
DÉDICACES

Je dédie ce modeste travail à :

Mes chers parents, Je mets entre vos mains le fruit de votre amour,
de votre tendresse, de vos sacrifices, tout au long de mes études.

Ma chère mère Henia Bey Zakhamia,

Toutes les expressions ne peuvent exprimer mes sentiments d'amour et de respect envers vous.
Vous êtes pour moi la meilleure des mères, qui a consacré sa vie au bonheur et à la réussite
de ses enfants.

Vous m'avez donné la vie et vous avez guidé tous mes pas, vous êtes toujours à mes côtés pour
me motiver, me pousser à devenir ce que je suis aujourd'hui.

En témoignage de vos fatigues, vos efforts et vos sacrifices, je vous dédie ce travail pour
avouer mes reconnaissances et ma profonde estime.

Puisse Dieu vous accorder la santé et une longue vie.

Mon cher père Ali MejriSSI,

Aucune dédicace ne peut exprimer l'amour, l'estime et le respect que j'ai toujours pour vous.
Veuillez trouver dans ce modeste travail la conclusion de vos efforts, mon amour et ma
gratitude.

Puisse ce jour vous apporter toute la joie et la fierté que vous avez tant attendues.

Puisse Dieu vous accorder la bonne santé et vous garder comme toujours à nos côtés.

Mes frères,

Que cette dédicace vous présente mes profonds sentiments d'amour et de sincérité.
Que le bonheur du monde envahisse votre avenir et vous apporte une bonne santé.

A mes chères amies,

Sana, Rahma, Zayneb, Sourour, Imen, Salma,

En témoignage de l'amitié qui nous unit et des souvenirs de tous les moments que nous avons
passés ensemble, je vous dédie ce travail et je vous souhaite une vie pleine de santé et de
succès.

Mariem

REMERCIEMENTS

Au terme de ce travail,
Je voulais remercier profondément mon encadrant **Dr. Mohamed Kassab**.
Autant de phrases aussi expressives soient-elles ne sauraient montrer ma gratitude d'avoir accepté de m'encadrer tout au long de ce projet.
Votre compétence professionnelle, votre pertinence pédagogique et vos qualités humaines suscitent toute mon admiration. Vos conseils judicieux et votre soutien, quand le découragement me guettait, m'ont été amplement bénéfiques.
Que Dieu vous bénisse, et vous donne le bonheur et le succès. Et que ce travail soit le reflet du respect et l'estime que j'ai envers vous.

J'adresse également des remerciements particuliers à **Mme Safa Sakly**, mon superviseur de Redlean.
Vous m'avez honoré en me confiant ce travail et je suis très fière de l'avoir élaboré avec vous.
Je suis infiniment reconnaissante pour tout le temps que vous m'avez consacré.
Que ce travail soit le reflet du respect et l'estime que j'ai pour vous.

Mes sincères remerciements à **M. Nader Bahroun, Mme Noura Elabed, M. Mohamed Bouzidi** mes collègues à Redlean pour votre disponibilité, et la générosité de vos efforts ainsi que pour tout le temps que vous m'avez consacré.

TABLE DES MATIÈRES

Liste des Figures	iv
Liste des Tableaux	vi
Introduction Générale	1
I Cadre général du projet	3
1 Cadre du projet	4
1.1 Présentation de l'organisme d'accueil	4
1.2 Problématique	6
2 La méthodologie Agile Scrum	6
3 Etude de l'existant	7
4 Solution proposée	10
5 Méthodologie de gestion du projet adoptée	10
6 Chronogramme prévisionnel	13
II Analyse et spécification des besoins	15
1 Analyse des besoins	16
1.1 Les utilisateurs du système	16
1.2 Analyse des besoins fonctionnels	17
1.3 Analyse des besoins non fonctionnels	18
2 Spécification semi-formelle des besoins	19
2.1 Les cas d'utilisation générale	19
2.2 Les cas d'utilisation développée	20
2.2.1 Spécification de cas d'utilisation «Gérer des projets»	20
2.2.2 Spécification de cas d'utilisation «Gérer les tâches et les sous-tâches des projets»	24
3 Diagramme de séquence Acteur/Système	25
4 Capture des besoins techniques	29
4.1 Salesforce	29
4.2 Choix des outils de développement	30

III Conception	33
1 Conception préliminaire	34
1.1 Architecture générale	34
1.2 Patron de conception	35
2 Conception de la base de données	36
2.1 Modèle conceptuel de données (MCD)	37
3 Conception détaillée	39
3.1 Diagramme de classes	39
3.1.1 Diagramme de classe partie contrôleur client - vue	39
3.1.2 Diagramme de classe partie contrôleur client - contrôleur serveur	41
3.1.3 Diagramme de classe partie modèle - contrôleur serveur	42
3.2 Diagrammes de séquence de conception	43
3.2.1 Diagramme de séquence "Modifier un projet"	43
3.2.2 Diagramme de séquence "Supprimer un projet"	44
4 Conception graphique	45
IV Réalisation	48
1 Environnement logiciel de travail	49
2 Implémentation de la base des données	49
3 Mise en place du back office	51
4 Implémentation du front office	53
4.1 Implémentation du front end	53
4.2 Implémentation du back end	55
5 Exemples d'interfaces	56
6 Plan de réalisation	58
Conclusion Générale et Perspectives	60
bibliographie	i
Annexe 1 : Spécification et raffinement des cas d'utilisation	ii

LISTE DES FIGURES

I.1	Logo de Redlean	4
I.2	Clients de Redlean	5
I.3	La méthode SCRUM	7
I.4	Interface de l'application Mavenlink	8
I.5	Interface de l'application Clarizen	9
I.6	Interface de l'application Asana	9
I.7	Le cycle de développement 2TUP	13
I.8	Planning prévisionnel	13
II.1	Diagramme de cas d'utilisation générale	19
II.2	Diagramme de cas d'utilisation "Gérer les projets"	20
II.3	Diagramme de cas d'utilisation "Gérer les tâches et les sous-tâches des projets" .	24
II.4	Diagramme de séquence système du cas d'utilisation «Authentification »	25
II.5	Diagramme de séquence système du cas d'utilisation «Ajouter un projet»	26
II.6	Diagramme de séquence système du cas d'utilisation «Modifier un projet»	27
II.7	Diagramme de séquence système du cas d'utilisation «Supprimer un projet» . .	28
II.8	Fonctionnement d'Apex	31
III.1	Architecture générale de l'application	34
III.2	Architecture MVCC	35
III.3	Modèle conceptuel des données	38
III.4	Diagramme de classe partie contrôleur client - vue	40
III.5	Diagramme de classe partie contrôleur client - contrôleur serveur	41
III.6	Diagramme de classe partie modèle-contrôleur Serveur	42
III.7	Diagramme de séquence " Modifier un projet"	43
III.8	Diagramme de séquence " Supprimer un projet"	44
III.9	Maquette de l'interface d'authentification	45
III.10	Maquette de l'interface d'accueil de Scrum Master	45
III.11	Maquette de l'interface de gestion des projets	46
III.12	Maquette de l'interface de calendrier des projets	46
III.13	Maquette de l'interface des statistiques	47
IV.1	Interface de création de l'objet Projet	50

IV.2 Interface de la liste des objets personnalisés	51
IV.3 Interface d'une règle de Workflow	51
IV.4 Interface d'un exemple générateur du processus	52
IV.5 Interface de paramétrage de site	52
IV.6 Interface de la liste des composants	53
IV.7 Interface de composant «ListProjet.cmp»	54
IV.8 Interface de Contrôleur «ListProjetController.Js»	54
IV.9 Interface de Contrôleur «ListProjetHelper.Js»	55
IV.10 Interface de la liste des classes Apex	55
IV.11 Fonctionnement d'Apex	56
IV.12 Interface de Contrôleur Apex	56
IV.13 Interface d'authentification	57
IV.14 Interface des rapports et dashboard	57
IV.15 Interface de la liste des projets	58
IV.16 plan de réalisation	58
A.1 Diagramme de cas d'utilisation "Gérer les ressources"	ii
A.2 Diagramme de cas d'utilisation "Consulter et gérer les priorités"	ii
A.3 Diagramme de cas d'utilisation "Gérer les sprints des projets"	iii
A.4 Diagramme de cas d'utilisation "Gérer les user stories des projets"	iii

LISTE DES TABLEAUX

I.1	Les Pôles de Redlean	5
II.1	La liste des principaux utilisateurs système	17
II.2	Description de la fonctionnalité "Gérer les projets"	20
II.3	Description de la fonctionnalité "Ajouter un projet"	21
II.4	Description de la fonctionnalité "Modifier un projet"	21
II.5	Description de la fonctionnalité "Supprimer un projet"	22
II.6	Description de la fonctionnalité "Consulter un projet"	22
II.7	Description de la fonctionnalité "Chercher un projet"	23
II.8	Description de la fonctionnalité " Consulter planning des projets"	23
II.9	Description de la fonctionnalité "Marquer état d'avancement d'une tâche et sous-tâche"	25

INTRODUCTION GÉNÉRALE

LE suivi de projet est une étude qui puise ses racines que depuis le XIXe siècle, bien que sa forme moderne ne se soit enrichie qu'à partir du début des années 60. L'évolution fulgurante en informatique a induit un renouveau à la gestion d'entreprise en permettant d'automatiser toute une série de tâches que l'on exécutait déjà auparavant à la main.

Au début, les chefs de projets se sont contentés de programmes génériques tels que des tableurs et ont simplement transposé les documents papiers en fichiers informatiques plus ou moins élaborés. Ces programmes étaient souvent très complexes et peu ergonomiques et seul le chef de projets avait accès aux données.

L'arrivée puissante d'internet change radicalement la situation : les systèmes des entreprises sont maintenant, dans leur quasi-totalité, reliés à un réseau (que ce soit à l'intranet ou à l'internet). L'information peut circuler rapidement à l'intérieur de l'entreprise et même à l'extérieur de celle-ci. Alors, il est donc devenu nécessaire d'avoir des logiciels de suivi de projets qui soient capables de visualiser le travail en temps réel dans un espace organisationnel marqué par des changements rapides et perpétuels impliquant plusieurs services.

Dans ce cadre, la société Redlean, éditeur de logiciels, ces équipes travaillent selon la méthodologie de gestion de projets Scrum, elle s'est proposé d'exploiter une application de gestion de projets pour assurer plus d'efficacité de suivi de ces projets.

C'est dans ce contexte que s'inscrit notre projet de fin d'études dont l'objectif de développer une application de gestion de projets spécifique à Redlean.

Le présent rapport synthétise tout le travail que nous avons effectué, il s'articule autour de quatre grands chapitres désignés comme suit :

Dans le premier chapitre « cadre général », nous traiterons la présentation de l'entreprise d'accueil, le cadre de projet, la problématique, l'étude de l'existant. Ainsi que, nous expliquerons notre solution proposée et le choix de la méthodologie de gestion de notre projet.

Dans le second chapitre « Expression des besoins », nous illustrerons le lancement du projet dans lequel nous déterminerons et spécifierons les besoins fonctionnels, non fonctionnels et techniques.

Dans le troisième chapitre « Conception », nous aborderons la phase de conception où nous allons présenter les différents diagrammes d'UML.

Dans le dernier chapitre intitulé « Réalisation », nous décrirons les outils utilisés lors de la réalisation de notre projet et nous illustrerons les fonctions de l'application développée en les illustrant par des captures d'écran d'interfaces.

Nous clôturerons notre rapport par une conclusion générale récapitulant tout le travail que l'on a réalisé et les perspectives envisagées en raison d'améliorer les performances de l'application.

CHAPITRE I

CADRE GÉNÉRAL DU PROJET

Plan

1	Cadre du projet	4
1.1	Présentation de l'organisme d'accueil	4
1.2	Problématique	6
2	La méthodologie Agile Scrum	6
3	Etude de l'existant	7
4	Solution proposée	10
5	Méthodologie de gestion du projet adoptée	10
6	Chronogramme prévisionnel	13

I.1 Cadre du projet

L'étude du cadre du projet est une phase primordiale au démarrage de tous projets. Elle permet une compréhension précise du contexte et une analyse complète de l'existant pour fixer les objectifs à atteindre.

Le présent chapitre a pour but de définir le contexte général afin de situer le projet dans son environnement organisationnel et méthodologique. Tout d'abord, nous présenterons l'organisme d'accueil. Par la suite, nous décrirons brièvement la problématique à résoudre. Ensuite, nous allons définir la méthodologie agile scrum. Après, nous aborderons l'analyse de l'existant suivie par une proposition de notre solution. Enfin, nous terminerons par le choix de la méthodologie adoptée.

1 Cadre du projet

La réalisation de ce projet s'inscrit dans le cadre d'un projet de fin d'études en vue de l'obtention du diplôme national d'ingénieurs en génie informatique. Il consiste à la conception et l'implémentation d'une application de gestion des projets en Salesforce.

1.1 Présentation de l'organisme d'accueil

Le stage est réalisé au sein de la société Redlean qui est spécialisée dans le développement de solutions informatiques innovantes et parfaitement adaptées aux besoins de ses clients.



Figure I.1 – Logo de Redlean

RedLean est un cabinet de conseil informatique, fondé en 2016 et situé à Monastir. Comme son nom l'indique, il suit pour son évolution la méthode Lean qui est un système conçu pour générer la valeur ajoutée maximale au moindre coût et le plus rapidement possible.

Elle cherche à concevoir et à mettre en œuvre les meilleures solutions technologiques afin d'améliorer la productivité, la rentabilité et la réactivité des entreprises sur le marché, en fa-

I.1 Cadre du projet

vorisant l'excellence, la réussite sécurisée et en améliorant la valeur des équipes de ses clients[1].

Elle met l'innovation au cœur de son développement et intervient dans des domaines liés à la transformation digitale des grands groupes.

On investit notamment sur ces leviers technologiques :

Nouvelles Technologies	Data	Salesforce
- Développement mobile - Développement des Apis - Développement Web	- Data collection - Data Management - Data Intelligence - Big Data - Data Science	- Administration - Développement Apex - Intégration des données - Développement des composants Lightning

Tableau I.1 – Les Pôles de Redlean

Redlean possède plusieurs clients satisfaits dans le monde entier grâce à ses solutions de haute qualité et une expérience client unique.

La figure I.2 ci-dessous résume les clients de Redlean.



Figure I.2 – Clients de Redlean

1.2 Problématique

Redlean utilise la méthodologie Agile Scrum pour la gestion de ses projets. Pour des besoins organisationnels et afin d'améliorer le suivi de ses projets, Redlean a été confronté à la nécessité d'utiliser une application informatique pour la gestion de ses projets et de développer son propre outil de gestion des projets en Salesforce en adoptant la méthodologie Scrum.

Dans ce cadre, Redlean nous a proposé de développer une application interne pour la gestion des projets répondant exactement à ses besoins.

2 La méthodologie Agile Scrum

Cette méthode Agile vise à subdiviser l'organisation en petites équipes auto-organisées et transversales et réduire les difficultés telles que le manque de planification, l'évolution constante des besoins, le manque d'investissement des clients et le manque de communication commun à la plupart des projets. Elle nécessite quatre types de réunion:

- **Les réunions quotidiennes appelés « DSM » (Daily Scrum Meeting)** : chaque jour, toute l'équipe se réunit pendant 15 minutes environ pour répondre aux trois questions suivantes : qu'ai-je fait hier ? Que vais-je faire aujourd'hui ? Y a-t-il un obstacle gênant aujourd'hui ?
- **Les réunions de planifications** : toute l'équipe se réunit pour décider des fonctionnalités qui vont composer le sprint suivant et mettre à jour la liste générale.
- **Les réunions de découpage des tâches** : lors de cette réunion toute l'équipe participe au découpage des tâches du sprint et à l'estimation de la charge de chaque tâche.
- **Les réunions de rétrospectives** : à chaque fin de sprint, l'équipe fait le point sur ce qui a bien fonctionné et sur ce qui a moins bien fonctionné. Lors de cette réunion d'une durée de 15 à 30 minutes tous les employés sont invités et chacun parle en son nom, un vote de confiance est organisé pour décider des améliorations à apporter[2].

La figure I.3 illustre le cycle de vie de la méthode Scrum.

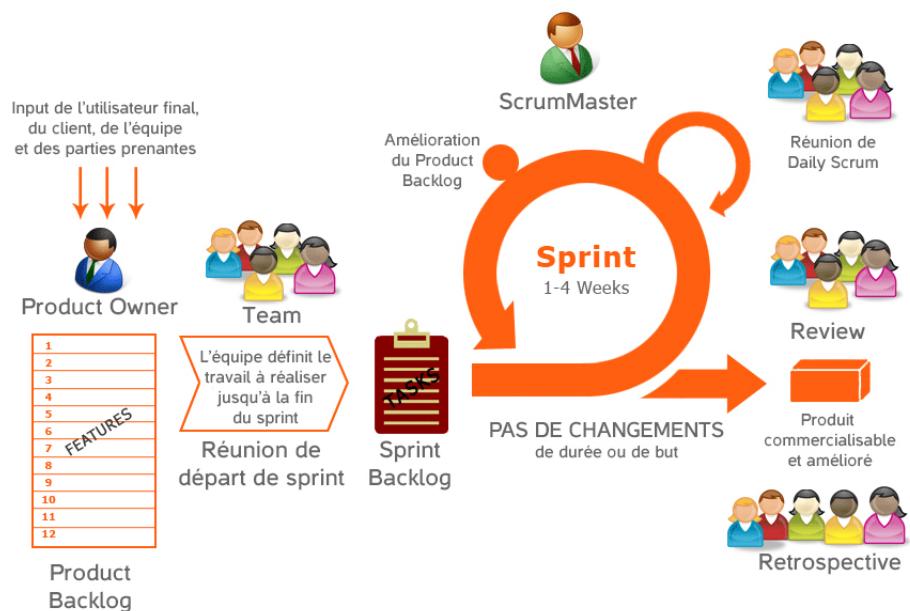


Figure I.3 – La méthode SCRUM
[3]

3 Etude de l'existant

Une étude de l'existant permet de se familiariser avec l'environnement étudié et de dégager les fonctionnalités d'un système actuel afin de pouvoir satisfaire les besoins du client.

