



Ministère de l'Enseignement  
Supérieur et de la Recherche  
Scientifique

Année Universitaire

2019 / 2020

# EPI-Polytec

## ÉCOLE POLYTECHNIQUE

ECOLE PRIVÉE INTERNATIONALE  
D'INGÉNIEURS

## RAPPORT DE MÉMOIRE DE FIN D'ETUDES

Pour l'obtention du Diplôme National d'Ingénieur  
en Informatique

Spécialité      Génie logiciel

### Intitulé

Développement d'un outil de gestion de projet basé sur la  
méthodologie Agile et Scrum

Lieu du stage

DOT-IT

Réalisé par

Rebhi Omar Farouk

Encadré par

Mr Chtioui Houssem  
Mr Mounastiri Abederrahim

# Dédicaces

**A feu mon père,**

*Que ce travail soit le meilleur cadeau que je puisse t'offrir. Comme j'ai tant souhaité que tu sois parmi nous pour partager ce moment. Repose en paix*

**A ma mère** Avec tout mon amour

**A mes frères et ma charmante et adorable sœur Zaineb** Avec toute mon affection

*A mes tantes et oncles pour leur soutien et encouragements inconditionnés  
Avec toute mon affection*

*A mes cousins et cousines A tous mes ami(e)s A tous ceux que j'aime je leur dit  
Merci d'avoir partagé le bonheur de ma réussite.*

**A mes encadreurs** pour leur disponibilité et leur patience

**A mes professeurs**, avec mon profond respect et mes remerciements pour la qualité de l'enseignement qu'ils m'ont prodigué A tout le personnel enseignant pour leur savoir et compétence A tout le personnel et cadre administratif Que ce modeste travail soit un hommage

**Omar Farouk**

# Remerciements

C'est avec plaisir que j'apporte ce témoignage écrit de ma profonde reconnaissance à tous ceux qui m'ont gratifiée de leur soutien et de leur confiance et qui ont contribué à l'aboutissement de ce travail.

Je profite de cette occasion pour remercier en premier lieu toute l'équipe pédagogique de l' **Ecole Pluridisciplinaire Internationale (EPI Sousse)**.

Mes remerciements sont adressés tout spécialement à **Mr Housseme Chtioui**, mon encadrant et mon enseignant à l'**EPI** qui a accepté de m'encadrer et qui a suivi mon projet durant ces mois pour sa patience, sa disponibilité et ses conseils toujours avisés.

A mon encadreur au sein de **DOT-IT, Mr abederrahim Mestiri** sans oublier toute l'équipe **DOT-IT**. Je vous remercie pour vos qualités humaines et professionnelles, vos directives, pour vos remarques constructives et votre disponibilité.

# Table des matières

<b>Introduction générale</b>	<b>1</b>
<b>Chapitre 1 Les concepts de base du projet</b>	<b>3</b>
Introduction . . . . .	3
1 Contexte académique . . . . .	3
2 Organisme d'accueil . . . . .	3
2.1 Présentation . . . . .	3
2.2 Fiche technique . . . . .	4
2.3 Services . . . . .	4
3 Problématique traitée . . . . .	5
4 Présentation du projet . . . . .	5
4.1 Introduction du projet . . . . .	5
4.2 Introduction de gestion du projet . . . . .	5
4.3 Introduction aux méthodes Agiles et Scrum . . . . .	6
4.4 Travail demandé . . . . .	7
4.5 Objectif du projet . . . . .	8
5 Spécification des besoins . . . . .	8
5.1 Besoins fonctionnels . . . . .	8
5.2 Besoins non fonctionnels . . . . .	10
6 Méthodologie du travail . . . . .	10
6.1 Scrum . . . . .	10
6.2 Répartitions des rôles dans Scrum : . . . . .	11
7 Langage de modélisation UML . . . . .	12
8 Diagramme de cas d'utilisation . . . . .	12
8.1 Identification des acteurs . . . . .	12
8.2 Diagramme de cas d'utilisation global . . . . .	13
8.3 Diagrammes cas d'utilisation détaillés . . . . .	15
Conclusion . . . . .	18
<b>Chapitre 2 Étude théorique</b>	<b>19</b>
1 Étude de l'existant . . . . .	19
1.1 Trello . . . . .	19
1.2 Asana . . . . .	20
1.3 Nutcache . . . . .	21
2 Solution proposée . . . . .	22
3 Tableau de Synthèse . . . . .	23
Conclusion . . . . .	24
<b>Chapitre 3 Étude conceptuelle</b>	<b>25</b>

1	Conception de l'application . . . . .	25
1.1	Architecture n-tiers . . . . .	25
1.2	Le Modèle MVC . . . . .	27
2	Conception détaillée . . . . .	28
2.1	Diagramme de classes . . . . .	28
2.2	Diagrammes de séquence . . . . .	30
	Conclusion . . . . .	38
	<b>Chapitre 4 Réalisation</b>	<b>39</b>
1	Environnement de travail . . . . .	39
1.1	Environnement matériel . . . . .	39
1.2	Outils de réalistaion . . . . .	39
2	Diagramme de déploiement . . . . .	43
3	Présentation des interfaces de l'application . . . . .	44
3.1	Interface Scrum Master . . . . .	45
3.2	Interface développeur . . . . .	53
3.3	Interface Back office . . . . .	56
	<b>Conclusion générale</b>	<b>58</b>
	<b>Webographie</b>	<b>60</b>

# Table des figures

1.1	Cycle de méthode Sprint . . . . .	11
1.2	Diagramme de cas d'utilisation global . . . . .	14
1.3	Diagramme cas d'utilisation raffiné du cas « gérer les projets » . . . . .	15
1.4	Diagramme cas d'utilisation raffiné du cas « gérer les sprints » . . . . .	16
1.5	Diagramme cas d'utilisation raffiné du cas « gérer les rapports d'un projet » . .	17
1.6	Diagramme cas d'utilisation raffiné du cas « gérer les comptes utilisateur » . . .	18
2.1	Interfaces de l'application Trello . . . . .	20
2.2	Interfaces de l'application Asana . . . . .	21
2.3	Interfaces de l'application Nutcache . . . . .	22
2.4	Tableau de Synthèse . . . . .	24
3.1	Présentation de l'architecture 3-tiers . . . . .	26
3.2	Présentation de Modèle MVC . . . . .	28
3.3	Diagramme de classes . . . . .	29
3.4	Diagramme de séquence « Authentification » . . . . .	31
3.5	Diagramme de séquence « Gérer Sprint » . . . . .	33
3.6	Diagramme de séquence « Sprint Active » . . . . .	35
3.7	Diagramme de séquence « Gérer les rapports » . . . . .	36
3.8	Diagrammes de séquence « Gère groupe de développeur » . . . . .	37
3.9	Diagramme de séquence « Gérer les tâches de sprint active » . . . . .	38
4.1	Diagramme de déploiement . . . . .	44
4.2	Interface login . . . . .	45
4.3	Interface Dashboard Scrum Master . . . . .	46
4.4	Interface Ajout un projet . . . . .	47
4.5	Interface Work Flow . . . . .	48
4.6	Interface Modifier l'emplacement de tâche . . . . .	48
4.7	Interface Détail tâche . . . . .	49
4.8	interface Activer Sprint . . . . .	50
4.9	Interface Activer Sprint . . . . .	50
4.10	Interface Terminer Sprint . . . . .	51
4.11	Interface liste des rapports . . . . .	52
4.12	Exporter rapports . . . . .	52
4.13	Capture de fichier Power Point . . . . .	53
4.14	Interface Dashboard développeur . . . . .	53
4.15	Interface changer l'état d'un tâche . . . . .	54
4.16	Interface date début et date fin . . . . .	55
4.17	Interface Détail tâche . . . . .	55
4.18	login Back office . . . . .	56
4.19	Interface Gérer l'application . . . . .	56

4.20 Interface Gérer l'application . . . . . 57

# Introduction générale

**L**es projets deviennent de plus en plus fréquents dans les compagnies, et les attentes sont de plus en plus grandes en termes de performance (délais, coûts, qualité...).

Généralement le projet a un caractère concret et un but défini, c'est l'ensemble des actions à entreprendre afin de répondre à un besoin bien déterminé dans des délais fixés au préalable. Ainsi, on peut définir un projet comme étant une action temporaire avec un début et une fin, mobilisant des ressources identifiées (humaines et matérielles) durant sa réalisation. Il possède également un coût et fait donc l'objet d'une budgétisation de moyens et d'un bilan indépendant de celui de l'entreprise et on appelle « livrables » les résultats attendus du projet.

Il est judicieux de mettre en place un processus de Gestion de Projet qui nous aide à atteindre nos objectifs. En général, les projets ont toujours des délais et des budgets serrés. Mais on peut toujours rencontrer des difficultés à maintenir la motivation de l'équipe, l'instabilité du budget alloué surtout qu'il y a un risque de changement en cours de conception. C'est pour ces raisons qu'il existe plusieurs méthodologies de gestion de projet « **Agiles** ». Ce terme " définit une approche de gestion de projet qui prend le contre-pied des approches traditionnelles prédictives et séquentielles de type cycle en V ou en cascade. La notion même de "gestion de projet" est remise en question au profit de "gestion de produit". De façon à raisonner davantage "produit" que "projet". Après tout l'objectif d'un projet consiste bien à donner naissance à un produit.

C'est dans ce cadre que s'inscrit mon projet de fin d'étude (PFE) au sein de La société de services informatiques "DOT-IT" qui a pensé à développer un outil de gestion de projet selon la méthode **Agile et Scrum** sur le nom "DO-IT" qui comprend quatre chapitres :

- Le premier chapitre présentation générale de la société d'accueil, le contexte général du projet, la problématique traitée, spécification des besoins, et la méthodologie adoptée.
- Le deuxième chapitre étude de l'existant.

## Introduction générale

- Le troisième chapitre l’architecture générale de la solution proposée et la conception par diagrammes.
- Enfin les choix technologiques, l’architecture logicielle adoptée et la description des captures d’écrans essentielles de la plateforme développée.

# Les concepts de base du projet

## Introduction

Ce chapitre présente d'une manière générale l'environnement du projet à réaliser. On commence par situer le projet dans son cadre puis présenter l'organisme d'accueil, pour conclure par donner une idée approximative du travail demandé .

### 1 Contexte académique

Mon stage de PFE a comme but de m'intégrer dans la vie professionnelle et m'y initier. La société m'a proposé de réaliser une application pour mettre en pratique mes connaissances théoriques acquises durant ma formation de diplôme national d'ingénieur en Génie Logiciel de l'École Pluridisciplinaire Internationale (EPI).

On a essayé de mettre en place une application web de WORKFLOW basée sur le Framework Symfony parti backend et le Framework Angular parti frontend.

Cette application va se réaliser au sein de la société «DOT-IT». Mon stage s'est déroulé sur une période de 6 mois consacrée principalement à l'étude conceptuelle de l'application, à l'apprentissage des outils et au développement.

### 2 Organisme d'accueil

#### 2.1 Présentation

"DOT IT" est une entreprise tunisienne spécialisée dans l'ingénierie logicielle et l'intégration de solutions de gestion pour les entreprises. Elle crée la mise en œuvre de méthodologies de

gestion de projet et d'expertise à partir des normes de l'industrie logicielle (ISO, CMMI, UP, RUP, XP, Méthodologies Agiles... ).

Annoncé en tant que membre du programme Microsoft Certified Partner, DOT IT travaille en partenariat avec BPA SOLUTIONS - société suisse développant des applications commerciales. Elle a contribué à la réalisation de projets majeurs et stratégiques pour de nombreux clients tels que Hacks House, Royal Kenz Hotel, Itac Tunisie, Forest Tunisia, Comet, Carthage Land, Ooredoo ...

## 2.2 Fiche technique

### — DOT-IT

Adresse : Immeuble Wejden, 4ème Etage B402-403 - Avenue perle du sahel – Khzema 4051 - Sousse

email : [contact@dotit.com.tn](mailto:contact@dotit.com.tn)

Téléphone : (+216) 70 834 163

Fax : (+216) 70 834 074

site web : <http://www.dotit-corp.com>

## 2.3 Services

DOT-IT a cinq grands domaines d'activité :

- Génie logiciel
- Solution d'affaires (CRM,Gestion de la qualité , Solutions de E-commerce)
- Sources informatique externes
- Solutions Web
- Solutions mobiles



### 3 Problématique traitée

Généralement les projets, en raison de ses caractéristiques d'unicité, d'incertitudes, ..., ne se déroule jamais de façon idéale. Le chef de projet se heurte donc en permanence à des difficultés qu'il est utile de recenser. Les difficultés dans la conduite du projet qui réside dans :

- La fin du projet ne se réalise jamais à la date prévue auparavant.
- Les spécificités du projet sont fréquemment modifiées, ce qui entraîne des corrections du projet initial et modifie le calendrier et les coûts.
- Un grand nombre de tâches doivent être reprises.
- dépassement du budget alloué initialement .

C'est pour ces raisons qu'on a opté pour SCRUM qui est une méthode pour les processus de gestion de projet . Elle ne peut être efficace que par le recours à framework scrum qui malheureusement présente quelques restrictions à savoir :

- La non gratuité.
- Une formation est vivement souhaitable.
- la limite des rapports scrum.
- respecte pas les normes agile scrum.

## 4 Présentation du projet

### 4.1 Introduction du projet

On appelle projet l'ensemble des actions à entreprendre afin de répondre à un besoin bien déterminer dans des délais fixés. Ainsi un projet étant une action temporaire avec un début et une fin, mobilisant des ressources identifiées (humaines et matérielles) durant sa réalisation, celui-ci possède également un coût et fait donc l'objet d'une budgétisation de moyens et d'un bilan indépendant de celui de l'entreprise. On appelle «livrables» les résultats attendus du projet.

### 4.2 Introduction de gestion du projet

La gestion de projet est donc une démarche visant à structurer, assurer et optimiser le bon déroulement d'un projet suffisamment complexe pour devoir :

- Être planifié dans le temps avec une date de début et de fin fixés.

- Faire intervenir de nombreuses parties prenantes : c'est l'objet des organisations qui identifient maîtrise d'œuvre et maîtrise d'ouvrage.
- Responsabiliser le chef de projet ou le directeur de projet, mettre en place un comité de pilotage ou de projet.
- Suivre des enjeux opérationnels et financiers importants. L'objectif est d'obtenir un résultat conforme à des normes de qualité et de performances prédéfinies, pour le moindre coût et dans le meilleur délai possible.

#### 4.3 Introduction aux méthodes Agiles et Scrum

La méthode Agile se base sur un cycle de développement qui porte le client au centre. Le client est impliqué dans la réalisation du début à la fin du projet. Grâce à la méthode **agile** le demandeur obtient une meilleure visibilité de la gestion des travaux qu'avec une méthode classique.

Il s'agit de décider quelle tâche est la plus importante à un instant « t » afin de la réaliser en priorité en divisant le projet en des multiples mini-projets selon les besoins.

La méthode SCRUM consiste à définir un cadre de travail permettant la réalisation de projets complexes. Cette méthode a été initialement prévue pour le développement de projets informatiques mais elle peut être appliquée à tout type de projet, du plus simple au plus innovant, et ce de manière très simple. Il existe trois rôles principaux à « pourvoir » : le responsable produit(Product Owner), le Scrum Master, et les membres de l'équipe.

- **Le responsable produit(Product Owner)** : Ce dernier définit les spécifications fonctionnelles et communique la vision globale du produit à l'équipe. Il établit la priorité des fonctionnalités à développer ou à corriger et valide les fonctionnalités développées. Il se doit de jouer le rôle client final, se mettre à sa place et donc de prioriser ses besoins.
- **Le Scrum Master** : Ce dernier agit en tant que facilitateur entre le responsable produit et l'équipe. Son rôle principal est d'éliminer tous les obstacles qui peuvent empêcher l'équipe d'atteindre les objectifs fixés pour chaque sprint de travail. Il s'assure que les principes et les valeurs Scrum sont respectés. Il facilite la communication au sein de l'équipe et cherche à améliorer la productivité et le savoir-faire de son équipe.
- **Les Membres de l'équipe** : Dans la méthode SCRUM, l'équipe est responsable de la réalisation opérationnelle des tâches. L'équipe est d'ailleurs généralement composée de

6 à 10 personnes mais pouvant aller jusqu'à 200 personnes. C'est toute l'équipe qui est responsable du résultat final de chaque sprint. La manière dont sont exécutées les tâches est très libre mais cette liberté doit être néanmoins cadrée par l'obligation de répondre aux objectifs du sprint.

#### **4.4 Travail demandé**

Afin de réaliser un produit complet, robuste et satisfaisant aux besoins de notre boîte DOT-IT, nous étions amenés à procéder les étapes suivantes :

- Spécification :
  - Analyser l'existant et dégager les inconvénients.
  - Création et mise à jour cyclique du document de spécification.
  - Création du cahier des charges.
- Conception : création des diagrammes UML nécessaires pour décrire l'application tels que :
  - Diagramme de classes décrivant la base des données.
  - Diagramme de cas d'utilisation.
  - Diagramme de séquences.
  - Diagramme de déploiement.
- Détermination et mise en place de l'environnement de développement :
  - Choix des technologies et des bibliothèques à utiliser pendant la développement.
  - Configuration de l'environnement de développement.
- Réalisation :
  - Développement des entités nécessaires et création de la base des données.
  - Création des interfaces et développement de code métier pour les différents modules.
    - Module de projet.
    - Module de sprint.
    - Module de backlog.
    - Module de tâche .

- Test et validation :
  - exécution des tests nécessaires pour la validation de l'application
  - Test unitaire.
  - Test d'intégration.
  - Test de fonctionnement.

#### 4.5 Objectif du projet

L'objectif de ce projet est de créer une application de scrum sur le nom DO-IT qui s'adapte au système de gestion de projet et respecte au maximum les normes de méthodes agiles. Cette application va permettre de créer d'un projet et le composer sous forme des user-story (Tâches), création de backlog et des sprints puis affecter des user-story pour ces derniers et présenter le sprint sous la forme d'un tableau scrum.

Cette application doit avoir des interfaces simples ergonomiquement et compréhensibles. Elle doit aussi permettre aux utilisateurs de saisir le minimum d'information possible et éviter au maximum les erreurs de saisie.

Notre application sera une application web, donc elle doit être simple à l'utilisation et facilite la navigation avec un temps de réponse optimisé.

