|  |  |  |
| --- | --- | --- |
| logo FSM | République Tunisienne Ministère de l'Enseignement Supérieur et de la  Recherche Scientifique  Université de Monastir  Faculté des Sciences  Département d'Informatique |  |

|  |
| --- |
| Numéro d’ordre 62-LGLSI3-2022 |

**RAPPORT DE PROJET DE FIN D’ÉTUDES**

Présenté en vue de l’obtention du

Diplôme National de Licence en sciences de l’informatique et systèmes d’informations

Spécialité : Génie Logiciel et Systèmes d’Information

Réalisé par

Mouheb Mehdoui

|  |
| --- |
| Mise en place d’une application web de gestion et de  suivi de projets selon la méthodologie Agile Scrum |

Réalisé au sein de

Département des sciences de l’informatique de la faculté des sciences de Monastir

Soutenu le

06 Juin 2022

Devant les membres du jury :

|  |  |
| --- | --- |
| Président : | Pr Mounir Zrigui |
| Rapporteur : | Dr Cyrine Neffati |
| Encadrant Universitaire: | Dr Aymen Trigui |

Année universitaire 2021-2022

Dédicaces

Je dédie ce travail :

À mes très chers parents, pour leur amour, leurs sacrifices, leurs encouragements durant tout mon parcours. Je ne trouve aucun moyen qui me permet d’exprimer toute ma joie, ma reconnaissance et ma gratitude pour tout ce qu’ils ont fait pour moi. Que dieu leur accorde la santé, la joie et la longue vie.

À mes sœurs Manel, Ameni et Wided, merci pour votre soutien, vos encouragements et votre motivation qui me permet de continuer ce parcours dur. Je vous souhaite tout le bonheur, la joie et la réussite.

À tous les membres de ma famille, pour leurs tendresses, leurs considérations et leur soutien permanent.

A ma partenaire Khouloud, qui sans le savoir m’a tantôt servi de source d’inspiration, tantôt de modèle, tantôt de repoussoir, et à partir duquel j’ai forgé mon identité et que je suis devenu qui je suis.

À mes chers amis sans exception, pour l’amitié qui nous a unies et pour les merveilleux moments et nos souvenirs et les bons moments que nous avons vécu ensemble.

Mouheb Mehdoui

Remerciements

C’est avec un grand honneur et un grand plaisir que j’écris ces quelques lignes afin d’exprimer mes sentiments d’honneur, de reconnaissance et de respect pour toute personne qui m’ont soutenu de loin ou de près à l’accomplissement de ce projet.

À mon encadrant académique, Monsieur Aymen Trigui, pour son assistance, son soutien et ses encouragements durant toute les phases de réalisation de ce projet.

À tous mes enseignants, dès le début de mon parcours scolaire qui m’ont soutenu durant les moments de doute et d’abandon, qui ont plus cru en moi.

Je suis très reconnaissant à toutes les personnes qui ont contribué à l’élaboration de ce travail à savoir Madame Sirine Ben Hamida et Monsieur Ahmed Ben Hammouda pour leur soutien moral, leur encouragement et leur conseil et pour tout le temps qu’ils ont sacrifiées.

Mouheb Mehdoui

Table des matières

Introduction générale 1

Chapitre 1 : Contexte générale…………………………………………..2

Introduction 3

I. L’organisme d’accueil 3

I.1 Présentation de l’organisme d’accueil 3

II. Étude de l’existant 3

Introduction 3

II.1 Description de l’existant 3

II.2 Critique de l’existant 4

II.2.1 Présentation des outils existantes de gestion de projets agiles 4

I.1.1 Comparaison entre les outils existants de gestion de projets agiles 5

III. Solution proposée 6

IV. Choix de la méthodologie de travail 6

V. Planning prévisionnel 7

Conclusion 8

Chapitre 2 : Etat de l'art……….…………………………………………..8

Introduction 9

I. Autour de la culture de gestion de projets 9

I.1 Notion de projet 9

I.2 Gestion et suivi de projets 9

II. Les méthodes classiques de gestion de projets 9

II.1 La méthode de cycle en cascade 9

II.2 La méthode de cycle en V 10

II.3 Les limites des méthodes classiques de gestion de projets 11

V. Les méthodes modernes de gestion de projets : les méthodes agiles 12

V.1 De l’approche orienté projet à l’approche orienté produit 12

V.2 Définition des méthodes agiles 12

V.2.1 Historique des méthodes agiles 12

V.2.2 Le manifeste Agile 13

III.3.2.1 Les valeurs du manifeste agile 13

III.2.2.2 Les principes du manifeste agile 13

III.3 Les avantages des méthodes agiles 14

IV. Étude de de la méthodologie adoptée : Scrum 14

IV.1 Définition du Framework Scrum 14

IV.2 Historique de Scrum 14

IV.3 Les principes Scrum 15

III.3.1 Le contrôle empirique de procès 15

III.3.2 L’auto-organisation 16

III.3.3 La collaboration 16

III.3.4 La priorisation basée sur la valeur 16

III.3.5 Le Time Boxing 16

III.3.6 Le développement itératif 16

III.4 Les rôles Scrum 17

III.4.1 Les rôles essentiels 17

IV.4.1 Les rôles non essentiels 18

IV.5 Les artefacts Scrum 18

IV.6 Les cérémonies Scrum 20

IV.7 Fonctionnement de Framework Scrum 20

Conclusion 21

Chapitre 3 : Analyse et Conception……………..…………………..21

Introduction 22

I. Identification des acteurs 22

I.1 Acteur 22

III.2 Spécification des acteurs 22

II. Spécification des besoins 22

II.1 Les besoins fonctionnels 22

II.2 Les besoins non fonctionnels 25

III. Diagramme de cas d’utilisation globale 25

IV. Raffinement des cas d’utilisation 27

IV.1 L’acteur utilisateur ordinaire 27

IV.1.1 Cas d’utilisation « Gestion de profil » 27

IV.1.2 Cas d’utilisation « Gestion de messages » 27

IV.1.3 Cas d’utilisation « consulter les notifications » 28

IV.1.4 Cas d’utilisation « gestion de réunions » 28

V.2 L’acteur Product Owner 28

V.2.1 Cas d’utilisation « Gestion de projets » 28

V.2.2 Cas d’utilisation «Gestion de stakeholders d’un projet» 29

V.2.3 Cas d’utilisation « gestion de la Persona d’un projet » 29

V.2.4 Cas d’utilisation « gestion de journal de risques » 30

V.2.5 Cas d’utilisation « gestion de journal d’obstacles d’un projet » 30

V.2.6 Les Cas d’utilisation « gestion des user stories d’un projet » et « gestion du backlog d’un projet » ………………………………………………………………………………………………………………………………….31

V.2.7 Cas d’utilisation « gestion de sprints d’un projet » 32

V.3 L’acteur développeur 32

V.3.1 Cas d’utilisation « gestion des user stories d’un projet » 32

V.3.2 Cas d’utilisation « consulter la liste de ses tâches » 33

V.3.3 Cas d’utilisation « gestion de sprints d’un projet » 33

V.4 L’acteur Scrum Master 33

V.3.1 Cas d’utilisation « gestion de l’équipe de développement d’un projet » 34

V.3.2 Cas d’utilisation « gestion de sprints d’un projet » 34

VI. Conception détaillée 34

VI.1 Diagramme de classes 35

VI.1.1 Présentation de diagramme de classes 35

VI.1.2 Descriptions des cardinalités 35

VII.2 Diagrammes de séquences 37

VII.2.1 Digramme de séquence supprimer la persona d’un projet 37

VII.2.2 Digramme de séquence créer la persona d’un projet 37

VII.2.3 Digramme de séquence modifier la persona d’un projet 38

VII.2.4 Digramme de séquence supprimer un membre de l’équipe de développement 38

VII.2.5 Digramme de séquence gestion de la persona d’un projet 38

Conclusion 39

Chapitre 4 : Réalisation………………………………………………….39

Introduction 40

I. Environnement de travail 40

I.1 Environnement matériel 40

I.2 Environnement logiciel 41

I.2.1 Les outils logiciels 41

I.3 Choix technologique 41

II. Travail Réalisé 42

II.1 Étape de construction de l’application 42

II.1.2 Installation et mise en place de l’architecture logique 43

III.1.2.1 L’architecture MVC 43

III.1.3 Conception de l’identité visuelle 43

III.1.4 Conception des interfaces graphiques 43

III.1.5 Développement et codage 43

III.1.6 Le déploiement de l’application 44

IV. Solution réalisée 45

Conclusion 47

Conclusion générale 48

Liste des figures

Figure 1: Le logo de Jira 4

Figure 2: Le logo de Trello 4

Figure 3: Le logo de Notion 5

Tableau 1: Tableau comparatif des outils de gestion de projets 6

Figure 4: planning prévisionnel 7

Figure 5: les phases de modèle de cycle en cascade 10

Figure 6: les phases de modèle de cycle en V 11

Figure 7: les principes de Scrum 15

Figure 8 : Le processus de développement itératif 17

Figure 9: la forme générale d’une user story 19

Figure 10 : la forme générale d’un backlog de produit 19

Figure 11: La forme générale d’un backlog de sprint 19

Figure 12: Le cycle de réalisation d’un projet selon l’approche Scrum 21

Figure 13: Diagramme de cas d’utilisation globale 26

Figure 14: Diagramme de cas d’utilisation Gestion de profil 27

Figure 15: Diagramme de cas d’utilisation Gestion de messages 27

Figure 16: Diagramme de cas d’utilisation Consulter les notifications 28

Figure 17: Diagramme de cas d’utilisation Gestion de réunions 28

Figure 18: Diagramme de cas d’utilisation Gestion de projets 29

Figure 19: Diagramme de cas d’utilisation Gestion de stakeholders 29

Figure 20: Diagramme de cas d’utilisation Gestion de la persona d’un projet 30

Figure 21: Diagramme de cas d’utilisation Gestion de journal de risques 30

Figure 22: Diagramme de cas d’utilisation Gestion de journal d’obstacles 31

[Figure 24: Diagramme de cas d’utilisation Gestion de sprints d’un projet de l’acteur Product Owner 32](#_Toc104288126)

Figure 25: Diagramme de cas d’utilisation Gestion des user stories 333

Figure 26: Diagramme de cas d’utilisation Consulter la liste de ses taches 33

Figure 27: Diagramme de cas d’utilisation Gestion de sprints d’un projet 33

Figure 28: Diagramme de cas d’utilisation Gestion de l’équipe de développement d’un projet 34

Figure 29: Diagramme de cas d’utilisation Gestion de sprints d’un projet 34

Figure 30: Diagramme de classes implémenté 35

Figure 31: Diagramme de séquence supprimer la persona d’un projet 37

Figure 32: Diagramme de séquence créer la persona d’un projet 37

Figure 33: Diagramme de séquence modifier la persona d’un projet 38

Figure 34: Diagramme de séquence supprimer un membre de l’équipe de développement d’un projet 38

Figure 35: Diagramme de séquence gestion de la persona d’un projet 39

Figure 36: le schéma des étapes de réalisation de projet 44

Figure 37: l’architecture de modèle MVC 43

Figure 38: Diagramme de déploiement 44

Figure 39: interface de gestion de taches 46

Figure 40: interface de gestion de profil 46

Figure 41: interface de gestion de réunions 47

Figure 42: interface de messagerie In Mail 47

Figure 43: interface de backlog d'un projet 48

Figure 44: interface de la page d’accueil 47

Liste des tableaux

Tableau 1: Tableau comparatif des outils de gestion de projets 6

Tableau 2 : La configuration minimale requise 41

Tableau 3 : La configuration utilisée 41

Introduction générale

Tout au long de l’histoire humaine, les crises sanitaires, économiques et sociales ont été des opportunités à l’innovation et à la créativité.

À la fin des années 90 s, la Dot Com Bubble, qui était une crise économique grave, était une forte motivation pour l’adoption des méthodes agiles au sein des entreprises de production digitale.

Cette forte motivation aura lieu de nouveau depuis décembre 2019, la date de l’apparition de virus COVID-19 en Chine et sa grande propagation par la suite dans le monde entier.