Dans cette section, nous menons une présentation et une analyse des systèmes existants.

- **Mavenlink**

est un logiciel en ligne de gestion de projet moderne conçu pour rationaliser les processus opérationnels des entreprises de services. Il rassemble les équipes, les aider à planifier, s'organiser et à exécuter les différentes étapes d'un projet de manière disciplinée.

Il offre une gestion des portefeuilles de projets et la planification des listes de tâches afin de garantir le bon suivi de projet pendant toutes les étapes du cycle de vie. Il offre aussi une gestion des ressources très efficace en leur fournissant des outils de collaboration et de communication. Il dispose notamment d'un diagramme de GANTT pour visualiser l'organisation du projet avec les différents deadlines du calendrier. Tous ces éléments sont regroupés dans un seul hub afin de s'assurer que les responsables et les membres de l'équipe restent sur le même espace. Une autre fonction notable c'est que Mavenlink est

I.3 Etude de l'existant

une plate-forme spécialement conçue pour intégrer en toute transparence le programme aux solutions CRM.

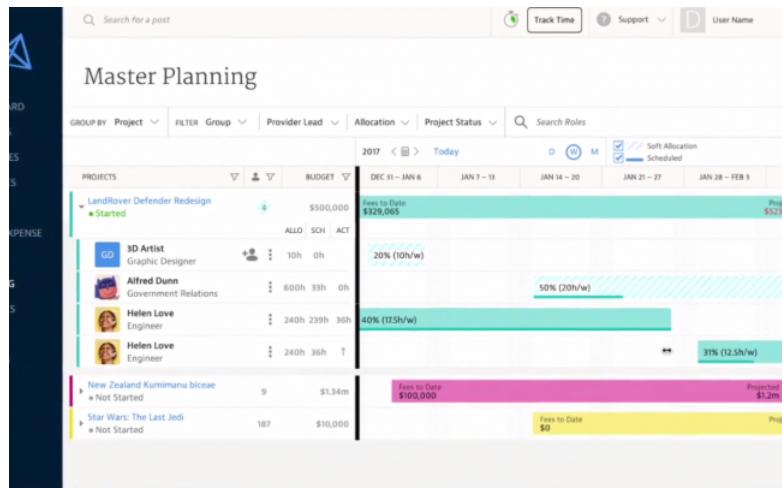


Figure I.4 – Interface de l'application Mavenlink
[4]

- **Clarizen**

est un logiciel de gestion de projet. Il s'adresse plus spécifiquement aux PME. Cette solution valorise les équipes en connectant les conversations, les tâches et les projets dans une seule et même application, pour simplifier leur travail et à atteindre leurs objectifs. Il s'agit d'une solution de gestion collaborative du travail d'entreprise qui permet aux utilisateurs de créer des groupes de discussion à tous les niveaux grâce à l'automatisation du workflow. Il expose des fonctionnalités de planification des capacités et de gestion des ressources qui fournissent des renseignements en temps réel sur les ressources disponibles et leur charge de travail actuelle. Il dispose de rapports et de tableaux de bord prédéfinis et personnalisables. En plus, il offre la gestion des tâches et des projets dont les chefs de projet surveillent l'état d'avancement de toutes les tâches et les réaffecter à des ressources selon les besoins. Il prend en charge les différentes méthodologies de travail qui permettent une efficacité accrue, fournissant une collaboration ciblée afin que les gens répondent plus rapidement.

I.3 Etude de l'existant

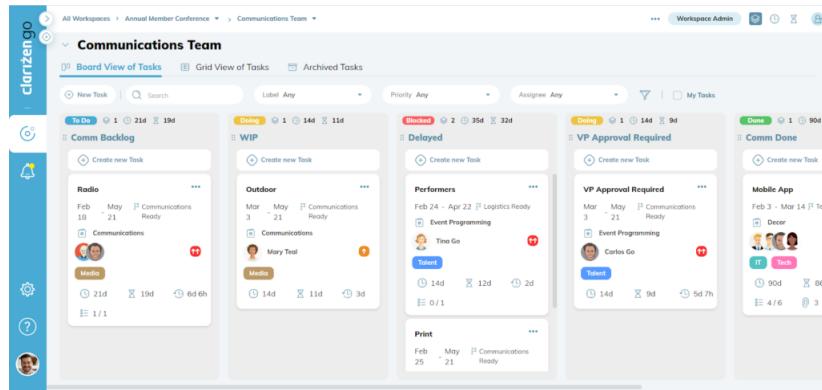


Figure I.5 – Interface de l'application Clarizen [5]

- **Asana** est une application de gestion de projet sophistiqué qui peut également être utilisée pour gérer des projets. Asana est idéal pour les grandes organisations où plusieurs équipes travaillent simultanément sur plusieurs projets complexes. Elle dispose d'une fonction de tableaux basée sur le système Kanban qui permet de visualiser le travail avec une progression visuelle. En plus, elle a son propre système de gestion de dépendance des tâches. Une autre caractéristique clé est la chronologie (Time-line) pour montrer l'assemblage des différentes parties de projet, visualiser la progression et respecter les échéances. Elle possède aussi une fonction de Calendrier qui facilite la visualisation de travail afin de repérer les conflits d'horaire. En plus, elle contrôle les accès grâce aux réglages de confidentialité pour assurer la sécurité.

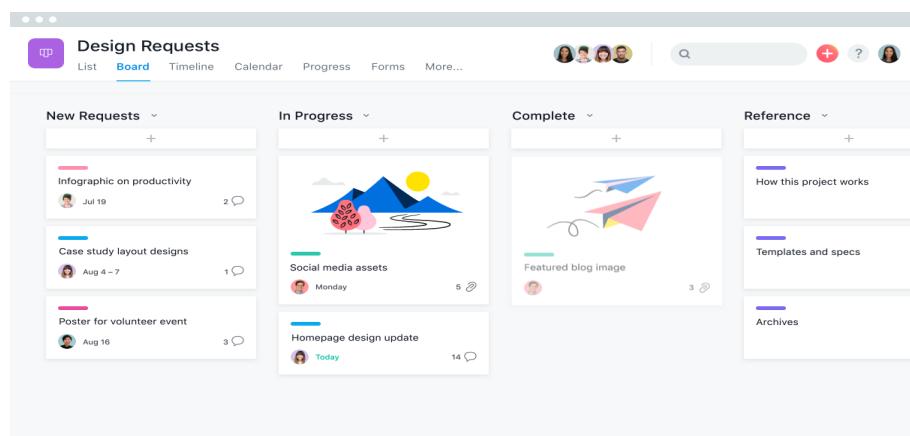


Figure I.6 – Interface de l'application Asana [6]

4 Solution proposée

À travers l'étude de l'existant, on a constaté que l'adoption de solutions de gestion de projet accélère l'achèvement des tâches et la livraison des projets.

Ces applications nous donnent une plus grande visibilité sur les futures fonctionnalités de notre solution et nous facilitent l'identification des besoins nécessaires pour gérer un projet.

Alors, notre solution consiste à réaliser un système de gestion des projets avec la méthodologie Scrum depuis la collecte des exigences jusqu'à la publication du logiciel de travail. Elle dispose d'une vue d'ensemble en terme de suivi des progrès et la centralisation de l'information pour mieux collaborer et faire réussir le projet. Elle offre une planification simplifiée et un suivi de temps précis. Cette visibilité aide à mieux visualiser l'avancement de projet et évaluer l'efficacité des ressources.

Cette solution fournit diverses fonctionnalités :

- Gestion de l'avancement et le suivi des projets
- La prise en charge du sprint et users stories
- Gestion efficace des tâches et sous tâches à l'aide d'un tableau Kanban
- Gestion des délais
- Analyse avec une panoplie des tableaux de bord
- Gestion et affectation des ressources internes
- Assistance et accélération des opérations de mise à jour

5 Méthodologie de gestion du projet adoptée

La gestion de projets joue un rôle crucial dans l'exécution, et l'accomplissement des objectifs. Elle aide à organiser le projet de manière structurée et rationalisée dans le but de parvenir au résultat le plus performant et ce, dans un climat d'efficacité et de convivialité. Souvent, plusieurs méthodologies de gestion de projet existent, mais ne sont pas toutes efficaces pour différentes tâches.

I.5 Méthodologie de gestion du projet adoptée

Alors, nous avons choisi la méthode 2TUP afin de concevoir et développer un système performant qui satisfait les exigences du client en moindre coût et délai. En effet, le processus 2TUP est la meilleure façon d'aborder un problème, c'est de s'y attaquer de front. C'est ce que propose le 2TUP en faisant une place à part entière à la technologie dans le processus de développement.

Ce processus se base lui-même sur l'UP (UnifiedProcess) Processus Unifié qui est devenu un standard général réunissant les meilleures pratiques de développement. Cette méthode ne se base aucunement sur un processus linéaire mais bien, sur un développement itératif et incrémental et adapté à une large classe de systèmes logiciels, dans différents domaines d'application, différentes entreprises, différents niveaux de compétences et différentes tailles de projets. Elle repose sur l'approche orientée-objet. Il utilise UML comme langage de modélisation visuelle, afin d'assurer la production d'un logiciel de bonne qualité pour la satisfaction des besoins de l'utilisateur.

« 2 Track » signifie littéralement que le processus suit deux chemins. Il s'agit des « chemins fonctionnels » et « d'architecture technique », qui correspondent aux deux axes de changement imposés au système d'information.

La méthodologie 2TUP propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels.

Le processus en Y s'articule autour de 3 phases :

- **Une branche technique**
- **Une branche fonctionnelle**
- **Une phase de réalisation (conception et développement)**

* **La branche gauche (fonctionnelle)**

Elle capitalise la connaissance du métier de l'entreprise. Elle constitue généralement un investissement pour le moyen et le long terme. Les fonctions du système d'information sont en effet indépendantes des technologies utilisées.

Cette branche comporte les étapes suivantes :

- La capture des besoins fonctionnels qui produit un modèle des besoins focalisés sur le métier des utilisateurs.
- L'analyse consiste à étudier précisément les spécifications fonctionnelles.

I.5 Méthodologie de gestion du projet adoptée

* **La branche droite (technique)**

Elle capitalise un savoir-faire technique. Elle constitue un investissement pour le court et le moyen terme. Les techniques développées pour le système peuvent l'être en effet indépendamment des fonctions à réaliser.

Cette branche comporte les étapes suivantes:

- La capture des besoins techniques qui permet de définir le modèle d'analyse technique
- La conception générique qui permet de définir les composants nécessaires à la construction de l'architecture technique

* **La branche du milieu (conception-réalisation)**

A l'issue des évolutions du modèle fonctionnel et de l'architecture technique, la réalisation du système consiste à fusionner les résultats des 2 branches. Cette fusion conduit à l'obtention d'un processus en forme de Y.

Cette branche inclut les étapes suivantes:

- **La conception préliminaire** est une étape qui permet de produire le modèle de conception système. Ce dernier organise le système en composants, délivrant les services techniques et fonctionnels. Ce modèle regroupe donc les informations de ces branches techniques et fonctionnelles.
- **La conception détaillée** permet d'étudier comment réaliser chaque composant. Cette étape produit le modèle de conception des composants.
- **Le codage** consiste à effectuer la production des composants et les tests des unités de code au fur et à mesure de leur réalisation.
- **La recette** consiste à valider les fonctionnalités du système développé

La figure I.7 détaille les étapes de développement des trois branches du processus 2TUP.

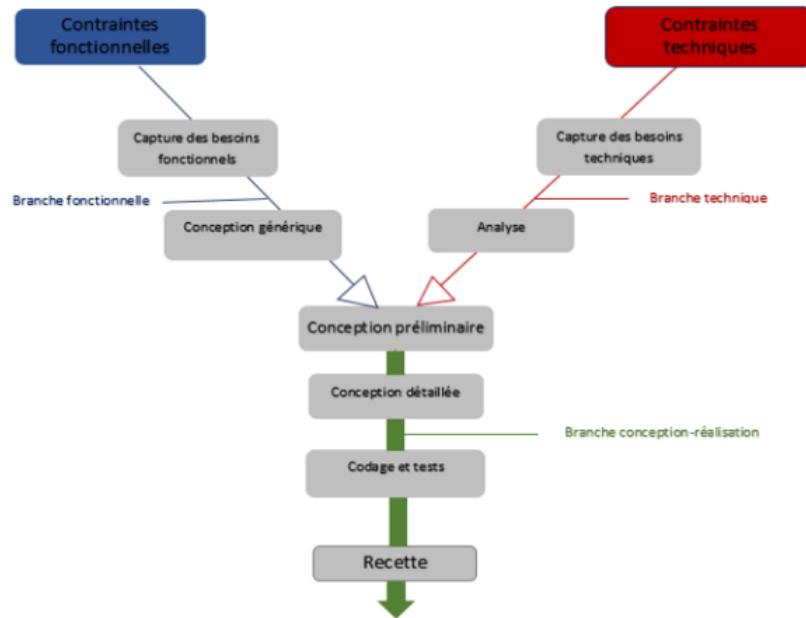


Figure I.7 – Le cycle de développement 2TUP

[7]

6 Chronogramme prévisionnel

Dans un souci d'organisation et afin de mener à bien le projet, nous avons établi un chronogramme qui détaille la répartition des tâches illustré par la figure I.8.

	2020											
	JAN	FEV	MAR	AVR	MAI	JUN	JUL	AOUT	SEP	OCT	NOV	DEC
Analyse et spécification des besoins												
Conception												
Implémentation												
Rédaction du rapport												

Figure I.8 – Planning prévisionnel

Conclusion

Dans ce premier chapitre, nous avons défini le champ d'étude. Ensuite, nous avons placé le cadre du projet avant de décider du contexte de la solution proposée après être passé par l'étude de l'existant, suivi des solutions existantes. Enfin, nous avons énoncé la méthodologie et le formalisme du travail adopté. Dans le chapitre suivant, nous entamerons la phase d'expression des besoins du système à développer.

CHAPITRE II

ANALYSE ET SPÉCIFICATION DES BESOINS

Plan

1	Analyse des besoins	16
1.1	Les utilisateurs du système	16
1.2	Analyse des besoins fonctionnels	17
1.3	Analyse des besoins non fonctionnels	18
2	Spécification semi-formelle des besoins	19
2.1	Les cas d'utilisation générale	19
2.2	Les cas d'utilisation développée	20
3	Diagramme de séquence Acteur/Système	25
4	Capture des besoins techniques	29
4.1	Salesforce	29
4.2	Choix des outils de développement	30

L'expression des besoins est une étape essentielle dans la réalisation d'une application. Elle est conçue pour mettre en relief les différentes fonctionnalités attendues du système et la fixation des buts à atteindre.

Dans ce chapitre, nous commencerons, en premier lieu, par identifier l'ensemble des utilisateurs du système ainsi que des besoins fonctionnels qui sont traduits en termes de cas d'utilisation. Nous enchaînerons ensuite par une présentation des besoins non fonctionnels auxquels devraient répondre notre solution. Puis, nous entamerons le raffinement et la spécification de chaque cas d'utilisation. Nous présenterons également quelques diagrammes de séquence système. Enfin, nous exposerons les différents besoins techniques de notre application.

1 Analyse des besoins

Dans cette partie, nous allons passer à l'identification des différents besoins fonctionnels et non fonctionnels qui présente l'une des étapes majeures sur lesquelles se forge notre projet.

1.1 Les utilisateurs du système

Tout système interactif doit assurer et faciliter l'interaction avec ses utilisateurs (utilisateur humain ou non).

Un tableau récapitulatif de la liste des principaux utilisateurs système de notre application qui se divisent en quatre catégories ainsi que l'explication du rôle de chacun d'entre eux :

Utilisateur	Rôle
Administrateur	C'est le superviseur qui a pour rôle de contrôler et rectifier le site web pour assurer le bon fonctionnement du système.
Scrum master	C'est un utilisateur qui garantit le processus de projet en mesurant les progrès, supprimant les obstacles et dirigeant les rencontres de l'équipe.
Product Owner	C'est le représentant des utilisateurs finaux et le porteur de vision de produit à réaliser dont il communique leurs exigences avec l'équipe de développement et définit les priorités pour assurer la réussite du projet.

Utilisateur	Rôle
Developer	Il transforme les besoins exprimés par le Product Owner en fonctionnalités utilisables.

Tableau II.1 – La liste des principaux utilisateurs système

1.2 Analyse des besoins fonctionnels

L’analyse des besoins fonctionnels consiste à traduire les objectifs du projet en un ensemble de fonctionnalités cibles par l’outil à réaliser. Ceci procurera une compréhension plus approfondie des tâches à mettre en œuvre.

Dans cette partie, nous allons exposer l’ensemble des besoins fonctionnels de notre application à développer.

- * **Gestion des ressources** : chaque Administrateur a la possibilité de consulter, ajouter, modifier et supprimer une ressource.
- * **Affecter ressources aux projets**
- * **Créer un nouveau projet**
- * **Gérer des projets** : englobe les opérations classiques qui peuvent être exécutées sur un projet tel que la modification, la suppression, la recherche, la consultation et l’archivage.
- * **Gérer des sprints des projets** : le système doit permettre de gérer les sprints auxquels appartiennent les users stories avec les fonctionnalités d’ajout, modification,consultation, suppression et recherche.
- * **Gérer des users stories des projets** : Une ressource a le droit d’accès de :
 - Ajouter de nouveaux users stories
 - Modifier des users stories
 - Supprimer des users stories
 - Consulter des users stories
 - Chercher des users stories
- * **Traiter une user story** : Dès la création d’une user story,le product owner a la possibilité de traiter chaque user story.

