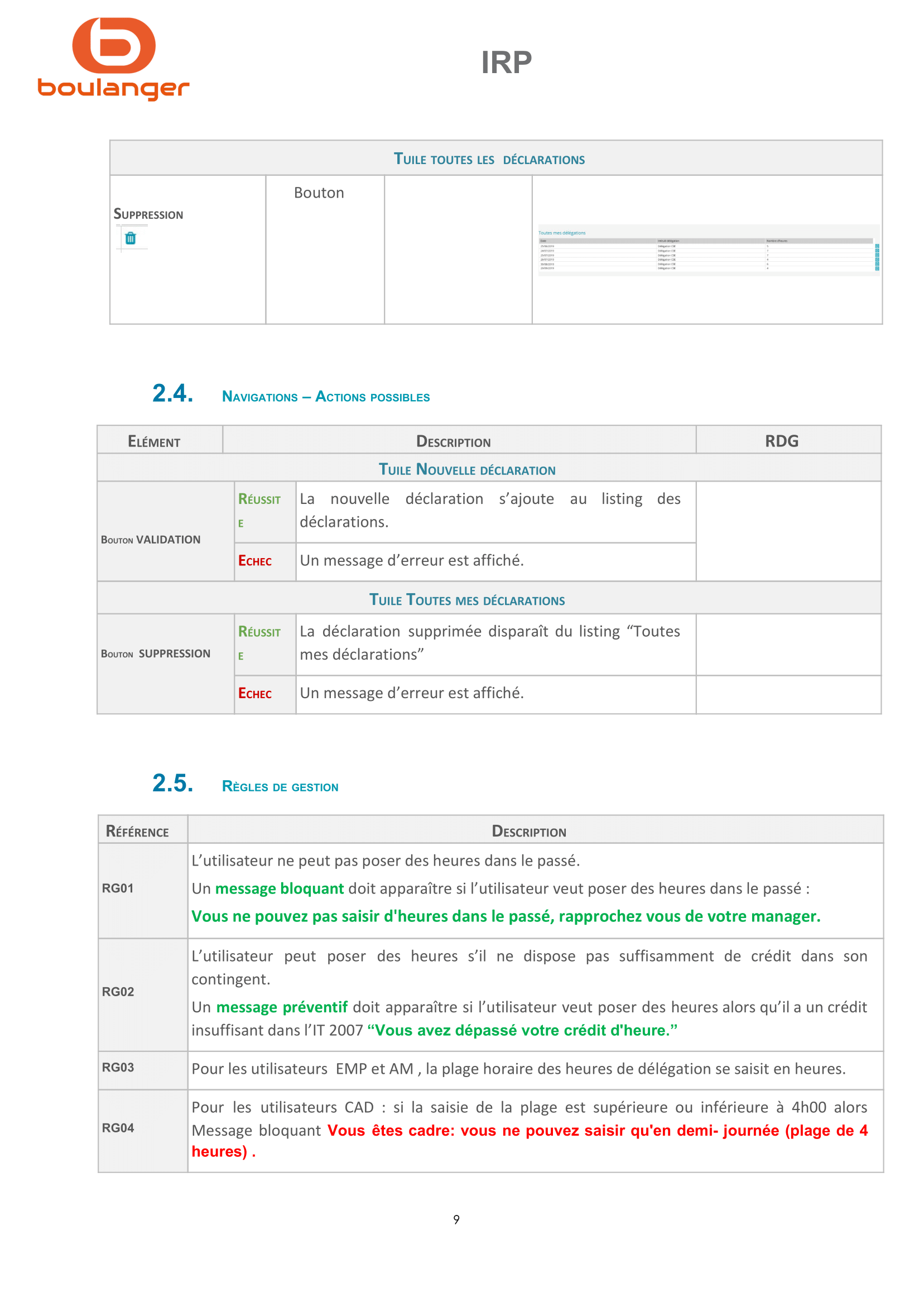
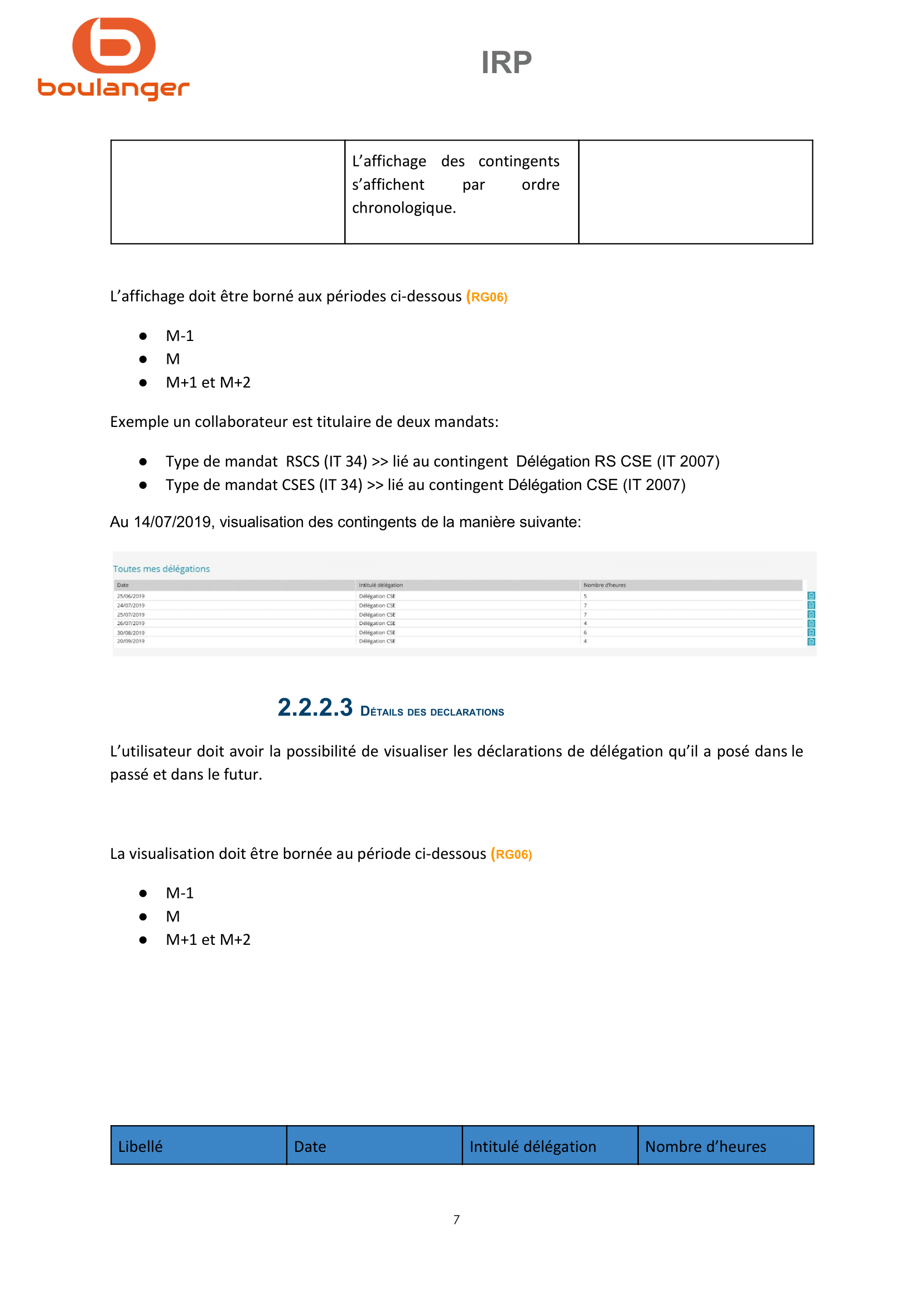
Annexe 2 : Détail d’un ticket sur MANTIS