### 5 Spécification des besoins

Notre application doit satisfaire les exigences de la totalité des utilisateurs. Nous exposons dans ce qui suit leurs besoins fonctionnels ainsi que les besoins non fonctionnels communs à tous les acteurs.

#### 5.1 Besoins fonctionnels

Ce sont les fonctionnalités et besoins indispensables auxquels doit répondre l'application, ils doivent être claire et nets le plus possible par rapport aux utilisateurs. En effet, cet outil permettra de réaliser les opérations suivantes :

##### 5.1.1 Du côté Scrum Master

- Consulter dashboard.
- Gestion de projet.

- Gestion des sprints.
- Valider sprint pour chaque sprint.
- Gestion de backlog.
- Gestion des tâches.
- Commenter les tâches.
- Ajouter fichier au projet ou tâches.
- Gestion des rapports de projets.
- Consulter les historiques de chaque tâches.
- Suivi de temps pour chaque tâche en train d'être développer.
- Gestion des groupes de développement.
- Gestion des comptes utilisateur.
- Effectuer un groupe pour un projet.

### **5. 1. 2 Du coté Développeur**

- Consulter la liste des projets qui lui correspond.
- Consulter la liste des tâches.
- Modifier le temps de développement de tâche qui lui correspond.
- Déplacement facile de ses tâches entre les 3 états ‘To do’, ‘Doing’ et ‘Done’ par glisser / déposer.
- Commenter les tâches.
- Consulter les historiques de chaque tâches.

### **5. 1. 3 Du coté Product owner**

- Consulter la liste des projets que lui correspond.
- Suivre l'avancement (changement d'états, Sprint, User Stories).
- Validation ou retour de projet.

## 5.2 Besoins non fonctionnels

- **Convivialité** : L'application doit être facile à utiliser. Les interfaces utilisateur doivent être conviviales, c'est-à-dire simple, ergonomique et adaptée à l'utilisateur.
- **Rapidité** : le système est une application web, donc il doit être léger à l'utilisation. Des optimisations doivent être réalisées sur le temps de calcul et sur l'accès à la base des données.
- **Intelligence** : l'application montre des signes d'intelligence dans la vérification des informations saisies. Elle évite les erreurs ou l'entrée des données qui ne sont pas conformes à la réalité ou à la base des données existante. Elle communique ces erreurs à l'aide des messages d'erreurs.
- **Maintenabilité** : le code doit être facile à maintenir pour des raisons de réutilisation et de maintenance.
- **Les droits d'utilisateur** : l'application sera utilisée par des différentes personnes et pour des besoins différents, on doit donc définir pour chaque type d'utilisateur les droits qui lui convient.
- **Efficacité** : Le système doit respecter les normes définies en termes de temps de réponse et en termes d'utilisation des ressources.
- **Sécurité** : L'application sera hébergée en web , une sécurisation par login et mot de passe ne sera pas suffisante, donc toute information confidentielle fournie par les clients via l'Internet sera cryptée.

## 6 Méthodologie du travail

Pour obtenir un résultat fiable il faut bien choisir un procès de suivi et une méthodologie de travail, avant la réalisation de n'importe quel projet. Il existe plusieurs méthodes dans ce qui suit nous intéressons aux méthodes agiles vues leurs apports pour des projets où les besoins évoluent. On a choisi Scrum pour son efficacité

### 6.1 Scrum

Scrum est de très loin la méthodologie la plus utilisée parmi les méthodes agiles existantes. C'est dans ce cadre que d'inscrit mon projet de stage de PFE qui porte sur le développement d'un outil de gestion de projet selon la méthode agile et Scrum.

Le produit final est développé par itération de temps (appelé des sprints). La méthode amène le client constamment au centre du produit en tentant de le faire intervenir régulièrement dans le courant du développement. On parle dans ce cas de Scrum comme méthode pour notre processus de gestion de projet.



FIGURE 1.1 – Cycle de méthode Sprint

Les sprints peuvent durer entre quelques heures et un mois (avec une préférence pour deux semaines). Chaque sprint commence par une estimation suivie d'une planification opérationnelle. Le sprint se termine par une démonstration de ce qui a été achevé. Avant de démarrer un nouveau sprint, l'équipe réalise une rétrospective. Cette technique analyse le déroulement du sprint achevé, afin d'améliorer ses pratiques.

### 6.2 Répartitions des rôles dans Scrum :

Les membres d'une équipe Scrum se répartissent trois rôles Scrum différents, sans aucun rapport hiérarchique entre eux. Le Product Owner va définir les besoins, modeler le produit et définir les priorités. L'équipe de développement va réaliser, au cours d'itérations ou sprint, les user stories définies comme prioritaires par le product owner. Enfin, le Scrum Master va s'assurer d'une part du respect de la méthodologie agile et faciliter le travail de l'équipe de développement et la communication avec le product owner. Son rôle principal est d'éliminer tous les obstacles qui peuvent empêcher l'équipe d'atteindre les objectifs fixés pour chaque sprint de travail.

les 3 rôles de scrum dans notre projet :

- **Product Owner :** Charfi houssem
- **Scrum Master :** Mastouri Abed Rahim
- **Les Membres de l'équipe :** Rebhi Omar Farouk, Zitir Malek , Mohamed ali Mosbeh

## 7 Langage de modélisation UML

La notation UML est un langage visuel constitué d'un ensemble de schémas, appelés des diagrammes, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par le logiciel, etc.

Donc, après le choix de la méthodologie, on a opté UML comme un langage de modélisation qui est utilisé dans tous les projets logiciels. Il comporte un ensemble des diagrammes, il permet de fournir une représentation informatique d'un ensemble d'objets et de problèmes standards du monde réel.[1]

## 8 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation sont des diagrammes UML utilisés pour donner une vision globale du comportement fonctionnel d'un système logiciel. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d'un projet, mais pour le développement, les cas d'utilisation sont plus appropriés. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur et un système. Il est une unité significative de travail. Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs , ils interagissent avec les cas d'utilisation.[1]

### 8.1 Identification des acteurs

Dans notre projet nous avons identifié un ensemble de tâches qui sont associées à trois acteurs principaux.

- **Scrum Master :** son rôle réside dans la gestion des projets, des tâches ,backlog et des utilisateurs.
- **Développeur :** peut consulter ses projets et ses tâches, et changer leurs états ainsi que poste des commentaires.
- **Product Owner :** peut consulter ses projets , et suivre l'avancement de projet et valider le projet.
- **Super Admin :** Gérer les projet ,les tâches , les types de tâches ,les Sprint et les compte d'utilisateurs.

## 8.2 Diagramme de cas d'utilisation global

Dans ce qui suit, nous allons présenter le diagramme de cas d'utilisation global pour notre application.

Le diagramme de cas d'utilisation global est le diagramme général du système qui modélise les fonctionnalités principales du système.

## Les concepts de base du projet

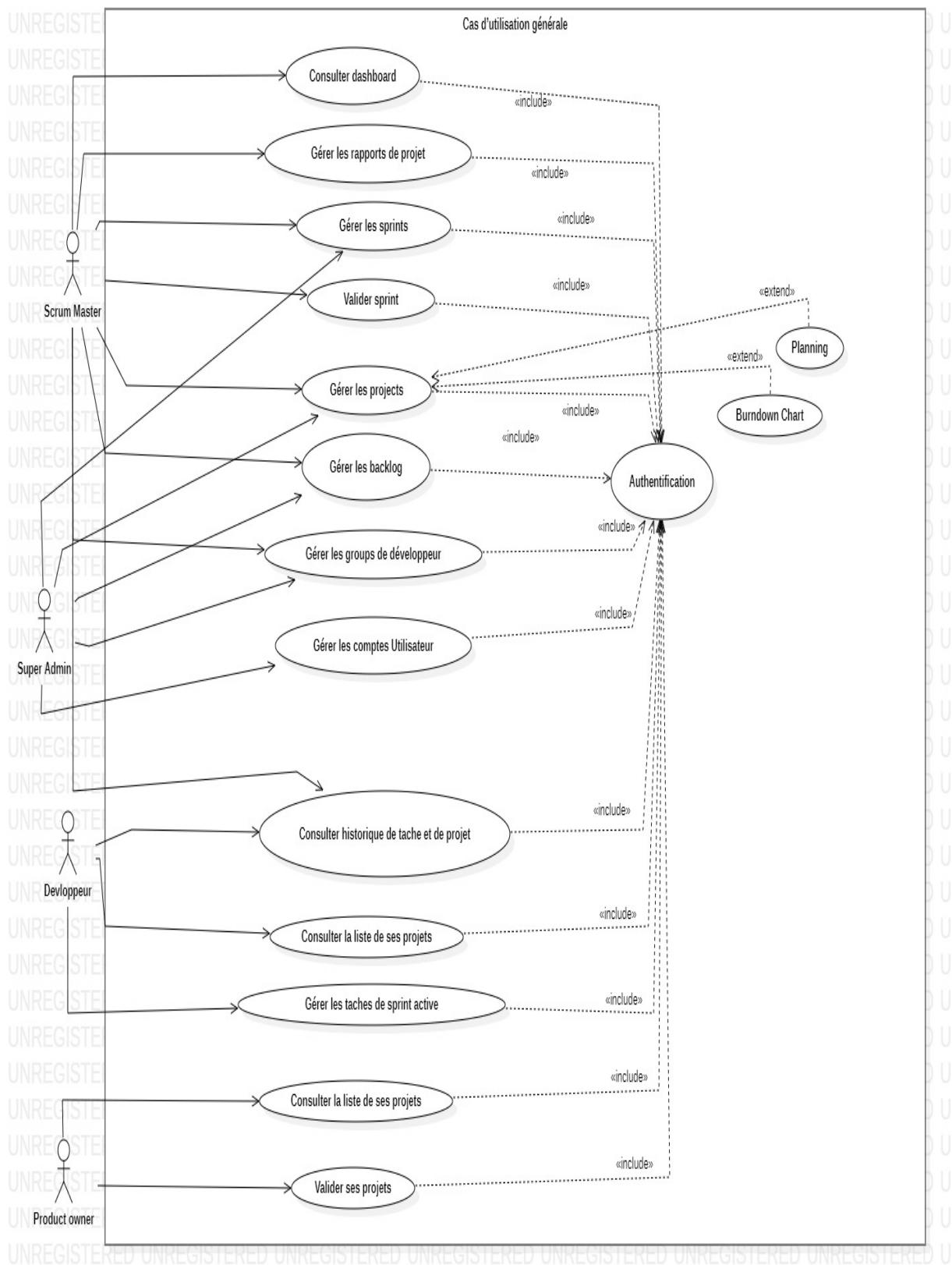


FIGURE 1.2 – Diagramme de cas d'utilisation global

### 8.3 Diagrammes cas d'utilisation détaillés

#### 8.3.1 Gérer les projets

Nous illustrerons à partir de la figure 1.3 le diagramme cas d'utilisation qui explique en détails la façon avec laquelle la fonctionnalité de gérer les projets.

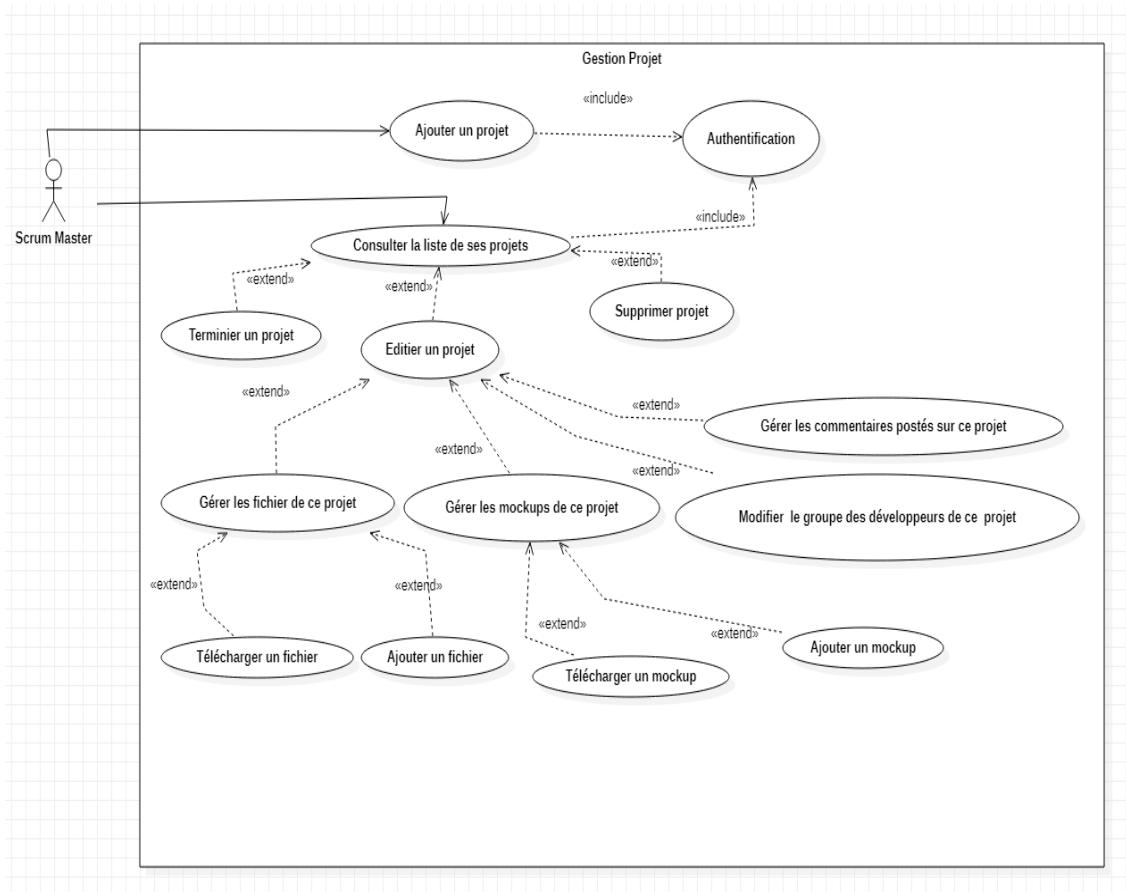


FIGURE 1.3 – Diagramme cas d'utilisation raffiné du cas « gérer les projets »

— **Acteur : Scrum Master**

— **Description :** DO-IT permet au Scrum Master de gérer tous ses projets. Donc ,il permet d'ajouter et supprimer un projet sélectionné ainsi que d'éditer un projet. Dans ce cas le Scrum Master peut gérer les fichiers les tâches les mockups, les commentaires ainsi qu'il peut modifier le groupe de développeurs.

### 8.3.2 Gérer les sprints

La figure suivante nous présente le diagramme cas d'utilisation de gérer les sprints.

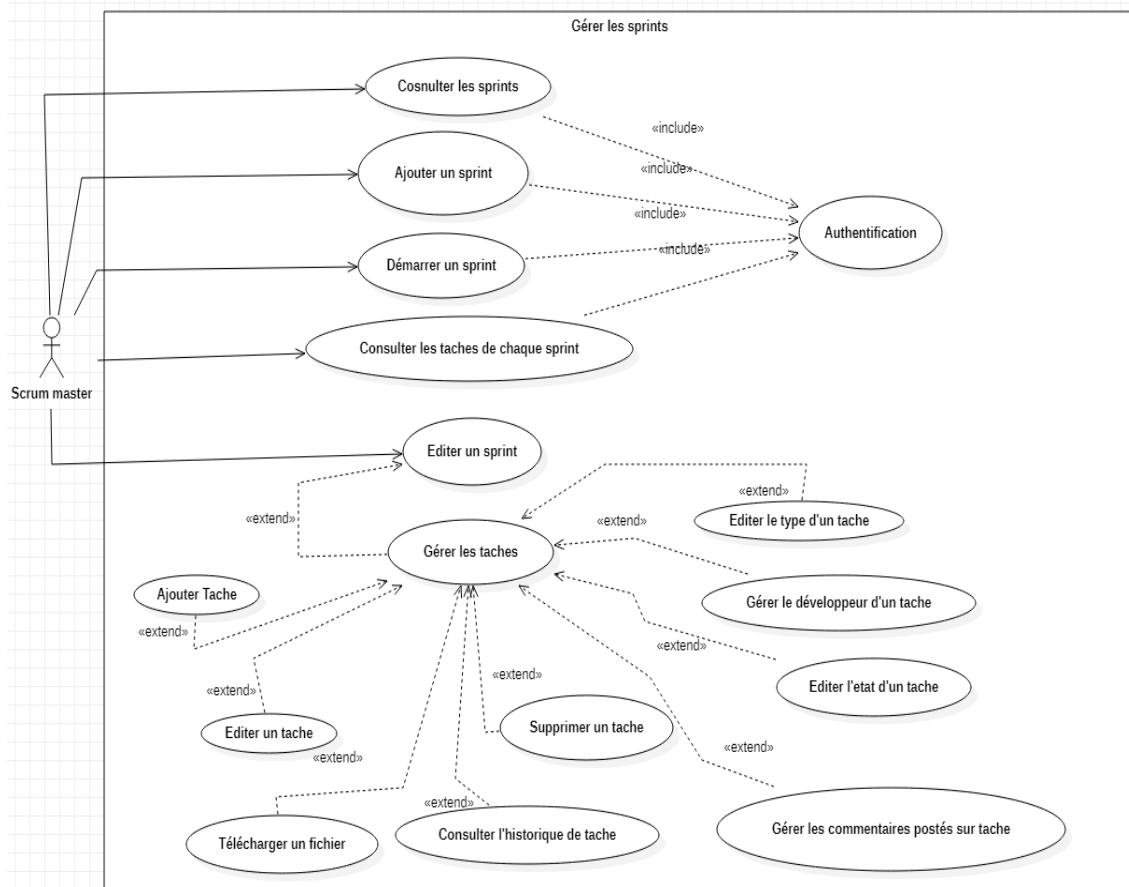


FIGURE 1.4 – Diagramme cas d'utilisation raffiné du cas « gérer les sprints »

— **Acteur :**Scrum Master

— **Description :** Scrum Master peut gérer les sprints ajouter, supprimer ou démarrer le sprint même en éditer. il a aussi la possibilité de gérer les tâches (ajouter, supprimer et même modifier ) et affecter un développeur dans le groupe en le modifiant voire même ajouter des fichiers ou mockaup aux tâches .

### 8.3.3 Gérer les rapports d'un projet

La figure suivante nous présente le diagramme cas d'utilisation de gérer les rapports d'un projet.