Le 11 mars 2020, l’Organisation mondiale de la santé a qualifié le virus de COVID-19 en pandémie et depuis ce jour-là et jusqu’à présent, les gouvernements dans le monde entier sont invités à mettre en place un protocole sanitaire et des mesures de sécurité restrictives afin de réduire les dégâts de ce virus. Malgré les grandes restrictions, les entreprises dans le monde entier et surtout en France, Brésil, l’Italie, les États-Unis, l’Inde et en Angleterre, ont marqué des grandes pertes qui ont pris de différentes formes : faillite, licenciement, pertes dans les chiffres d’affaires…

Pour s’adapter avec cette nouvelle variation, les entreprises ont fortement cherché des outils informatiques pour supporter la culture de télétravail tout en conservant les méthodes agiles comme méthode de suivi et de gestion de projets.

Le présent projet de fin d’études se doit présenter un service efficace permettant aux équipes dans le monde entier de se collaborer et de gérer leurs tâches et de suivre aussi l’avancement de leurs projets et qui les aident à s’adapter aux nouvelles conditions de travail imposées par la pandémie de COVID-19.

Dans le cadre de cette évolution, il nous a été proposé de mettre en place une solution de suivi et de gestion de projets en adoptant le Framework Scrum étant une méthodologie souple et moderne.

Ce présent rapport fait l’objet d’une documentation expliquant les étapes suivies pour la réalisation de ce projet et qui compte cinq chapitres qui font le rôle d’une simple introduction du contexte général du projet suivi par un chapitre d’état d’art où nous faisons une étude détaillée des différentes méthodes de gestion et de suivi de projets et une étude du Framework Scrum adopté. Puis nous faisons l’étude technique de notre projet résumée dans les deux chapitres d'analyse et de spécification de besoins et de conception et enfin, nous consacrons le dernier chapitre pour expliquer la phase de réalisation de notre projet.

Introduction

Dans le présent chapitre, nous entamons le contexte général de notre projet. Nous commençons tout d’abord par la présentation de l’organisme qui nous a accueillis pendant la réalisation de notre projet. La deuxième partie sera consacrée à l’étude de l’existant au cours de laquelle nous ferons la critique des solutions existantes de gestion et de suivi de projet suivi de notre solution pour résoudre ces problèmes. La dernière partie est dédiée à la méthodologie adoptée ainsi que l’élaboration d’un planning prévisionnel qui nous a permis d’organiser nos tâches et à mettre en place un plan de travail efficace.

1. L’organisme d’accueil

## Présentation de l’organisme d’accueil

Le département des sciences de l’informatique est l’un des quatre départements qui composent la Faculté des sciences de Monastir. Ce département assure la formation d’une licence en sciences de l’informatique, d’un mastère de recherche en système de raisonnement informatique, de deux mastères professionnels le premier en sciences de données et le deuxième en ingénierie des systèmes d’informations et assure aussi une formation de doctorat et d’habilitation universitaire en informatique.

1. Étude de l’existant

## Introduction

Une bonne planification et l'étude détaillée sont insuffisantes pour assurer la réussite d'un projet. En effet, malgré d'éventuelles difficultés externes, le planning doit être strictement suivi. Pour des besoins organisationnels et pour améliorer le processus de réalisation et le suivi d'un projet et le suivi de sa progression, il nous a été proposé de construire une solution pour suivre les projets et les gérer afin de répondre exactement aux besoins des entreprises.

## Description de l’existant

Malgré le grand choix des solutions logicielles de gestion et suivi de projets, plusieurs entreprises gérer jusqu'à aujourd'hui leurs projets d'une façon manuelle ce qui peut engendrer des pertes de temps des retards et une mal-organisation des responsabilités entre les membres de l'équipe de réalisation d'un projet. Pour résoudre cela, de nombreuses solutions ont été réalisées pour résoudre ces problèmes, mais à cause de la crise sanitaire depuis décembre 2019, la plupart des entreprises ont opté la culture de travail à distance ce qui augmente alors le trafic sur les solutions existantes de gestion et suivi de projets. Cette migration vers les outils en ligne a conduit à changer les mesures de coût qualité vers des mesures de performance, de qualité et de scalabilité face à la grande demande de ce genre de solutions.

## Critique de l’existant

Lorsqu'une entreprise décide d'opter un outil de gestion et le suivi de projets agiles, l'entreprise constatera que de nombreux outils ciblent le même besoin, mais de manière et des méthodes très différentes pour la mise en œuvre. Ces différences rendent certains outils meilleurs que d'autres. Pour cela, dans les deux sections suivantes nous nous intéressons par la présentation des outils les plus populaires de gestion de projets agiles puis nous ferons une de comparaison pour présenter les divergences de chaque outil par rapport aux autres.

* + 1. Présentation des outils existants de gestion de projets agiles

Les outils les plus populaires de gestion et suivi de projets sont :

* JIRA

Jira est une application utilisée pour le suivi des problèmes et la gestion de projets. L'outil, développé en 2002, est devenu largement utilisé par les équipes de développement agiles pour suivre les erreurs, les histoires, les épopées et d'autres détails. [1]



Figure 1: Le logo de Jira

* Trello

Trello est un outil de gestion de projets en ligne, inspiré de la méthode Kanban. Il est organisé en fonction des éléments des cartes de la liste du tableau, chaque carte représentant une tâche. [2]



Figure 2: Le logo de Trello

* Notion

Notion est un logiciel de gestion de projets et de prise de notes conçus pour aider les membres d'une entreprise ou d'une organisation à coordonner les délais, les objectifs et les tâches afin d'augmenter l'efficacité et la productivité. [3]



Figure 3: Le logo de Notion

* + 1. Comparaison entre les outils existants de gestion de projets agiles

Après la phase de recherche des outils populaires de gestion de projets agiles qui a conduit à relever Jira, Trello et Notion nous devons ensuite expliquer les critères de comparaison que nous utilisons.

La recherche a abouti à relever les critères suivants :

* Le coût

Pour les entreprises, un service de qualité ne suffit pas, le cout d’utilisation est un facteur assez important.

* L’évolutivité

L’accès des utilisateurs d’un service en ligne peut être simultané (l’heure de pointe). Ce qui rend nécessaire d’utiliser un modèle de déploiement qui permet d’équilibrer la charge d’accès pour garantir un service de qualité et stable indépendamment de nombre de demandeurs de service.

* La clarté de l’application de l’approche agile

Ce critère est le plus important parmi tous les autres critères, car comment il peut être un excellent outil de gestion de projets s'il ne garde pas une vision claire des artefacts et des fondements des méthodes agiles ?

* Les outils de communication

Ces outils sont des outils de base nécessaires pour garder un fil de communication entre les membres d’une équipe sans être obligés d’utiliser un service externe qui entraine la perte de temps et de budget pour l’entreprise.

* La génération automatique des chartes

Les chartes agiles sont très importantes pour le succès d'un projet. L'utilisation d'un outil externe à l'entreprise présente plusieurs inconvénients tels que le coût supplémentaire, la perte de temps et un taux d'erreurs élevé.

Après la recherche des critères capables de faire une comparaison équitable entre les différents outils de gestion de projets, nous devons maintenons générer un tableau comparatif qui met l'accent sur les avantages et les inconvénients de chaque solution comme le montre le tableau **1**.

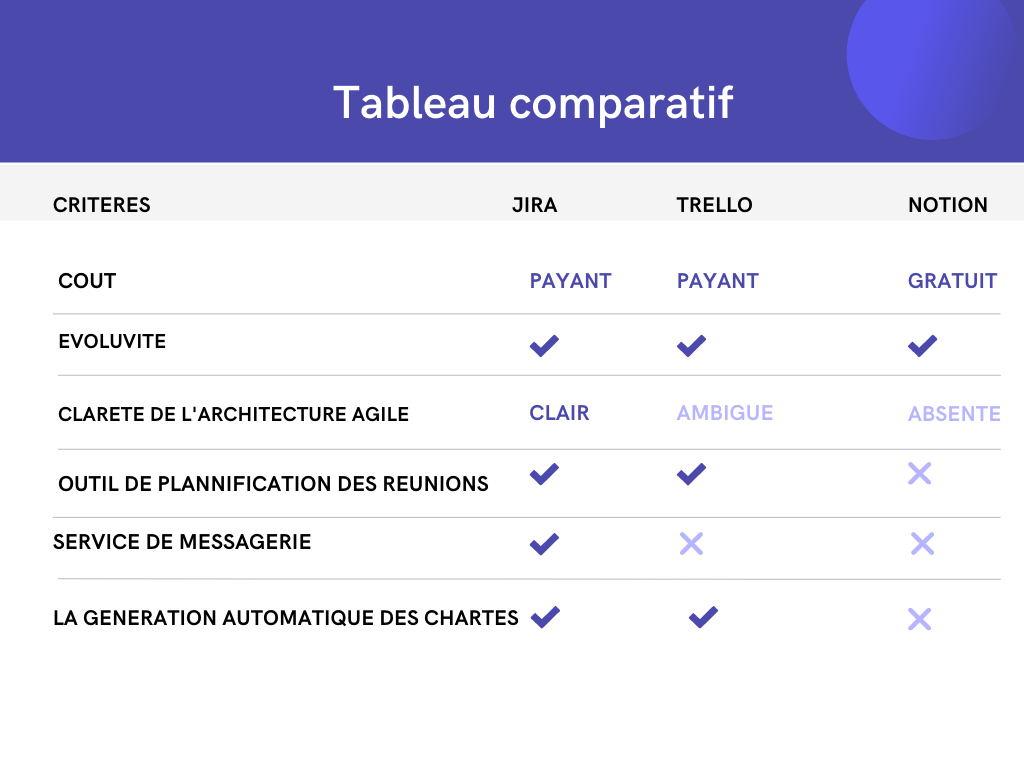


Tableau 1: Tableau comparatif des outils de gestion de projets



## Solution proposée

La première remarque que nous devons apporter est que les attentes de différentes parties prenantes par rapport à cette solution sont sensiblement différentes.

En effet, les parties prenantes telles que les clients cherchent principalement un outil pour superviser le travail des équipes de développement de leurs projets d'une manière aisée qui part d'une vue globale et arrivera aux plus petits détails. Ils cherchent aussi de se bénéficier d'un service efficace, stable et garantit un certain niveau de sécurité et d'organisation autour de leurs projets.

En ce qui concerne les développeurs de projets, l'idéal pour eux serait de disposer d'un système efficace pour la collaboration en temps réel afin de pouvoir superviser l'avancement de leurs tâches et d'accompagner leurs projets depuis la création de vision globale jusqu'à la livraison de produit finale au client. Notre solution dont également afficher à chaque développeur ses tâches dont il est responsable de le terminer et affiche aux chefs de projet une interface lui permet de suivre l'avancement de leurs équipes, de communiquer avec eux et de gérer aussi les tâches externes relatives à leurs projets tels que leurs réunions. Ils cherchent aussi à avoir un certain niveau de stabilité et un endroit pour sauvegarder leurs informations dans un endroit sécurisé.

4. Choix de la méthodologie de travail

Pour avoir une meilleure compréhension de notre système et de tirer le maximum possible d'informations à partir de notre expérience personnelle avec l'approche Agile Scrum, nous décidons d'opter les approches Agiles en tant que méthodologie de travail pour des diverses raisons.

En effet, ce modèle garantit une meilleure collaboration et une meilleure communication avec les différentes parties prenantes de notre projet.

Dans notre cas, où le projet est réalisé en monôme, nous essayons donc d'adapter l'approche Agile à nos besoins. Ainsi nous faisons usage de bonnes pratiques suivantes :

* La réalisation du projet sur un ensemble des sprints.
* Des réunions quotidiennes avec l’encadrant pour le suivi de l’avancement.
* Une réunion chaque q intervalle régulier de temps ou nous essayons de répondre à ces trois questions :
* Quelles sont les tâches terminées ?
* Quels sont les obstacles rencontrés ?
* Quelles sont les tâches que nous planifions à réaliser au futur ?

1. Planning prévisionnel

L'élaboration du planning prévisionnel des différentes tâches de notre projet est une phase très importante qui nous permet d'organiser nos tâches et de savoir aussi l'ordonnancement des tâches selon leurs priorités. Pour ces raisons, nous décidons d’utiliser le diagramme de Gantt pour faire la mise en place d’un planning prévisionnel pour la réalisation de ce projet comme le montre la figure **4**.