- * **Gérer des tâches et sous-tâches des projets** : le système offre la possibilité de gérer les tâches et sous-tâches de chaque user story.
- * **Consultation et gestion des priorités** : Ordonner les sprints, users stories, tâches et sous tâches pour mieux réaliser les objectifs et optimiser la valeur du travail effectué par les développeurs.
- * **Consulter statistiques** : des tableaux de bord seront affichés afin de visualiser les statistiques.

1.3 Analyse des besoins non fonctionnels

Outre les besoins fonctionnels précédemment établis, notre application devra respecter la liste des besoins non fonctionnels qui suit :

- **La simplicité**

Plus l'application est simple dans l'apprentissage et dans l'utilisation, plus ses fonctionnalités sont utilisées.

- **Fiabilité**

Notre application doit fonctionner de façon cohérente sans erreurs.

- **Ergonomie**

L'application doit offrir une interface conviviale et ergonomique exploitable par l'utilisateur

- **Maintenabilité**

Les différents modules de l'application doivent être faciles à maintenir pour une détection rapide des éventuelles anomalies.

- **Extensibilité**

L'architecture de l'application permettra l'évolution et la maintenance (ajout ou suppression ou mise à jour) au niveau de ses différents modules d'une manière flexible.

- **Sécurité**

L'utilisateur doit être connecté s'il veut accéder à l'une des fonctionnalités ajoutées.

2 Spécification semi-formelle des besoins

2.1 Les cas d'utilisation générale

Le diagramme de cas d'utilisation générale permet d'englober les principaux acteurs qui sont les utilisateurs du système et la visualisation des besoins fonctionnels de chaque utilisateur, afin de produire un résultat observable et intéressant pour un acteur particulier.

Une représentation graphique des différents cas d'utilisation de notre application est assurée par ce diagramme de cas d'utilisation général illustré dans la figure II.1

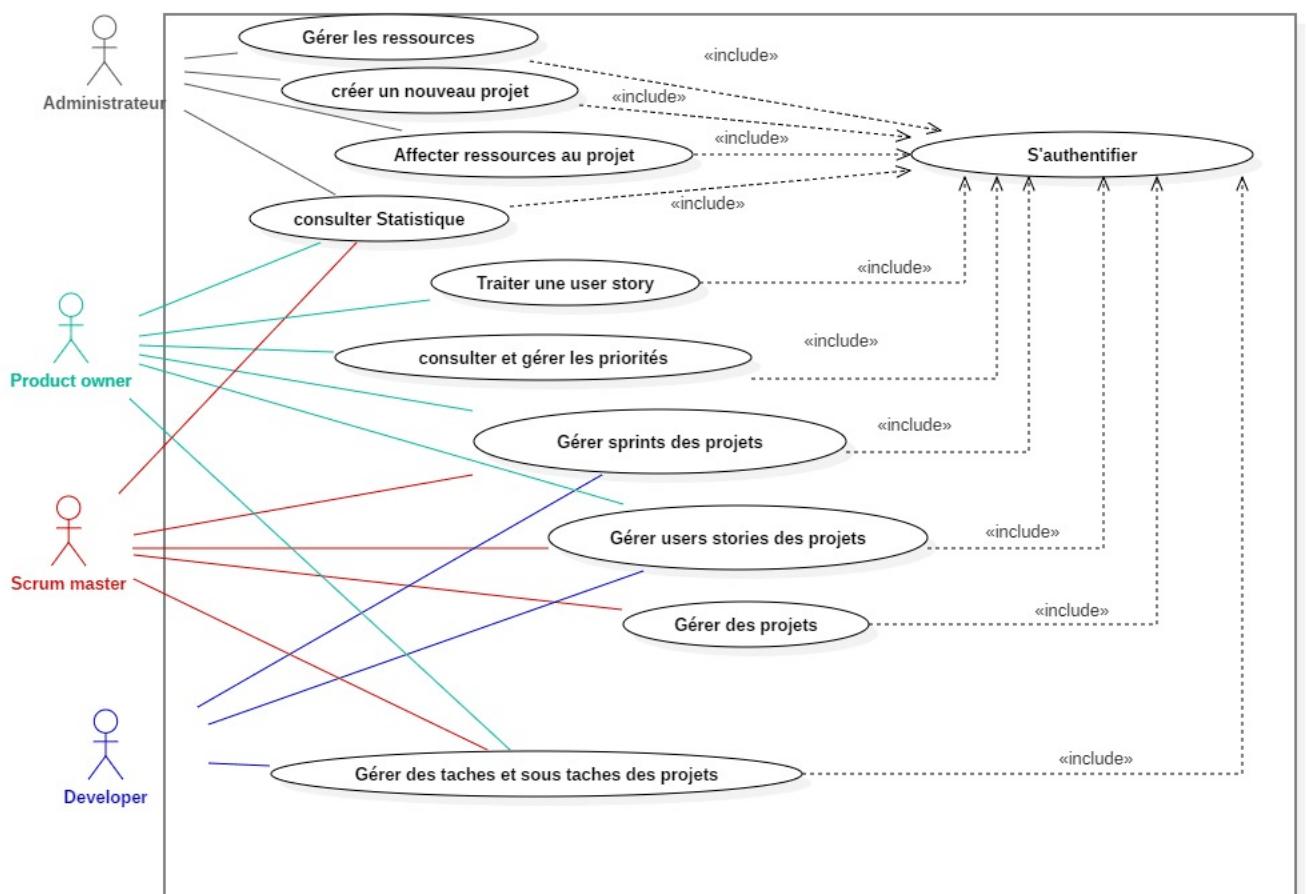


Figure II.1 – Diagramme de cas d'utilisation générale

II.2 Spécification semi-formelle des besoins

2.2 Les cas d'utilisation développée

Dans cette partie, on va présenter les différents diagrammes de cas d'utilisation en décrivant le raffinement de chaque cas en détail afin de mieux visualiser notre application.

2.2.1 Spécification de cas d'utilisation «Gérer des projets»

- Diagramme raffiné

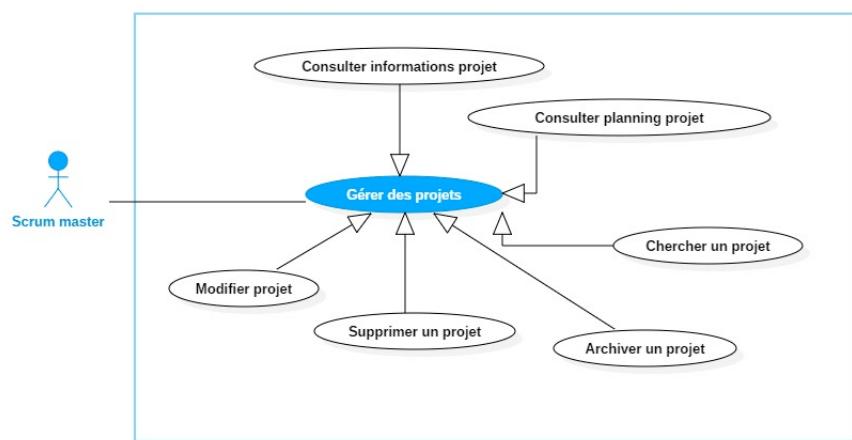


Figure II.2 – Diagramme de cas d'utilisation "Gérer les projets"

- Spécification

* Spécification et raffinement des cas d'utilisation « Gérer projets »

Cas d'utilisation	Gérer les projets
Acteurs	Scrum Master
Préconditions	Le Scrum master doit s'authentifier
Postconditions	La liste des projets est affichée
Scénario de base	1- Le système affiche l'interface d'accueil 2- Le Scrum master clique sur le sous menu « Gérer Projets » 3- Le système affiche une page contenant la liste des projets existants

Tableau II.2 – Description de la fonctionnalité "Gérer les projets"

II.2 Spécification semi-formelle des besoins

* Spécification et raffinement des cas d'utilisation « Ajouter un projet »

Cas d'utilisation	Ajouter un projet
Acteurs	Administrateur
Préconditions	L'administrateur s'authentifie, il choisit l'opération d'ajout
Postconditions	Le projet est ajouté
Scénario de base	1- L'administrateur clique sur le bouton « Ajouter un nouveau projet » 2- Le système affiche le formulaire à remplir 3- L'administrateur introduit les informations nécessaires 4- L'administrateur valide l'ajout en cliquant sur le bouton « Enregistrer » 5- Le système affiche un message de succès
Scénario alternatif	3- L'administrateur saisit des données manquantes ou erronées a. Le système affiche un message d'erreur b. Le scénario reprend à partir de 2

Tableau II.3 – Description de la fonctionnalité "Ajouter un projet"

* Spécification et raffinement des cas d'utilisation « Modifier un projet »

Cas d'utilisation	Modifier un projet
Acteurs	Scrum Master
Préconditions	Le Scrum master s'authentifie, il choisit l'opération de modification
Postconditions	Le projet est modifié
Scénario de base	1- Scrum master choisit le projet à modifier et clique sur le bouton « modifier » 2- Le système affiche l'interface de modification 3- Scrum master saisit les données désirées 4- Scrum master confirme la modification 5- Le système retourne à l'affichage des projets
Scénario alternatif	3- Si un champ est incorrect ou non valide a. Le système affiche un message d'erreur b. Le scénario reprend à partir de 2

Tableau II.4 – Description de la fonctionnalité "Modifier un projet"

II.2 Spécification semi-formelle des besoins

* Spécification et raffinement des cas d'utilisation « Supprimer un projet »

Cas d'utilisation	Supprimer un projet
Acteurs	Scrum master
Préconditions	Scrum master s'authentifie, il choisit l'opération de suppression
Postconditions	Le projet est supprimé
Scénario de base	1- Scrum master choisit un projet 2- Scrum master appuie sur le bouton « Supprimer » 3- Le système affiche un message de confirmation et retire le projet 4- Le système indique que la suppression s'est déroulée avec succès
Scénario alternatif	3- Si le Scrum master ne confirme pas : a. Le système annule l'action de suppression

Tableau II.5 – Description de la fonctionnalité "Supprimer un projet"

* Spécification et raffinement des cas d'utilisation « Consulter un projet »

Cas d'utilisation	Consulter un projet
Acteurs	Scrum Master
Préconditions	Le Scrum master s'authentifie, il choisit l'opération de consultation
Postconditions	Le projet est consulté
Scénario de base	1- Le Scrum Master sélectionne un projet 2- Le Scrum Master clique sur le bouton «détail» 2- Le système affiche les détails du projet

Tableau II.6 – Description de la fonctionnalité "Consulter un projet"

II.2 Spécification semi-formelle des besoins

* Spécification et raffinement des cas d'utilisation « Chercher un projet»

Cas d'utilisation	Chercher un projet
Acteurs	Scrum Master
Préconditions	Le Scrum master s'authentifie, il choisit l'opération de recherche
Scénario de base	1- Le Scrum master saisit le nom de projet 2- Le système affiche le projet correspondant
Scénario alternatif	1- Le Scrum master entre un nom invalide a. Le système n'affiche pas le projet souhaité

Tableau II.7 – Description de la fonctionnalité "Chercher un projet"

* Spécification et raffinement des cas d'utilisation « Consulter planning des projets»

Cas d'utilisation	Consulter planning des projets
Acteurs	Scrum Master
Préconditions	Le Scrum master doit s'authentifier
Postcondition	Le calendrier du planning des projets est affiché
Scénario de base	1- Le Scrum master demande l'affichage du planning 2- Le système affiche le planning de tous les projets sous forme d'un calendrier
Scénario alternatif	2- S'il n'y a pas de projet a. Le système affiche un calendrier vide

Tableau II.8 – Description de la fonctionnalité " Consulter planning des projets"

II.2 Spécification semi-formelle des besoins

2.2.2 Spécification de cas d'utilisation «Gérer les tâches et les sous-tâches des projets»

- Diagramme raffiné

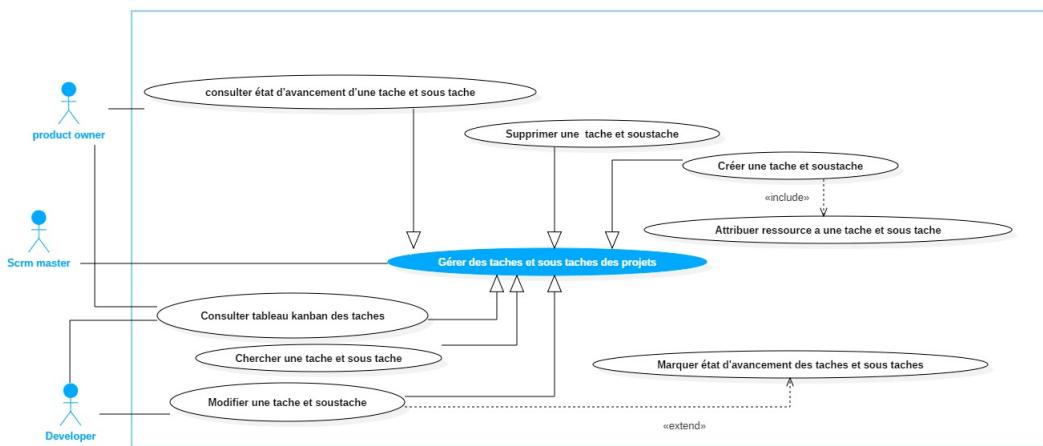


Figure II.3 – Diagramme de cas d'utilisation "Gérer les tâches et les sous-tâches des projets"

- Spécification

* Spécification et raffinement des cas d'utilisation « Marquer avancement d'une tâche et sous-tâche »

Cas d'utilisation	Marquer état d'avancement d'une tâche et sous-tâche
Acteurs	Developer
Préconditions	Le Developer doit s'authentifier
Postconditions	L'avancement est marqué
Scénario de base	1- Le système affiche la liste des tâches 2- Le Developer clique sur le bouton « Temps » 3- Le système affiche un formulaire 4- Le Developer saisit les informations nécessaires correctement et clique sur le bouton « Enregister»

II.3 Diagramme de séquence Acteur/Système

Cas d'utilisation	Marquer état d'avancement d'une tâche et sous-tâche
Scénario alternatif	4- Si un champ est incorrect ou non valide a. Le système affiche un message d'erreur b. Le scénario reprend à partir de 4

Tableau II.9 – Description de la fonctionnalité "Marquer état d'avancement d'une tâche et sous-tâche"

Les autres raffinements sont annexés au présent rapport (Annexe Spécification et raffinement des cas d'utilisation).

3 Diagramme de séquence Acteur/Système

Le diagramme de séquence acteur/système c'est un diagramme en boite noir qui montre les interactions entre les acteurs et le système. Ils servent aussi à illustrer un cas d'utilisation.

- **Authentification**

La figure II.4 présente le diagramme de séquence système qui décrit le cas d'utilisation «Authentification» et les interactions entre le système et une ressource.

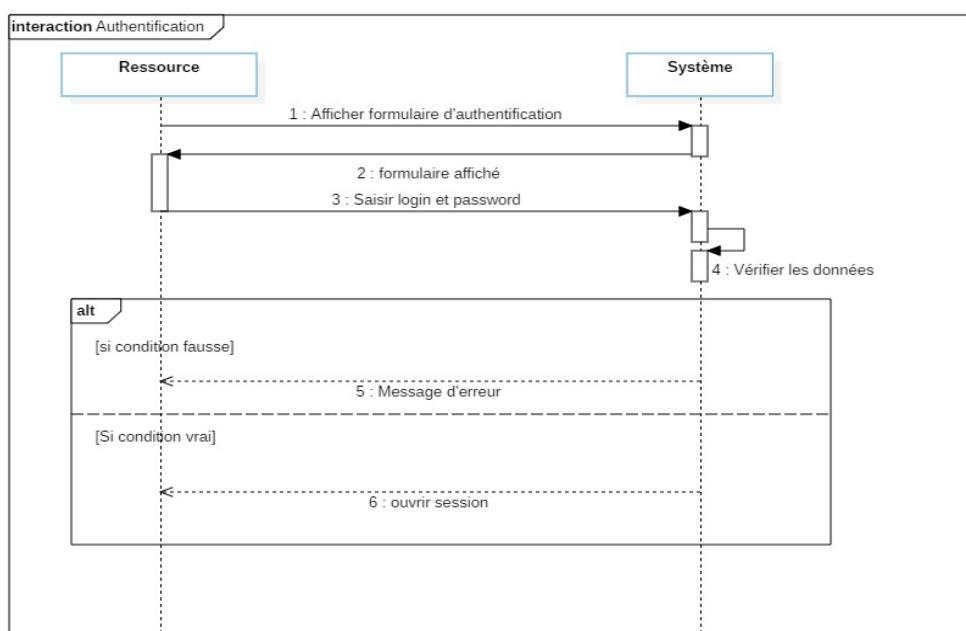


Figure II.4 – Diagramme de séquence système du cas d'utilisation «Authentification »

II.3 Diagramme de séquence Acteur/Système

- Ajouter un projet

La figure II.5 ci-dessous décrit le scénario d'ajout d'un projet.