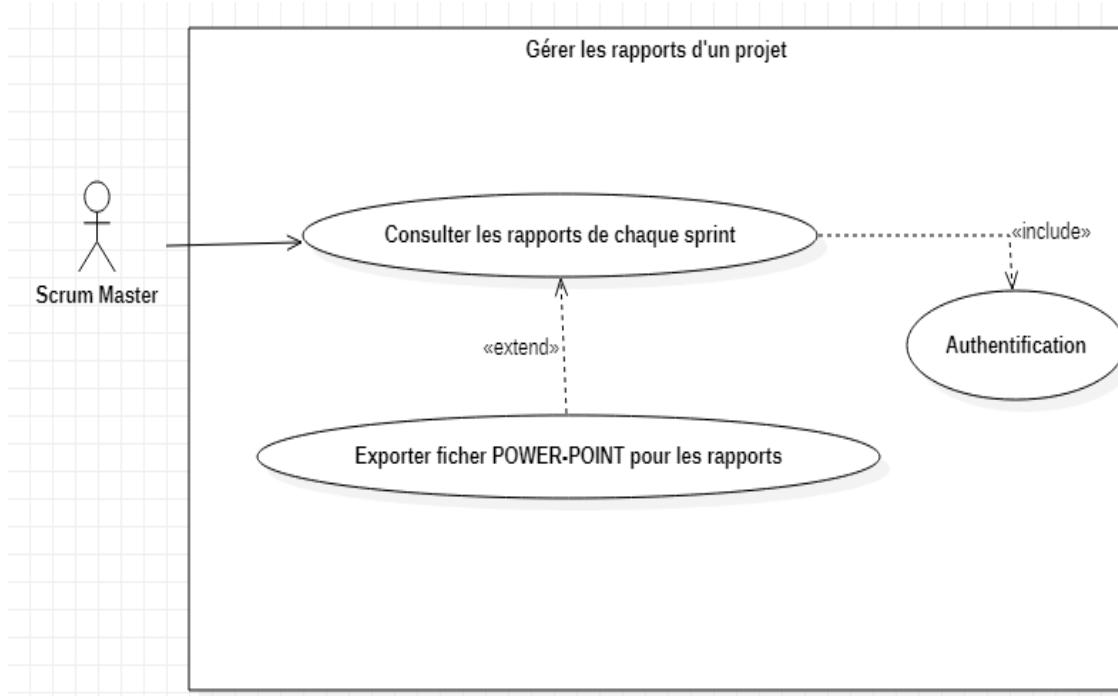


FIGURE 1.5 – Diagramme cas d'utilisation raffiné du cas « gérer les rapports d'un projet »

- **Acteur :**Scrum Master
- **Description :** Le Scrum Master peut consulter les rapports de chaque sprint et un fichier powerpoint qui contient ces rapports.

### 8.3.4 Gérer les comptes Utilisateur

La figure suivante nous présente le diagramme cas d'utilisation de Gérer les rapports d'un projet.

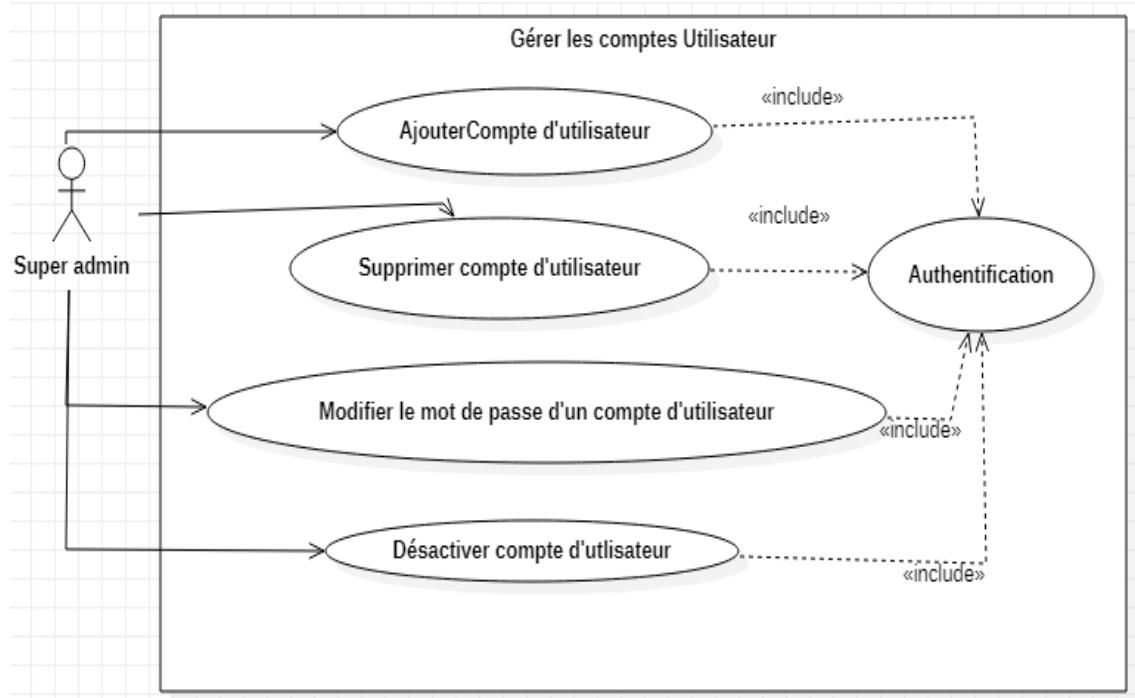


FIGURE 1.6 – Diagramme cas d'utilisation raffiné du cas « gérer les comptes utilisateur »

- **Acteur :** Super Admin
- **Description :** DOT-IT permet au Super Admin de gérer tous les comptes de l'utilisateur. Ce qui permet d'ajouter, désactiver et supprimer un compte. Il peut modifier le mot de passe du compte d'utilisateur.

## Conclusion

Dans ce chapitre, nous avons présenté le contexte du stage et la problématique traitée en spécifiant les besoins fonctionnels et non fonctionnels. On a présenté le diagramme de cas d'utilisation en indiquant la méthodologie du travail adopté d'une façon générale.

Dans le chapitre suivant, nous présenterons l'analyse de l'existant avec un tableau de synthèse et la solution proposée.

# Chapitre 2

## Étude théorique

### Introduction

Dans ce chapitre nous allons mettre le projet dans son cadre général, en détaillant l’analyse de l’existant avec un tableau de synthèse des approches, des solutions et des technologies existantes, et nous concluons par des propositions notre solution.

### 1 Étude de l’existant

Cette étape est l’une des principales pour la réalisation d’un projet. Elle nous permet d’identifier les points faibles des solutions actuelles afin de pouvoir y pallier et concevoir une application qui réponde aux besoins de ses utilisateurs en offrant des fonctionnalités plus riches et plus développées qui permettent à l’utilisateur une utilisation simple et fluide.

Parmi les applications qui offrent des fonctions similaires à celles que nous souhaitons implémenter dans notre application Asana, Trello et Nutcache.

Asana, Trello et Nutcache sont des outils de gestion de projet très populaires dans les entreprises travaillant de manière collaborative et Agile quotidiennement ou pour des projets ponctuels. Ces trois logiciels en ligne sont très souvent comparés, car leur approche de la gestion de projet est visuelle, ludique et collaborative.

#### 1.1 Trello

Trello est un outil universel d’organisation pour les projets personnels et professionnels. Trello offre une approche très spécifique de la gestion de projet en remplaçant les tâches par des

"cartes" et les projets par des "tableaux". Cette approche basée sur la méthode Kanban offre un environnement très visuel pour les projets à la manière de Post-it sur un tableau blanc. Cet outil de gestion de projet est pertinent aussi bien pour planifier et organiser la création d'un site web que pour organiser son emménagement dans un nouvel appartement.[5]

Trello permettre de :

- Gérer les tâches.
- Créer les tableaux.
- Ajouter des commentaires.
- Ajouter des fichiers aux tâches.
- Drag and Drop facile pour les tâches.

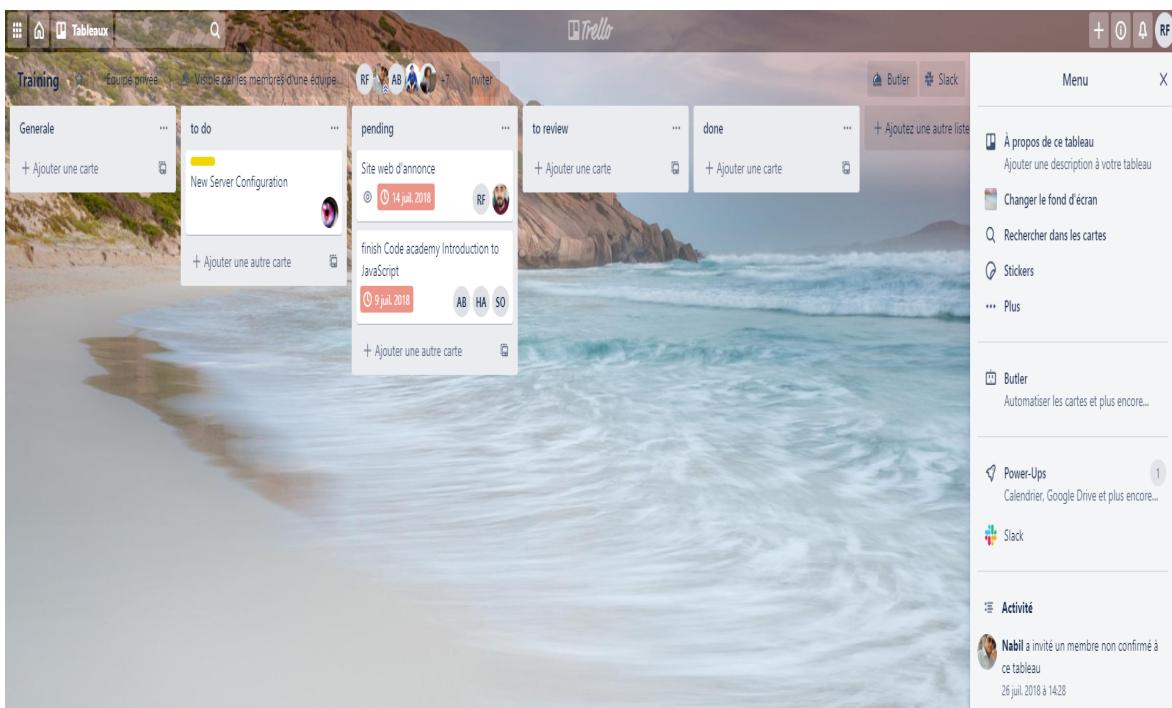


FIGURE 2.1 – Interfaces de l'application Trello

### 1.2 Asana

Asana est un logiciel de gestion de tâches et de gestion de projet conçu pour la collaboration au quotidien et pour la planification de projets ponctuels. Asana reprend les fondamentaux des outils de gestion de projet afin d'offrir un cadre structurant aux équipes tout en offrant une expérience souple et collaborative.[5]

Asana permettre de :

- Progression et suivi de projets.
- lister les tâches.
- Décomposez le travail en tâches faciles à gérer par les membres de l'équipe.
- Divisez les tâches en plusieurs éléments plus petits ou indiquez les étapes nécessaires pour les terminer.

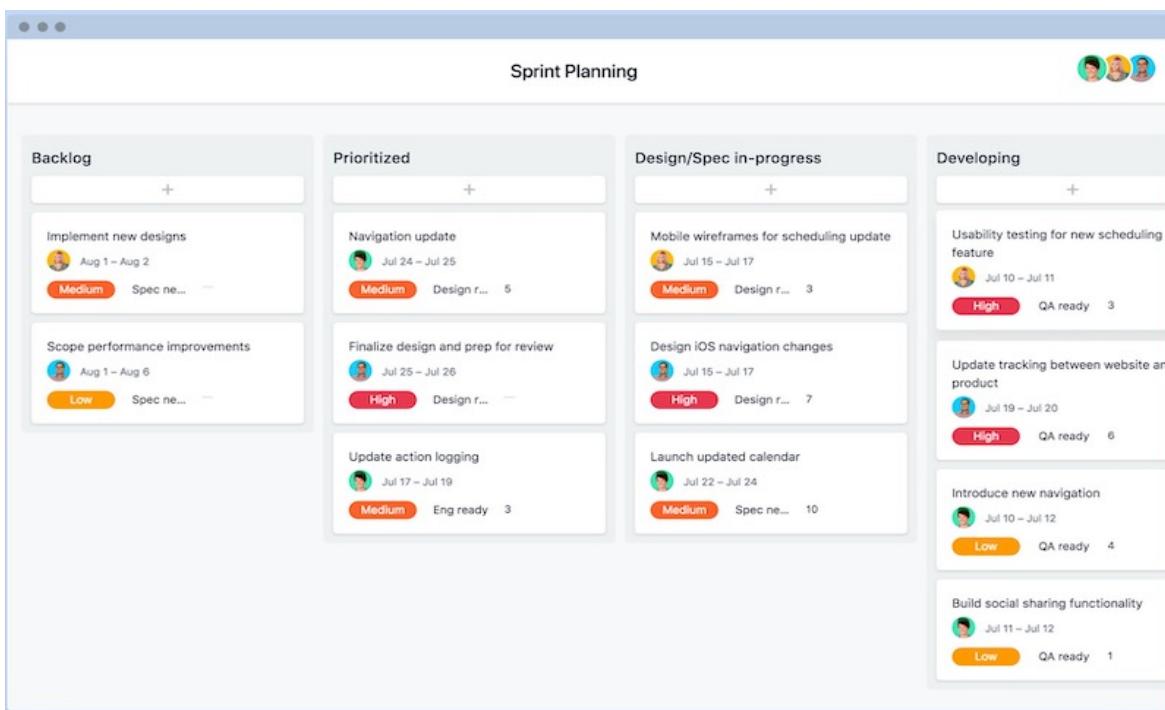


FIGURE 2.2 – Interfaces de l'application Asana

### 1.3 Nutcache

Nutcache est une application en ligne développée par Dynacom Technologies, une entreprise canadienne qui cumule 25 ans d'expérience éprouvée en matière de développement d'applications de facturation, de comptabilité et de gestion. L'objectif principal de Nutcache est de faciliter les opérations de gestion pour les travailleurs autonomes et petites entreprises, avec des outils simples et conviviaux.[9]

Nutcache permettre de :

- Affectez plusieurs personnes à une même tâche.
- Construisez des processus de travail personnalisés.

- Budgéter les projets.
- Organiser les tâches.

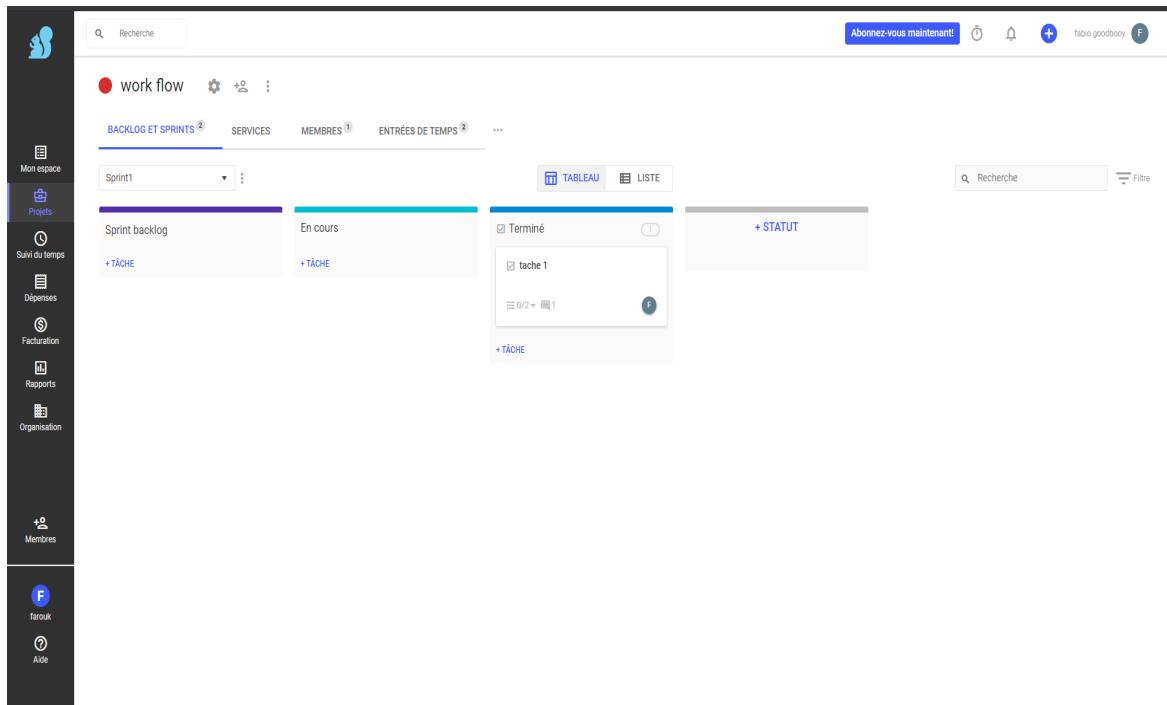


FIGURE 2.3 – Interfaces de l’application Nutcache

## 2 Solution proposée

Le but de ce stage est d’arriver à concevoir et à développer une application web qui doit répondre à un besoin existant, en gestion de projets informatique pour conduire un projet en appliquant les valeurs communes et les principes fondamentaux des méthodes agiles et Scrum.

La solution proposée devra répondre à plusieurs critères dont les principaux sont :

- Gestion de tâches pour chaque utilisateur.
- Gestion de temps pour chaque tâche.
- Comparaison du temps passé pour chaque tâche par rapport au temps estimé.
- Gestion des Sprint et des User Stories pour chaque utilisateur **Scrum Master, Product Owner, Équipe**
- Calcul de durée pour chaque tâche ou User Story.
- Dessiner un Burn Down Chart pour chaque Sprint.
- Dessiner multi-Rapport pour le projet , tâches , sprints, sprint active

- Exporter un fichier powerpoint pour les rapports.
- Gestion des Product Owners et des développeurs.
- l'application multi interface pour chaque utilisateur.
- les tâches peuvent être divisées en sous-tâches.
- Gestion des projets.
- importer un fichier pour une tâche ou un projet.