# 

Annexe 3 : Tableau d’avancement



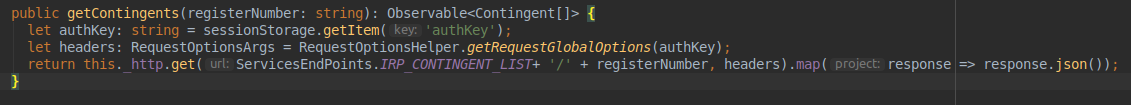
Annexe 4 ; Extrait du cahier des charges****

****

Annexe 5 ; Code du composant mère MyMandatesComponent



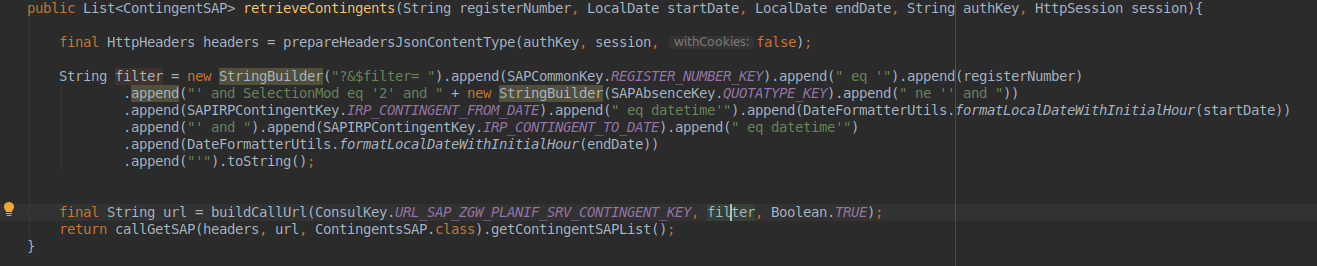
Annexe 6 ; Code de la méthode du service pour récupérer les contingents