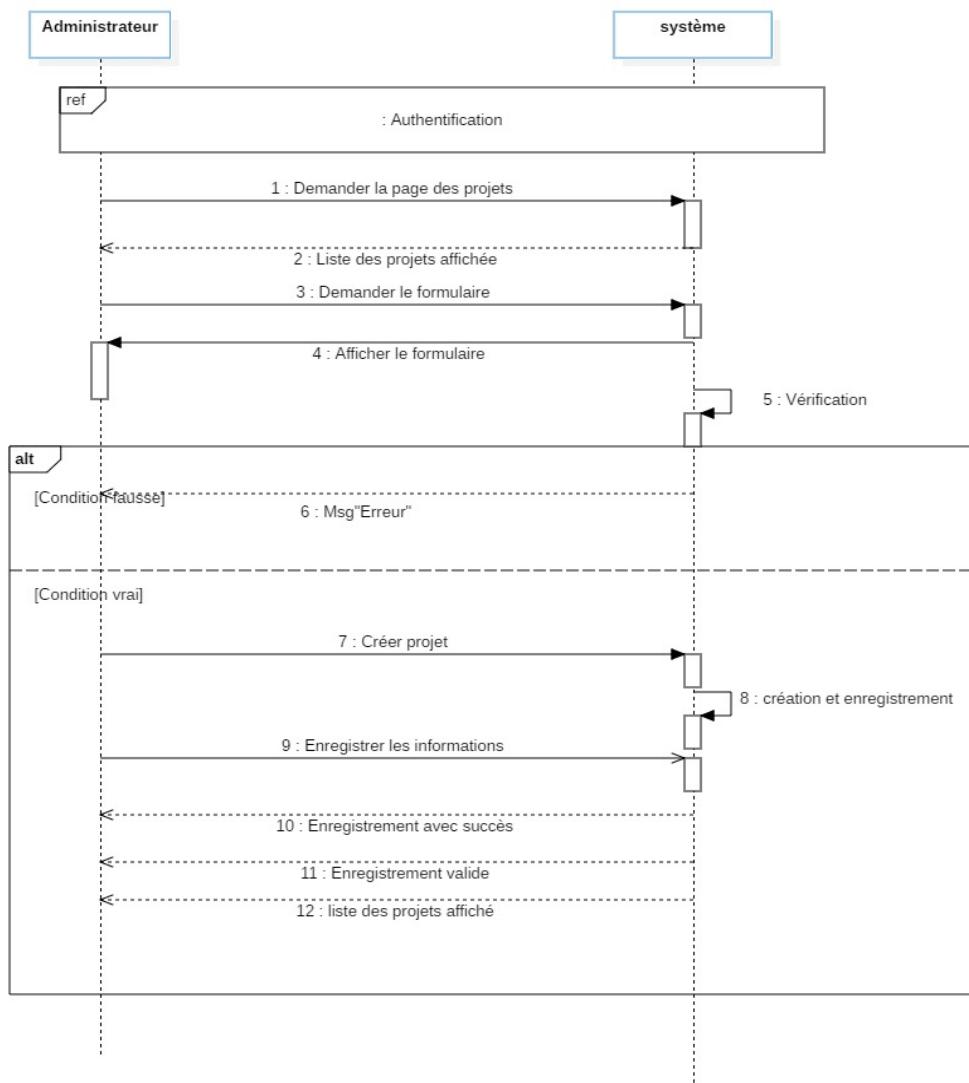


Figure II.5 – Diagramme de séquence système du cas d'utilisation «Ajouter un projet»

II.3 Diagramme de séquence Acteur/Système

- Modifier un projet

La figure II.6 représente le scénario de modification d'un projet.

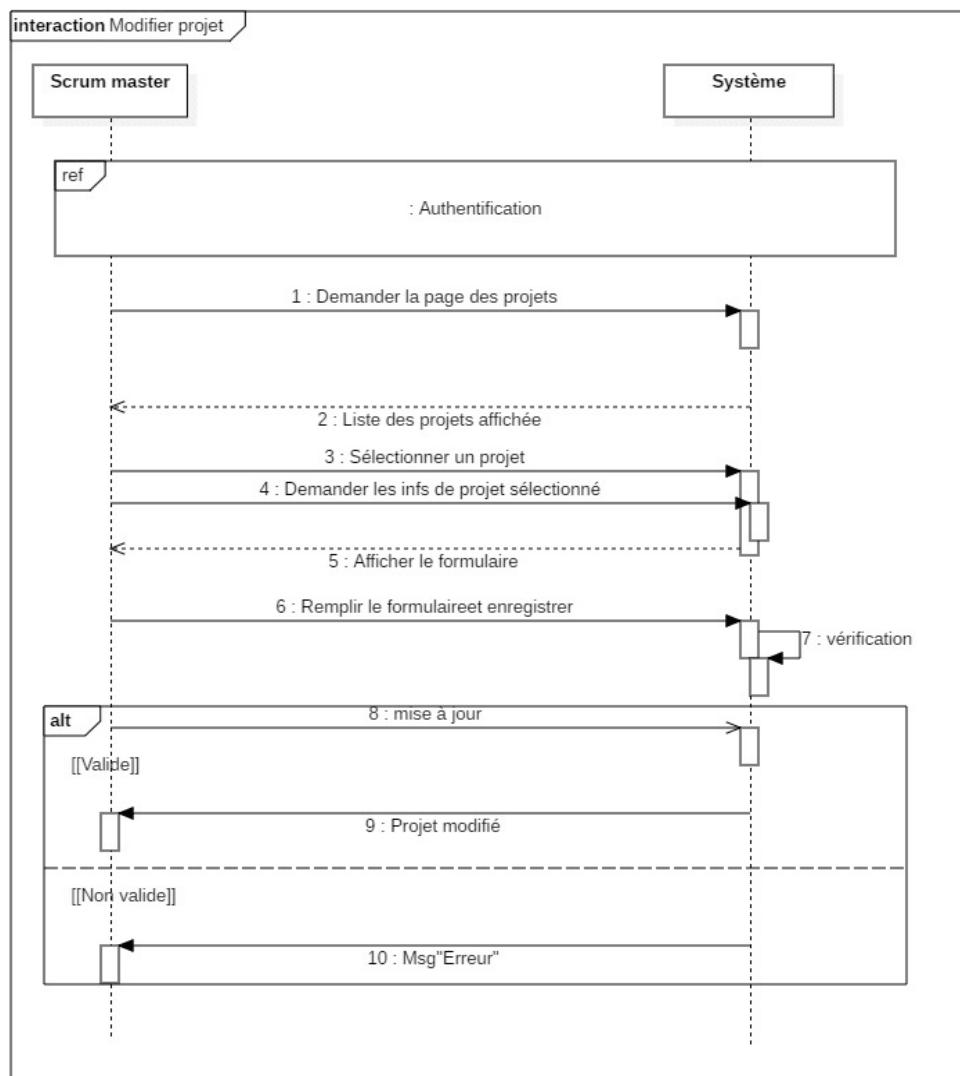


Figure II.6 – Diagramme de séquence système du cas d'utilisation «Modifier un projet»

II.3 Diagramme de séquence Acteur/Système

- Supprimer un projet

La figure II.7 ci-dessous illustre le scénario de suppression d'un projet.

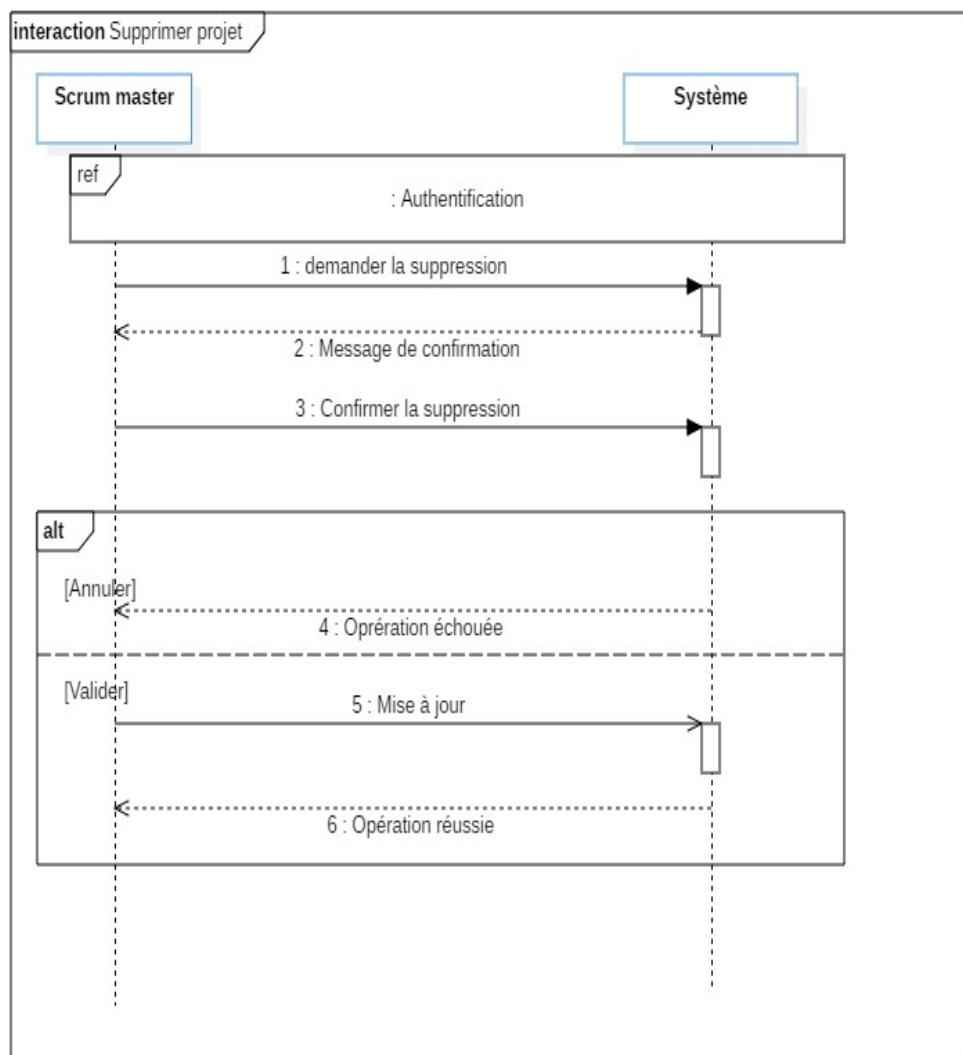


Figure II.7 – Diagramme de séquence système du cas d'utilisation «Supprimer un projet»

4 Capture des besoins techniques

Le choix technologique représente l'étape la plus critique où plusieurs critères tels que le coût, le temps de réaction et le framework de programmation l'impactant.

Dans la partie qui suit nous allons identifier les technologies que nous allons exploiter pour notre projet.

4.1 Salesforce

Redlean utilise la plateforme SaaS (Service as a Service) Salesforce pour développer ses solutions informatiques dans le domaine de la gestion de la relation client (CRM).

La gestion de la relation client (CRM) est une stratégie de gestion de l'interaction d'une entreprise avec ses clients, dans le but d'améliorer la satisfaction de la clientèle, sa fidélisation et la croissance de ses ventes.

Un système de gestion de la relation client aide une entreprise à atteindre ses objectifs, en rationalisant la communication sur différents supports et en fournissant des informations sur le client aux principaux décideurs de l'entreprise.

Aujourd'hui, Salesforce est un leader du marché du Cloud CRM (gestion de la relation client). C'est un CRM modulaire. Ses modules Sales Cloud, Service Cloud et Marketing Cloud peuvent être adaptés à toutes entreprises en démarrage, PME ou grandes entreprises et peut être modifié en fonction de l'évolution de chaque entreprise.

De plus, sa richesse fonctionnelle lui permet, quel que soit le secteur d'activité, de disposer d'un CRM sur mesure en embellissant des développements spécifiques et adaptés à l'activité.

En tant que SaaS, Salesforce offre des outils d'administration qui permettent d'automatiser et de rationaliser des processus tels que la gestion des utilisateurs, gestions des rôles et des accès, etc. Cet outil offre également des indicateurs utiles pour contrôler l'activité.

4.2 Choix des outils de développement

Outres cette fonctionnalité principale de CRM, Salesforce offre des outils en mode PaaS qui permettent de développer de nouvelles fonctionnalités. Voici une présentation de ses outils:

- **Force.com** : est un PaaS (Platform as a Service) conçu pour simplifier le développement et le déploiement d'applications d'entreprise et de sites Web en profitant de la puissance Lightning. Les développeurs peuvent créer des applications et des sites Web par le biais de l'environnement de développement intégré (IDE) dans le cloud, et les déployer rapidement sur les serveurs multi-tenants de Force.com.[\[8\]](#)
- **Visualforce** : est un cadre de développement Web qui permet aux développeurs de créer des interfaces utilisateur personnalisées et sophistiquées pour les applications pouvant être hébergées sur la plate-forme Lightning. Il peut intégrer n'importe quelle technologie Web standard ou infrastructure JavaScript pour créer une interface utilisateur plus animée et plus riche.
- **Framework de Composants Lightning** : est un Framework d'interface utilisateur qui permet de développer des applications pour les appareils mobiles et les ordinateurs du bureau. Il est similaire à AngularJS ou React. En plus, il est un framework moderne qui permet de créer des applications monopages à interface utilisateur réactive et dynamique pour les applications Lightning Platform. Il repose sur JavaScript côté client et Apex côté serveur. Les composants Lightning se composent de nombreuses ressources :
 - **Composant ou application** : la seule ressource requise dans un bundle. Contient un balisage pour le composant ou l'application. Chaque bundle ne contient qu'un seul composant ou ressource d'application.
 - **Styles CSS** : style pour le composant
 - **Controller Js** : Méthodes de contrôleur côté client pour gérer les événements dans le composant.
 - **Controller Helper Js** : Fonctions JavaScript pouvant être appelées à partir de n'importe quel code JavaScript dans le bundle d'un composant.
 - **Design** : obligatoire pour les composants utilisés dans Lightning App Builder ou Lightning Pages.
 - **Renderer** : Une description, un exemple de code et une ou plusieurs références à des exemples de composants.

II.4 Capture des besoins techniques

- **Bootstrap** : est un Framework CSS gratuit qui propose une collection d'outils pour créer et styliser des sites et des applications web en mode responsive. Il contient du HTML et des modèles de conception à base de CSS pour la typographie, des formes, des boutons, la navigation et d'autres composants de l'interface, ainsi que des extensions optionnelles.
- **JavaScript** : nous avons utilisé cette collection pour la création des différentes interfaces de notre application. Ce langage a la particularité de s'activer sur le poste client, en d'autres mots c'est l'ordinateur qui va recevoir le code et qui devra l'exécuter. C'est en opposition à d'autres langages qui sont activés côté serveur. L'exécution du code est effectuée par le navigateur internet.[9]
- **SOQL** : Salesforce fournit le SOQL pour rechercher des informations spécifiques dans les données Salesforce de votre organisation. Il est similaire au langage SQL standard mais est personnalisé pour Lightning Platform.
- **Apex** : est un langage de programmation orienté « objet » fortement typé. Il permet aux développeurs d'implémenter et de traiter des transactions complexes coté back end en utilisant une syntaxe qui ressemble à Java. Il est conçu pour ajouter de la logique métier aux applications et pour traiter de grandes quantités de données.

Il fonctionne entièrement sur la plateforme Force.com. Lorsqu'un développeur enregistre son code dans la plateforme, il est compilé et stocké sous forme de métadonnées dans les serveurs Salesforce. Ensuite, l'utilisateur final envoie une demande à partir de l'interface utilisateur et les résultats seront récupérés à partir des serveurs Salesforce comme illustré dans la figure suivante.[10]

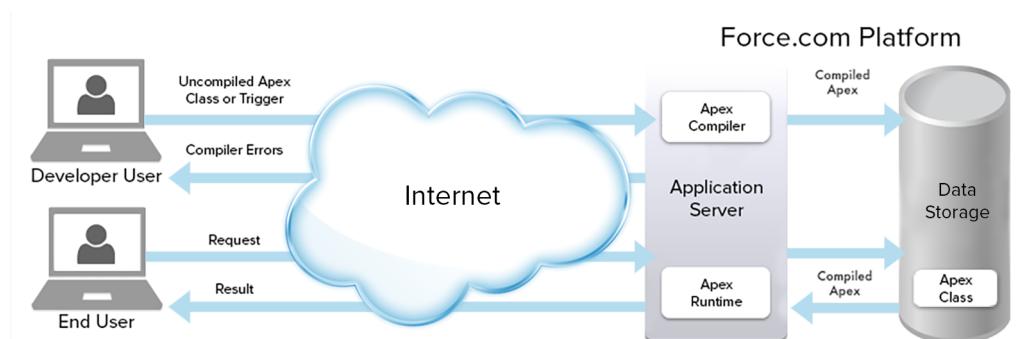


Figure II.8 – Fonctionnement d'Apex

Au vu de la richesse de la plateforme Salesforce et sa centralité dans les activités de Readlean, nous avons fait le choix d'exploiter les outils de Salesforce pour développer notre application

II.4 Capture des besoins techniques

de gestion de projet. Nous allons en particulier, exploiter les outils de développement en mode PaaS pour développer notre Front-office. Nous allons en suite exploiter le CRM en mode SaaS pour la gestion en Back-office des utilisateurs de notre application.

Conclusion

L’analyse des besoins procure une vision plus claire du sujet et une compréhension plus profonde des tâches à réaliser.

Dans ce chapitre, nous avons exprimé d’une manière semi-formelle les besoins fonctionnels et non fonctionnels de notre système. Nous avons fourni une analyse plus détaillée de ces besoins grâce aux diagrammes des cas d’utilisation. Ces spécifications serviront de base à l’étape de conception de l’application, objet du chapitre suivant.

CHAPITRE III

CONCEPTION

Plan

1	Conception préliminaire	34
1.1	Architecture générale	34
1.2	Patron de conception	35
2	Conception de la base de données	36
2.1	Modèle conceptuel de données (MCD)	37
3	Conception détaillée	39
3.1	Diagramme de classes	39
3.2	Diagrammes de séquence de conception	43
4	Conception graphique	45

La conception est une étape cruciale dans le cycle de vie d'une application. Elle présente un pont entre l'analyse et l'implémentation. Elle permet une compréhension approfondie des besoins de point de vue interne.