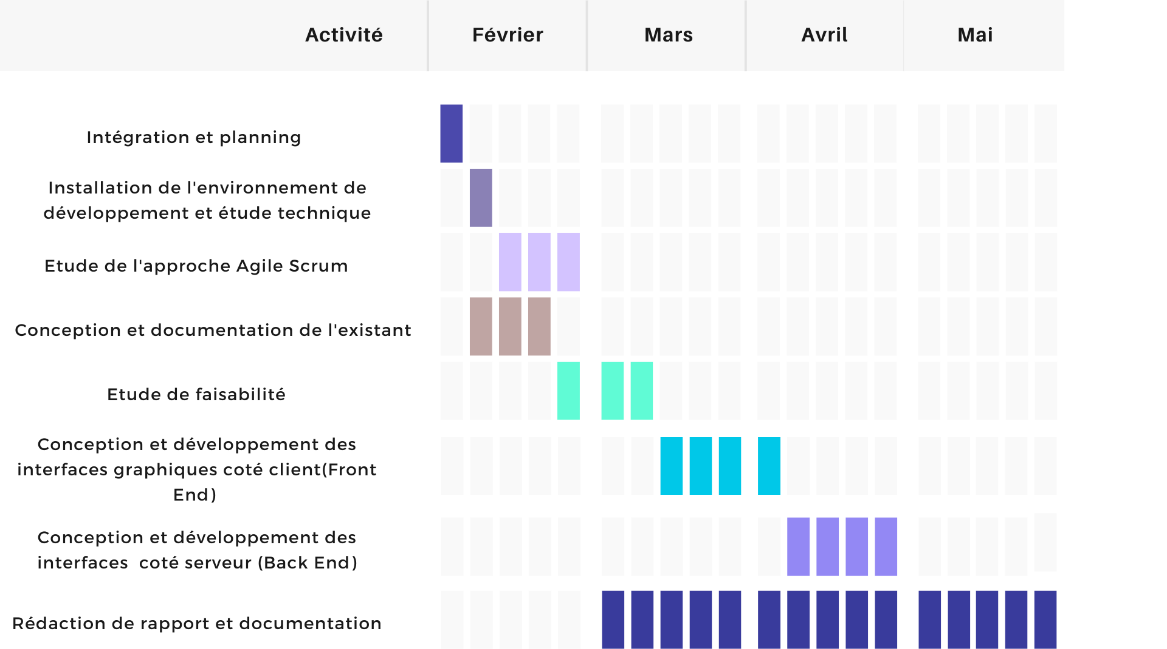


Figure 4: planning prévisionnel

Conclusion

Dans le présent chapitre, nous avons présenté l’organisme qui nous a accueillis pendant la réalisation de ce projet puis nous avons étudié l’existant et enfin nous avons fourni une description de la méthodologie de travail que nous avons adoptée pour réussir à réaliser ce projet ainsi que l’élaboration d’un planning prévisionnel des différentes activités de notre projet. Dans le chapitre qui suit, nous allons faire l’exposition de l’état de l’art.

Introduction

Dans le présent chapitre nous nous intéressons par souligner les grands axes sur lesquelles s’articule notre projet. Nous commencer par aborder des définitions clé .Ensuite, nous expliquant les méthodes classiques et les méthodes modernes de gestion de projets en enfin nous étudions le Framework Scrum.

1. Autour de la culture de gestion de projets

## Notion de projet

Un projet est une entreprise unique et transitoire qui vise à atteindre des objectifs programmés qui peuvent être définies en termes de produits, avantages ou encore des résultats dans les limites de budget, de qualité et de délai.

## Gestion et suivi de projets

Ces deux termes sont généralement interconnectés lorsqu’il s’agit d’exprimer l’application de processus, des méthodes et de connaissance et d’expériences dont l’objectif et d’atteindre des objectifs relatifs à la spécification des besoins de projet et qui assure à la livraison d’un produit final conforme aux critères d’acceptation du projet. La différence fondamentale entre la gestion de projets et la gestion classique est que la gestion de projets s’agit d’un produit final à livrer a contrainte de temps, de qualité et coût.

1. Les méthodes classiques de gestion de projets

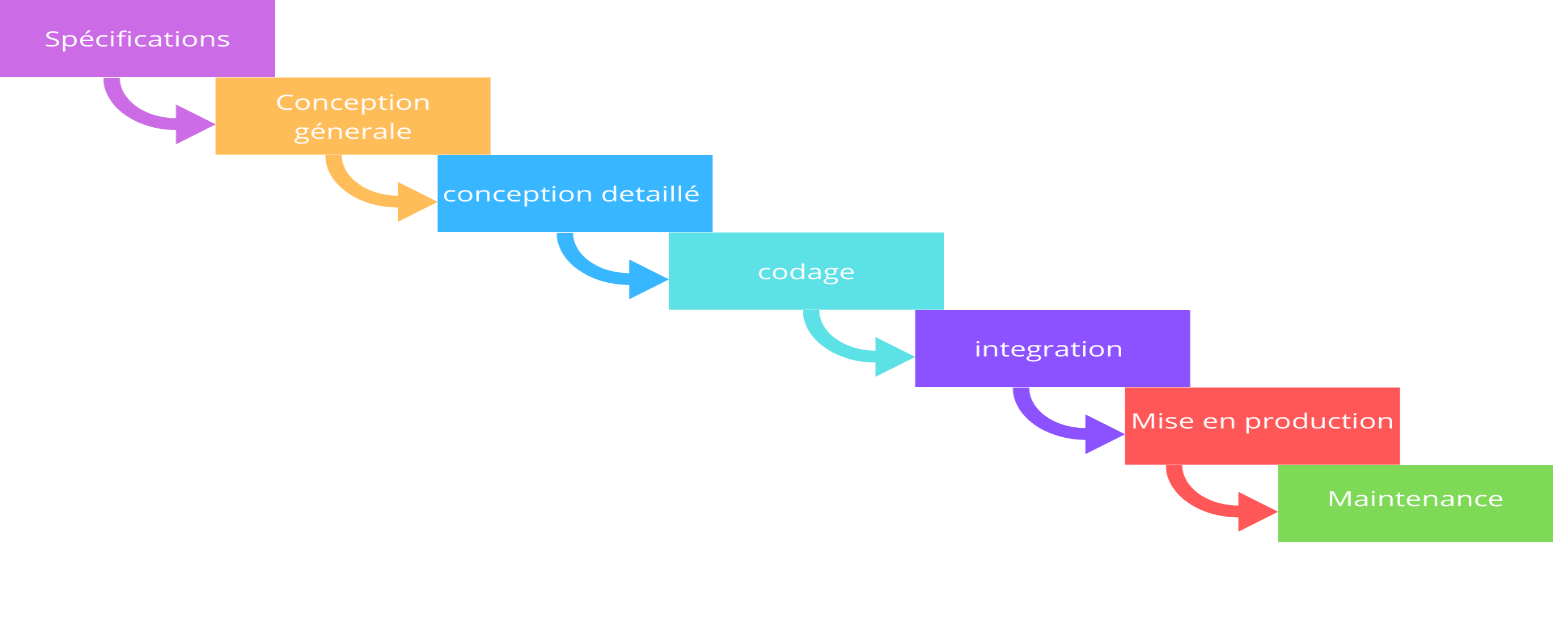
Lors de l’utilisation d’une approche traditionnelle de gestion de projets, le projet, ses fonctionnalités et ses fonctions finales sont définis à l’avance. Cette méthodologie se caractérise par l’utilisation d’un processus strict la rédaction d’une documentation très raffinée et détaillée et une intervention très faible de client.

Lorsqu’on parle des méthodes classiques de gestion de projets, on parle nécessairement de la méthode de cycle en cascade et la méthode de cycle en V qui sont les méthodes les plus utilisées.

## La méthode de cycle en cascade

Le principe de la méthode de cycle en cascade s’articule sur une succession des phases où chacune de ces phases doit être terminée avant le passage à la phase suivant. Lors de l’utilisation d’une méthode de cycle en V, toute demande de changement dans la spécification du projet est très couteuse pour l’entreprise.

La figure **5** modélise les phases de réalisation d’un projet selon la méthode de cycle en cascade.

Figure 5: les phases de modèle de cycle en cascade

* La phase de spécification

Le maitre d’ouvrage exprime ses besoins et ses attentes.

* La phase de conception générale

L’équipe de projet analyse les besoins du client et procède aux spécifications fonctionnelles ainsi que le traçage des grands axes du projet.

* La phase de conception détaillée

L’équipe de projet s’attaque à la spécification formelle du projet et l’expression des besoins à travers un langage de modélisation.

* La phase de codage

Durant cette phase, l’élaboration du produit en lui-même est effectuée.

* La phase d’intégration

Tout au long de cette phase, plusieurs tests sont effectués afin de vérifier la conformité du produit avec la spécification.

* La phase de mise en production

Le produit est mis en service afin d’être délivré au client.

* La phase de maintenance

Durant l’utilisation réelle du produit, des erreurs et des vulnérabilités peuvent être apparues ou l’équipe de projet doit appliquer des corrections évolutives ou correctives au niveau du produit.



## La méthode de cycle en V

Le modèle de cycle en V reprend le même principe de succession des séquences

sauf qu’on associe à chaque étape descendante (corresponds à la conception de produit) une étape ascendante (corresponds à la validation de produit).

La figure ci-dessous explique l’ordre des phases dans le modèle de cycle en V.

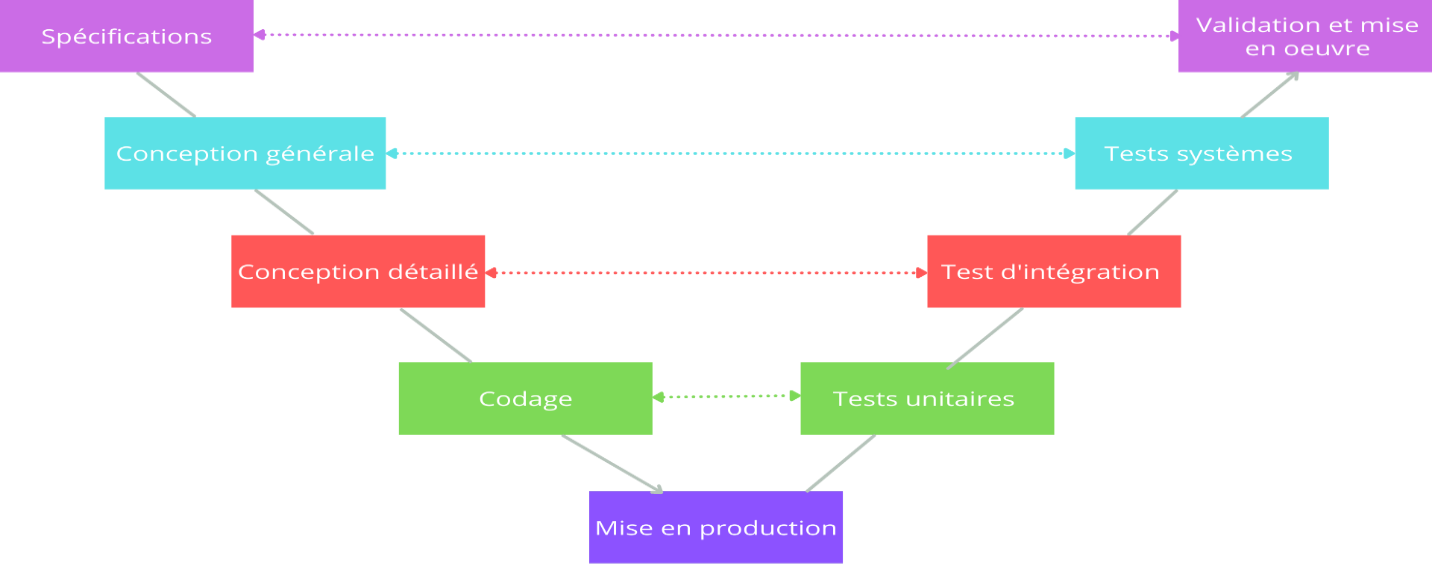


Figure 6: les phases de modèle de cycle en V

* La phase de maintenance

Durant l’utilisation réelle du produit, des erreurs et des vulnérabilités peuvent être apparues ou l’équipe de projet doit appliquer des corrections évolutives ou correctives au niveau du produit.

* La phase des tests unitaires

Le principe des tests unitaires s’agit de petits tests qui valident l’attitude d’une

portion de code afin de dégager les vulnérabilités.

* La phase des tests d’intégration

L’équipe de développement construit le produit en faisant l’assemblage des parties ensemble et puis elle effectue des tests pour mesure la fiabilité de produit.

* La phase des tests système

Les tests système sont effectués en partenariat avec les futurs utilisateurs du produit qui doivent vérifier la conformité du produit implémenté avec leurs besoins.

## Les limites des méthodes classiques de gestion de projets

Même si les méthodes classiques présentent certains points forts, elles présentent aussi certaines limites qui les rendent inutiles dans certains cas.