### 3 Tableau de Synthèse

Le tableau de synthèse ci-dessous présentant les applications Asana , Trello, Nutcache va nous permettre de comparer leurs fonctionnalités avec la nôtre.

Fonctionnalité	Asana	Trello	Nutcache	DO-IT
• Gestionnaire de tâches	✓	✓	✓	✓
• Sous-tâches	✓	✗	✓	✓
• Type tâche	✗	✗	✗	✓
• Gestion de planning	✓	✗	✓	✓
• Progression et suivi de projets	✓	✗	✓	✓
• Tableau de bord	✓	✗	✓	✓
• Scrum support	✗	✗	✗	✓
• Liste des rapports	✗	✗	✗	✓
• Gestion d'équipe	✓	✓	✓	✓
• Gestion de rôle	✗	✓	✓	✓
• Open Source / logiciel libre	✗	✗	✗	✓
• Version gratuite	✗	✗	✗	✓

FIGURE 2.4 – Tableau de Synthèse

## Conclusion

Dans ce chapitre, nous avons présenté une étude de l'existant et la solution proposée pour la gestion de projet. Dans le chapitre suivant, nous présenterons la phase d'études conceptuelle et l'architecture générale de la solution proposée.

# Étude conceptuelle

## Introduction

La conception est une étape primordiale dans le cycle de vie d'un logiciel. Elle permet de décrire les solutions adoptées pour résoudre des problématiques d'une application ou d'un système. Dans ce chapitre, nous présentons tout d'abord la conception générale de notre projet en détaillons son architecture basée sur l'architecture n-tiers et le model MVC. Ensuite nous entamons le diagramme de classe et terminer par le diagramme de séquence.

## 1 Conception de l'application

Afin d'avoir une visibilité parfaite sur l'architecture de l'application et pour une décomposition permettant une distinction des sous modules pour des raisons de maintenance et de réutilisabilité, nous nous intéressons à l'architecture de la plateforme et ses différentes composantes

### 1.1 Architecture n-tiers

L'architecture n-tiers a été faite pour pallier aux limitations des architectures simples telles que l'architecture 2-tiers, et concevoir des applications puissantes et simples à maintenir. Ce type d'architecture permet de distribuer plus librement la logique applicative, ce qui facilite la répartition de la charge entre tous les niveaux.

Cette évolution des architectures qualifie la distribution d'application entre multiples services et non la multiplication des niveaux de service. Cette distribution est facilitée par l'utilisation de composants « métier », spécialisés et indépendants, introduits par les concepts orientés objets

(langages de programmation et middleware). Elle permet de tirer pleinement partie de la notion de composants métiers réutilisables.

Pour notre application, nous adopterons une architecture 3-tiers décrite par la figure 3.1 et qui se présente comme suit :

- **La présentation des données** : contient les différents types de clients, léger (Web, JSP, ASP, PHP, etc.) ou lourd (Swing, SWT, AWT, etc.) comme elle peut correspondre à la restitution de données en un format standard capable d'être interprété par différentes machines.
- **Le traitement métier des données** :correspondant à la logique applicative.
- **L'accès aux données persistantes** :correspondant à la couche d'accès et de stockage des données.

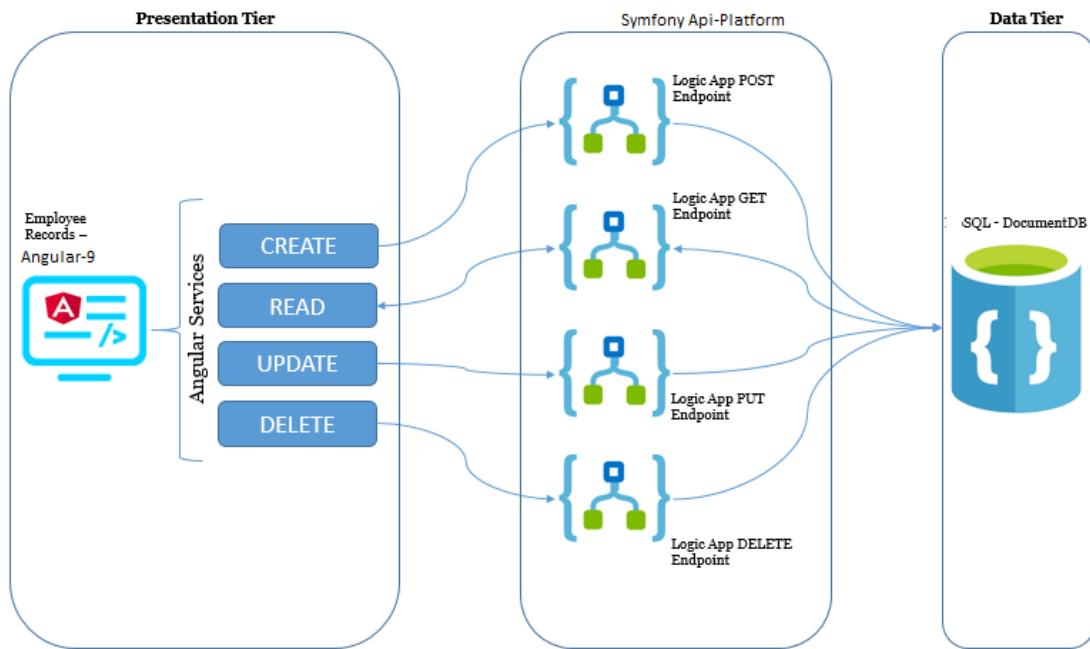


FIGURE 3.1 – Présentation de l'architecture 3-tiers

Dans cette approche, les couches communiquent entre elles à travers un “ modèle d'échange ” et chacune d'entre elles propose un ensemble de services rendus. Les services d'une couche sont mis à la disposition de la couche supérieure. On s'interdit par conséquent qu'une couche invoque les services d'une couche plus basse que la couche immédiatement inférieure ou plus haute que la couche immédiatement supérieure (chaque niveau communique qu'avec ses voisins immédiats).

Le rôle de chacune des couches et leurs interfaces de communication étant bien définis, les fonctionnalités de chacune d'entre elles peuvent évoluer sans induire de changements dans les autres.

Ce modèle d'architecture n-tiers a pour objectif de répondre aux préoccupations suivantes :

- **La flexibilité** : les applications sont sujettes à des changements structurels fréquents à cause de la mutation rapide des technologies c'est pourquoi l'ajout de nouveaux modules est primordiale pour la pérennité de l'application.
- **La sociabilité** : c'est la capacité qu'a l'architecture pour évoluer en cas de montée en charge si nécessaire. En effet, chaque module peut être déployé indépendamment des autres sur une machine, ou un cluster de machines. Ce qui permet d'ajouter des ressources si nous avons besoin.
- **La modularité** : c'est une approche structurante qui sépare un logiciel en petites unités rassemblées qui composeront l'ensemble d'un logiciel. Cette approche permet non seulement une certaine réutilisation de certaines unités de traitements mais aussi une très grande souplesse dans la modification puisque le changement d'un module n'implique pas le changement de tous les autres.

## 1.2 Le Modèle MVC

Le Modèle-Vue-Contrôleur (en abrégé MVC, de l'anglais (Model-View-Controller) est une architecture logicielle et une méthode de conception logicielle. Ce modèle MVC présenté dans la figure 4.2 impose la séparation entre les données, les traitements et la présentation. C'est pour cette raison que l'application est divisée en trois composants fondamentaux : le modèle, la vue et le contrôleur. Chacun de ces composants tient un rôle bien défini.[3]

- **Un modèle (Model)** contient les données à afficher.
- **Une vue (View)** contient la présentation de l'interface graphique.
- **Un contrôleur (Controller)** contient la logique concernant les actions effectuées par l'utilisateur.

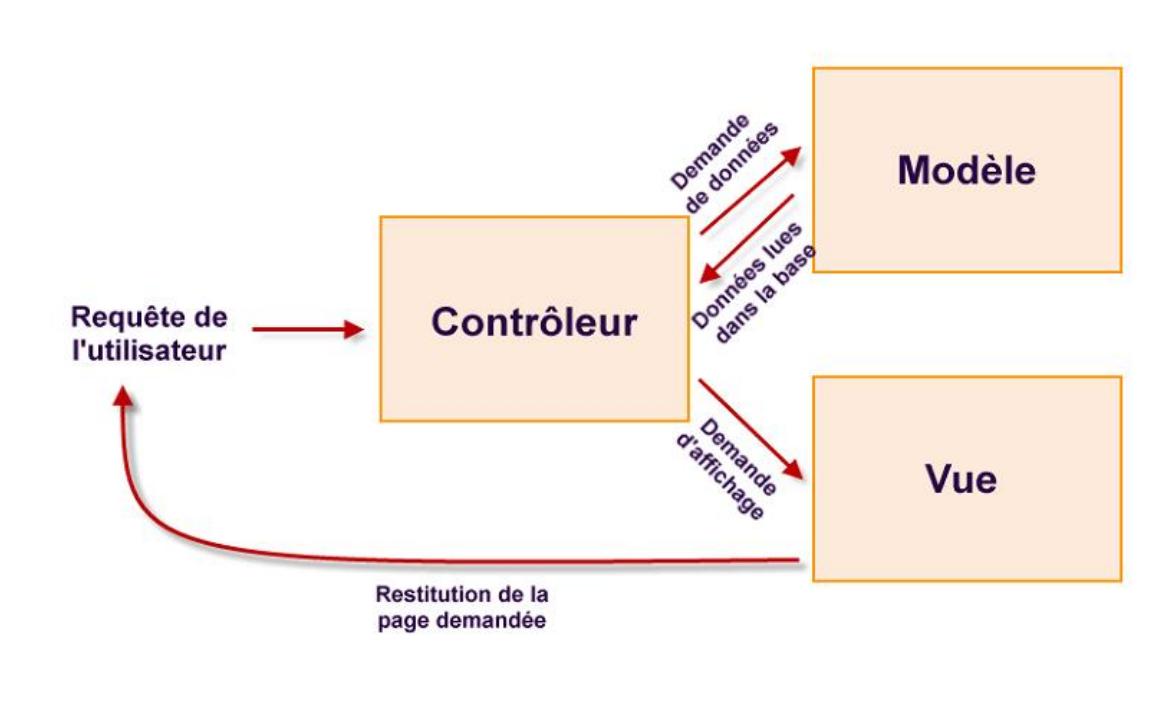


FIGURE 3.2 – Présentation de Modèle MVC

## 2 Conception détaillée

Nous décrivons dans cette partie la conception détaillée de notre application, commençant par la définition de la méthodologie UML, passant par le diagramme de classe, ensuite les diagrammes de séquences.

### 2.1 Diagramme de classes

Le diagramme de classe est une représentation statique des éléments du système et de leurs relations. Chaque application qui va mettre en œuvre le système sera une instance des différentes classes qui le compose. À ce titre il faudra bien garder à l'esprit qu'une classe est un modèle et l'objet sa réalisation.[11]

Dans ce qui suit nous présentons les principales classes et les relations entre elles.

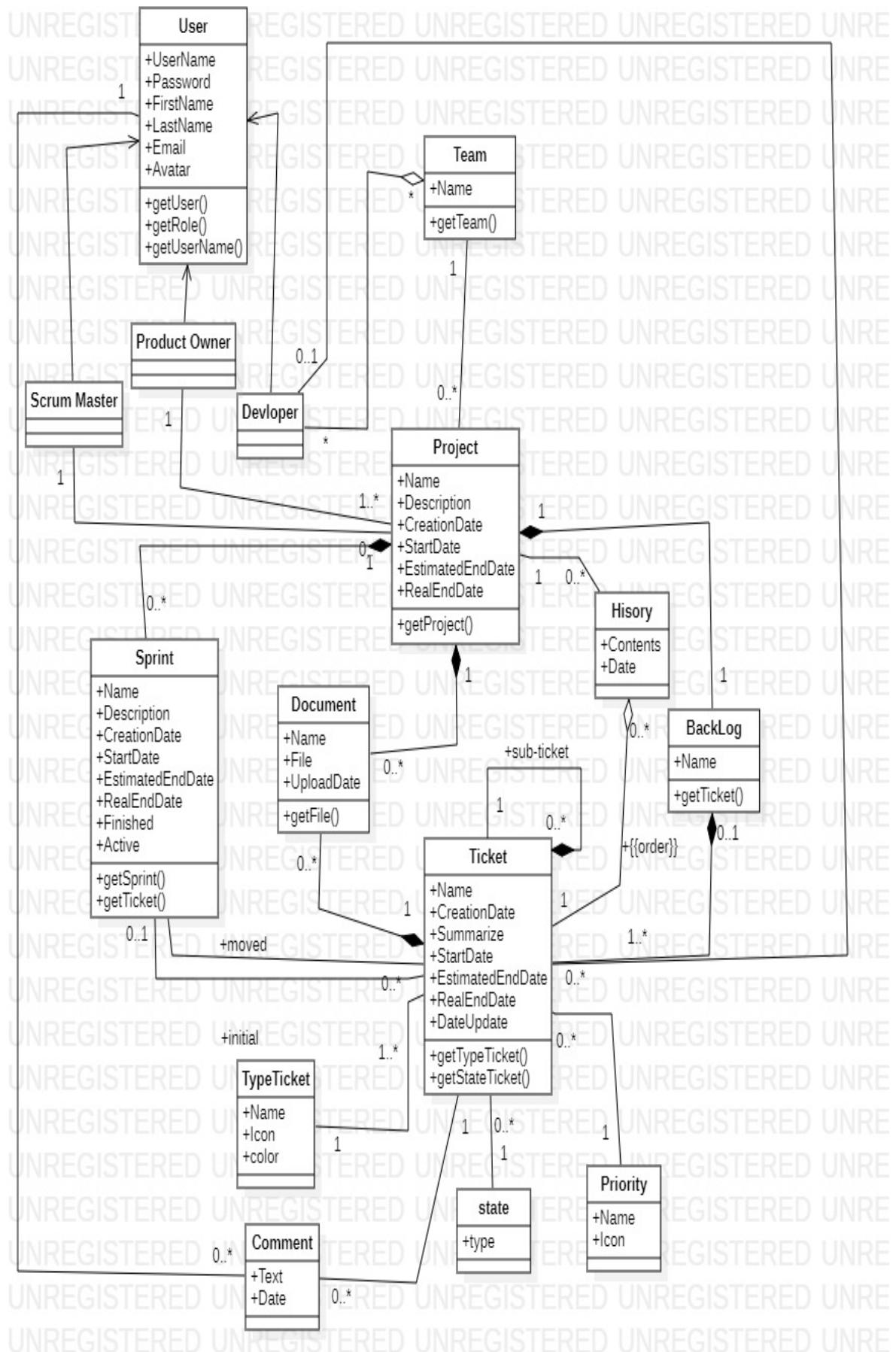


FIGURE 3.3 – Diagramme de classes

## 2.2 Diagrammes de séquence

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique, ils permettent de définir plus précisément que dans le cas d'utilisation le principe de fonctionnement de certaines phases de l'application. Dans un souci de simplification, le but étant de décrire comment se déroulent les actions entre les acteurs ou objets.[2]

### 2.2.1 Diagramme de séquence « Authentification »

Afin de pouvoir accéder à la plateforme, l'utilisateur est invité à saisir ses coordonnées. Le composant web « Login » valide le formulaire en appelant la méthode isValid (). En cas de validation du formulaire, « Login » envoi une requête HTTP de type GET au contrôle «AuthController ». Le contrôle fait appel à la méthode LoginCheck (Request) avec les paramètres login et le mot de passe qui communique à son tour avec le model USER, si le utilisateur n'existe pas dans notre base, le model renvoi un message d'erreur au contrôle et puis le contrôle envoi au login page une requête HTTP de type Response Error Message, le vue afficher un message d'errer au l'utilisateur. Dans le cas où l'utilisateur existe le model renvoi un token qui contient le rôle de l'utilisateur et sont user name, après l'interface Dashboard affiche dans l'écran.

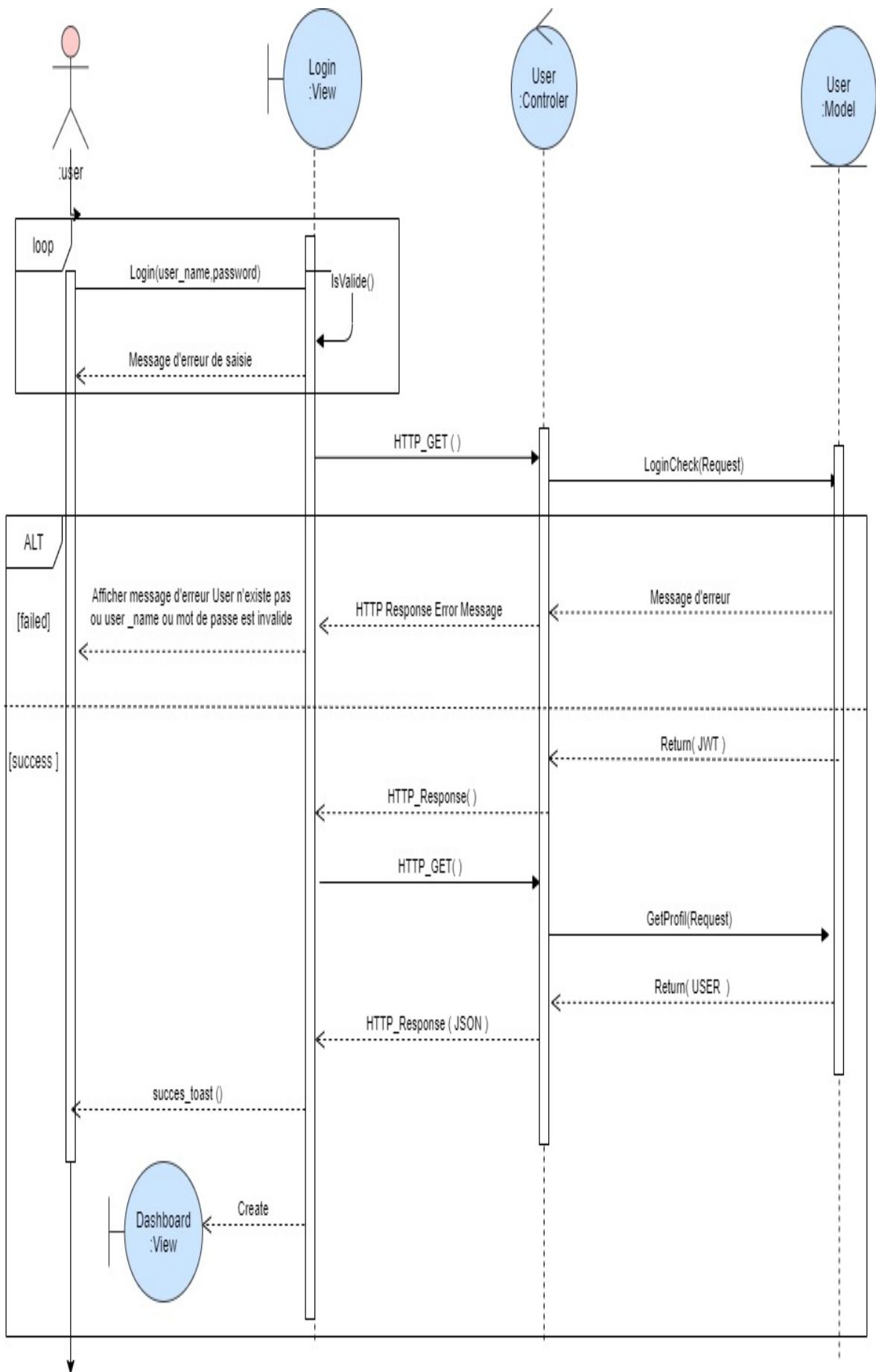


FIGURE 3.4 – Diagramme de séquence « Authentification »

## 2. 2. 2 Diagrammes de séquence « Gérer Sprint »

Dans le figure 3.5 on a présenter le diagramme de séquence Gérer Sprint qui composer sur 4 parti, consulter sprint le Scrum Master demande la page « Work Flow », le vue envoi un requête GET avec le paramètre l'id de projet, après les sprint qui sont terminés sera afficher dans la vue. Deuxième parti ajouter un ticket au Sprint , le Scrum Master clique sur le bouton « Add Ticket » , le vue « Dialog Add Ticket » s'affiche. Le Scrum Master saisit les données puis clique sur bouton « Save ». troisième parti Gérer ticket , permet au Scrum Master de modifier le ticket, commenter , ajouter fichier ou changer le type de ticket. Dernier parti c'est activer Sprint le vue vérifie si il'y a un sprint est déjà active ou non, si oui le vue cacher le bouton « Star Sprint », si non un vue « Start Sprint Dialog » s'affiche puis le scrum Master sélect la période de Sprint puis clique sur bouton « Start ».