Dans ce chapitre, nous définirons tout d'abord la conception préliminaire de notre application. Ensuite, nous procéderons à la présentation de la conception de notre base de données. Puis, nous détaillerons la conception détaillée de notre application en élaborant les diagrammes de classes et en illustrant les diagrammes de séquence. Enfin, nous présenterons la conception graphique de notre application.

1 Conception préliminaire

La conception préliminaire représente le cœur du processus 2TUP vu qu'elle intègre le modèle d'analyse dans l'architecture technique de manière à tracer la cartographie des composants du système à développer.

1.1 Architecture générale

L'architecture générale permet d'analyser les fonctionnalités attendues ou existantes du système, l'organisation des différents éléments ainsi que la relation entre eux.

La figure III.1 met l'accent sur l'architecture globale de notre solution proposée.

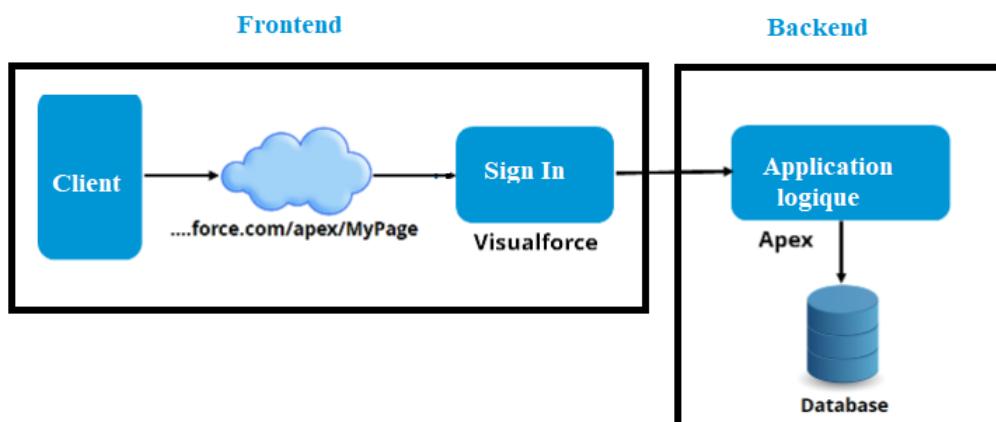


Figure III.1 – Architecture générale de l'application

Le client demande des informations à l'application Salesforce grâce au Visualforce. Ces informations sont ensuite transmises à la couche logique d'application, écrite en Apex. Selon les informations, les données sont insérées ou supprimées de la base de données. Salesforce offre également la possibilité d'utiliser des services Web pour accéder directement à la logique d'application.

1.2 Patron de conception

La conception consiste à scinder les tâches de l'application en différentes petites parties afin de mieux organiser et développer le logiciel. Pour établir un programme correctement conçu, il faut se pencher sur les patrons de conception qui vont nous guider tout au long du projet.

Salesforce définit une architecture logicielle de type MVCC (Modèle-Vue-Contrôleur Client-Contrôleur Serveur). Ce choix est argumenté par Salesforce afin de garantir une assurance de la maintenabilité, la modularité de l'application et la rapidité de développement.

De plus MVCC est présenté comme un modèle puissant pour répondre aux besoins des applications interactives. Il impose la séparation entre les données, la présentation et les traitements ce qui permet le regroupement des fonctions nécessaires en quatre éléments fondamentaux dans l'application finale MVCC.

Le développement en Salesforce s'appuie sur cette architecture logiciel afin de répondre aux besoins des applications interactives en séparant les problématiques liées aux différents composants en les regroupant par couches.

La figure III.2 présente l'architecture logicielle de la solution proposée en MVCC.

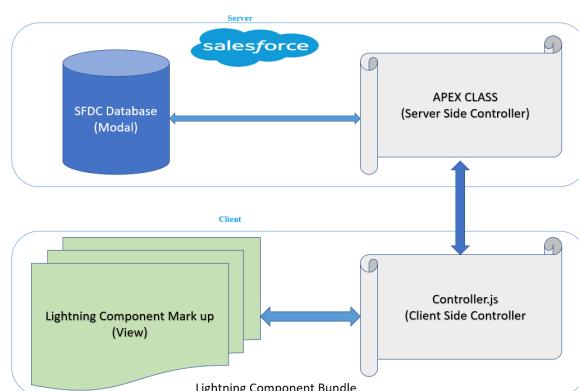


Figure III.2 – Architecture MVCC

L'architecture MVCC est donc décomposée en quatre éléments :

- **La Vue** : cette partie se concentre sur les éléments d'interface utilisateurs sophistiquées. Elle représente la présentation des données. Dans Salesforce, elle constitue les pages, les composants, les mises en page et les onglets de Visualforce. Elle communique uniquement avec le contrôleur Javascript.
- **Le Modèle** : le modèle correspond aux objets de données, champs et relations Salesforce. Il s'agit d'objets standard (compte, opportunité, etc.) et personnalisés (objets créés). Il prend également en charge tous les objets personnalisés créés selon les exigences du client. Il ne communique qu'avec le contrôleur Apex.
- **Le Contrôleur côté client** : un contrôleur côté client gère les événements au sein d'un composant. Il s'agit d'une ressource JavaScript qui définit les fonctions de toutes les actions du composant. Il communique avec le contrôleur Apex (contrôleur côté serveur) s'il a vraiment besoin d'un traitement côté serveur ou de toute autre opération de données du modèle (Sobjects).
- **Le Contrôleur côté serveur** : est la couche intermédiaire entre le modèle et la vue dont la logique métier est implémentée dans le contrôleur. Ce sont les éléments constitutifs de la logique réelle utilisant le langage Apex qui interagit avec la base de données en fonction des besoins de la vue.

2 Conception de la base de données

Les bases de données constituent le cœur du système d'information. Alors, la conception de ces bases est la tâche la plus ardue du processus de développement du système d'information.

Les méthodes de conception préconisent une démarche en étapes et font appel à des modèles pour représenter les objets qui composent les systèmes d'information, les relations existantes entre ces objets ainsi que les règles sous-jacentes.[\[11\]](#)

2.1 Modèle conceptuel de données (MCD)

Le modèle conceptuel de données est une représentation schématique la plus abstraite des données d'un système d'information de l'entreprise qui met en évidence sa sémantique. Il a pour but d'écrire de façon formelle les données qui seront utilisées par le système d'information.

Dans la présentation de notre modèle conceptuel des données, nous nous sommes basés sur le modèle entité-association qui a pour but de décrire de façon formelle des données qui seront utilisées par le système d'information. Il s'agit donc d'une représentation des données, facilement compréhensible.

La figure III.3 fournit une représentation graphique du modèle conceptuel des données de notre projet, qui contient toutes les relations entre les tables de notre base de données.

Notre modèle conceptuel des données illustré par la figure ci-dessous III.3 comporte huit entités.

Parmi ces entités on cite :

- L'entité «**Ressource**» présente les informations d'une ressource tels que son identifiant unique, son nom, son numéro de téléphone, son login ,son mot de passe et son rôle qui présente sa responsabilité au sein d'une équipe (Scrum master ou product owner ou developer).

Le scrum master ou le product owner peuvent gérer zéro ou plusieurs projets.

- L'entité «**sous tache**» présente ses informations tels que son identifiant, son nom, sa description,sa date de début,sa date de fin, sa note, sa priorité, son temp estimé, son état et son total temp passé.

Une entité ressource qui a pour rôle developer se charge de gérer 1 ou plusieurs sous taches.

III.2 Conception de la base de données

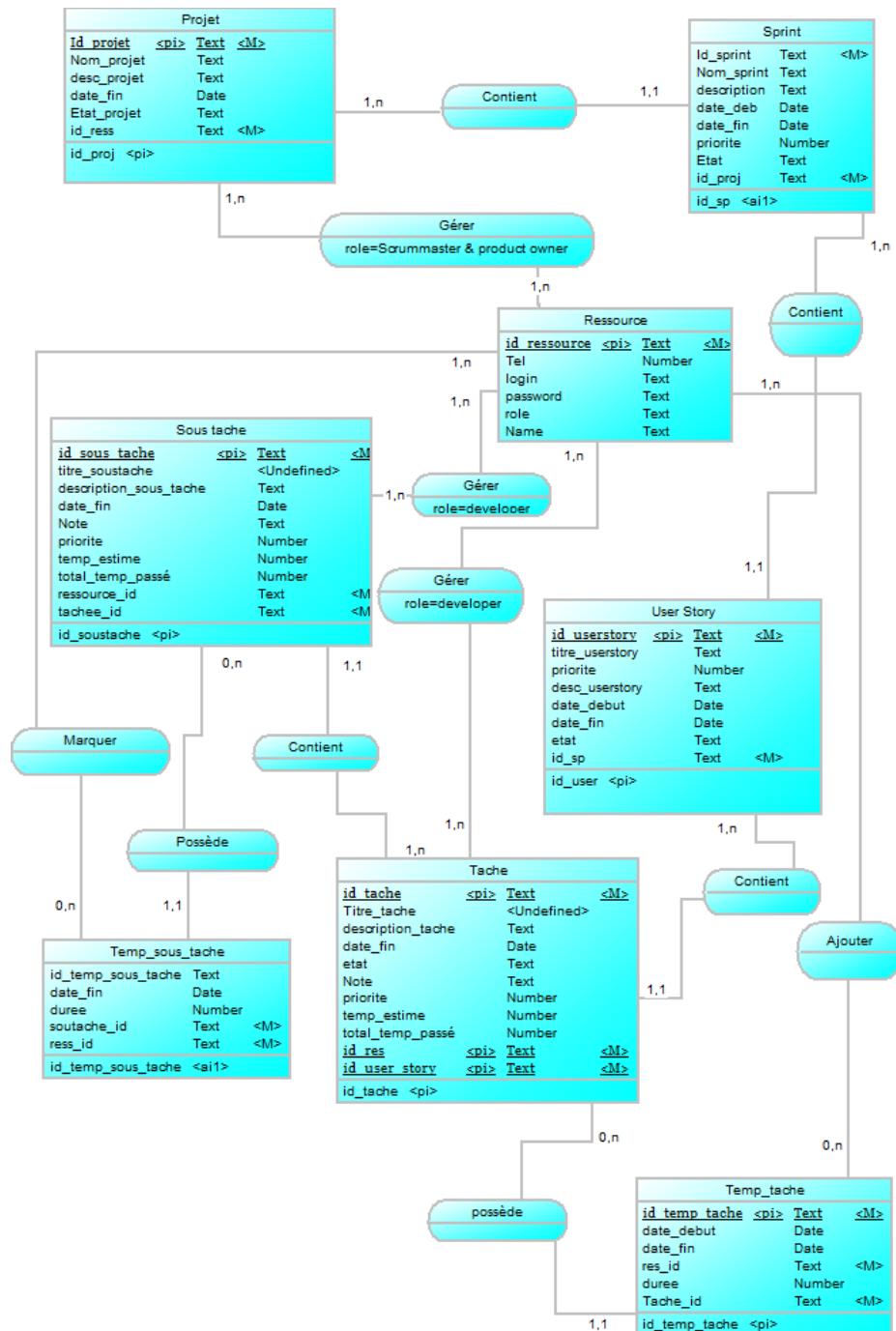


Figure III.3 – Modèle conceptuel des données

3 Conception détaillée

Pour détailler davantage la conception de notre solution, nous présenterons le diagramme de classes et les diagrammes de séquence qui décrivent les différents composants et fonctionnalités de notre solution.

3.1 Diagramme de classes

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet. Il permet de fournir une représentation abstraite des objets du système qui vont interagir ensemble pour réaliser les cas d'utilisation. Il s'agit d'une vue statique car on ne tient pas compte du facteur temporel dans le comportement du système. Donc, sa représentation peut être illisible.

Pour cette raison, on a divisé notre diagramme de classe en trois parties :

- partie contrôleur client - vue
- partie contrôleur client - contrôleur serveur
- partie modèle - contrôleur serveur

3.1.1 Diagramme de classe partie contrôleur client - vue

La figure III.4 présente la première partie du diagramme de classe qui inclut la partie contrôleur client et quelques relations avec la partie vue.

Notre partie vue est un ensemble regroupant les composants d'extension «.cmp» qui communiquent avec les contrôleurs client d'extension «.js».

III.3 Conception détaillée

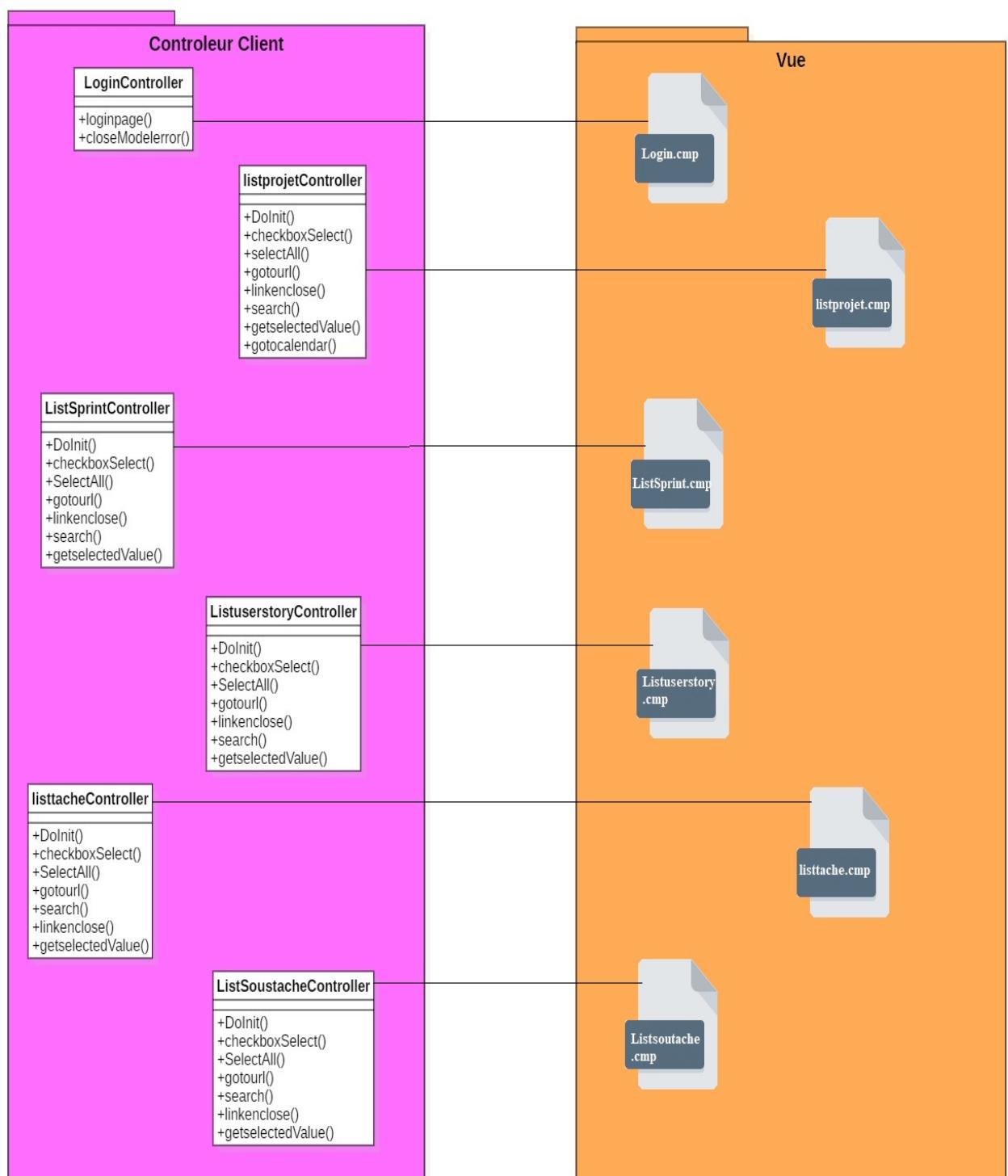


Figure III.4 – Diagramme de classe partie contrôleur client - vue

III.3 Conception détaillée

3.1.2 Diagramme de classe partie contrôleur client - contrôleur serveur

La figure III.5 montre la deuxième partie du diagramme de classe qui inclut les éléments de la partie contrôleur client et quelques relations avec la partie contrôleur serveur.

Les contrôleurs côté client d'extension « .js » gèrent les événements au sein d'un composant. Ils définissent les fonctions de toutes les actions du composant en faisant appel à des contrôleurs Apex (contrôleur côté serveur) d'extension « .apxc ».