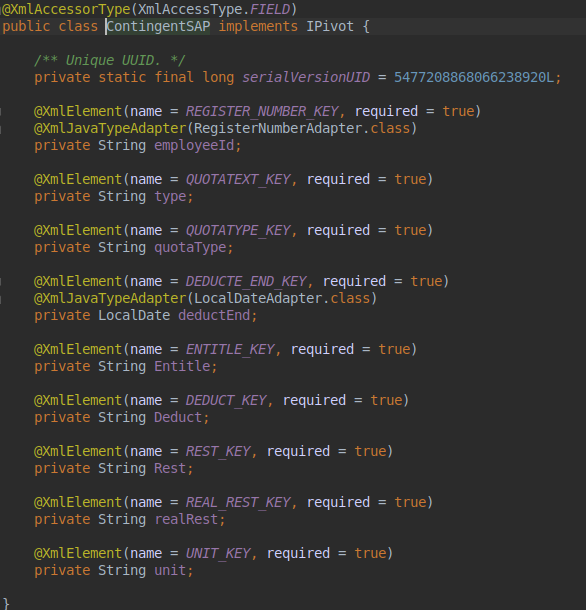
Annexe 7 ; Code du contrôleur back



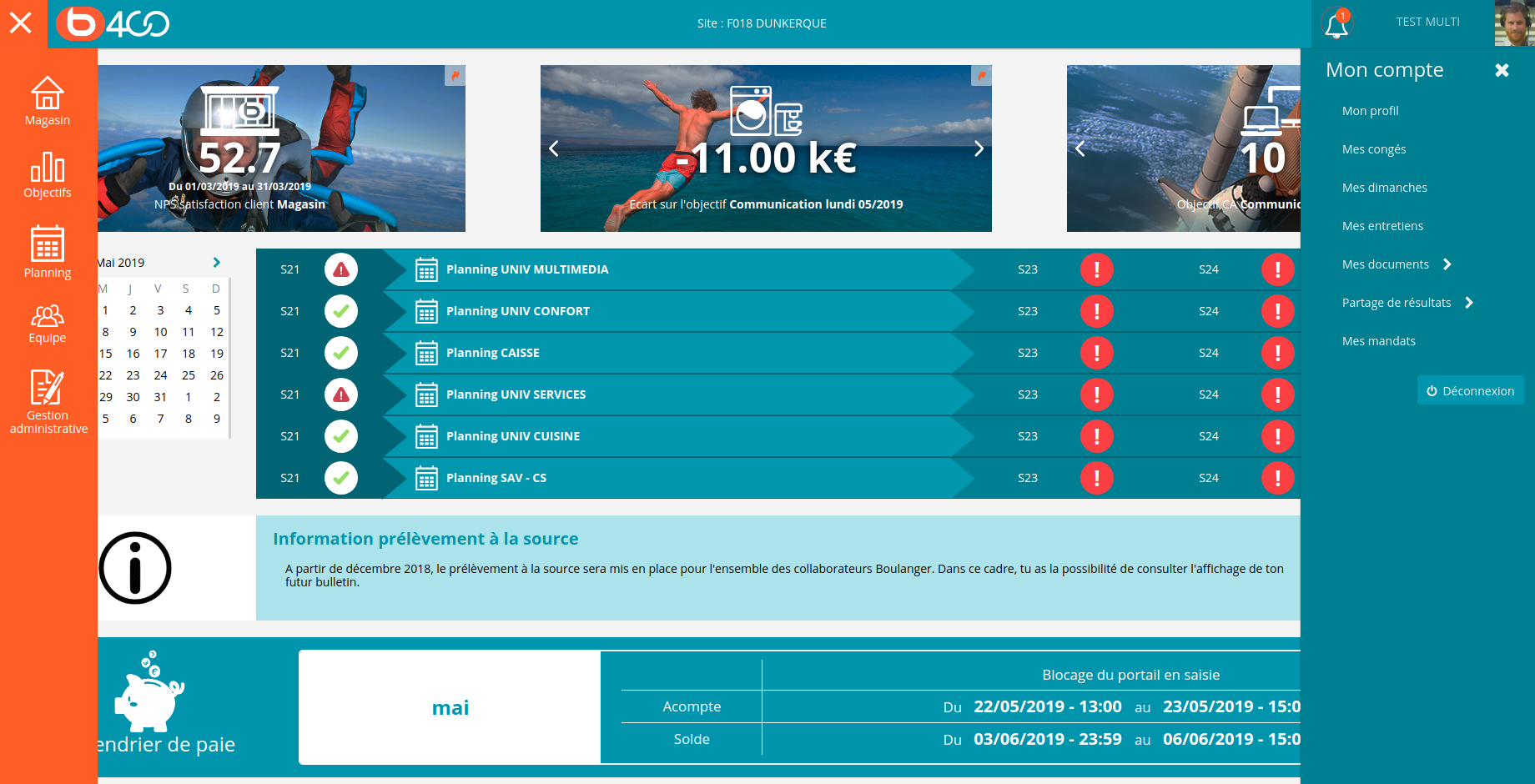
Annexe 8 ; Code du service back



Annexe 9 ; Code de l’entité SAP pour les contingents



Annexe 10 ; Page d’accueil de B4CO



# 

# 

# 

# 

# Glossaire

**Agilité** : Cela veut signifier que je parle des méthodes agiles, qui sont des méthodes de gestion de projets modernes qui ont pour but de réduire l’effet tunnel engendré par les méthodes plus classiques (comme le cycle en V).

**Build** : Il s’agit des évolutions du projet.

**Directive angular :** C’est un moyen simple et rapide d'ajouter un élément à différents endroits de l'application.

**Effet tunnel :** C’est lorsque le client et l’équipe ne se parlent pas, l’équipe travaille seulement en fonction d’une charte de projet.

**Fichier JSON** : Il s’agit d’un fichier qui contient des données sous forme textuelle et représente ces données de manière structurée.

**GET :** C'est la méthode HTTP la plus courante pour demander une ressource. Une requête GET est sans effet sur la ressource, il doit être possible de répéter la requête sans effet.

**HTTP :** Protocole permettant à l'utilisateur d'accéder à des pages web par l'intermédiaire d'un navigateur.

**IDE** : C’est un logiciel qui comporte des outils permettant de développer d’autres logiciels et qui comporte différents outils comme un éditeur de code, un compilateur, un débogueur …

**Jour-homme** : Unité de mesure qui correspond au temps que passe une personne au travail par jour.

**Pair programming :** Méthode de travail qui consiste à travailler en binôme sur le même poste de travail.

**POST :** Cette méthode doit être utilisée lorsqu'une requête HTTP modifie la ressource.