Entre autres :

* Un cadre de travail rigide

Lorsque vous utilisez une des méthodes classiques, il y a peu de possibilités de changement ou de retour en arrière à cause du cout très élevé de ce genre d’actions.

* Des tests trop tardifs

Pour les méthodes classiques, les tests interviennent généralement à la fin de processus de développement. Ainsi, tout résultat négatif pourrait déclencher la nécessité d’une révision importante et très couteuse.

1. Les méthodes modernes de gestion de projets : les méthodes agiles

## De l’approche orientée projet à l’approche orientée produit

La culture produit est plus qu’une nouvelle mode, c’est une réponse pertinente, au développement des certaines idées et à la nécessité de réfléchir pour et avec le client. En mode produit, le but n’est ni la date ni le coût, mais la satisfaction de client et l’amélioration permanente du produit pour qu’il soit en phase avec son marché. En mode projet, le projet s’arrête quand l’équipe projet fait la livraison du produit, par contre en mode projet, le projet commence vraiment lorsqu’on fait la livraison du produit et puis l’amélioration continue en se basant sur les retours du client.

## Définition des méthodes agiles

Les méthodes agiles caractérisent une approche de gestion de projets privilégiant le dialogue entre toutes les parties prenantes, les partenaires, les développeurs et autres professionnels du projet, la souplesse en cours de réalisation du projet, la capacité à modifier les plans et la rapidité de livraison et l’élasticité de s’adapter aux changements.

L’approche agile vise à livrer le bon produit en suivant un processus incrémental et itératif par contradiction aux méthodes classiques qui utilisent un processus de développement incrémental et itératif à la fois.

* + 1. Historique des méthodes agiles

Sans trop compliquer les choses, les premières apparitions des méthodes agiles étaient entre **1930** et **194**0 à travers les recherches de **Walter Schwehat** et **William Edward Deming**. Ces recherches étaient introduites au développement des produits informatiques environ **1950**. En fait, les méthodes agiles sont à la collaboration des nombreuses approches de conduite de projets tels que ceux de **Tom Gilb**, **Scott Shultz**, **Brian Gallagher** et celui d’**Alex Balchin**.

La première officialisation et documentation de la première méthode agile est celle de **Evo** de **Tom Gilb** date de **1976**.

Une des dates très importantes dans l’historique des méthodes agile était le weekend de **13** et **14** **février 2001**, une réunion de **17** personnes expertes de gestion de projets à **Utah** aux **États** **unis** pour une mission de recherche d’une base commune aux différentes méthodes agiles.

Cette réunion a abouti à une documentation officielle des principes et les valeurs agiles qui deviennent connues aujourd’hui sous l’appellation de **Manifeste** **Agile**. [4]

* + 1. Le manifeste Agile

Connu aussi sous l’appellation de manifeste pour le développement Agile de logicielles, le manifeste Agile est une présentation formelle des axes principaux de l’approche agile de conduite de projet pour une approche orientée produit, itérative et incrémentale et centrée sur les personnes. Le manifeste agile est une combinaison de quatre valeurs et douze principes fondamentaux.

3. 3. 2. 1. Les valeurs du manifeste agile

Le manifeste agile s’étend sur quatre valeurs principales qui sont :

* Les individus et leurs interactions plus que les processus et les outils
* Des logiciels opérationnels plus qu’une documentation exhaustive
* La collaboration avec les clients plus qu’une négociation contractuelle
* L’adaptation au changement plus que le suivi d’un plan

3. 2. 2. 2. Les principes du manifeste agile

Le manifeste agile s’étend aussi sur douze principes qui font des méthodes agiles une approche efficace pour la gestion de projets.

* Satisfaire le client en livrant rapidement et régulièrement des fonctionnalités a grande valeur ajoutée.
* Décomposer les travaux importants en composants plus petits rapidement réalisables.
* Ne reconnaitre que les meilleures conceptions émergentes d’équipes auto-organisées.
* Fournir à des personnes motivées l’environnement et le soutien dont elles besoin de leur donner confiance pour atteindre les objectifs fixés.
* Créer des processus qui encouragent un rythme de travail soutenable.
* Maintenir un rythme de développement constant.
* Les utilisateurs ou leurs représentants et les développeurs doivent travailler ensemble quotidiennement tout au long du projet.
* A intervalles réguliers, l’équipe réfléchit aux moyens de devenir plus efficaces puis règlent et modifient leurs comportements en conséquence.
* Mesurer l’avancement en fonction du travail accompli.
* Rechercher continuellement l’excellence technique.
* Exploiter le changement pour donner un avantage compétitif au client.
* Accueillir positivement les changements de besoins même tard dans le projet

## Les avantages des méthodes agiles

Étant être l’approche la plus adoptée à nos jours pour faire la gestion de projets, les méthodes agiles présentent un certain nombre davantage. Entre autres nous citons :

* Moins de documentation et donc plus de productivité et celui-ci est le plus importante pour le client.
* Souplesse, élasticité et réactivité pour les demandes de changements
* Le client reçoit à intervalles réguliers une version fonctionnelle du produit ce qui lui permet alors d’améliorer le produit en se basant sur les retours du client.

1. Étude de la méthodologie adoptée : Scrum

## Définition du Framework Scrum

Scrum est une démarche de gestion de projets qui intègre le client en tant que le principal leader du projet et qui collabore avec l’équipe responsable du développement du projet.

## Historique de Scrum

La première apparition de métaphore de Scrum était en **1986** à travers l’œuvre de **Hirotaka Takeushi** et **Ikujro Nonaka** sous le tuteur **The New Product Development Game** qui fait l’objet d’un guide qui introduit l’adoption d’une nouvelle approche de développement de nouveaux produits. En **1995**, **Ken** **Schwaber** et **Jeff Sutherland** ont présenté au monde les fondements de la méthode de Scrum à L’**Oopsla** à **Austin** aux **États unis**. [5]

## Les principes Scrum

Scrum exigent un règlement composé de six principes fondamentaux qui sont :

* Le contrôle empirique du processus
* L’auto-organisation
* La collaboration
* La priorisation basée sur la valeur
* Le time-boxing
* Le développement itératif

La figure ci-dessous résume les principes de bases du Framework Scrum.



Figure 7: les principes de Scrum

* + 1. Le contrôle empirique de procès

Ce principe est basé sur les trois idées principales de transparence, d’adaptation et de l’inspection.

* Adaptation

Scrum exige que l’équipe de développement s’adapte aux demandes de changements de la part de clients et modifier les objectifs finaux afin de satisfaire le client.

* Inspection

L’équipe de développement doit se consulter quotidiennement à intervalle

régulier afin de dégager d’éventuels écarts entre les objectifs finaux du projet et le travail réalisé.

* Transparence

Durant tout le cycle de développement d’un produit, tous les membres de l’équipe de développement et leurs partenaires ont accès à un espace commun ou ils ont accès aux informations relatives aux projets.

* + 1. L’auto-organisation

Lorsque les membres d’une équipe de développement sont auto-organisés, celui-ci va contribuer à créer un environnement de travail simple et confortable donc plus de productivités.

* + 1. La collaboration

La communication, l’entraide et la collaboration sont des valeurs clés de la méthode

Scrum. La gestion de projets est considérée comme un processus qui exige une communication efficace entre tout individu afin d’avoir une vision globale et commune de projet.

* + 1. La priorisation basée sur la valeur

L’équipe Scrum vise à offrir le maximum de valeur ajoutée au produit dans une période de temps très limités. Afin de garantir cet objectif,

Scrum utilise la priorisation basée sur la valeur comme l'un des principes fondamentaux régissant la structure et la fonctionnalité de l'ensemble du cadre Scrum. Il profite aux projets grâce au développement adaptatif et itératif d'un produit ou d'un service. Plus important encore, Scrum vise à fournir un produit ou un service de valeur au client de manière précoce et continue.

* + 1. Le Time Boxing

Le temps est considéré comme une contrainte qui doit être utilisée et respectée afin de gérer l’organisation et la réalisation de projet. Dans Scrum, les sprints, les réunions et les mêlées sont tous des évènements limités dans le temps afin d’imposer des contraintes temporelles.

* + 1. Le développement itératif

Ce principe définit l’approche de développement en itération et met l’accent sur comment l’équipe de développement gère efficacement les changements et

comment livrer des produits qui satisfait les exigences des parties prenantes.

La figure **8** montre les phases nécessaires lors de l’application d’une approche de développement itératif.

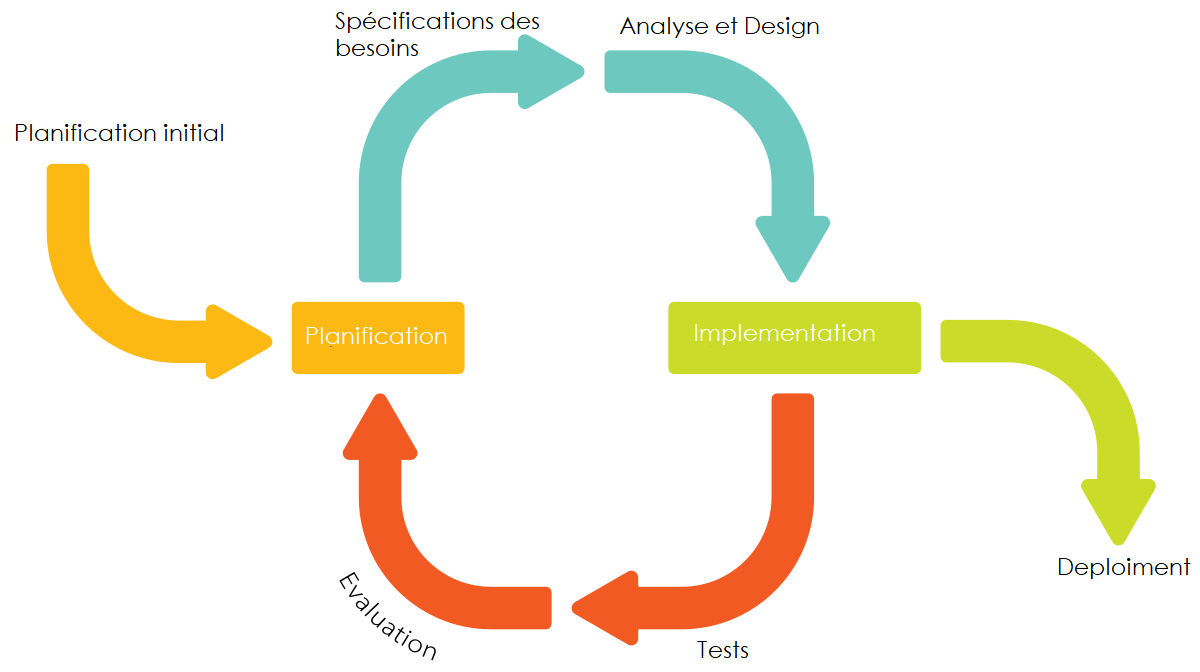


Figure 8 : Le processus de développement itératif

## Les rôles Scrum

Le Framework Scrum compte deux catégories fondamentales des rôles : les rôles essentiels et les rôles non essentiels.

* + 1. Les rôles essentiels

Les individus ayant des rôles essentiels sont pleinement impliqués dans le projet et responsable de la réussite du projet et aussi à la livraison d’un produit fonctionnel à chaque laps de temps.

Les rôles essentiels incluent :

* Le Product Owner

C’est la voix de client Le Product Owner est la personne responsable d’atteindre la valeur commerciale maximum de projet et il est chargé de transmettre les exigences du client à l’équipe responsable du développement de produit.

* Le Scrum Master

Le Scrum Master est la personne la plus expérimentée dans le Framework Scrum et qui enseigne les pratiques Scrum à toutes les personnes qui participent du projet et élimine tous les obstacles pour l’équipe et exige le respect de processus Scrum.

* L’équipe Scrum

C’est le groupe responsable à interpréter les exigences imposées par le Product

Owner et travaille pour livrer un produit fonctionnel à chaque intervalle de temps.

* + 1. Les rôles non essentiels

Cette catégorie inclut des individus intéressés par le projet et qui n’ont pas un rôle défini, mais qui doivent avoir un moyen pour communiquer avec l’équipe de développement.

* Les parties prenantes

Connu aussi sous le nom de stakeholders, c’est un terme qui regroupe les clients, les utilisateurs et les sponsors.

* Le Scrum guidance Body

C’est un rôle facultatif qui peut être soit un ensemble de documents ou un groupe d’expert qui s’intéresse à la définition des mesures de qualité, aux légalisations gouvernementales.