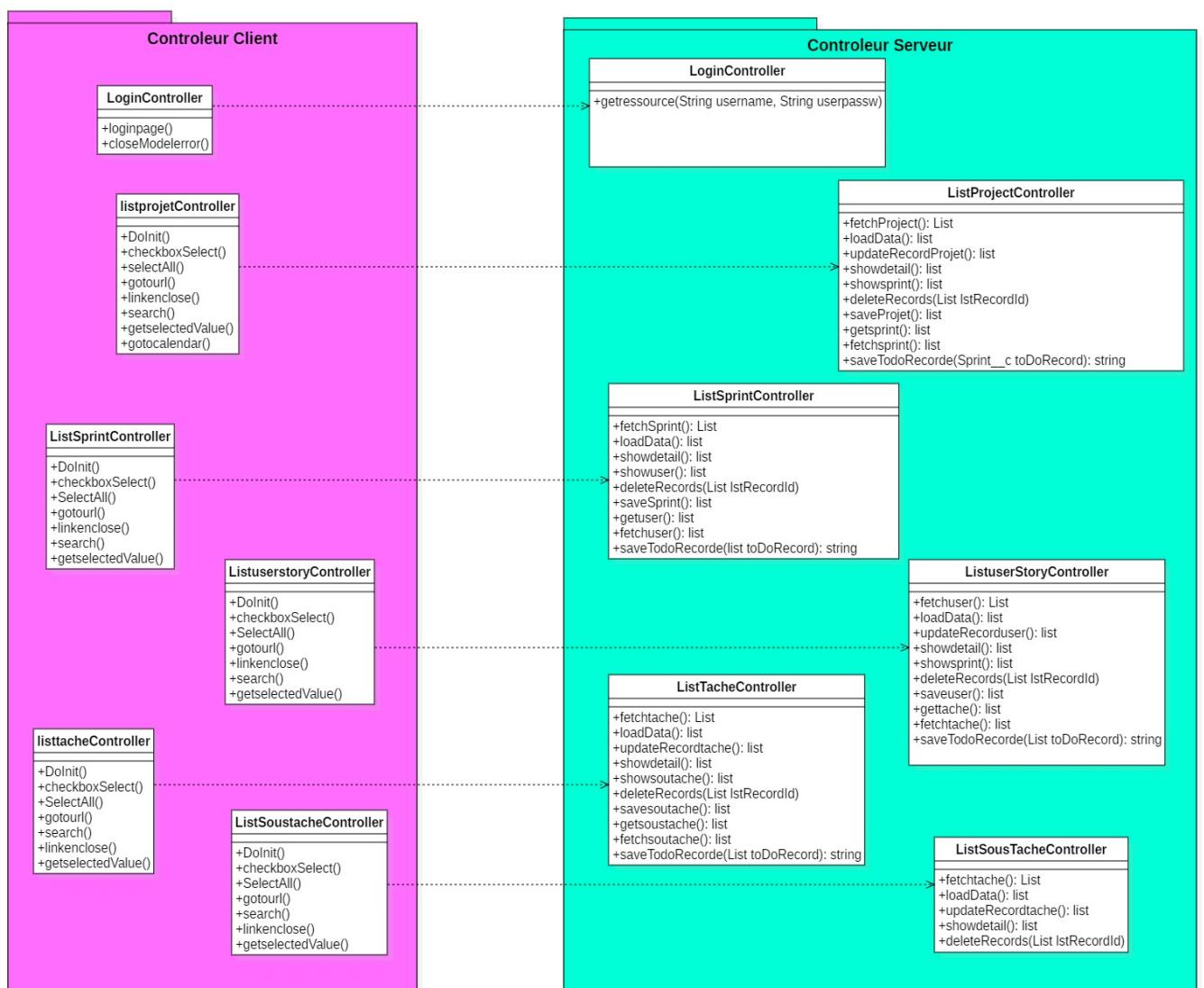


Figure III.5 – Diagramme de classe partie contrôleur client - contrôleur serveur

III.3 Conception détaillée

3.1.3 Diagramme de classe partie modèle - contrôleur serveur

La figure III.6 illustre la troisième partie du diagramme de classe qui inclut les éléments de la partie contrôleur serveur et quelques relations avec la partie modèle.

La partie modèle correspond à la transposition du modèle conceptuel de données de notre application. Elle ne communique qu'avec les contrôleurs Apex d'extension «.apxc» dont ils représentent la couche intermédiaire entre le modèle et la vue.

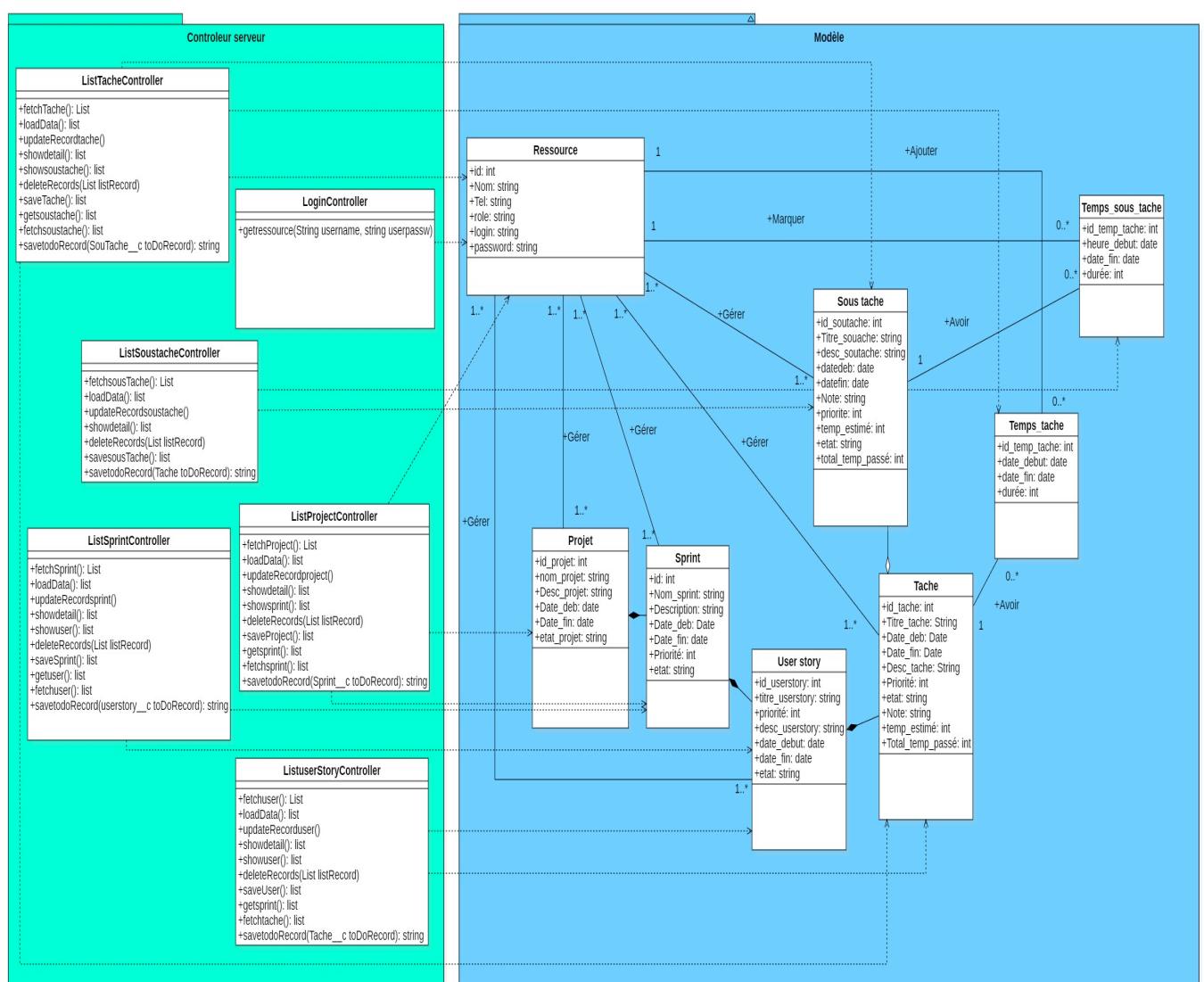


Figure III.6 – Diagramme de classe partie modèle-contrôleur Serveur

3.2 Diagrammes de séquence de conception

Les diagrammes de séquence permettent de décrire le déroulement des scénarios bien définis afin de présenter graphiquement les interactions entre les acteurs et les différents composants d'un système selon un ordre chronologique.

Dans cette section, nous illustrons l'aspect dynamique de notre application en présentant quelques diagrammes de séquence du formalisme UML et en détaillant les interactions entre les différents objets et l'utilisateur.

3.2.1 Diagramme de séquence "Modifier un projet"

La figure III.7 présente le diagramme de séquence conceptuel qui fait référence au diagramme de séquence système «Modifier un projet» du chapitre

Le scrum master clique sur le bouton « modifier» sur la vue «listprojet.cmp». Alors, la méthode «editprojet()» va déclencher au contrôleur «ListprojetcController.js» qui fait appel à la méthode «updateRecords()» de la classe Apex «ListProjetContrller.apxc» qui interagit avec la base de données.

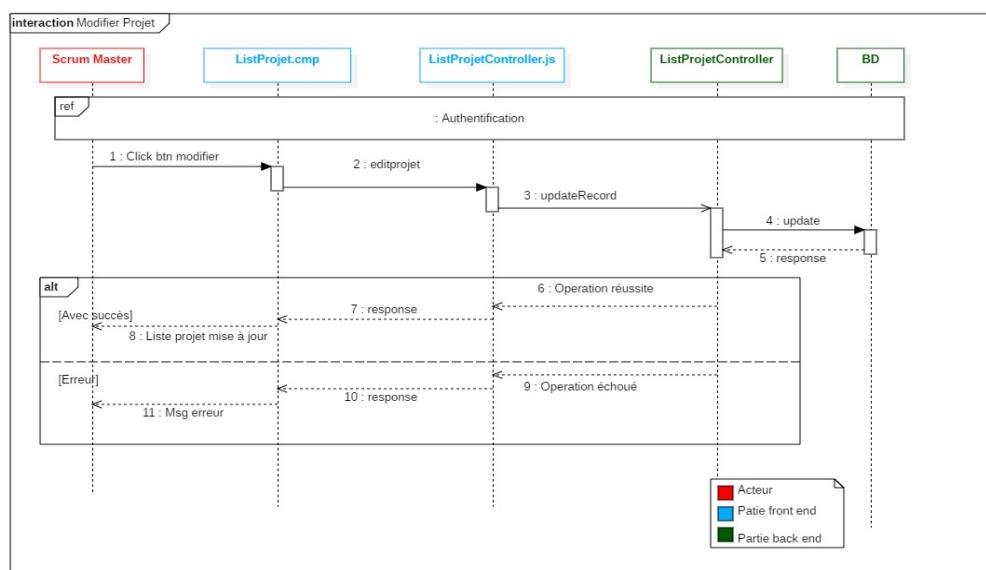


Figure III.7 – Diagramme de séquence " Modifier un projet"

3.2.2 Diagramme de séquence "Supprimer un projet"

La figure III.8 présente le diagramme de séquence conceptuel qui fait référence au diagramme de séquence système «Supprimer un projet» du chapitre 2.

Le scrum master clique sur le bouton «Supprimer» sur la vue «listprojet.cmp» qui appelle la méthode «handleconfirmDialog()» du contrôleur «listprojetController.js». Cette dernière déclenche la méthode «deleteRecords()» de la classe Apex «listprojetController.apxc» qui fait accès à la base de données.

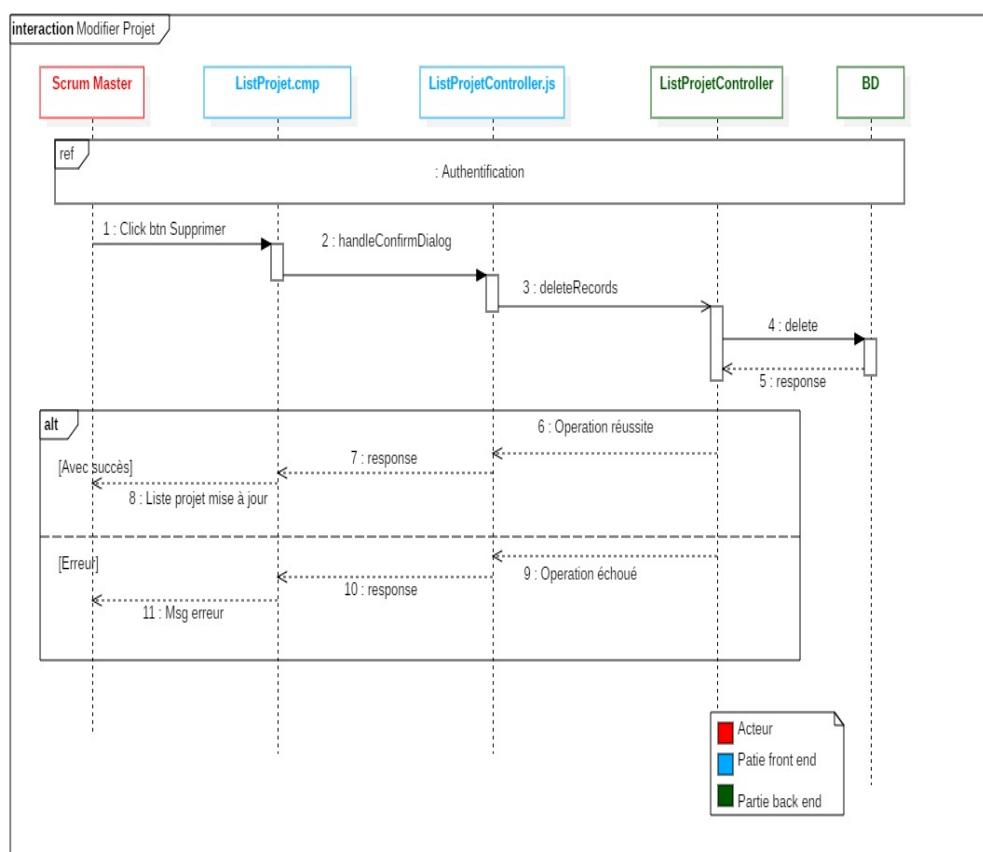


Figure III.8 – Diagramme de séquence " Supprimer un projet"

4 Conception graphique

Dans cette section,nous allons établir quelques maquettes de notre application.

La figure III.9 présente l'interface qui permet aux ressources d'authentifier et d'accéder aux différentes fonctionnalités selon ses droits d'accès.

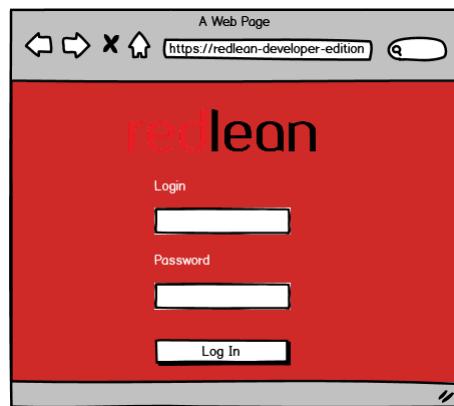


Figure III.9 – Maquette de l'interface d'authentification

La figure III.10 illustre l'interface d'accueil de Scrum Master. A partir de cet interface il peut avoir une idée globale sur ses fonctionnalités.

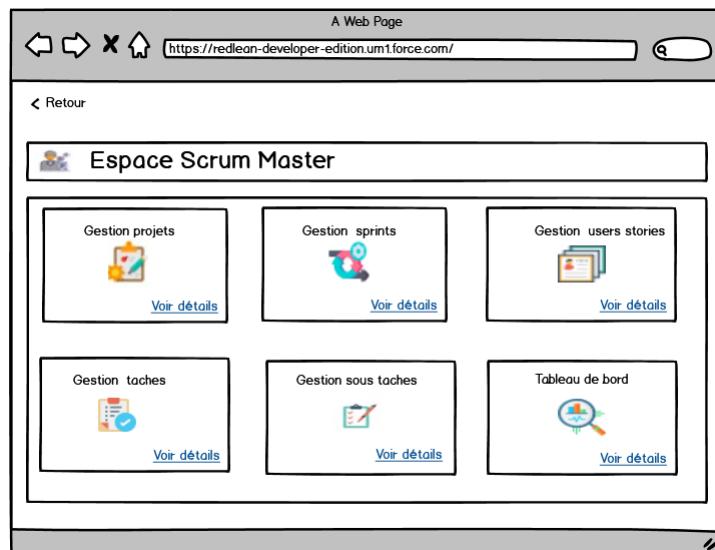
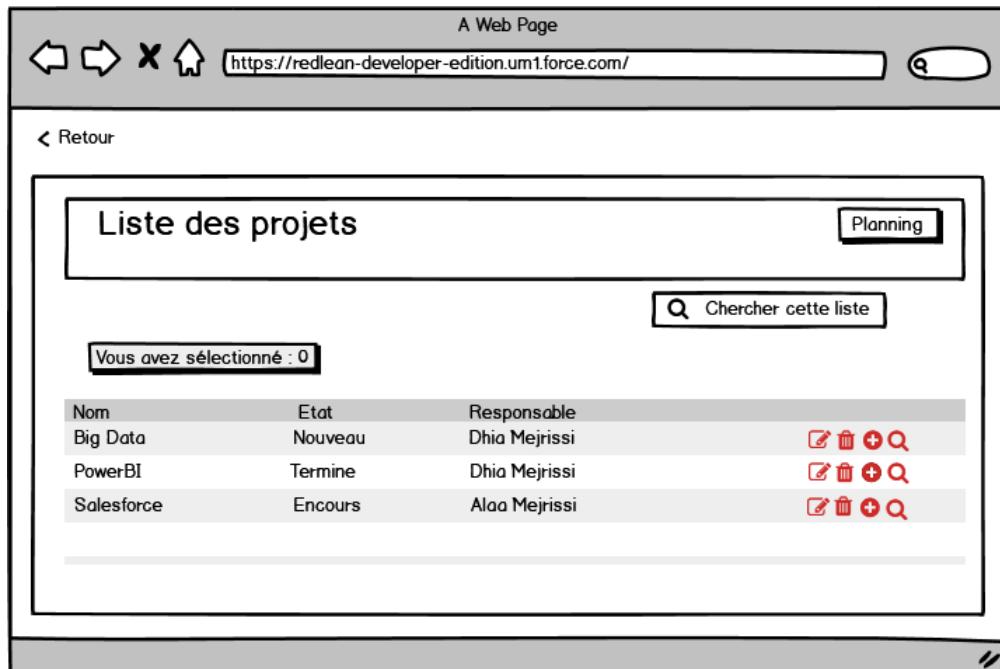


Figure III.10 – Maquette de l'interface d'accueil de Scrum Master

III.4 Conception graphique

La figure III.11 présente l'interface qui permet au Scrum Master de gérer ses projets.