**REST** : Type d’architecture permettant de construire des applications WEB. Il reprend les règles du protocole HTTP. Il suit un ensemble de règles : l’URI comme identifiant des ressources, les verbes HTTP comme identifiant des opérations, les réponses HTTP comme représentation des ressources, les liens comme relation entre ressources et un paramètre comme jeton d’authentification.

**Run** : Il s’agit de la maintenance du projet.

**Sprint agile** : Il s’agit d’une période définie entre le client et l’équipe projet durant laquelle l’équipe va devoir réaliser les différentes tâches établies. Chaque sprint doit apporter des fonctionnalités supplémentaires à l’application en cours de développement qui doivent être livrées lorsqu’il se termine.

**TMA:** Tierce maintenance applicative, il s’agit de la maintenance d’une application par une entreprise externe.

**URI :** Uniform Resource Identifier, c’est une courte chaîne de caractères qui va identifier une ressource sur un réseau.

# 

# 

# 

# Table des figures

|  |  |
| --- | --- |
| **Nom** | **Page** |
| Fig 1: Liste des agences | 6 |
| Fig 2: Logos de quelques clients (anciens ou actuels) de Norsys | 7 |
| Fig 3: Schéma de la performance globale | 8 |
| Fig 4: Schéma de la procédure de la TMA | 10 |
| Fig 5: Ticket sur le tableau d’avancement | 12 |
| Fig 6: Projets sur le GIT | 13 |
| Fig 7: Gestion des branches GIT | 14 |
| Fig 8: Procédure de validation | 15 |
| Fig 9: Maquette de la page employée | 16 |
| Fig 10: Maquette de la popin manager | 17 |
| Fig 11 : Gestion des composants | 18 |
| Fig 12: Echange de données dans l’application | 19 |
| Fig 13: Messages d’erreurs lors de la création | 20 |
| Fig 14: Popin de confirmation | 20 |
| Fig 15: Test du contrôleur de récupération de statut | 21 |
| Fig 16: Siège social de Boulanger à Lesquin | 22 |