* Les fournisseurs

Ce rôle inclus les individus ou organisations externes qui fournissent des produits et/ou des services qui ne pas inclues dans les compétences de base de projet.

## Les artefacts Scrum

Le mot artefact dans le contexte Scrum fait référence aux composants de base permettant une transmission fluide, claire et précise d’informations entre les membres de l’équipe de développement d’un projet.

Les artefacts Scrum sont :

* User Story

Une User Story est une explication informelle et générale d’une fonctionnalité logicielle écrite du point de vue de l’utilisateur final de produit pour expliquer comment une fonctionnalité logicielle ou physique apportera de la valeur au client.

La figure **9** présente le format général d’une User Story.

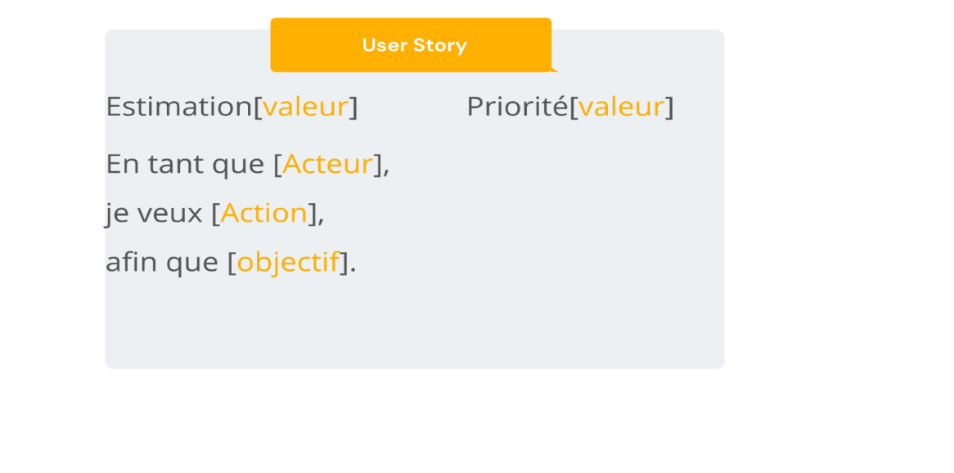


Figure9: la forme générale d’une user story

* Le Backlog de produit

Un backlog de produit est une liste de travaux prioritaires pour l’équipe de développement, établie à partir de la feuille de route et de ses exigences. Les éléments présents dans le Backlog de produit sont ordonnés et triés selon leur priorité.

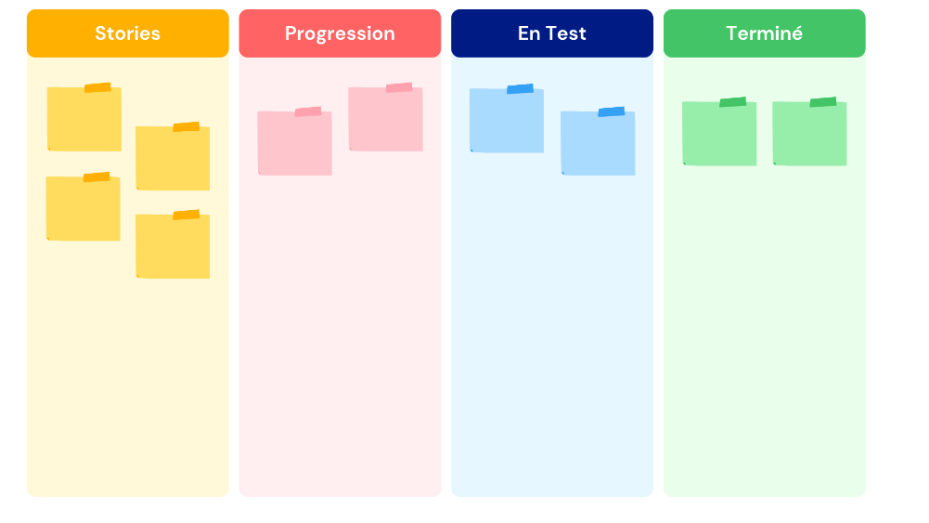


Figure 10 : la forme générale d’un backlog de produit

* Le Backlog de Sprint

Un backlog de sprint est l’ensemble des éléments qu’une équipe de production sélectionne dans son backlog de produit pour travailler sur le sprint à venir.

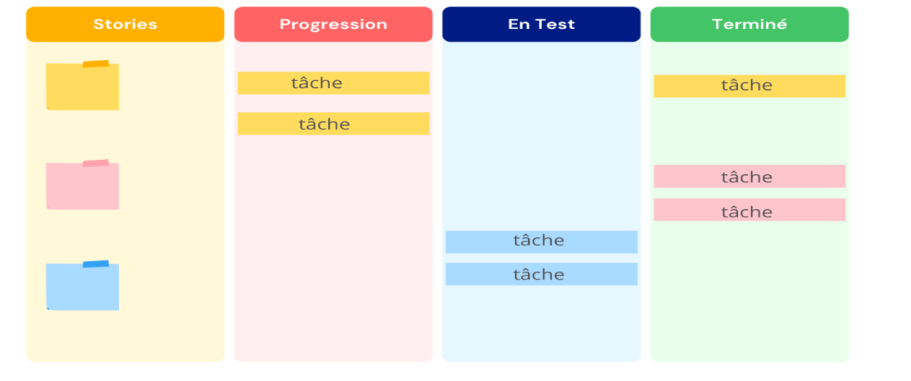


Figure 11: La forme générale d’un backlog de sprint

## Les cérémonies Scrum

Scrum présente des évènements essentiels dans chaque projet et qui sont limites dans le temps. Les délais Scrum sont :

* Sprint

Un sprint est une itération qui dure entre une et six semaines au cours de

laquelle l’équipé de développement et le Scrum Master collaborent afin de créer

des incréments fonctionnels de produit.

* Mêlée quotidienne

La mêlée quotidienne ou le Daily meeting est une réunion quotidienne de 15 minutes. Le but de cette mêlée est de répondre aux trois questions suivantes :

Que ce que j’ai fait depuis la dernière réunion ?

Que ce que je planifie de faire d’ici à la prochaine réunion ?

Y a-t-il des obstacles que j’ai rencontrés en ce moment ?

* Réunion de planification de sprint

Cette réunion aura horaire juste avant le déclenchement d’un nouveau sprint. Elle est limitée à 8 heures pour un sprint d’un mois. Le but de cette réunion est de définir les objectifs de sprint ainsi que l’identification et l’estimation de tâches.

* Réunion de revue de sprint

Au cours de cette réunion, l’équipe Scrum expose en présence du Product Owner le résultat de sprint et présente les incréments de sprint. À la fin de cette réunion, les incréments seront comparés aux critères d’acceptation du projet et seront classifiés soit en acceptées ou rejetées. Pour un sprint de quatre mois, une réunion de revue de sprint est limitée à quatre heures.

* Réunion de rétrospective de sprint

Cette réunion se limite à quatre heures pour un sprint de quatre semaines et qui est effectuée afin de mettre l’accent sur les processus suivis au cours de sprint passé, de les évaluer, l’équipé évalue aussi les outils utilisés, expose les obstacles rencontrés afin de tirer des renseignements.

## Fonctionnement de Framework Scrum

Une fois les user stories sont collectés en se basant sur la vision du produit de client, elles seront classées par ordre de priorité dans ce qu’on appelle le backlog de produit.

Dès que le backlog de produit est priorisé, les user stories seront subdivisés en un ensemble de tâches.

La prochaine étape de Scrum sur l’étape d’exécution de sprint durant laquelle l’équipe Scrum travaille sur l’accomplissement des tâches afin de livrer un produit fonctionnel à la fin de sprint .La progression des exigences de backlog de sprint est évaluée lors des réunions d’équipe .

Une fois le sprint est terminé, tous les membres de l’équipe Scrum participent aux revues de sprint et aux rétrospectives de sprint.

La phase finale de chaque cycle de sprint est la phase d’estimation du produit ou le produit est évalué par le client

Pour résumer tout ce que nous avons dit, la figure ci-dessous présente le cycle de réalisation d’un projet suivant l’approche Scrum.

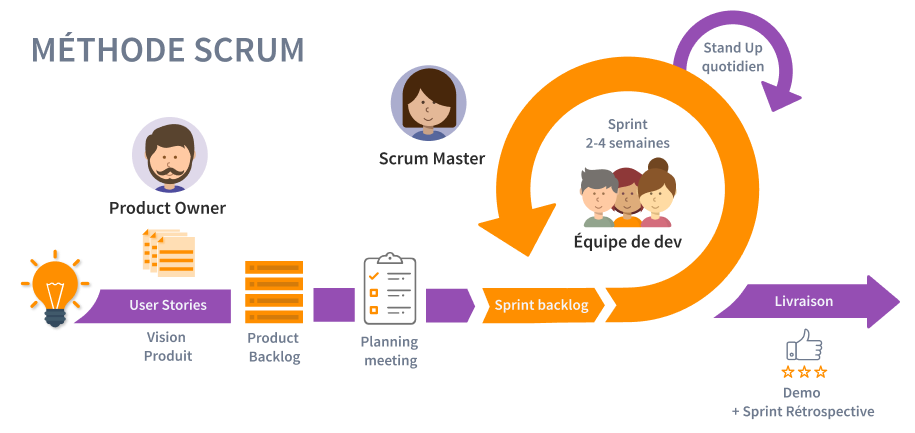


Figure 12: Le cycle de réalisation d’un projet selon l’approche Scrum[6]

Conclusion

Dans le présent chapitre, nous essayons de mettre l’accent sur les méthodes classiques et modernes de gestion de projets ainsi que leurs limites et avantages et nous finissons par l’étude de Framework Scrum. Dans le chapitre qui suit, nous allons commencer l’étude technique de notre système.

## Introduction

La phase de spécification des besoins est nécessaire pour la détermination des fonctionnalités attendues du système. En effet, dans le présent chapitre, nous entamons tout d’abord les acteurs de notre système. Puis nous élaborons l’étude des besoins fonctionnels et les besoins non fonctionnels qui seront modélisés sous forme des diagrammes de cas d’utilisation. Enfin nous allons présenter le diagramme de classes et les diagrammes de séquences.

1. Identification des acteurs

## Acteur

Tout système interactif doit interagir avec ses intervenants. Un acteur est une entité qui définit le rôle joué par un utilisateur (être humain ou un autre système) qui interagit avec le système à travers les différentes interfaces fournies.

## Spécification des acteurs

Dans le cas de notre projet, nous spécifions les acteurs du système qui se divergent en quatre catégories de base :

* Utilisateur ordinaire
* Le Product Owner
* Le Scrum Master
* Le développeur

1. Spécification des besoins

Dans cette section, nous allons mettre l’accent sur les besoins fonctionnels ainsi que les besoins non fonctionnels de notre système.

## Les besoins fonctionnels

Les besoins fonctionnels sont les besoins qui doivent répondre aux exigences du futur système de point de vue fonctionnalités.