## Étude conceptuelle

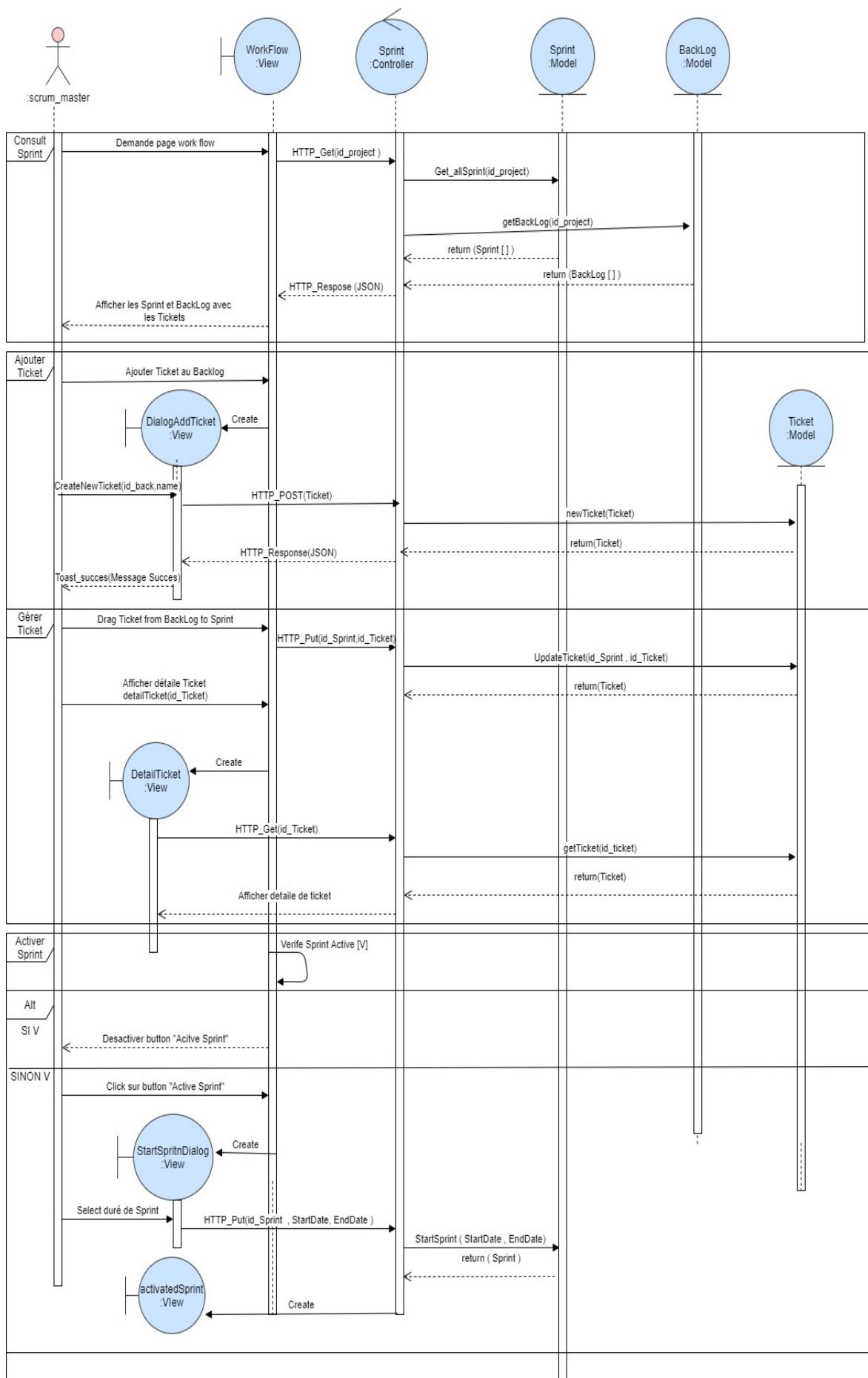


FIGURE 3.5 – Diagramme de séquence « Gérer Sprint »

### 2.2.3 Diagramme de séquence « Sprint Active »

Le Scrum Master demande la vue Active Sprint, la vue envoi une requête GET avec le paramètre active puis le contrôle fait appel à la méthode getSprint(active) qui communique avec le model Sprint. Le model renvoie le résultat au contrôle, puis le contrôle forme le résultat sous format JSON, après la vue présente le JSON dans l'interface.

Le scrum clique sur le bouton « finish », la vue vérifie si tous les tickets sont terminés ou non, si tous les tickets sont terminés. La vue envoi une requête HTTP de type POST avec paramètre l'id de sprint au contrôle puis le contrôle fait appel à la méthode updateSprint(id), le model applique la méthode, avec un message succès . Après l'interface rapport affiche dans l'écran. Sinon si il y a des tickets non terminés le lance une requête HTTP de type GET, le model sélectionne tous les sprints, et l'affiche dans un nouveau vue appelé Dialog Finish Sprint, puis le Scrum Master sélectionne un sprint ou les tickets sera déplacer, puis une requête POST lancer vers le contrôleur pour modifier le ticket.

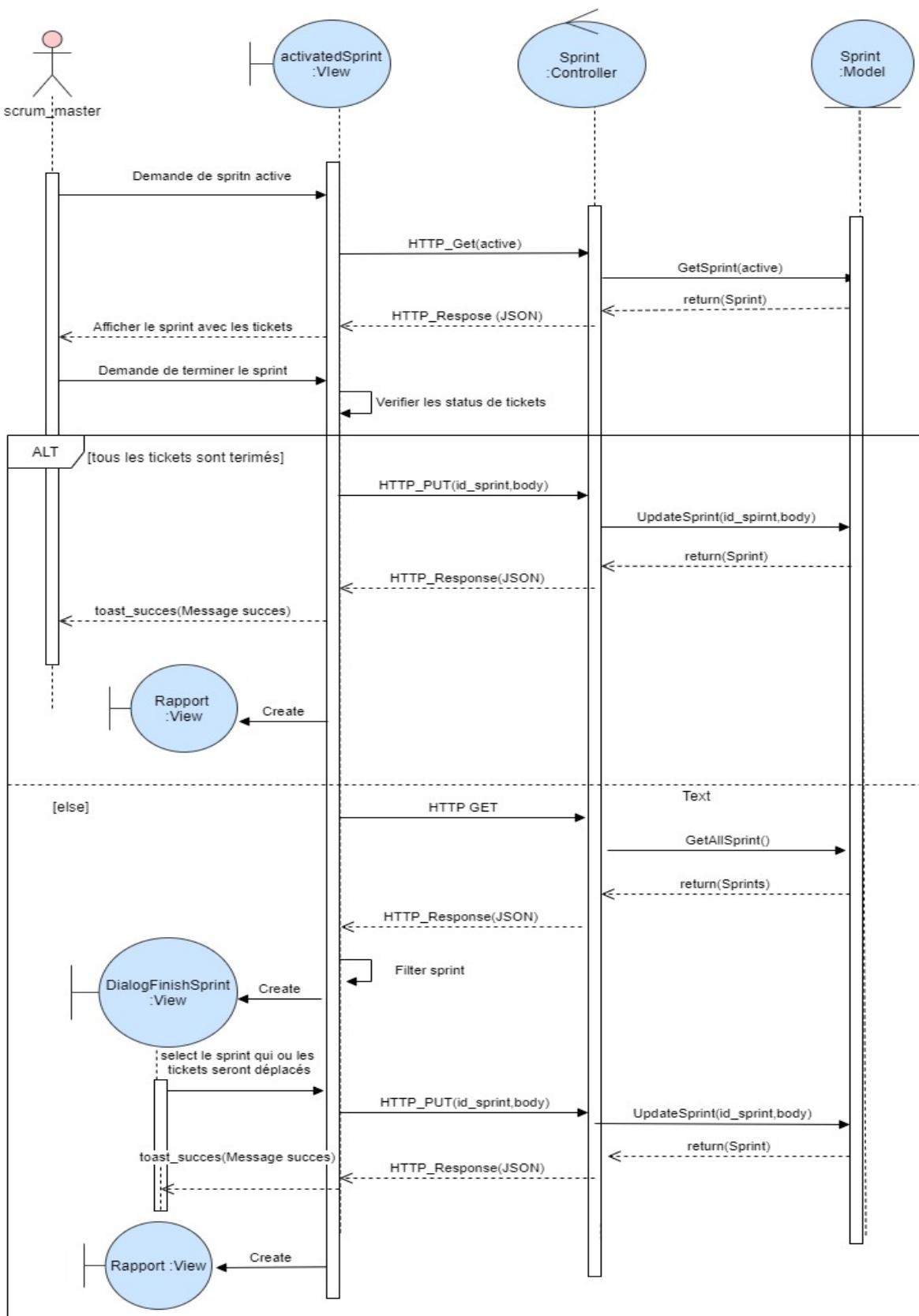


FIGURE 3.6 – Diagramme de séquence « Sprint Active »

## 2.2.4 Diagramme de séquence « Gérer les rapports »

Gérer les rapports est une étape importante dans l'utilisation de la plateforme. Le Scrum Master peut le consulter les rapports de chaque sprint terminer, le Model Sprint retourne au vue liste des sprint qui sont terminé, le Scrum Master choisit un sprint pour consulter les graphes et il peut aussi de télécharger un fichier power-point qui contient les rapport de sprint .

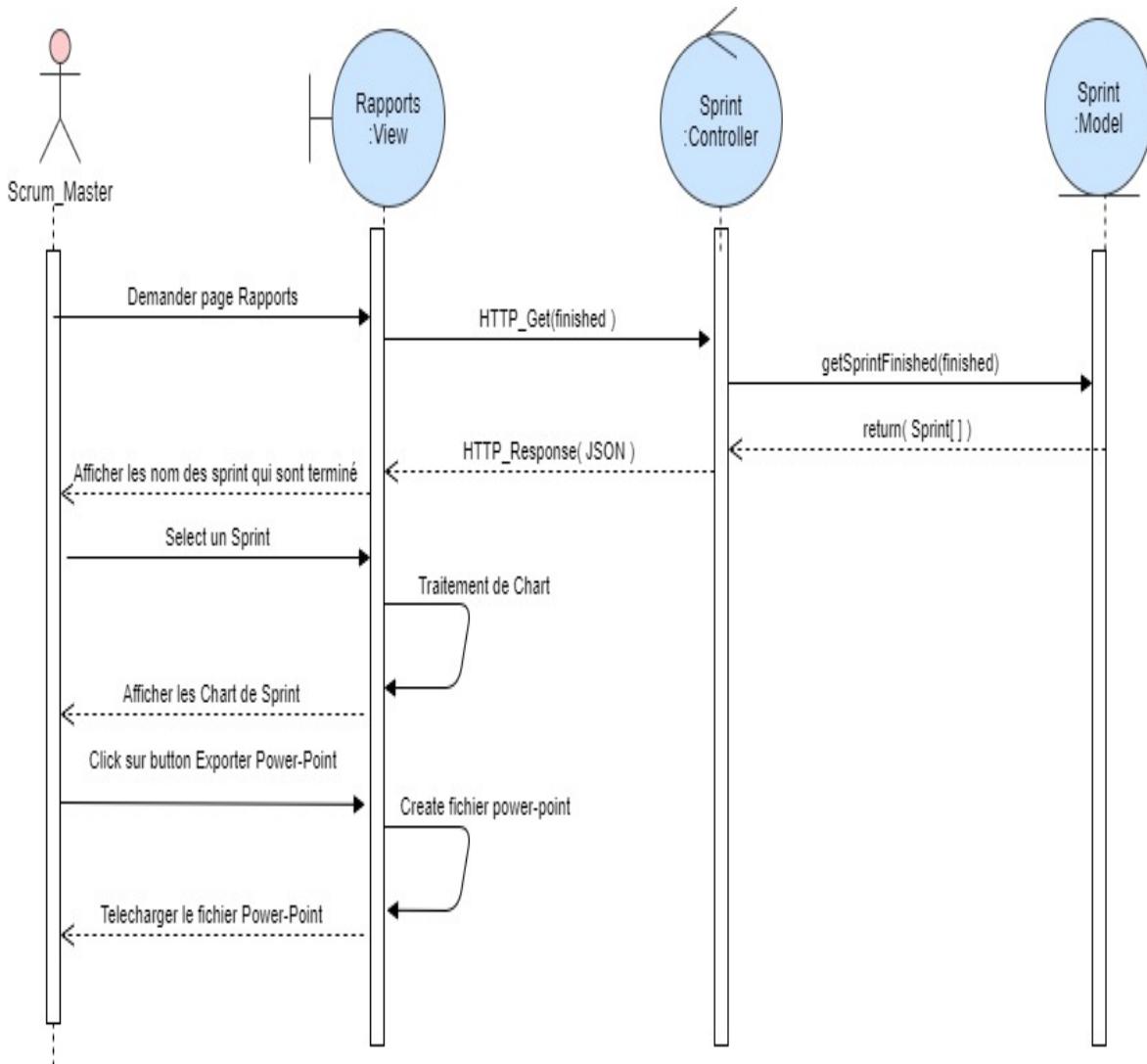


FIGURE 3.7 – Diagramme de séquence « Gérer les rapports »

## 2.2.5 Diagramme de séquence « Gère groupe de développeurs »

Le Scrum Master demande l'interface Team, l' model Team retourne au contrôle résulta de tous les groups, puis le résultat affiche dans le vue. Le Scrum Master clique sur le bouton «Add new Team » le vue envoi une requête HTTP GET, au contrôle, puis le model User renvoi

un résultat de tous les user son rôle développeur, puis un nouveau vue s'affiche, puis le Scrum Master sélect les user de groupe, et effectue un nom au groupe et terminer par le clique sur bouton «Create new Team ». Le vue lance une requête HTTP Post avec des paramètres et le model ajouter le groupe dans la base de donnée.

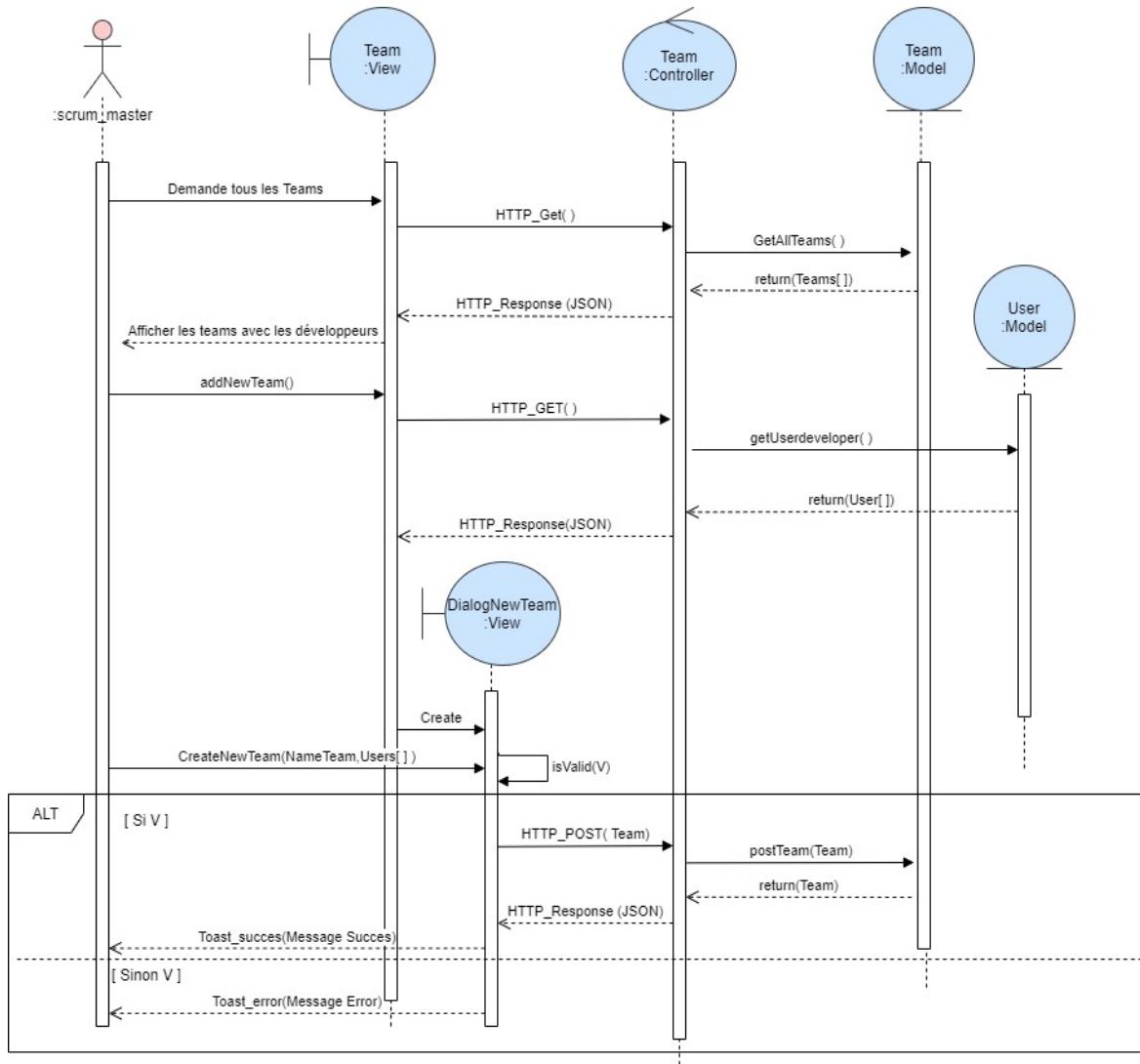


FIGURE 3.8 – Diagrammes de séquence « Gère groupe de développeur »

## 2. 2. 6 Diagramme de séquence « Gérer les tâches de sprint active »

Afin d'activer le sprint, le développeur pour gérer les tâches de ce sprint, le développeur fait appel à le sprint active. Le vue lance une requête HTTP de type GET au sprint controller. le controller fait l'appel à la méthode getSprintActive(). La méthode a une retour de Sprint avec les tâches, le vue fait le classement de ticket selon leur états puis il l'affiche. Le développeurs glisse un tâche vers un autre état.