A Web Page
https://redlean-developer-edition.um1.force.com/

< Retour

Liste des projets

Planning

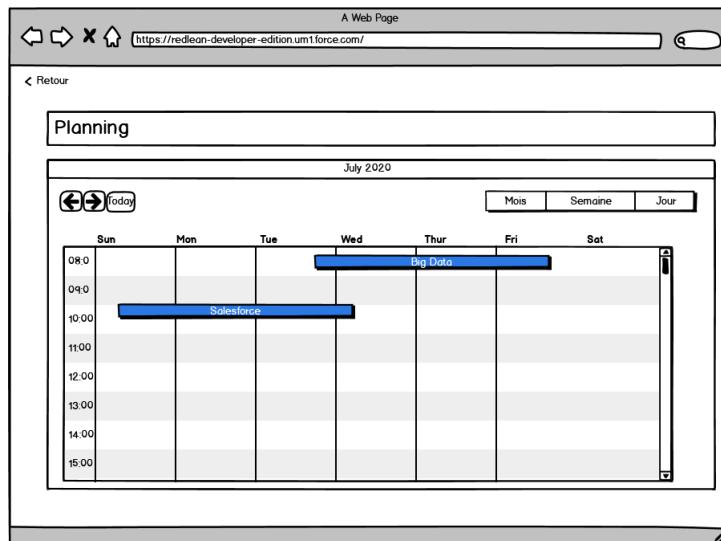
Chercher cette liste

Vous avez sélectionné : 0

Nom	Etat	Responsable	
Big Data	Nouveau	Dhia Mejrissi	<input type="checkbox"/> <input type="button" value="Supprimer"/> <input type="button" value="Ajouter"/>
PowerBI	Termine	Dhia Mejrissi	<input type="checkbox"/> <input type="button" value="Supprimer"/> <input type="button" value="Ajouter"/>
Salesforce	Encours	Alaa Mejrissi	<input type="checkbox"/> <input type="button" value="Supprimer"/> <input type="button" value="Ajouter"/>

Figure III.11 – Maquette de l'interface de gestion des projets

La figure III.12 montre l'interface de calendrier des projets.



A Web Page
https://redlean-developer-edition.um1.force.com/

< Retour

Planning

July 2020

today

Mois Semaine Jour

Sun	Mon	Tue	Wed	Thur	Fri	Sat
08:00				Big Data		
09:00						
10:00			Solesforce			
11:00						
12:00						
13:00						
14:00						
15:00						

Figure III.12 – Maquette de l'interface de calendrier des projets

III.4 Conception graphique

La figure III.13 illustre l'interface des statistiques des projets,sprints,users stories et sous tâches.

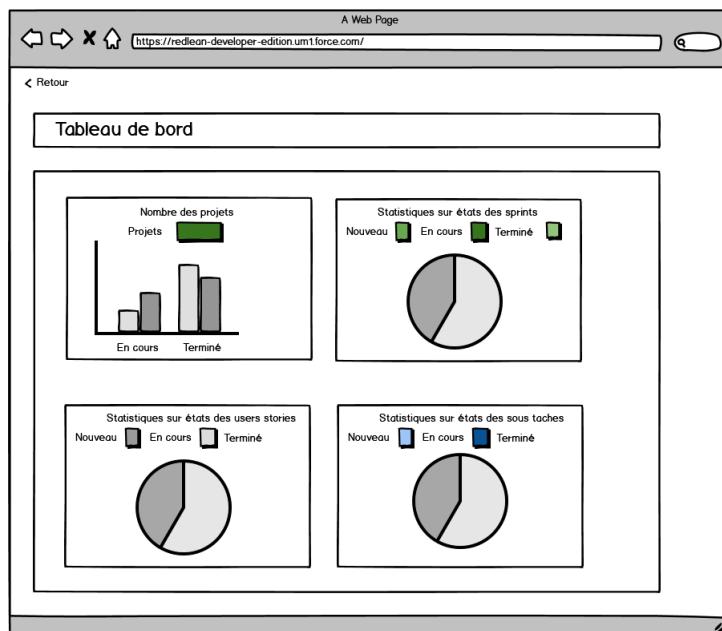


Figure III.13 – Maquette de l'interface des statistiques

Conclusion

Dans ce chapitre, nous avons présenté la conception de notre système à travers différents diagrammes afin que ça soit le plus simple et le plus aisé pour l'implémentation du futur système qui fera l'objet du prochain chapitre.

CHAPITRE IV

RÉALISATION

Plan

1	Environnement logiciel de travail	49
2	Implémentation de la base des données	49
3	Mise en place du back office	51
4	Implémentation du front office	53
4.1	Implémentation du front end	53
4.2	Implémentation du back end	55
5	Exemples d'interfaces	56
6	Plan de réalisation	58

Ce chapitre représente le dernier volet de ce rapport, qui a pour objectif d'illustrer les différents axes de la réalisation de ce projet.

Pour ce faire, nous commencerons par décrire les outils utilisés pour le développement de la solution proposée. Par la suite, nous détaillerons la mise en œuvre de quelques composants. Enfin, nous allons présenter quelques exemples d'interfaces de notre système.

1 Environnement logiciel de travail

Au cours de cette partie, nous énumérerons quelques outils utilisés tout au long de ce projet pour l'étude et la mise en place de notre application :

- **Salesforce** : c'est un éditeur de logiciels. Il distribue des logiciels de gestion basés sur Internet et héberge des applications d'entreprises.[\[12\]](#)
- **Workbench** : c'est un outil web qui aide les administrateurs et les développeurs à interagir avec Salesforce pour l'insertion, la mise à jour, la conversion, la suppression et l'exportation de données. C'est un outil avancé qui est construit en utilisant JavaScript et PHP, où il prend en charge plus de 5 millions d'enregistrements pour la manipulation de données.[\[13\]](#)
- **StartUML** : c'est un outil complet de modélisation UML en passant par les étapes d'analyse, les modèles de conception et les étapes de test et d'entretien. Nous avons utilisé cet outil pour la présentation des différents diagrammes de ce rapport. [\[14\]](#)
- **Overleaf** : c'est un éditeur latex en ligne, facile à utiliser et permettant la création de nouveaux documents ou bien le téléchargement d'un document existant. Il offre la possibilité d'avoir le code source du fichier ou le télécharger sous format pdf. Cet outil nous a permis de rédiger ce rapport.

2 Implémentation de la base des données

Salesforce offre une base de données mutualisée dans le cloud dans laquelle l'objet est similaire à une table de bases de données, les colonnes comme des champs et les lignes comme des enregistrements afin de collecter des informations.

Cette base de données permet de :

IV.2 Implémentation de la base des données

- **Créer des objets :** tels que des tables relationnelles et d'utiliser les métadonnées pour décrire ces objets et leur utilisation.

La figure IV.1 montre l'interface de création de l'objet personnalisé Projet dans l'organisation de développeur

The screenshot shows the Salesforce Object Manager interface for the 'Projet' object. On the left, there's a sidebar with various options like Details, Fields & Relationships (which is selected), Page Layouts, Lightning Record Pages, etc. The main area is titled 'Fields & Relationships' and lists 12 items. The table has columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Currency	CurrencyIsoCode	Picklist		
date_debut	date_debut__c	Date		
date_fin	date_fin__c	Date		
Description_projet	Description_projet__c	Long Text Area(32768)		
Etat	Etat__c	Picklist		
Iconstatus	Iconstatus__c	Formula (Text)		
Last Modified By	LastModifiedById	Lookup(User)		

Figure IV.1 – Interface de création de l'objet Projet

- **Créer deux types de relations entre des objets de données :**

- **Relation Lookup :** relie principalement deux objets, ce qui permet de référencer un objet depuis les éléments associés d'un autre objet à travers une relation un-à-un ou un-à-plusieurs
- **Relation Master Détail :** est plus étroite dans laquelle l'objet principal (maître) contrôle certains comportements de l'objet de détail (subordonné). Ainsi que la suppression d'objet maître provoque la suppression des objets subordonnés.

La figure IV.2 présente les objets personnalisés que nous avons créés ainsi que les relations entre eux.

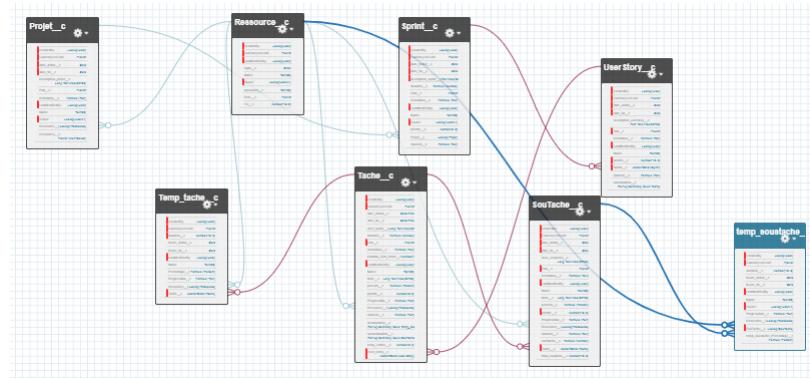


Figure IV.2 – Interface de la liste des objets personnalisés

3 Mise en place du back office

Salesforce fournit divers outils pour une administration efficace. Parmi ces outils figurent :

- **Envoi automatique des emails**

Pour l'envoi automatique des emails, nous avons utilisé :

- **Règle de Workflow** : permet d'automatiser les processus et procédures internes standards afin de gagner du temps dans l'ensemble de l'organisation, tels que l'envoi d'alertes par courrier électronique, la mise à jour de champs sur les exigences basées sur des critères de déclenchement ou l'attribution de tâches.

Lorsque tous les critères d'une règle de workflow correspondent à un enregistrement, les actions de cette règle sont immédiatement exécutées.[15]

Un exemple est illustré par la figure IV.3 suivante

The screenshot shows the 'Edit Rule' interface for an 'Account' object. The rule is named 'Rule1' and is described as 'Rule for Generating the E-mail alert'. The 'Evaluation Criteria' section indicates that the rule runs when a record is created and edited. The 'Rule Criteria' section contains a single criterion: 'Fax' field contains '12345'. The bottom of the screen features standard navigation buttons: 'Previous', 'Save & Next', and 'Cancel'.

Figure IV.3 – Interface d'une règle de Workflow

IV.3 Mise en place du back office

- **Générateur de processus (Process Builder)** : est un outil de développement de processus qui aide à automatiser les Workflows. Un générateur de processus est plus avancé que les flux de travail. La principale différence entre eux est le nombre d'opérations/actions pour lesquelles ils sont utilisés. Le générateur de processus est plus adaptable avec des cas plus complexes.

En fait, on peut dire que le Process Builder est une chaîne de workflows.[\[16\]](#)

La figure IV.4 illustre un exemple simple d'un générateur de processus qui génère une alerte par courrier électronique lorsqu'une ressource a été créée dans notre système.

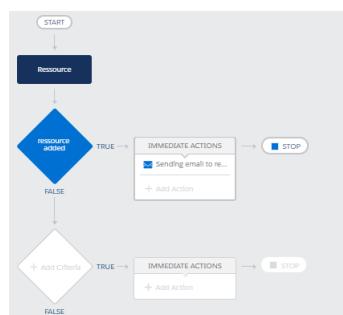


Figure IV.4 – Interface d'un exemple générateur du processus

- **Site Salesforce** : offre la possibilité de créer des sites Web publics sans que les utilisateurs aient à se connecter avec un nom d'utilisateur et un mot de passe.

La figure IV.5 présente le paramétrage d'un site dans l'organisation développeur.



Figure IV.5 – Interface de paramétrage de site

4 Implémentation du front office

Dans cette partie, nous allons présenter la mise en oeuvre de la partie front end grâce à Developer Console qui nous permet de créer, déboguer et tester notre application dans l'organisation développeur du Salesforce.

4.1 Implémentation du front end

Dans l'implémentation du front end, nous allons nous baser sur les composants qui vont nous permettre d'avoir des compartiments réutilisables de notre application et cela afin de réduire la quantité de codes à générer.

La figure IV.6 résume la liste des composants implémentés dans notre application.

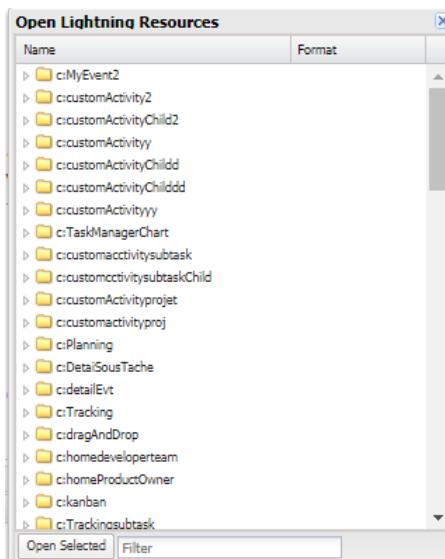


Figure IV.6 – Interface de la liste des composants

Dans ce qui suit, nous allons présenter un exemple d'implémentation d'un composant «List projet» qui fait partie de notre projet.

Ce composant comporte de nombreuses ressources :

* Composant «ListProjet.cmp»

La figure IV.7 présente le composant « ListProjet » qui contient les balises HTML.

IV.4 Implémentation du front office

```
<lightning:layout multipleRows="true">  
  <lightning:layoutItem size="12" padding="around-small">  
    <div class="slds-grid slds-grid--align-left" >  
      <lightning:buttonIcon iconName="utility:chevronleft" variant="bare" onclick=" {! c.handleClick } " /> &nbsp;  
      <div class="slds-text-heading_small slds-text-color_default"><center><b>Retour</b></center></div>  
    </div>  
  <br/><br/>
```

Figure IV.7 – Interface de composant «ListProjet.cmp

* Contrôleur «ListProjetController.Js»

Le contrôleur «ListProjetController.Js» côté client gère les événements au sein de composant «List projet» comme illustre la figure IV.8.

```
doInit :function(component,event,helper){  
  // Apex method definition  
  var action = component.get("c.loadData");  
  // callbak function  
  action.setCallback(this,function(response){  
    //get state  
    var state = response.getState();  
    // check if state is 'SUCCESS'  
    if(state == 'SUCCESS'){  
      var result = response.getReturnValue();  
      //set value to "UnfilteredData" attaribute  
      component.set("v.UnfilteredData",result);  
      console.log(result);  
      // set value to "data" attaribute  
      component.set("v.data",result);  
    }else{  
      console.log('something bad happend! ');  
    }  
  });
```

Figure IV.8 – Interface de Contrôleur «ListProjetController.Js»

* Contrôleur «ListProjetHelper.Js»

La figure IV.9 montre l'une des fonctions utilitaires qui peut être réutilisée dans le contrôleur de composant.

IV.4 Implémentation du front office

```
onLoad: function(component, event) {
    console.log('onLoad call');
    //call apex class method
    var action = component.get('c.fetchProject');
    action.setCallback(this, function(response) {
        //store state of response
        var state = response.getState();
        if (state === "SUCCESS") {
            //set response value in ListOfContact attribute on component.
            component.set('v.ListOfContact', response.getReturnValue());
            // set deafult count and select all checkbox value to false on load
            component.set("v.selectedCount", 0);
            component.find("box3").set("v.value", false);
        }
    });
    $A.enqueueAction(action);
},
```

Figure IV.9 – Interface de Contrôleur «ListProjetHelper.js»

4.2 Implémentation du back end

Afin d'avoir une partie back end interopérable, nous avons implémenté des classes Apex. La figure IV.10 présente la liste des classes Apex de notre application.

Open		
Entity Type	Entities	Related
Entity Type	Name Namespace ▾	
Classes	DashboardPalTest Dashboard_Pal	
Triggers	DashboardPalCtrl Dashboard_Pal	
Pages	ReallocateTasksCntrl tastratoo	
Page Components	ReallocateTasksCntr... tastratoo	
Objects	LeadKanban	
Static Resources	LeadLightningKanba...	
Packages	kanban2Controller	
	PopupFormC	
	MyControl	

Figure IV.10 – Interface de la liste des classes Apex

* Contrôleur Apex «ListProjetController.apxc»

La figure IV.11 explique l'architecture de notre composant Lightning qui se lie aux méthodes d'action Apex sur la plateforme Salesforce.