* La gestion de profil

Cette tâche consiste à :

* Consulter son profil et ses informations
* Éditer les informations de son profil.
* La gestion des messages

Cette tâche consiste à :

* Consulter la liste de ses messages (les messages reçus, envoyés et supprimés).
* Rédiger un nouveau message.
* Supprimer un message.
* Consulter un message spécifique.
* La gestion des réunions

Cette tâche consiste à :

* Consulter la liste de ses réunions.
* Planifier une nouvelle réunion.
* Annuler une réunion.
* Inviter des utilisateurs à assister une réunion.
* La gestion des notifications

Cette tâche consiste à :

* Consulter la liste de ses notifications.
* Supprimer une notification.
* La gestion de projets

Cette tâche consiste à :

* Créer un nouveau projet.
* Supprimer un projet.
* Modifier un projet.
* Terminer un projet.
* Consulter la liste de ses projets.
* La gestion des tâches

Cette tâche peut être consiste à :

* Créer des nouvelles tâches.
* Consulter la liste de ses tâches.
* Marquer l’état d’une tâche (terminé, en cours, abandonné).
* La gestion de journal d’obstacles

Cette fonctionnalité doit garantir les actions suivantes :

* Consulter la liste d’obstacles déclarés.
* Déclarer un nouvel obstacle.
* Résoudre un obstacle.
* Supprimer un obstacle.
* Modifier la description d’un obstacle.
* La gestion de journal des risques

Cette fonctionnalité doit garantir les actions suivantes :

* Consulter les risques déclarés d’un projet.
* Modifier un risque déclaré.
* Supprimer un risque.
* La gestion des user stories

Cette fonctionnalité doit garantir les actions suivantes :

* Consulter la liste des user stories d’un projet.
* Créer une nouvelle user story.
* Supprimer une user story.
* Prioriser les user stories.
* Générer le backlog de produit à partir des user stories.
* La gestion de l’équipe de développement

Cette fonctionnalité doit garantir les actions suivantes :

* Consulter la liste des développeurs d’un projet.
* Ajouter un membre à l’équipe de développement.
* Modifier le rôle d’un membre de l’équipe.
* Dissocier un membre de l’équipe.
* Rechercher un profil spécifique.
* La gestion des sprints

Cette fonctionnalité consiste à :

* Consulter l’historique des sprints d’un projet.
* Consulter les chartes de sprint en cours.
* Annuler un sprint.
* Créer un nouveau sprint.
* Déclencher un nouveau sprint.
* La gestion des tâches relatives aux user stories

Cette fonctionnalité doit garantir les actions suivantes :

* Consulter la liste des tâches relatives aux user stories.
* Créer une nouvelle tâche.
* Supprimer une tâche.
* Modifier une tâche.
* La gestion des stakeholders

Cette tâche consiste essentiellement à :

* Consulter la liste des stakeholders d’un projet.
* Désigner un nouveau Stakeholder.
* Supprimer un Stakeholder.
* Modifier le rôle d’un Stakeholder.
* La gestion de persona de projet

Cette tâche consiste essentiellement à :

* Consulter la persona d’un projet.
* Créer la persona d’un projet.
* Supprimer la persona d’un projet.
* Modifier la persona d’un projet.

## Les besoins non fonctionnels

Les besoins non fonctionnels sont mesures de performance, de type de matériel ou aussi le type de conception et qui peuvent concerner les contraintes d'implémentation (langage de programmation, type SGBD, de système d'Exploitation...). Pour cela notre application devra se caractériser par :

* La maintenabilité

Le code source de notre application doit être lisible, bien commenté dont le but est d’assurer son état évolutif et extensible par rapport aux besoins de l’entreprise et aussi extensible pour des demandes de correction et de maintenance ultérieures

* La fiabilité

Notre application doit satisfaire les besoins des parties prenantes en offrant un système fonctionnel sans erreur ou vulnérabilité.

* La scalabilité ou l’évolutivité

La scalabilité de notre application est définie comme sa capacité à augmenter ou à diminuer ses performances en réponse à des changements dans les demandes et les requêtes de traitement.

1. Diagramme de cas d’utilisation globale

L'étude approfondie des spécifications de notre système nous permet de

dégager les cas d'utilisation présentés dans la figure **13**.

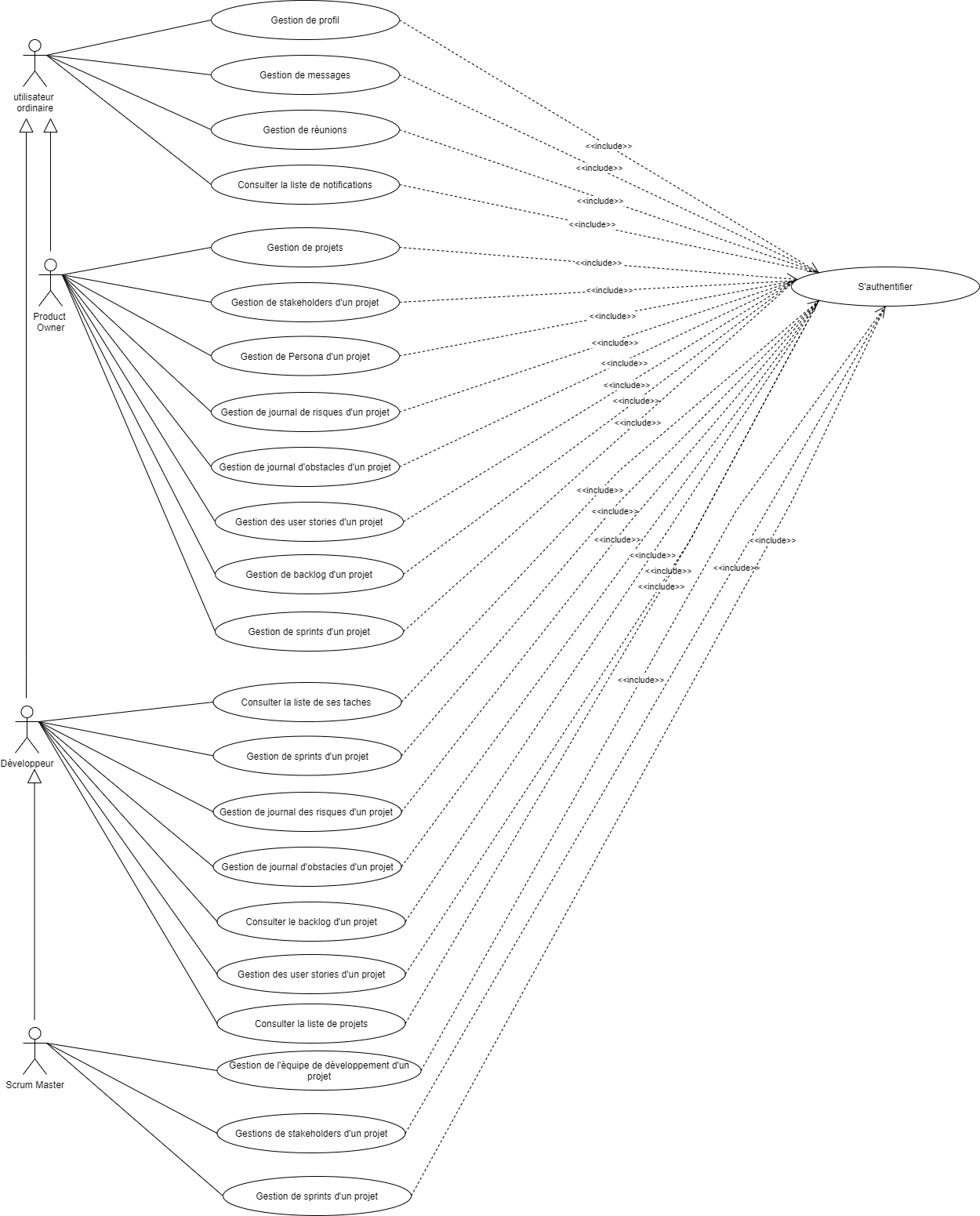


Figure 13: Diagramme de cas d’utilisation globale

1. Raffinement des cas d’utilisation

Dans cette partie, nous allons présenter les diagrammes de cas d’utilisation raffinés classés par acteur.

## L’acteur utilisateur ordinaire

L’acteur ordinaire est défini dans notre système comme un utilisateur n’ayant aucun projet et aucun rôle.

* + 1. Cas d’utilisation « Gestion de profil »

La figure ci-dessous représente le diagramme de cas d’utilisation relatif à **la gestion de profil.**

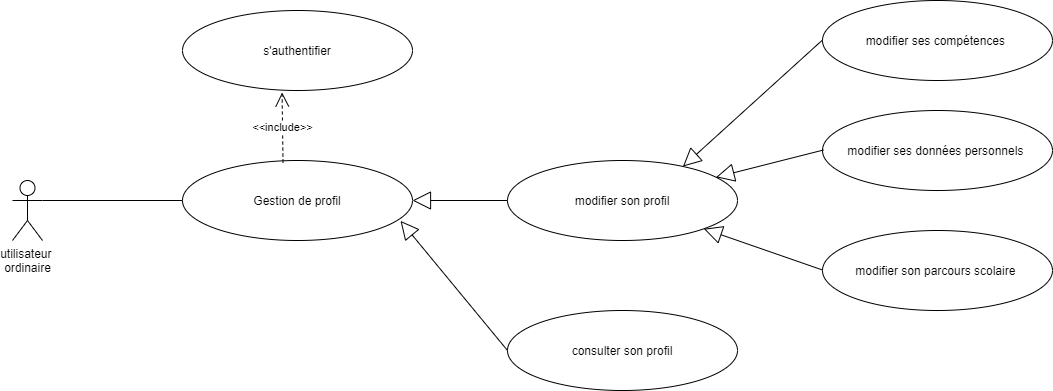


Figure 14: Diagramme de cas d’utilisation Gestion de profil

* + 1. Cas d’utilisation « Gestion de messages »

La figure suivante représente le diagramme de cas d’utilisation relatif à **la gestion de messages.**

Figure 15 : Diagramme de cas d’utilisation Gestion de messages

* + 1. Cas d’utilisation « consulter les notifications »

La figure 16 représente le diagramme de cas d’utilisation relatif à **la gestion de notifications.**

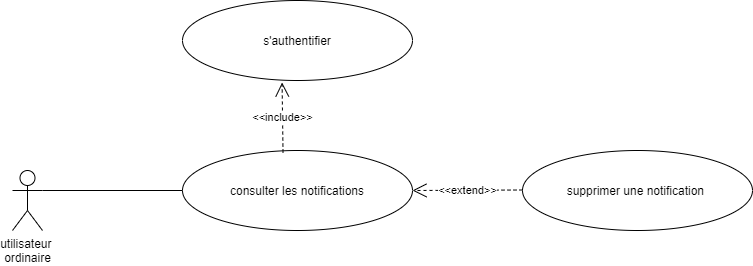


Figure 16: Diagramme de cas d’utilisation consulter les notifications

* + 1. Cas d’utilisation « gestion de réunions »

La figure ci-dessous représente le diagramme de cas d’utilisation relatif à **la gestion de réunion.**

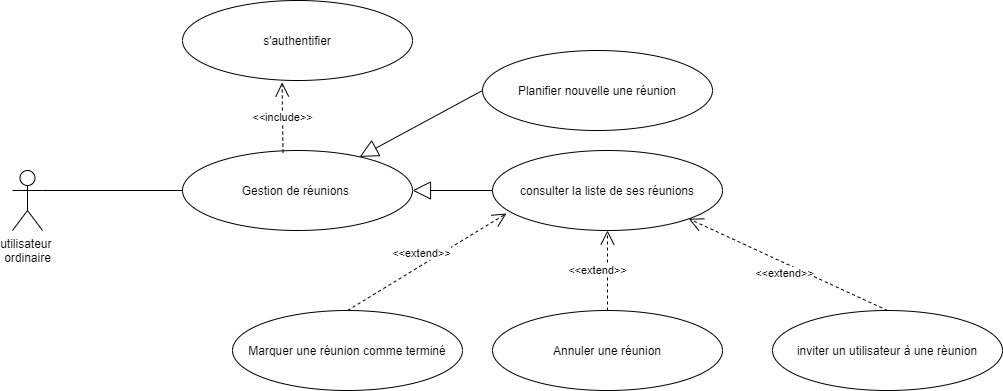
****

Figure 17: Diagramme de cas d’utilisation Gestion de réunions

## L’acteur Product Owner

Le Product Owner d’un projet est défini dans notre système comme le créateur de ce projet.

1. 2. 1. Cas d’utilisation « Gestion de projets »

La figure **18** représente le diagramme de cas d’utilisation **gestion de projets.**

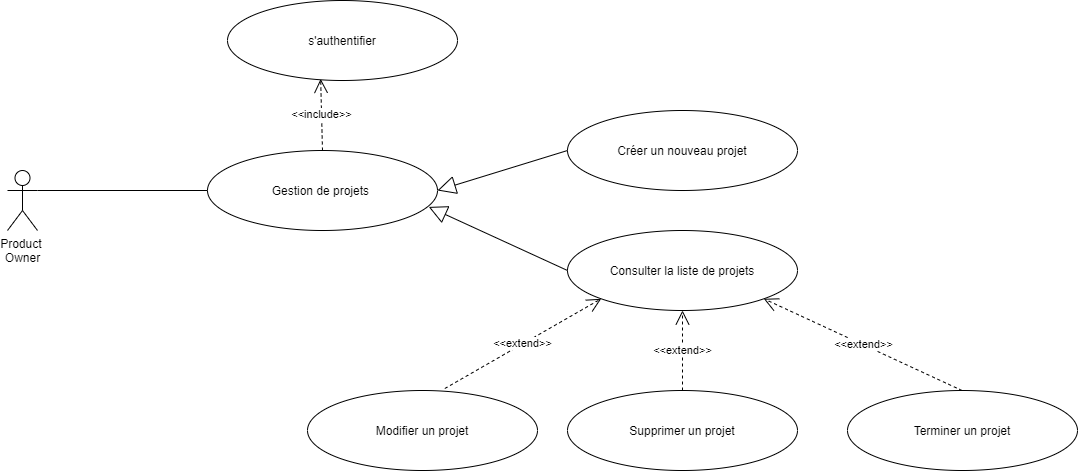


Figure 18: Diagramme de cas d’utilisation Gestion de projets

* + 1. Cas d’utilisation «Gestion de stakeholders d’un projet»

La figure **19** présente le diagramme de cas d’utilisation **Gestion de stakeholders d’un projet.**

Ce cas d’utilisation peut être effectué par les deux acteurs Scrum Master et Product Owner de la même façon ce qui est présenté dans le diagramme par une relation d’héritage.