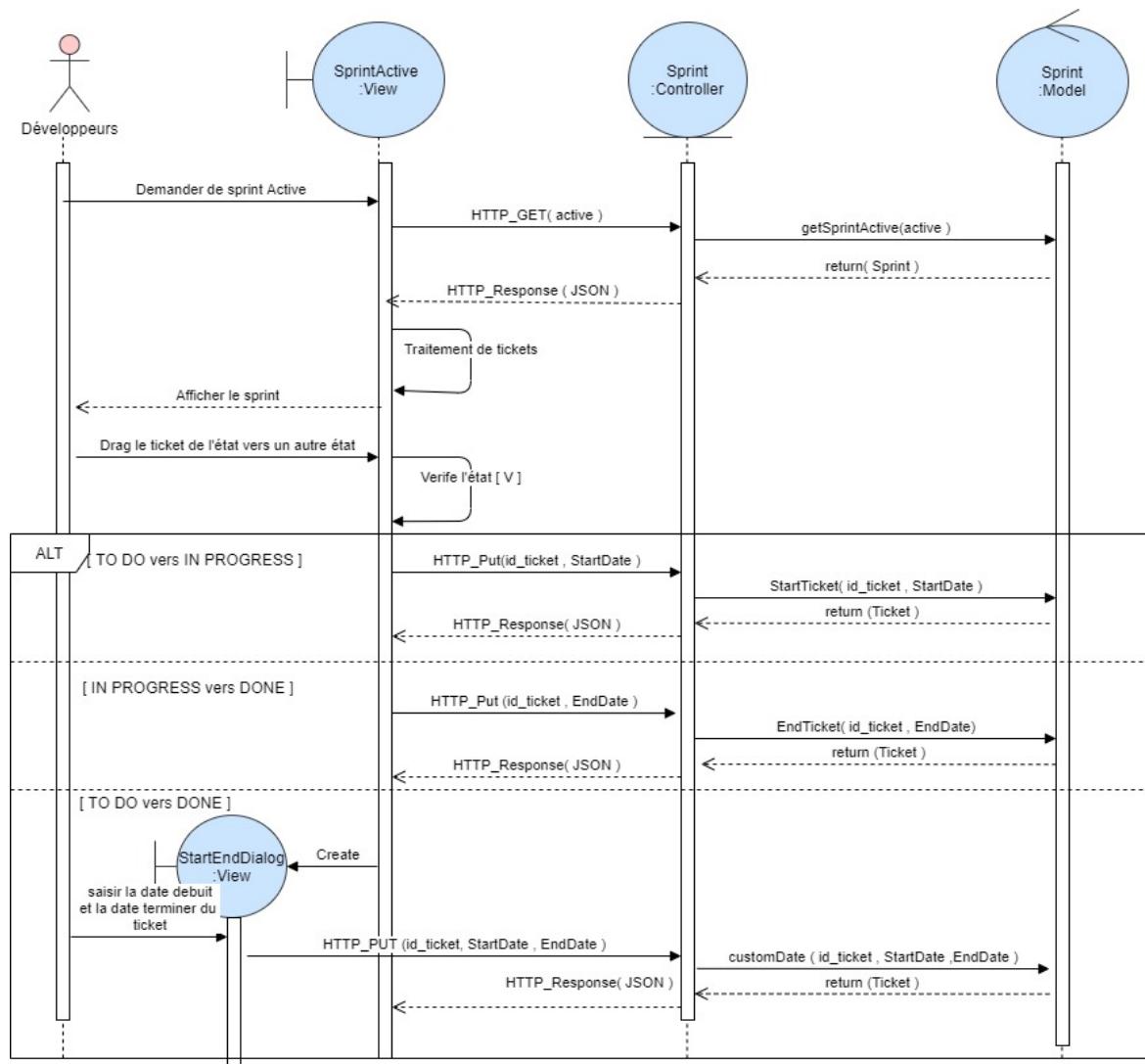


FIGURE 3.9 – Diagramme de séquence « Gérer les tâches de sprint active »

## Conclusion

Dans ce chapitre, nous avons présenté l'architecture de l'application et décrit la conception suivant la démarche UML. Cette conception nous permet d'entamer la réalisation et l'implémentation de notre système.

# Réalisation

## Introduction

Après avoir clôturé la phase de conception, la solution étant déjà choisie et bien étudiée, je présente dans ce chapitre quelques détails de réalisation. Tout d'abord, nous présentons un aperçu sur les technologies et les outils utilisés pour réaliser mon application. Finalement, nous allons élaborer une présentation des différentes interfaces créées.

## 1 Environnement de travail

### 1. 1 Environnement matériel

Les caractéristiques de la machine utilisée sont :

- Nom : Asus
- Processeur : AMD rayzen 5 3550x
- RAM : 32GO
- Disque Dur : 1.5To
- Type système : Windows 64 bits

### 1. 2 Outils de réaliataion

Au niveau du système d'exploitation et des logiciels utilise.

### 1.2.1 Draw.io

Draw.io est un logiciel gratuit de création de diagrammes en ligne pour la création d'organigrammes, de diagrammes de processus, d'organigrammes, de diagrammes UML, ER et de réseaux.[4]



### 1.2.2 Visual Studio Code

Visual Studio Code est un éditeur de code open-source, gratuit et multiplateforme (Windows, Mac et Linux), développé par Microsoft. Principalement conçu pour le développement d'application avec JavaScript, TypeScript et Node.js, l'éditeur peut s'adapter à d'autres types de langages grâce à un système d'extension bien fourni.



### 1.2.3 ngrok

Ngrok est une petite application gratuite qui va vous permettre par exemple de partager un site web en cours de développement sur votre machine locale avec n'importe qui à travers le monde... et ceci sans vous prendre la tête avec le paramétrage de votre routeur ou encore le firewall de votre entreprise.[6]



### 1. 2. 4    **phpstorm**

PhpStorm est un éditeur pour PHP, HTML, CSS et JavaScript, édité par JetBrains. Il permet d'éditer du code PHP pour les versions allant de la 5.3 à la 7.4[8]



### 1. 2. 5    **MySQL**

MySQL est un serveur de bases de données relationnelles Open Source qui sert à stocker les données dans des tables séparées plutôt de les rassembler dans une seule table. [7]



### 1. 2. 6    **Symfony**

Symfony est un framework qui représente un ensemble de composants (aussi appelés librairies) PHP autonomes qui peuvent être utilisés dans des projets web privé ou open source. Mais c'est également un puissant Framework PHP. On l'a utilisé pour le développement de ce Framework pour le développement BackEnd de l'application.[10]



#### **Pourquoi choisir Symfony ?**

Symfony étant basé sur une architecture HTTP, son utilisation est idéale pour créer des webservices RESTful. Tout est natif dans le framework, de la conception des endpoints jusqu'à

la sécurité des accès. Il est possible d'aller plus loin grâce à des Bundles open source dédiées pouvant être intégrés dans le projet..

Pour créer les API de notre application nous utilisons Bundel « API Platform » :

Qui est un framework libre (licence MIT) écrit en PHP7 et basé sur Symfony destiné à la création d'API Web modernes, puissantes et sécurisées. Cet outil est particulièrement adapté à la construction de systèmes d'informations « API-centric » basés sur l'hypermédia et le Web des données.



Voici encore d'autre raisons pour notre choix :

- Développement rapide et facile.
- Stable et durable.
- Des nombreuses bibliothèques sont déjà intégrées.
- Facilite les tests unitaires.

### 1.2.7 Angular

Angular est un Framework côté client open source basé sur TypeScript dirigé par l'équipe du projet Angular à Google et par une communauté de particuliers et de sociétés. On l'a utilisé pour le développement frontend de l'application. Le Framework est basé sur une architecture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités. Un type d'architecture qui a largement fait ses preuves et qui permet une forte maintenabilité et une amélioration du travail collaboratif.



### Pourquoi choisir Angular ?

Angular fournit donc nativement tout le nécessaire pour produire une application entière avec une configuration standard. Parmi les raisons pour notre choix on peut citer :

- TypeScript : Angular profite de la rigueur et flexibilité du langage TypeScript.
- Mvc : Angular permet de mieux séparer les responsabilités avec une approche MVC (Model / View / Controller) et l'injection de dépendances.
- Architecture et maintenabilité : Angular impose une approche mieux structurée à base de composants et une façon plus claire d'échanger les données entre les composants.
- Des améliorations constantes et régulières (compilation inférieure à 3 secondes pour Angular 9)

## 2 Diagramme de déploiement

L'application DO-IT est une application qui se connecte à un serveur de base de données via Internet afin de récupérer les données, ce qui nécessite l'intégration d'un serveur web entre l'application client et le serveur de bases des données. C'est pour cela que nous avons conçu le diagramme de déploiement pour notre application client-serveur.

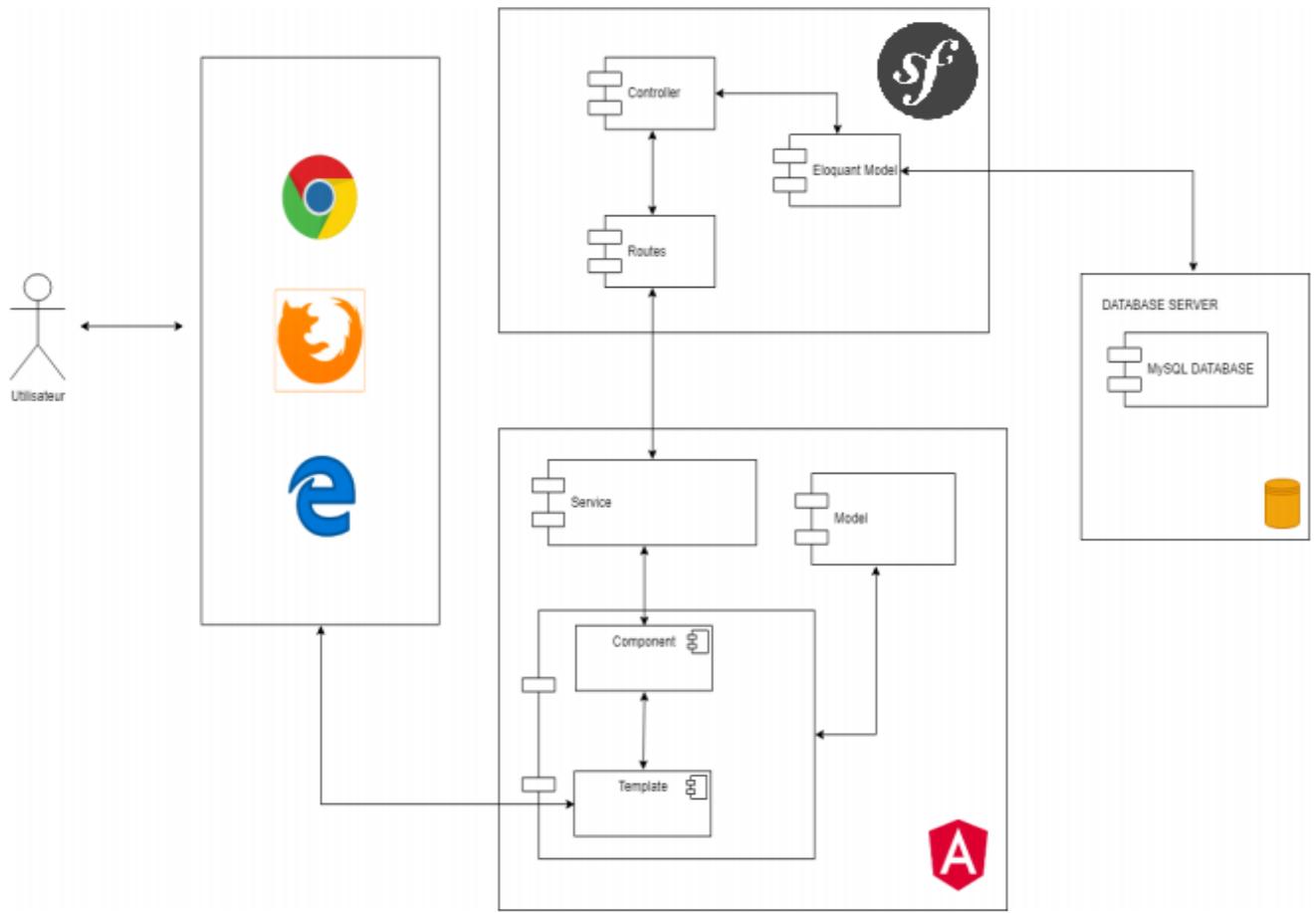


FIGURE 4.1 – Diagramme de déploiement

### 3 Présentation des interfaces de l'application

nous présenterons dans ce qui suit un scénario complet d'utilisation pour tous les acteurs qui inclut toutes les interfaces de mon application tout en suivant une séquence bien définie.

### 3.1 Interface Scrum Master

#### 3.1.1 Authentification

La figure suivante présente une fenêtre de connexion. La première fenêtre affichée lors de l'exécution de l'application. L'utilisateur peut saisir le user name et le mot de passe. Cette page contient également un bouton "Login", Si le user name et le mot de passe sont vrais, on peut accéder à la fenêtre principale sinon un message d'erreur s'affiche, trois modes d'accès (Scrum Master, Développeur ou Product Owner).



FIGURE 4.2 – Interface login

### 3.1.2 Dashboard

Si la connexion a été établie par le Scrum Master, elle est affichée comme première interface qui est composée de deux parties, la première comprend la liste des projets de Scrum Master avec le nom et description de chaque projet et la deuxième partie est composée d'un tableau avec un recherche rapide, le tableau détaille le projet avec un pourcentage des tâches complétées.

The screenshot shows the Scrum Master dashboard interface. On the left is a vertical sidebar with icons for navigation. The main area is divided into two sections: a top section showing project cards and a bottom section showing a detailed project table.

**Top Section: Your projects**

- Work Flow (create application scrum "WORK FLOW")
- pfe (pfe to do)
- Conception (diagramme de classe et diagramme use case)
- Back Log (Back log)
- Back log Sprint (Sprint Back log)
- SPRINT (SPRINT)
- Application Prototyp (creation d'application qui crée des prototypes d'applications iOS.)

**Bottom Section: Detailed Project Table**

#	Name	Type	Creation Date	Project Manager	Team	tasks completed	...
0	Work Flow	software	Feb 25, 2020	farouk rebhi	Team	40%	...

A search bar at the top of the table is partially filled with "wor".

FIGURE 4.3 – Interface Dashboard Scrum Master

### 3.1.3 Ajout un projet

L'interface ajoute un projet qui permet au Scrum Master d'en créer un, l'interface s'affiche lorsque le Scrum Master clique sur le bouton « Create Project », puis le Scrum Master remplira le formulaire, et choisira le type de projet et le groupe de développeurs en terminant par un clique sur le bouton « Create ».