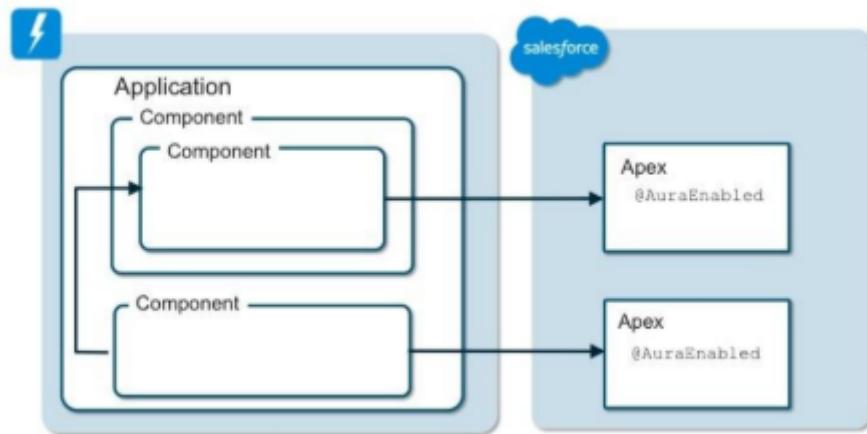


Figure IV.11 – Fonctionnement d'Apex

La figure IV.12 est une capture d'écran présentant la classe Apex « `ListProjetController.apxc` » avec ses différentes méthodes qui comprend l'instruction de bouclage, l'instruction de flux de contrôle et la requête SOQL.

```
public with sharing class ListProjetController {
    @AuraEnabled
    public static list < Projet__c > fetchProject() {
        list < Projet__c > returnConList = new List < Projet__c > ();
        list < Projet__c > lstCon = [SELECT Name, iconstatus__c, Description_projet__c, date_debut__c, date_fin__c, Etat__c, Ressource__r.Name From Projet__c ORDER BY date_fin__c DESC NULLS LAST LIMIT 50];
        for (Projet__c c : lstCon) {
            returnConList.add(c);
        }
        return returnConList;
    }
    @AuraEnabled
    public static List < Projet__c > loadData(){
        return [SELECT Name, iconstatus__c, Description_projet__c, date_debut__c, date_fin__c, Etat__c, Ressource__r.Name From Projet__c ORDER BY date_fin__c DESC NULLS LAST LIMIT 50];
    }
    @AuraEnabled
    public static List<Projet__c> UpdateRecord(String conid){
        return [SELECT Id, Name, Description_projet__c, date_debut__c, date_fin__c, Etat__c FROM Projet__c WHERE Id =:conid];
    }
    @AuraEnabled
    public static List<Projet__c> showdetail (String conid){
        return [SELECT Id, Name, Description_projet__c, date_debut__c, date_fin__c, Etat__c FROM Projet__c WHERE Id =:conid];
    }
}
```

Figure IV.12 – Interface de Contrôleur Apex

5 Exemples d'interfaces

Lors de la réalisation de notre application, nous n'avons pas adopté une version complète de CRM Salesforce puisqu'elle coûte environ \$500 par mois et augmentent en fonction du nombre d'utilisateurs.

A cet égard, nous avons utilisé la notion de site comme mécanisme technique rapide de Salesforce permettant de créer des sites web publics et des applications directement intégrés

IV.5 Exemples d'interfaces

dans l'organisation Salesforce gratuitement mais avec une limitation au niveau des outils utilisés de gestion de la relation client.

La figure IV.13 illustre l'interface d'accès à l'application. Elle implémente la fonction qui consiste à vérifier l'identité de l'utilisateur par un login et un mot de passe afin de lui autoriser l'accès aux différentes fonctionnalités accordées de l'application.

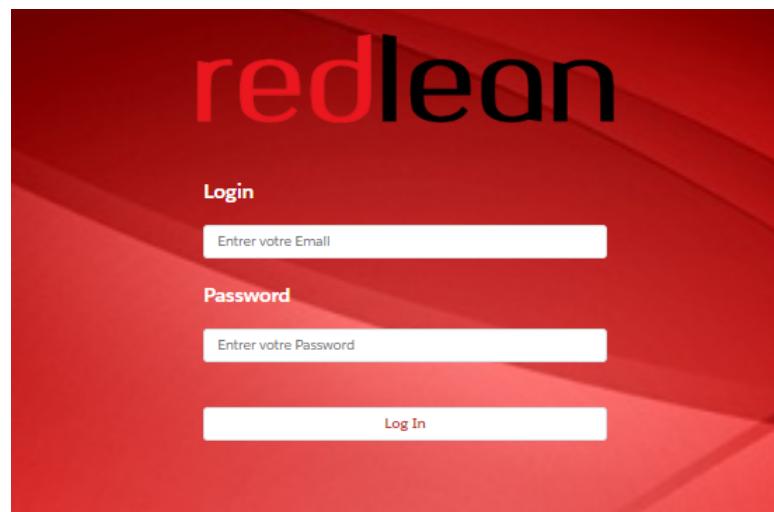


Figure IV.13 – Interface d'authentification

Lors de l'authentification, le Scrum master peut consulter les statistiques des projets, users stories, sprints, tâches et sous-tâches comme montre la figure IV.14

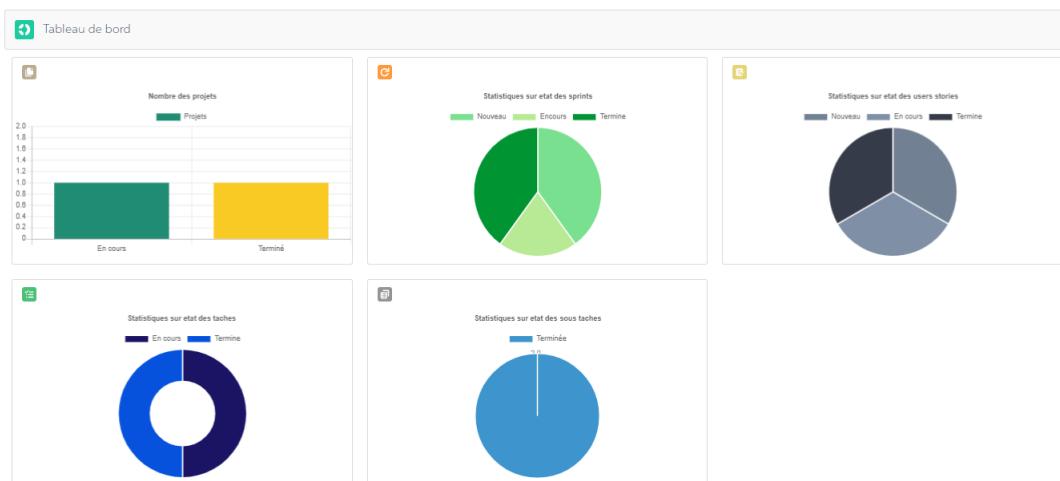


Figure IV.14 – Interface des rapports et dashboard

IV.6 Plan de réalisation

La figure IV.15 ci-dessous illustre l'interface de la liste des projets où le Scrum master est en mesure de consulter tous les projets, et de les gérer soit par modification soit par suppression.

The screenshot shows a web-based project management application. At the top left is a 'Retour' button. In the center, there's a header with a folder icon and the text 'Liste des projets'. On the right side of the header are 'Planning' and 'Chercher cette liste' buttons. Below the header, a message says 'Vous avez sélectionné: 0'. A table follows, displaying two projects:

NOM	ETAT	RESPONSABLE	Actions
PowerBI	Terminé	Dhla Mejrlssl	
salesforce	En cours	Alaa Mejrlssl	

Figure IV.15 – Interface de la liste des projets

6 Plan de réalisation

Vu les circonstances de la pandémie « Coronavirus », nous n'avons pas pu appliquer notre planning prévisionnel tout au long de la réalisation de notre projet.

Par conséquent, ce travail a été réalisé durant une période de 5 mois. Nous avons tout d'abord commencé par une étude approfondie de l'existant pour pouvoir comprendre les besoins et débuter le travail. Ensuite, après avoir compris ce qui était demandé, nous avons pu dégager les spécifications fonctionnelles et non fonctionnelles (chapitre 2). Puis, nous avons entamé la phase de conception et implémenté les modules demandés (chapitre 3 et 4).

La figure IV.16 ci-dessous représente le plan de réalisation de déroulement du stage.

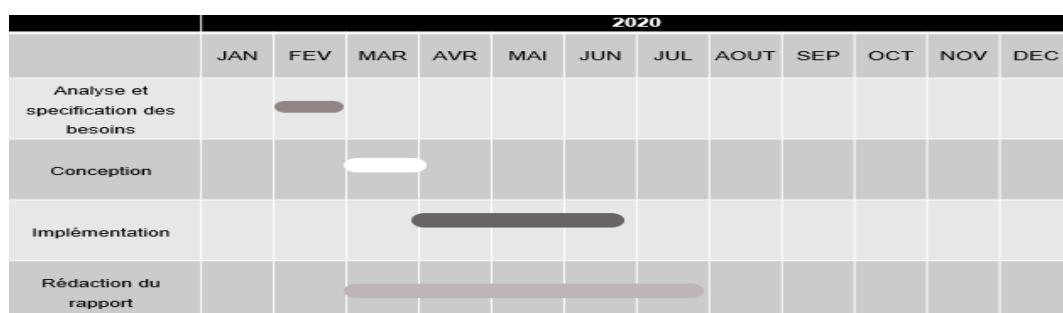


Figure IV.16 – plan de réalisation

Conclusion

A travers ce dernier chapitre, nous avons illustré quelques scénarios de ce travail, à travers des captures d'écran témoignant des différentes interfaces que contient notre application. La section suivante sera consacrée à donner une conclusion sur notre travail ainsi que quelques perspectives.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Nous venons de réaliser une application de gestion de projets selon la méthodologie Scrum pour la société Redlean .

Dans ce rapport, nous avons présenter les différentes étapes permettant d'aboutir à la réalisation de ce projet.

La première étape consiste à préciser le contexte du projet, la problématique ainsi que la solution proposée.

Dans une deuxième étape, nous avons fixé les besoins fonctionnels et nom fonctionnels de notre application.Nous avons capturé les besoins techniques.

La troisième étape consiste à exposer la conception détaillée de notre application en modélisant la solution suivant la démarche UML.

Finalement nous avons abordé la réalisation pour mettre en œuvre notre modélisation conceptuelle.

Sur le plan méthodologique, l'utilisation de la méthode 2TUP nous a permis de réaliser un système stable et évolutif.

D'un point de vue technologique, le projet a été très enrichissant puisqu'il nous a donné l'occasion d'étudier et utiliser un vaste panel de technologies et d'outils (e.g. JavaScript, Lightning Component, Apex, etc...).

En termes de perspectives, nous pouvons envisager plusieurs améliorations pour notre projet.
Nous proposons :

- Intégration d'un diagramme de Gantt afin de mieux visualiser les dépendances entre les tâches et les sous-tâches.
- Amélioration du Timing des tâches et sous-tâches et enrichissement du tableau Kanban.
- Ajout d'un module de discussion instantanée pour une communication efficace et fluide entre les collaborateurs.
- Génération des rapports au format PDF.

BIBLIOGRAPHIE

- [1] Présentation de Redlean. <http://redlean.io/>, (2016). [En ligne; consulté le 26-Juin-2020]. 5
- [2] FLORENT Y. *Gestion de projet agile*. (2015). 6
- [3] M.FLORIS. *Conception participative, agile et scrum*. (2015). 7
- [4] Mavenlink. <https://www.mavenlink.com/>. [En ligne; consulté le 26-Juin-2020]. 8
- [5] Clarizen. <https://www.clarizen.com/>. [En ligne; consulté le 04-Juillet-2020]. 9
- [6] Asana. <https://asana.com/fr/>. [En ligne; consulté le 26-Juin-2020]. 9
- [7] Méthodologie 2TUP. <http://imilsoftware.blogspot.com/2017/01/processus-de-developpement-en-y.html>. [En ligne; consulté le 26-Juin-2020]. 13
- [8] Force.com. <https://www.lemagit.fr/definition/Forcecom>. [En ligne; consulté le 10-Juin-2020]. 30
- [9] Java Script. <http://glossaire.infowebmaster.fr/javascript/>. [En ligne; consulté le 17-Juin-2020]. 31
- [10] Apex. https://trailhead.salesforce.com/fr/content/learn/modules/platform_dev_basics/. [En ligne; consulté le 16-Juin-2020]. 31
- [11] Conception de la base de données. http://tecfaetu.unige.ch/staf/staf-h/tassini/staf2x/Heidi/last_bd.htm. [En ligne; consulté le 02-Juillet-2020]. 36
- [12] Salesforce. <https://www.actionco.fr/LivreBlanc/Partenaire/salesforce-386643/>. [En ligne; consulté le 02-Juillet-2020]. 49
- [13] Workbench. <https://www.appshark.com/blog/workbench-in-salesforce/>. [En ligne; consulté le 02-Juillet-2020]. 49
- [14] Star Uml. <https://www.sparxsystems.eu/>. [En ligne; consulté le 02-Juillet-2020]. 49
- [15] Règle de Workflow. https://trailhead.salesforce.com/en/content/learn/modules/business_process_automation/. [En ligne; consulté le 27-Juin-2020]. 51
- [16] Générateur de processus. https://trailhead.salesforce.com/en/content/learn/modules/business_process_automation/. [En ligne; consulté le 27-Juin-2020]. 52

ANNEXE 1 : SPÉCIFICATION ET RAFFINEMENT DES CAS D'UTILISATION

* Spécification de cas d'utilisation «Gérer les ressources»

- Diagramme raffiné

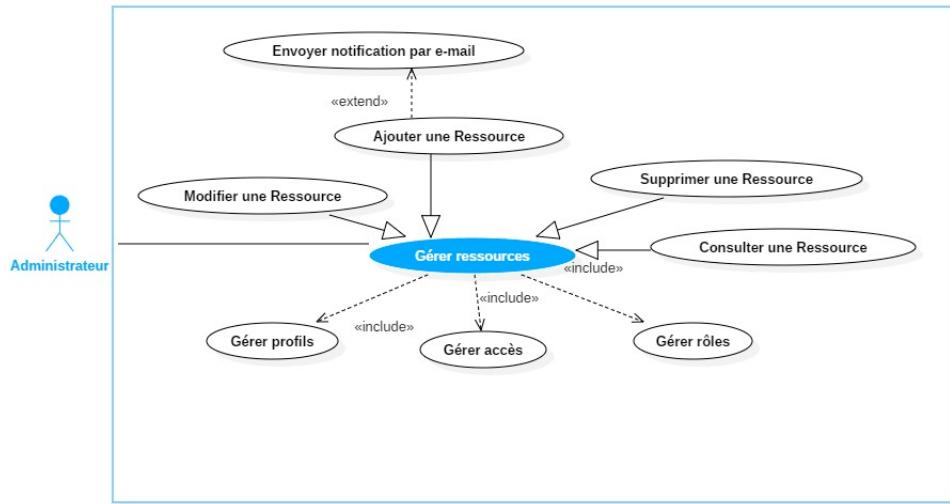


Figure A.1 – Diagramme de cas d'utilisation "Gérer les ressources"

* Spécification de cas d'utilisation «Consulter et gérer les priorités»

- Diagramme raffiné

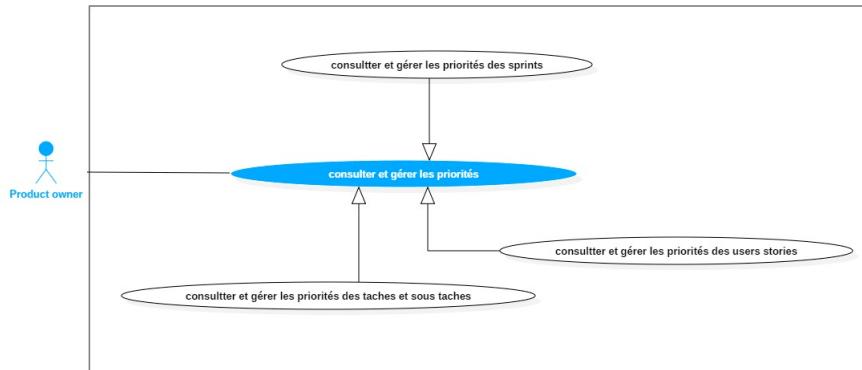


Figure A.2 – Diagramme de cas d'utilisation "Consulter et gérer les priorités"

* Spécification de cas d'utilisation «Gérer les sprints des projets»

- Diagramme raffiné

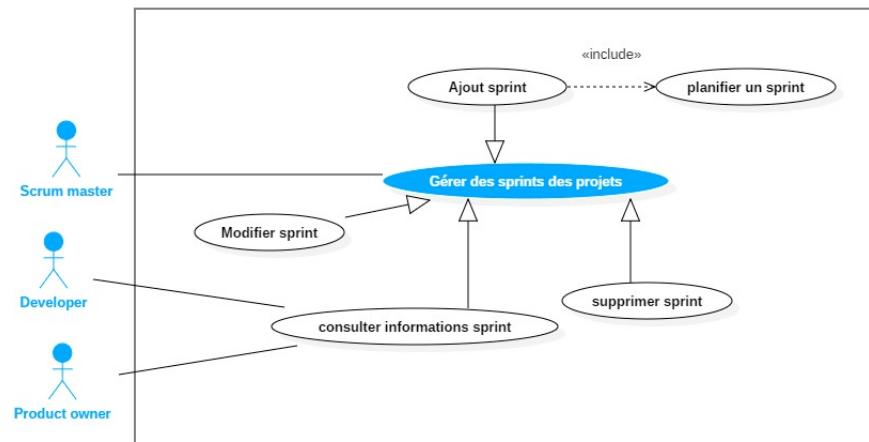


Figure A.3 – Diagramme de cas d'utilisation "Gérer les sprints des projets"

* Spécification de cas d'utilisation «Gérer les users stories des projets»

- Diagramme raffiné

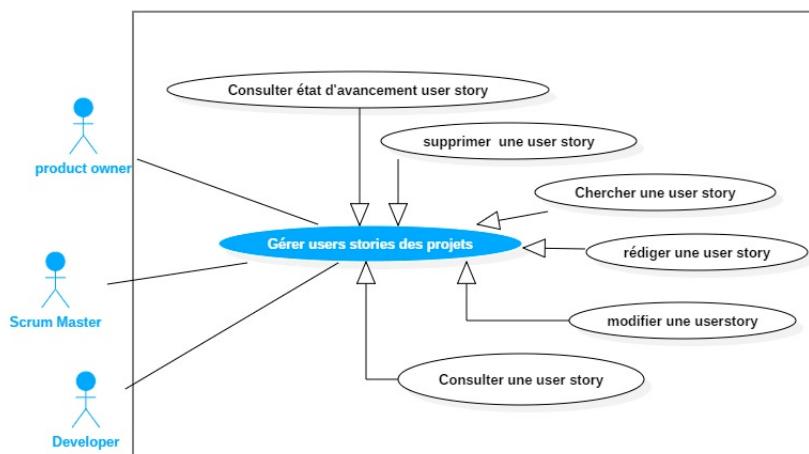


Figure A.4 – Diagramme de cas d'utilisation "Gérer les users stories des projets"