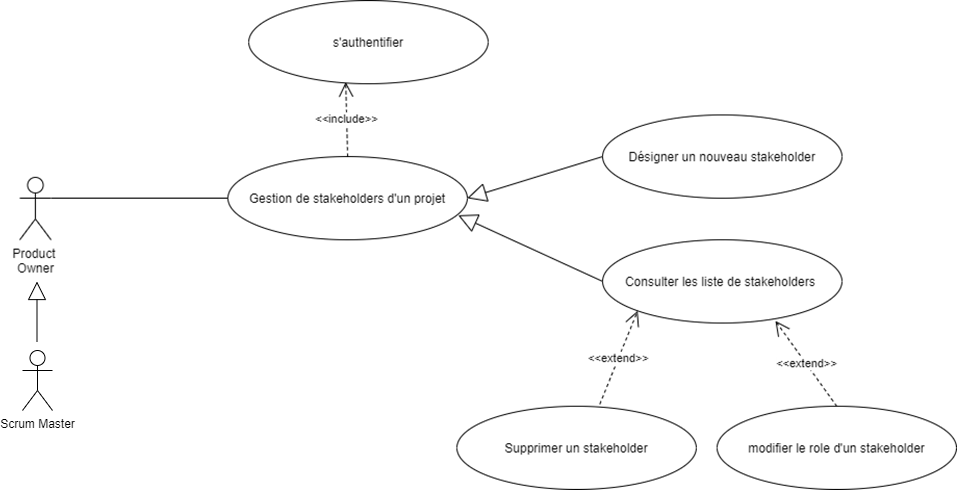


Figure 19: Diagramme de cas d’utilisation Gestion de stakeholders

* + 1. Cas d’utilisation « gestion de la Persona d’un projet »

La figure **20** le diagramme de cas d’utilisation **Gestion de la Persona d’un projet.**

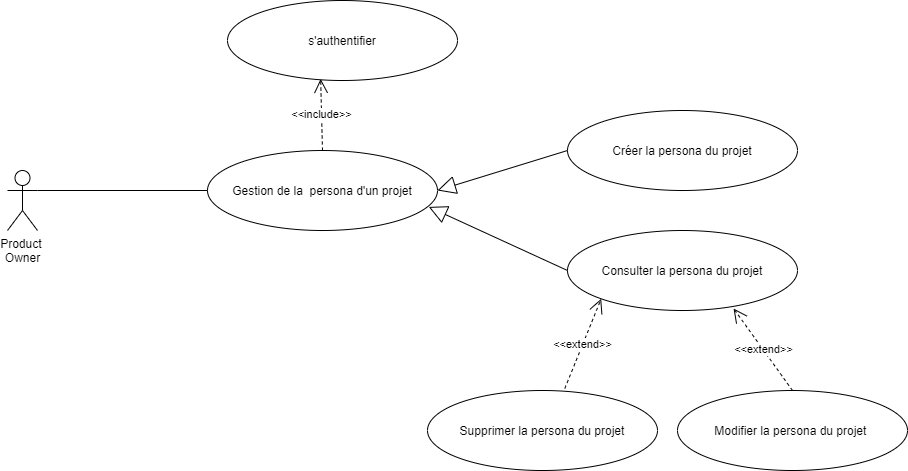


Figure 20: Diagramme de cas d’utilisation Gestion de la persona d’un projet

* + 1. Cas d’utilisation « gestion de journal de risques »

La figure **21** représente le diagramme de cas d’utilisation **Gestion de journal de risques.**

Ce cas d’utilisation peut être effectué par les acteurs Scrum Master, Product Owner et l’acteur développeur de la même façon ce qui est présenté dans le diagramme par une relation d’héritage.

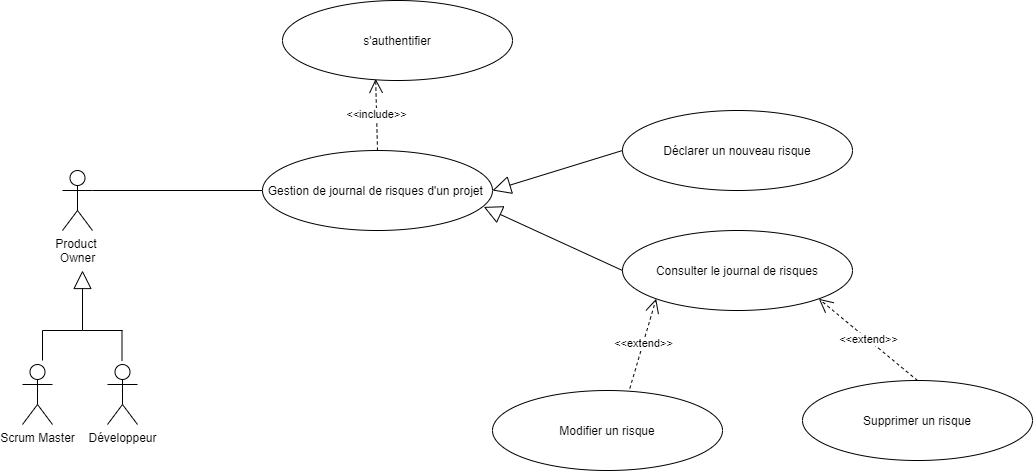


Figure 21: Diagramme de cas d’utilisation Gestion de journal de risques

* + 1. Cas d’utilisation « gestion de journal d’obstacles d’un projet »

La figure **22** représente le diagramme de cas d’utilisation **Gestion de journal**

**d’obstacles d’un projet.**

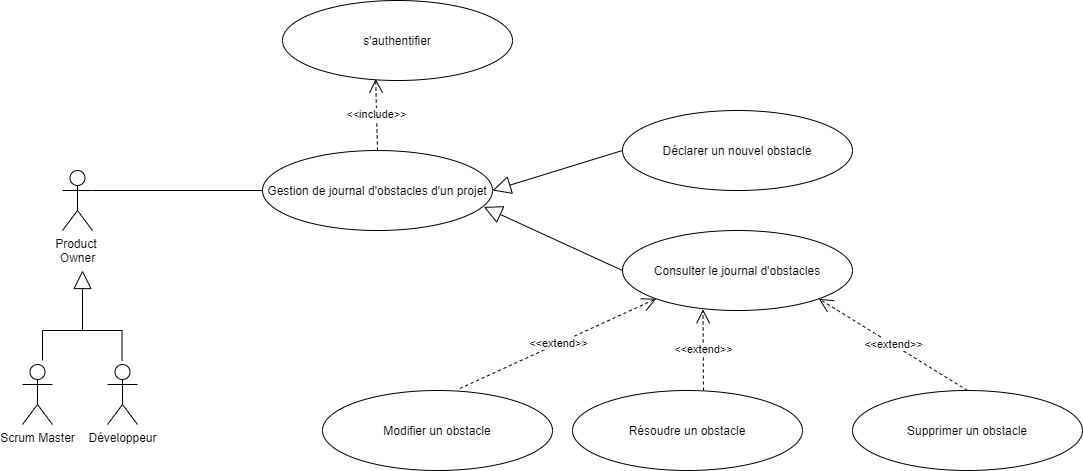
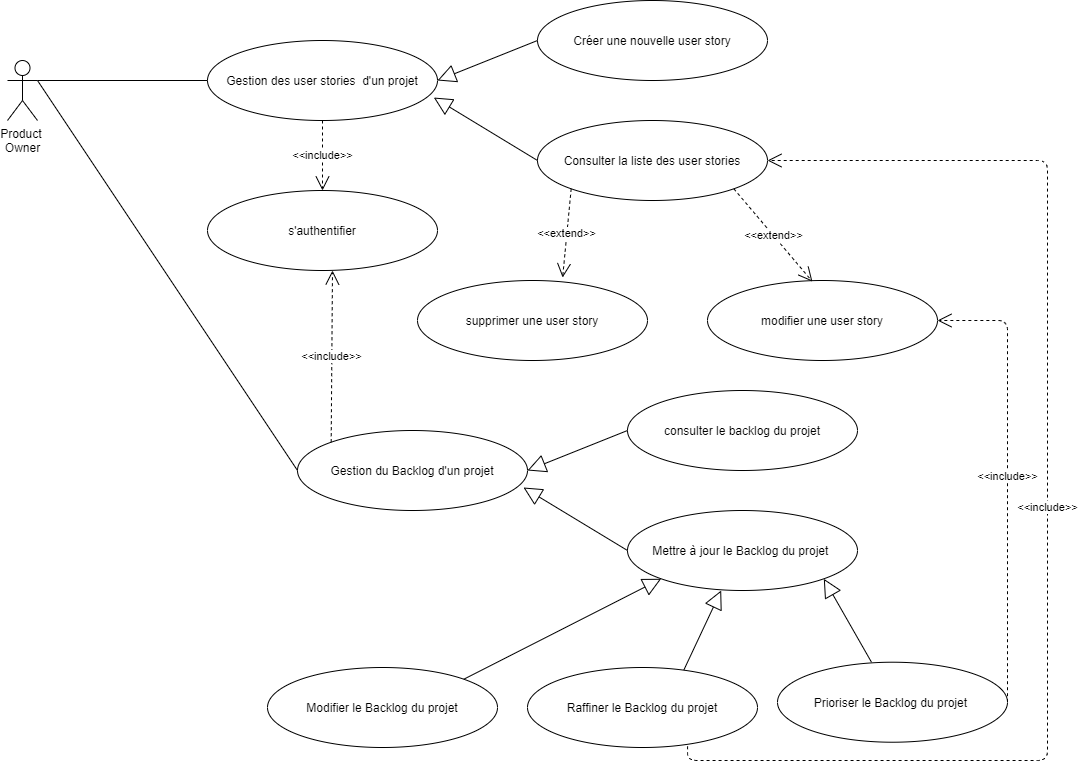


Figure 22: Diagramme de cas d’utilisation Gestion de journal d’obstacles

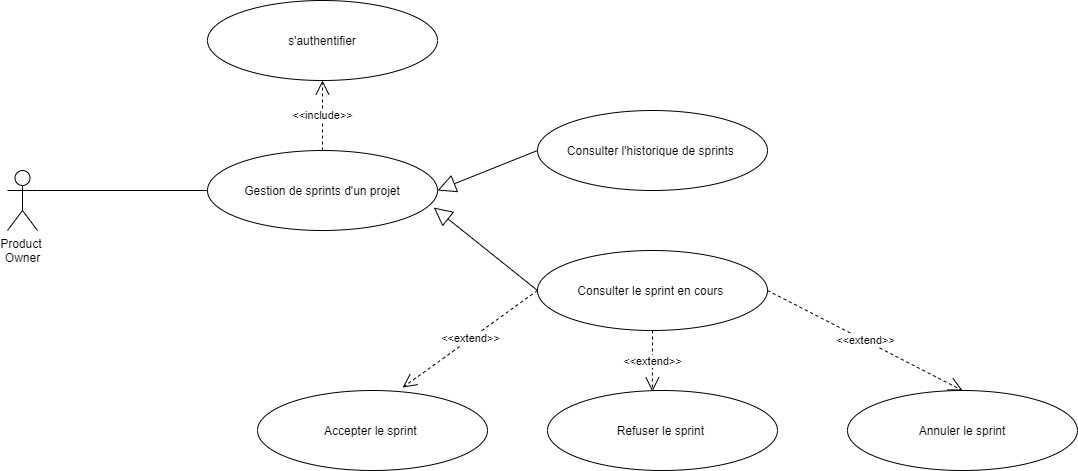
* + 1. Les Cas d’utilisation « gestion des user stories d’un projet » et « gestion du backlog d’un projet »

La figure suivante représente le diagramme de cas d’utilisations **Gestion des user stories d’un projet.**

Figure 23: Diagramme des cas d’utilisation Gestion des user stories et gestion du backlog d’un projet