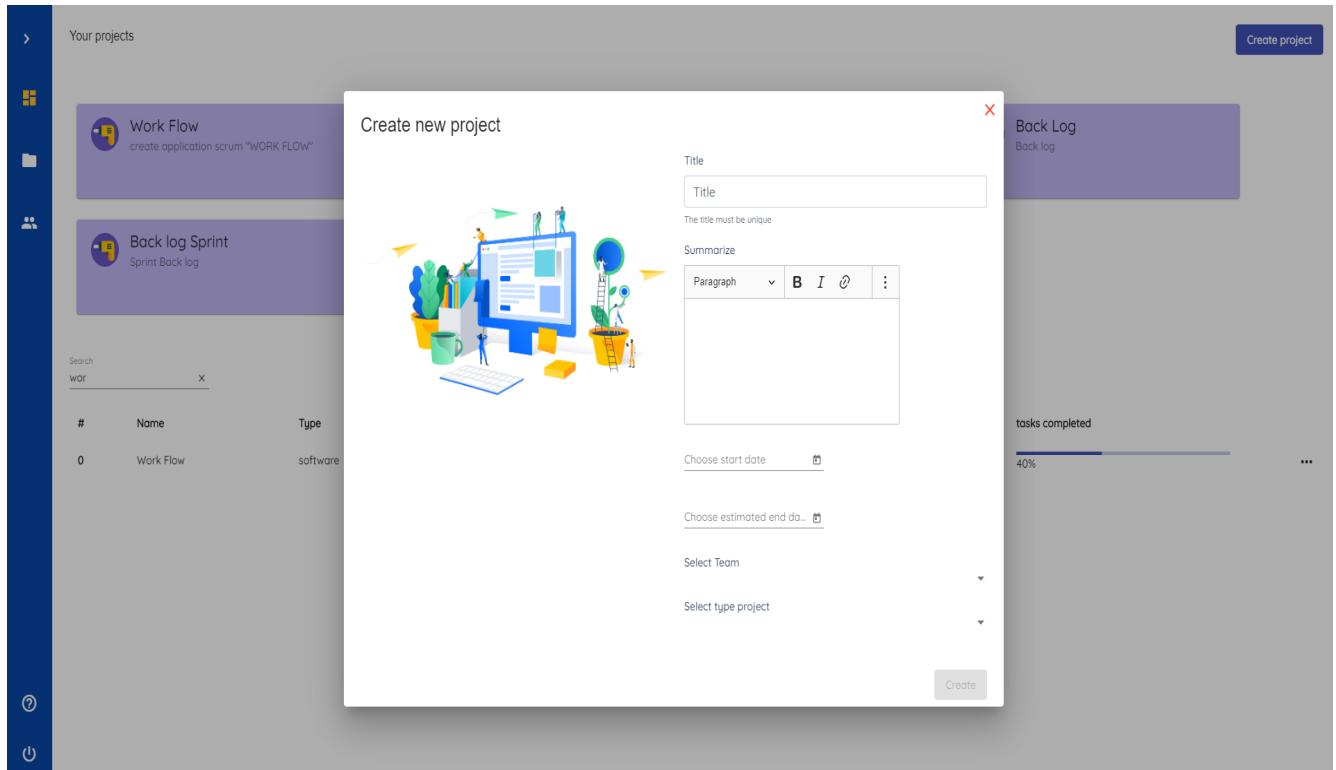


FIGURE 4.4 – Interface Ajout un projet

### 3.1.4 Work Flow

Cette interface est plus important pour le Scrum Master, a partir de cette interface il peut composer le projet sur des User story qui appartient au backlog et créer les sprint, c'est à dire la planification de projet. Cette interface est composée de trois parties .

**Sprint 1** 2 Ticket  
Jul 16, 2020 \*

- function logout
- Demo application

**Sprint 2** 3 Ticket  
Jul 16, 2020 \*

- Create new page profile User
- page login
- Create Token Back office

**Back log** 1 Ticket  
Jul 16, 2020 \*

- page auth

**Plan the work of your team**  
Here is your team's Sprint. Create and estimate new tickets, and prioritize them using drag and drop.

**Create sprint** **Start sprint**

**Ticket detail** **Create new page profile User**

**Create new page profile User**

**TO DO**

**User**

**Sub-task**

**Developer**

**Not assigned**

**Comments** **History**

**FR** Add a comment

Created date : Jul 16, 2020, 11:28:00 PM  
Last update :

farouk rebhi moved Demo application to Sprint 1 • 2 hours 0 minutes ago

farouk rebhi moved function logout to Sprint 1 • 2 hours 0 minutes ago

farouk rebhi moved page auth to Sprint 2 • 2 hours 0 minutes ago

farouk rebhi moved page login to Sprint 2 • 2 hours 0 minutes ago

FIGURE 4.5 – Interface Work Flow

Dans la première partie on a la liste des Sprints avec leur tâches et le Backlog avec les User Story, le Scrum Master peut créer des sprints et des tâches et aussi peut modifier l'emplacement des tâches, pour faire cette action il suffit de glisser le tâche vers le sprint.

**sprint 1** 2 Ticket  
Jul 14, 2020 \*

- create auth page
- create page login

**BACK log** 0 ticket

+ Create Ticket

FIGURE 4.6 – Interface Modifier l'emplacement de tâche

Dans la deuxième partie, lorsque le Scrum Master clique sur la tâche, l'interface détaille la tâche qui sera affichée avec les informations dans cette interface, le Scrum Master peut modifier le nom, l'état, la description ou le type. Dans cette partie il peut ajouter des fichiers ou des sous-tâches. Il suffit de sélectionner un développeur qui existe dans le groupe de développeurs, y ajouter des commentaires et consulter l'historique des tâches.

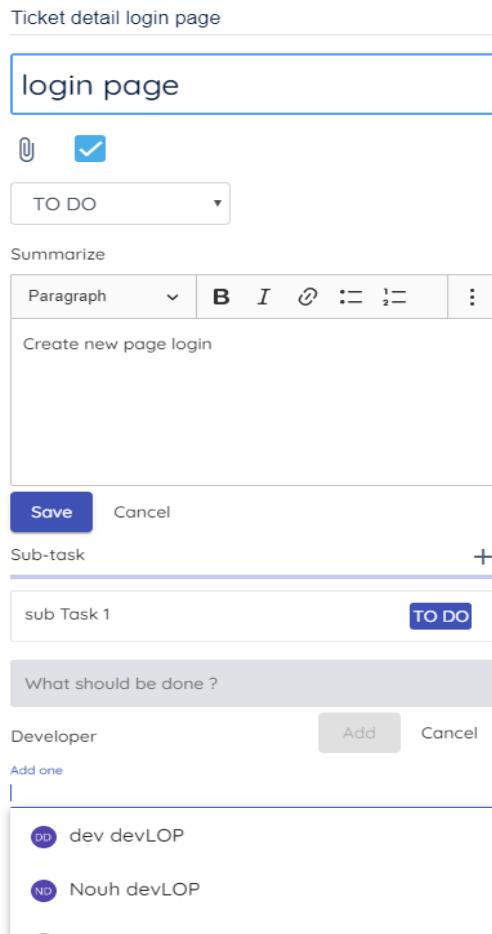


FIGURE 4.7 – Interface Détail tâche

Dans la troisième partie, l'historique de projet doit afficher avec les informations de chaque historique.

### 3.1.5 Sprint Active

Pour accéder à l'interface sprint active, le Scrum Master doit cliquer sur le bouton "Start Sprint", l'interface "start sprint" sera démarrer après effectuer une durée du sprint. Cette durée peut être une semaine, deux semaines, quatre semaines ou par option.

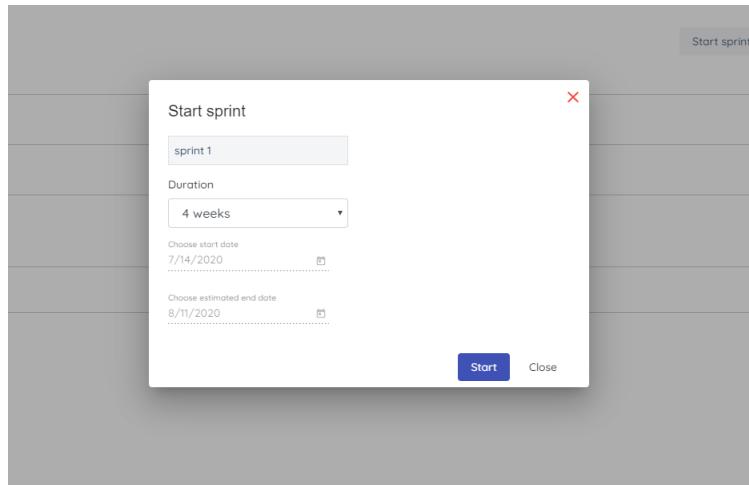


FIGURE 4.8 – interface Activer Sprint

Après d'activer le sprint, l'interface "Sprint Active" sera affiché, avec leurs tâches classée selon leurs états.

A screenshot of a project management interface titled 'Tableau sprint 9'. On the left is a sidebar with icons for Project, Workflow, and sprint 9. The main area shows a Kanban board with three columns: 'TO DO', 'IN PROGRESS', and 'DONE'.

- TO DO:**
  - task with Summarize
  - Update Ticket 2 (status: ND)
  - tickl 15
- IN PROGRESS:**
  - ticket
  - new ticket
  - sprint
  - Conception
- DONE:**
  - spd 25 (status: ND)
  - xo
  - sa
  - ez
  - gz
  - da

A status bar at the top right indicates '6 days 2 hours left' and a 'Finish sprint' button.

FIGURE 4.9 – Interface Activer Sprint

### 3.1.6 Terminer Sprint

Pour terminer le sprint, il suffit de cliquer sur le bouton "Finish Sprint" le système fait un traitement, si les tâches ne sont pas toutes terminées, une interface sera affichée avec les sprints non activés et backlog. Le Scrum Master a le choix de placer les tâches non terminées vers le sprint ou backlog après avoir cliqué sur terminer. Si les tâches sont terminées alors le sprint l'est aussi.

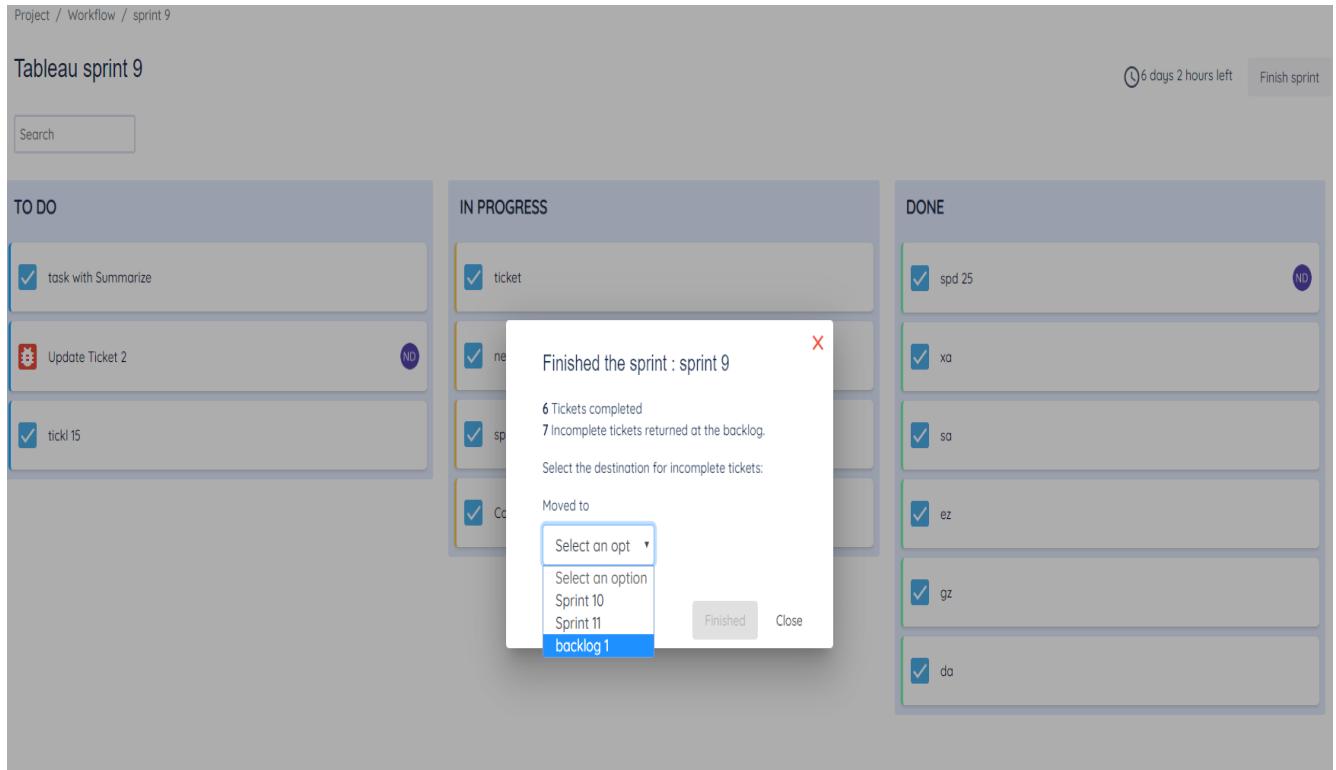


FIGURE 4.10 – Interface Terminer Sprint

### 3.1.7 Rapports

Le Scrum Master peut sélectionner un sprint terminé et afficher les rapports.

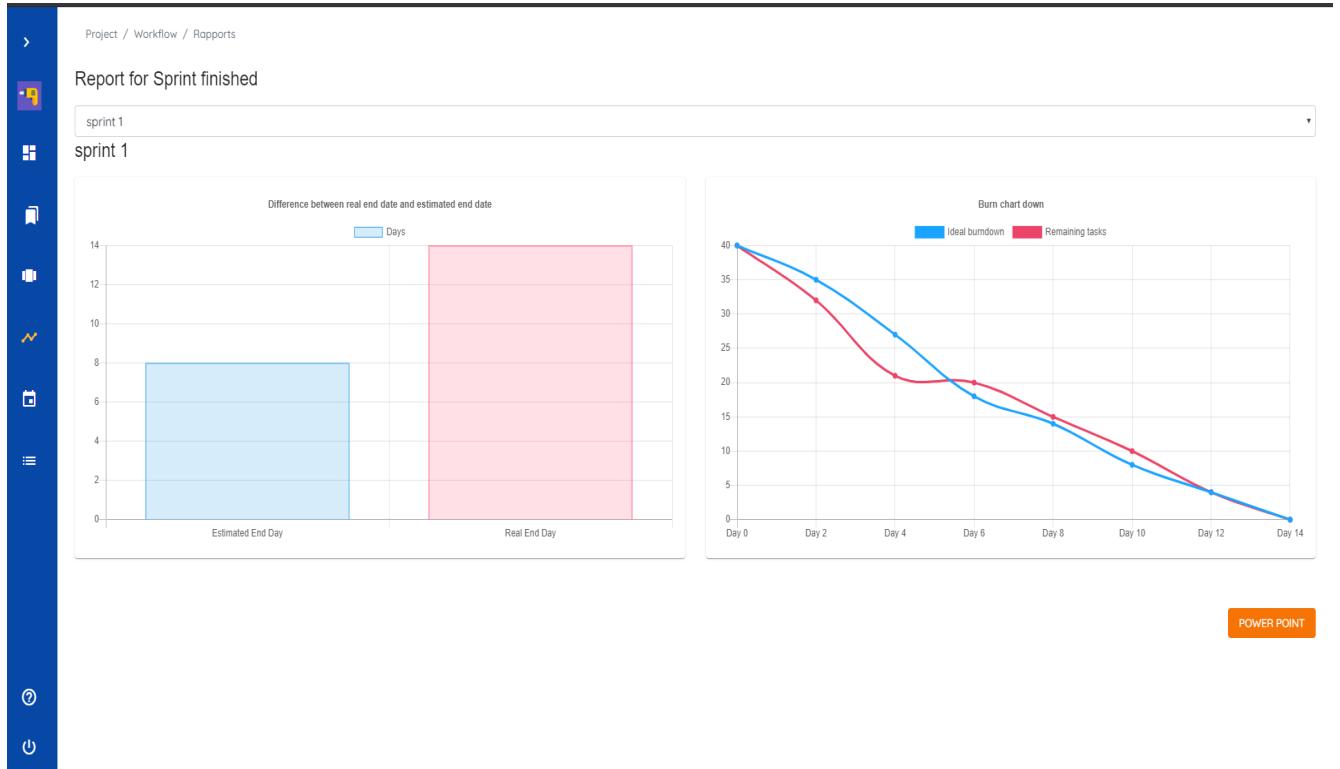


FIGURE 4.11 – Interface liste des rapports

Si le Scrum Master besoin de présenter les rapport de chaque sprint il peut exporter les rapports sous forme d'un fichier power point.

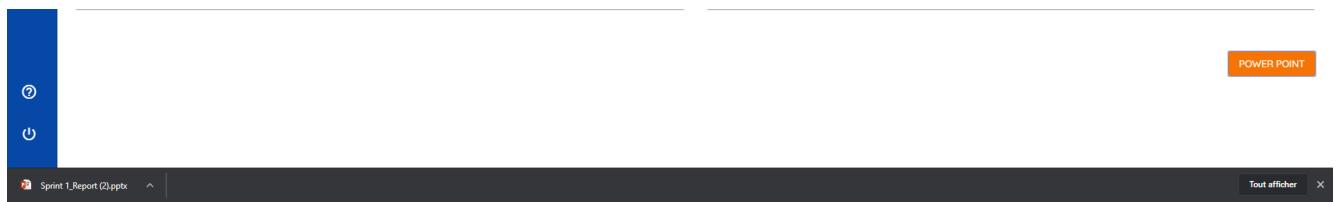


FIGURE 4.12 – Exporter rapports

## Étude conceptuelle



FIGURE 4.13 – Capture de fichier Power Point

## 3.2 Interface développeur

### 3.2.1 Interface Dashboard développeur

Dans le cas où un développeur vient de se connecter, l'interface présentée ci-dessous sera sa première interface.

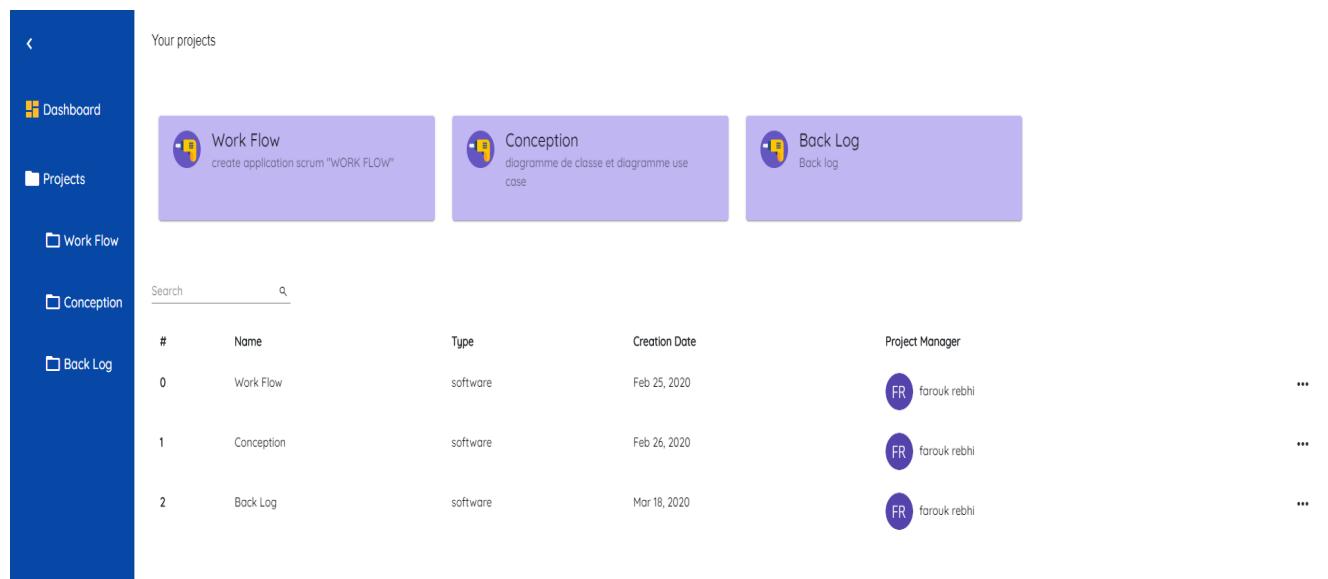


FIGURE 4.14 – Interface Dashboard développeur

Le développeur ne peut donc consulter que la liste du projet dont il appartient et il n'a que la permission d'édition (ni ajout, ni suppression).

### 3.2.2 Gérer les tâches de sprint actif

L'édition d'un tâche pour un développeur lui donne le pouvoir seulement de changer l'état d'un tâche juste il glisse le tâche vers un autre état.

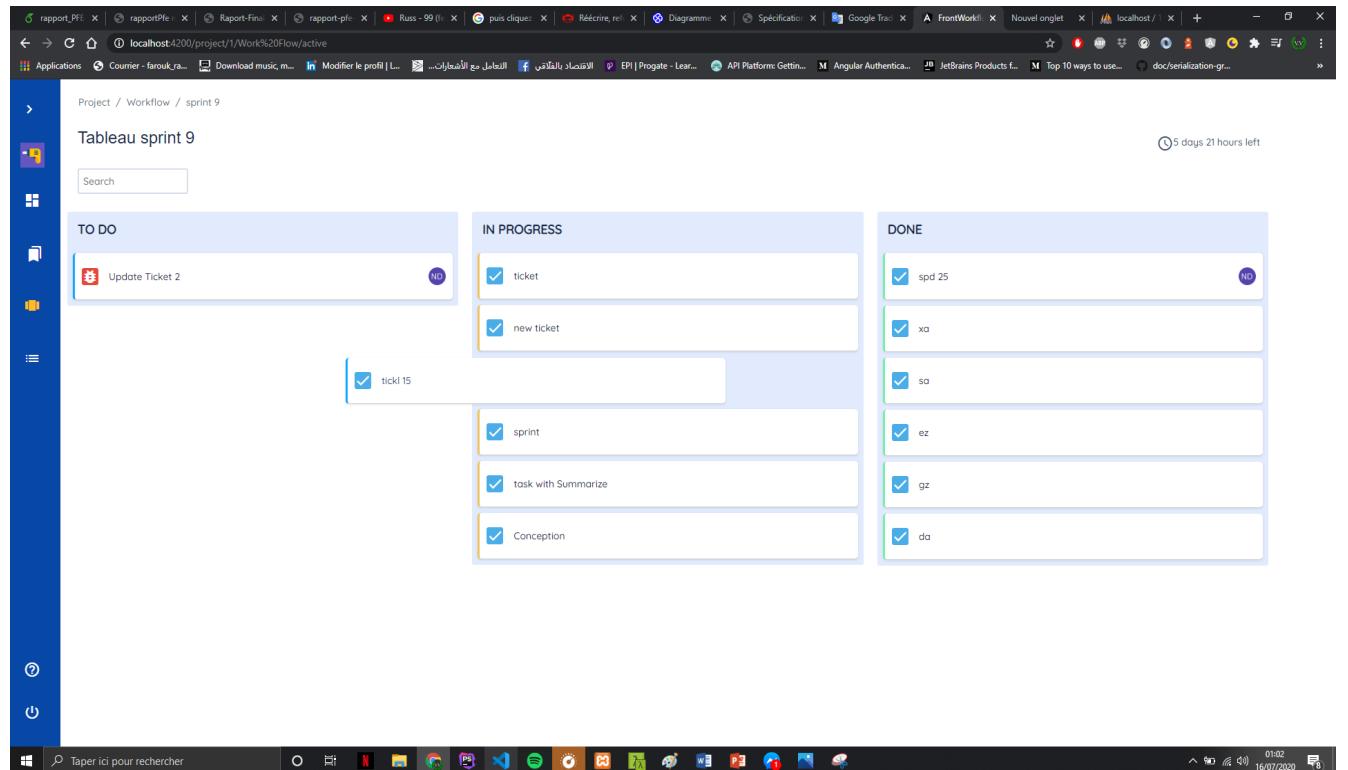


FIGURE 4.15 – Interface changer l'état d'un tâche

On a 3 cas d'édition :

- si le développeur glisse la tâche de l'état "To Do" vers "In Progress", le système lance la date du début de la tâche,
- s'il glisse l'état de "In Progress" vers "Done" , le système affectera une date de fin de tâche,
- s'il glisse de de l'état "TO Do" vers "Done", le système affiche une interface qui demande au développeur d'affecter une date de début et une date fin de tâche.

## Étude conceptuelle

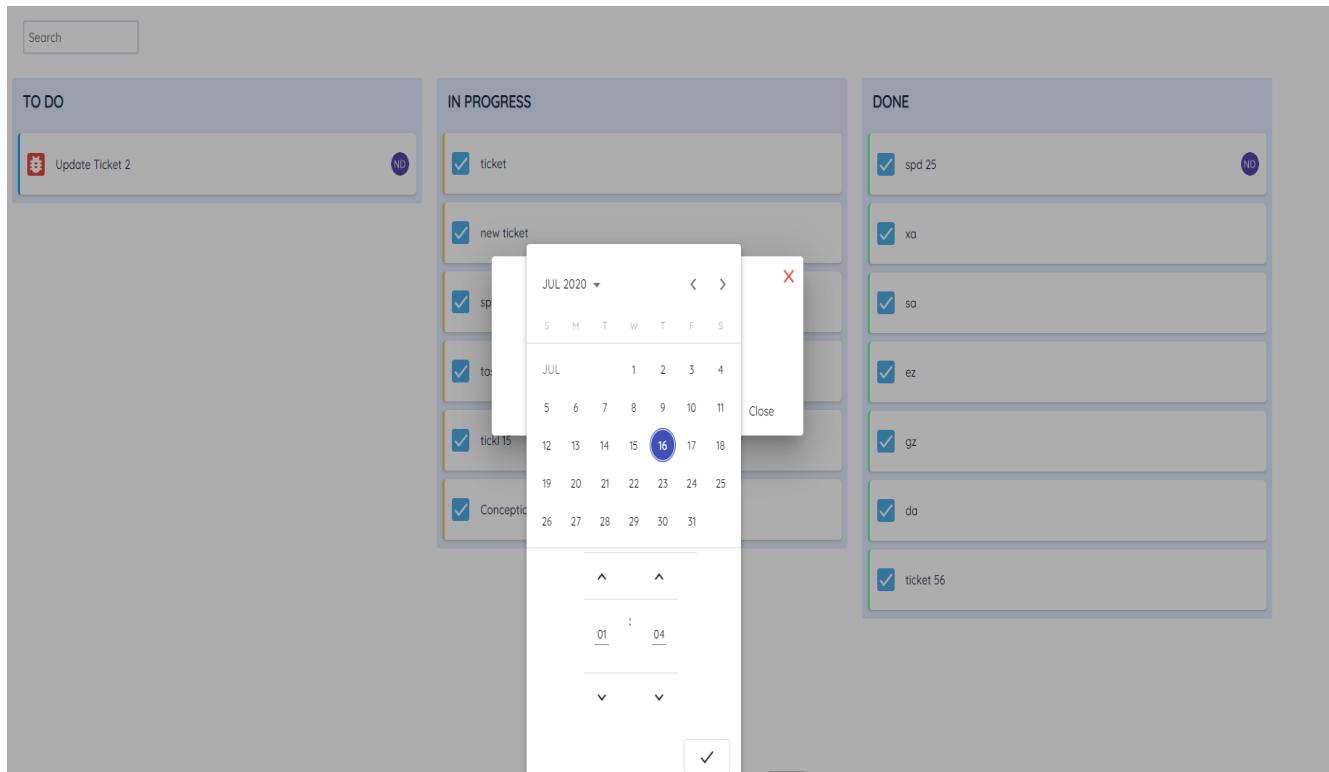


FIGURE 4.16 – Interface date début et date fin

### 3.2.3 Détail tâche

Pour consulter les détails de la tâche ou son commentaire il n'a qu'à cliquer sur la tâche et une interface s'affiche avec les détails.

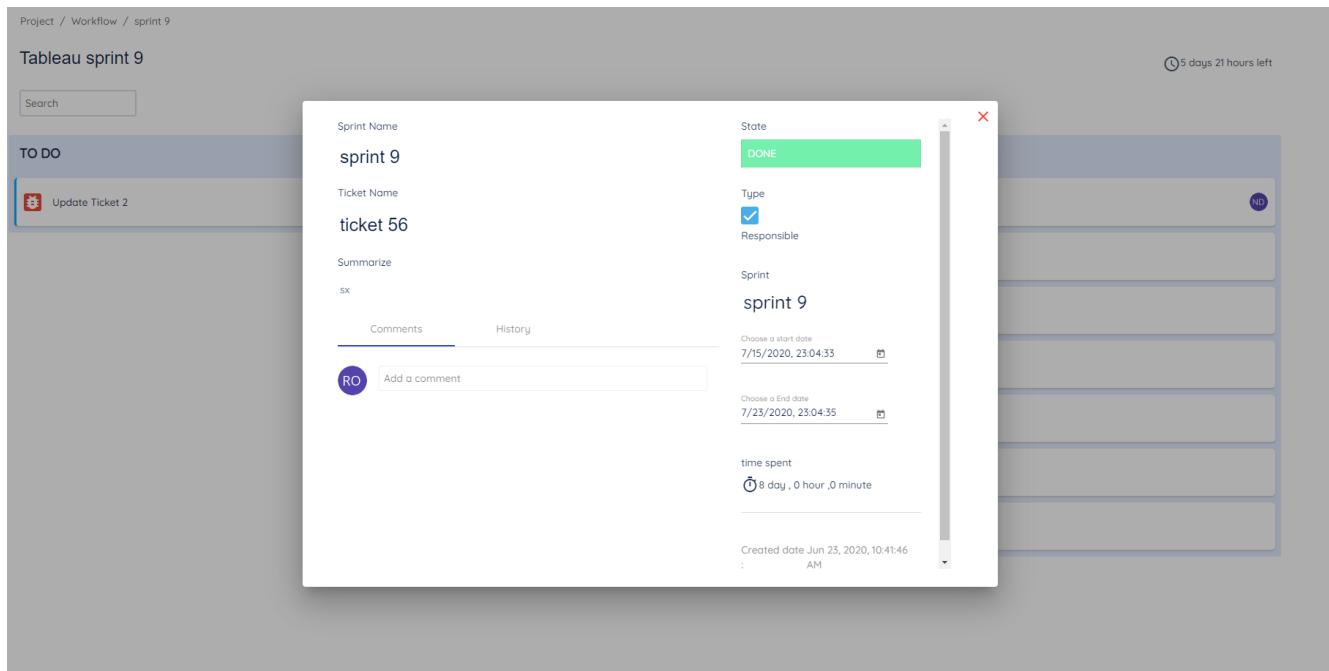


FIGURE 4.17 – Interface Détail tâche

### 3.3 Interface Back office

#### 3.3.1 Authentification

Afin d'accéder à l'administration de la plateforme, l'administrateur devra s'authentifier en saisissant son nom d'utilisateur son mot de passe.



FIGURE 4.18 – login Back office

#### 3.3.2 Gérer l'application

Lorsque l'administrateur se connecte, l'interface de back office s'affiche ; il peut donc gérer l'application.

ID	Name	Creationdate	Startdate	Estimatedenddate	Realenddate	Update date	
20	Application Proty	2020-07-14	2020-07-13	2020-07-30	Null	Null	<a href="#">Edit</a> <a href="#">Delete</a>
19	SPRINT	2020-03-18	2020-03-17	2020-03-30	Null	Null	<a href="#">Edit</a> <a href="#">Delete</a>
18	Back log Sprint	2020-03-18	2020-03-17	2020-03-30	Null	Null	<a href="#">Edit</a> <a href="#">Delete</a>
17	Back Log	2020-03-18	2020-03-17	2020-03-30	Null	Null	<a href="#">Edit</a> <a href="#">Delete</a>
9	Conception	2020-02-26	2020-02-25	2020-02-28	Null	Null	<a href="#">Edit</a> <a href="#">Delete</a>
2	pfe	2020-02-12	2020-02-20	2020-02-21	-0001-11-30	Null	<a href="#">Edit</a> <a href="#">Delete</a>
1	Work Flow	2020-02-25	2020-02-25	2020-02-29	2020-02-29	Null	<a href="#">Edit</a> <a href="#">Delete</a>

FIGURE 4.19 – Interface Gérer l'application

### 3.3.3 Ajouter un utilisateur

L'administrateur peut créer un nouveau utilisateur quand il aura rempli le formulaire ; un email sera envoyé à l'utilisateur contenant le user name et le mot de passe.

Add Use to DO-IT

Omarrabhi8@gmail.com

Omar

Rebhi

.....

ROLE\_DEVELOPER

Registre

FIGURE 4.20 – Interface Gérer l’application

## Conclusion

Dans ce chapitre, nous avons présenté l'environnement de développement de l'application et les différentes technologies utilisées. A la fin du chapitre nous avons présenté les interfaces de l'application.

# Conclusion générale

Notre stage dans DOT-IT- société de renommée - nous a permis non seulement de mettre en valeur et en pratique ce que nous avons appris mais de découvrir le milieu et l'ambiance professionnels. Ce stage m'a été très bénéfique, car nous avons beaucoup appris. En effet, les apports que j'ai tirés de cette expérience professionnelle peuvent être regroupés autour de deux éléments essentiels : les compétences acquises ainsi que le travail « en groupe ».

Nous avons donc, pu grâce aux expériences d'une équipe confirmée en matière de développement web, assimiler les différentes facettes de développement de logiciels et joindre le savoir au savoir-faire et enrichir, sans doute, mes connaissances techniques, avec en plus un auto-apprentissage actif et continu de l'environnement de développement basé sur JavaScript.

Ce projet de fin d'étude a été également une très belle occasion pour mieux comprendre et répondre à la question fondamentale :

Quelle démarche pour passer des besoins utilisateur au code de l'application ?

Cette période de 6 mois a été pour nous l'occasion de mettre en valeur une application qui permet d'approfondir nos investigations pour trouver d'autres solutions plus efficaces.

Tout au long de ce passage, j'ai pu développer une application ayant pour but de respecter les normes de méthodologie Agile et Scrum comme il a été analysé. Pour y parvenir, on a insisté sur les rapports Scrum pour pallier aux problèmes et minimiser les lacunes.

Nous avons franchi et décrit toutes les étapes nécessaires pour la réalisation du travail, allant de la contribution à la réalisation du cahier des charges, ensuite l'étude de l'existant, conception, pour finir avec la réalisation.

Comme perspective nous envisageons la mise en œuvre de paramétriser les projets avec droit

d'accès pour chaque utilisateurs, afficher les données en temps réels. Nous souhaitons également intégrer la discussion en temps réel entre le Scrum Master et les développeurs et le Scrum Master et Product Owner.

Finalement, nous espérons que notre travail dans l'ensemble sera à la hauteur de la confiance que la société DOT-IT nous a témoignée.

# Webographie

- [1] [HTTPS ://FR.WIKIPEDIA.ORG/WIKI/DIAGRAMME<sub>d</sub>'ecas<sub>d</sub>%27utilisation](https://fr.wikipedia.org/wiki/DIAGRAMME_d'ecas_d%27utilisation). Date de dernière consultation : 12/04/2020.
- [2] [HTTPS ://FR.WIKIPEDIA.ORG/WIKI/DIAGRAMME<sub>d</sub>'esquence](https://fr.wikipedia.org/wiki/DIAGRAMME_d'esquence). Date de dernière consultation : 12/04/2020.
- [3] [HTTPS ://FR.WIKIPEDIA.ORG/WIKI/MODLE-VUE-CONTRLEUR](https://fr.wikipedia.org/wiki/MODLE-VUE-CONTRLEUR). Date de dernière consultation : 07/04/2020.
- [4] [HTTPS ://OUTILSTICE.COM/2018/04/DRAW-IO-OUTIL-GRATUIT-POUR-DESSINER-DES-DIAGRAMMES-EN-LIGNE/](https://outilstice.com/2018/04/draw-io-outil-gratuit-pour-dessiner-des-diagrammes-en-ligne/). Date de dernière consultation : 12/04/2020.
- [5] [HTTPS ://WWW.APPVIZER.FR/MAGAZINE/OPERATIONS/GESTION-DE-PROJET/ASANA-VS-TRELLO](https://www.appvizer.fr/magazine/operations/gestion-de-projet/asana-vs-trello). Date de dernière consultation : 22/02/2020.
- [6] [HTTPS ://WWW.CACHEM.FR/NGROK-TUNNEL-APPLICATIONS-LOCALES](https://www.cachem.fr/ngrok-tunnel-applications-locales). Date de dernière consultation : 21/05/2020.
- [7] [HTTPS ://WWW.FUTURA-SCIENCES.COM/TECH/DEFINITIONS/INTERNET-MYSQL-4640/](https://www.futura-sciences.com/tech/definitions/internet-mysql-4640/). Date de dernière consultation : 27/04/2020.
- [8] [HTTPS ://WWW.JETBRAINS.COM/FR-FR/PHPSTORM/](https://www.jetbrains.com/fr-fr/phpstorm/). Date de dernière consultation : 18/06/2020.
- [9] [HTTPS ://WWW.NUTCACHE.COM/FR/](https://www.nutcache.com/fr/). Date de dernière consultation : 25/02/2020.
- [10] [HTTPS ://WWW.PURE-ILLUSION.COM/LEXIQUE/DEFINITION-DE-SYMFONY](https://www.pure-illusion.com/lexique/definition-de-symfony). Date de dernière consultation : 10/02/2020.
- [11] [HTTPS ://WWW.UML-SYSML.ORG/DIAGRAMMES-UML-ET-SYSML/DIAGRAMME-UML/DIAGRAMME-DE-CLASSE](https://www.uml-sysml.org/diagrammes-uml-et-sysml/diagramme-uml/diagramme-de-classe). Date de dernière consultation : 12/04/2020.