* + 1. Cas d’utilisation « gestion de sprints d’un projet »

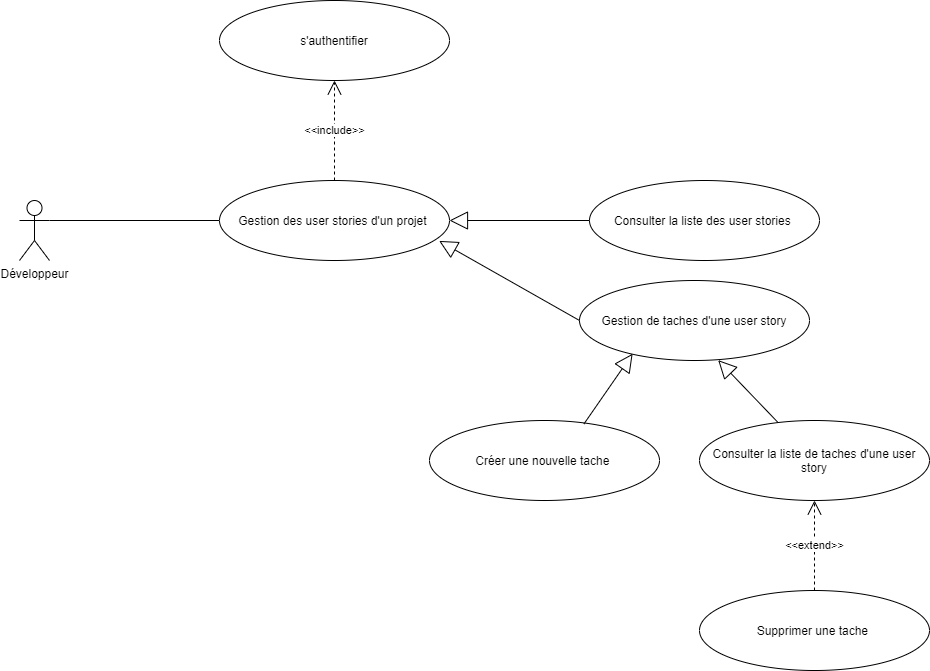
La figure suivante représente le diagramme de cas d’utilisation **Gestion de sprints d’un projet.**

Figure 24: Diagramme de cas d’utilisation Gestion de sprints d’un projet de l’acteur Product Owner

## L’acteur développeur

* + 1. Cas d’utilisation « gestion des user stories d’un projet »

La figure suivante représente le diagramme de cas d’utilisation **gestion des user stories d’un projet.**

Figure 25: Diagramme de cas d’utilisation Gestion des user stories

* + 1. Cas d’utilisation « consulter la liste de ses tâches »

La figure suivante représente le diagramme de cas d’utilisation **Consulter la liste de ses taches.**

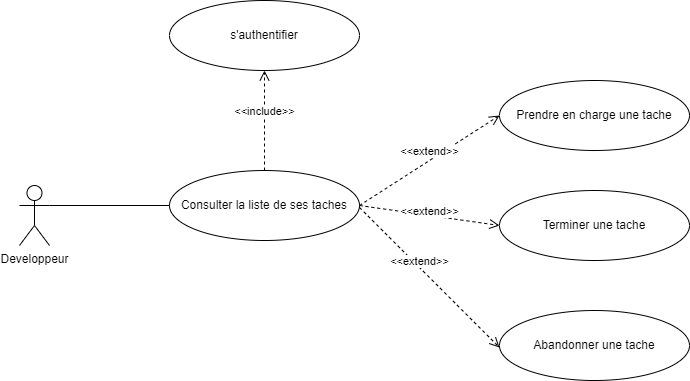


Figure 26: Diagramme de cas d’utilisation Consulter la liste de ses taches

* + 1. Cas d’utilisation « gestion de sprints d’un projet »

La figure suivante représente le diagramme de cas d’utilisation **gestion de sprints d’un projet.**

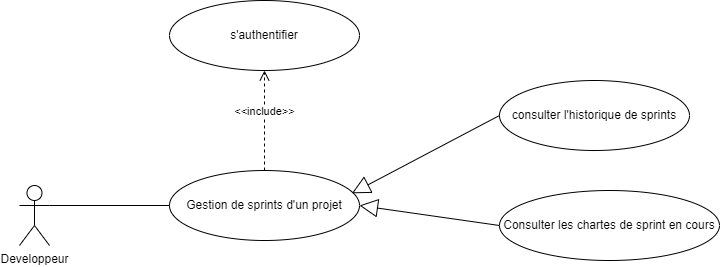


Figure 27: Diagramme de cas d’utilisation Gestion de sprints d’un projet

## L’acteur Scrum Master

Le Scrum Master est défini dans notre système comme un développeur avec des droits supplémentaires. Il peut gérer tous les cas d’utilisation qu’un développeur peut gérer avec d’autre cas d’utilisation supplémentaire.

* + 1. Cas d’utilisation « gestion de l’équipe de développement d’un projet »

La figure suivante représente le diagramme de cas d’utilisation **Gestion de l’équipe de développement d’un projet.**

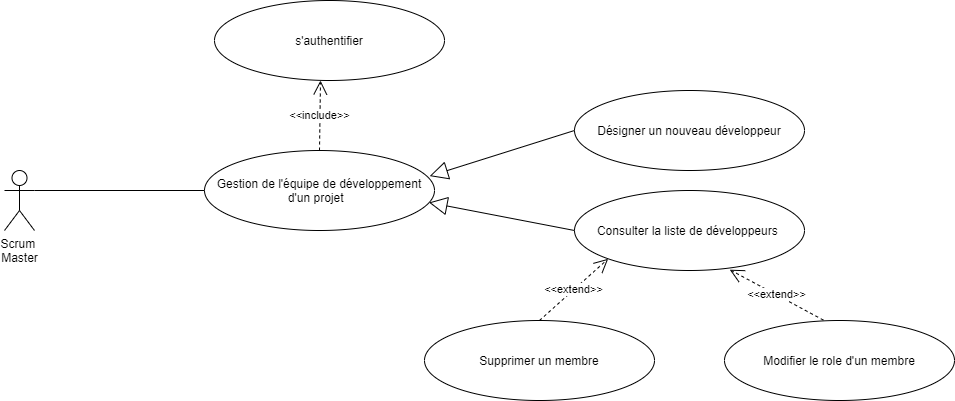


Figure 28: Diagramme de cas d’utilisation Gestion de l’équipe de développement d’un projet

* + 1. Cas d’utilisation « gestion de sprints d’un projet »

La figure suivante représente le diagramme de cas d’utilisation **Gestion de sprints d’un projet.**

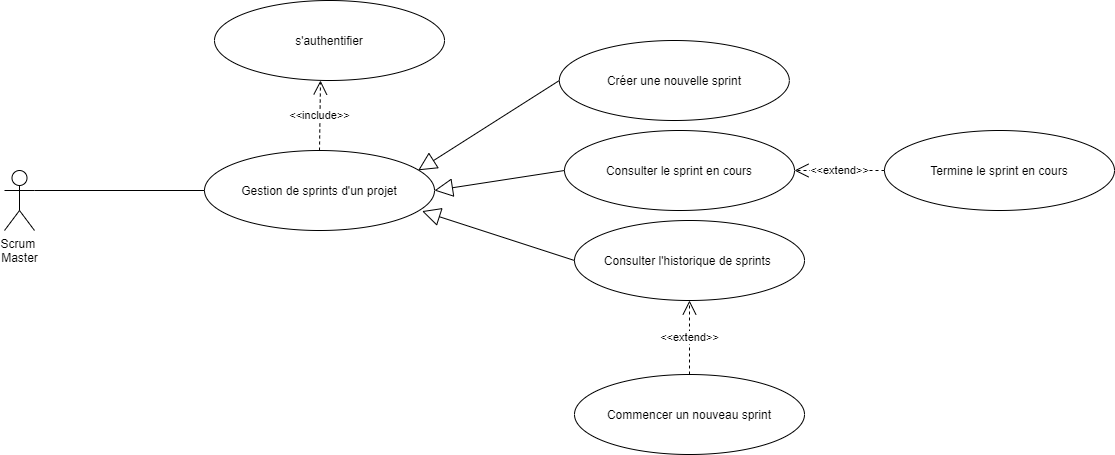


Figure 29: Diagramme de cas d’utilisation Gestion de sprints d’un projet

1. Conception détaillée

A ce stade, nous allons entamer la partie conception au cours de laquelle nous faisons la présentation de diagramme de classes et des diagrammes de séquences.



## Diagramme de classes

* + 1. Présentation de diagramme de classes

Le diagramme de classes est un schéma utilisé dans le domaine de génie logiciel pour visualiser les classes, les objets, les interfaces et les relations entre eux. [1]



La figure suivante représente le diagramme de classes de notre système.

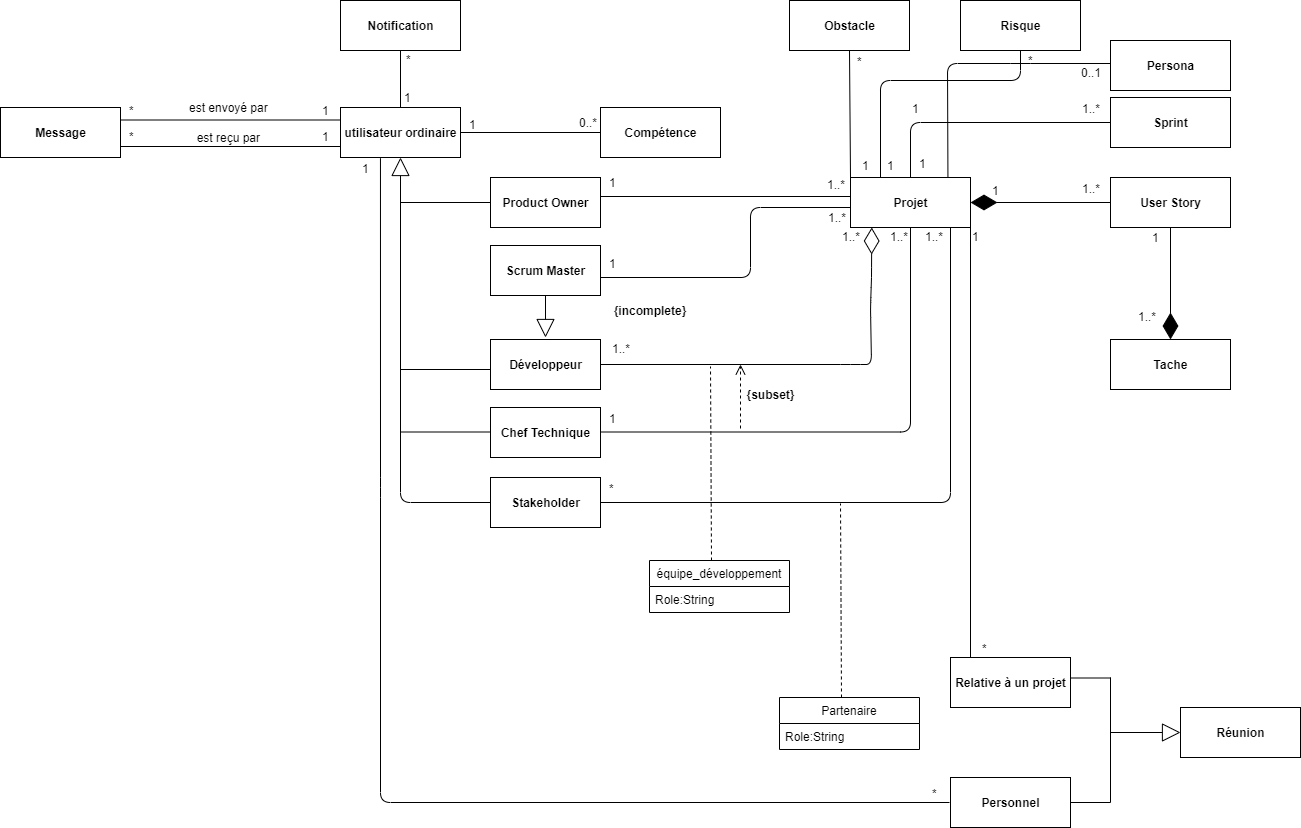


Figure 30: Diagramme de classes implémenté

3. * 1. Descriptions des cardinalités

Dans cette partie, nous allons décrire les différentes cardinalités présentes dans le diagramme de classes.

**R1 : un utilisateur peut envoyer plusieurs messages et un message est envoyé par un et un seul utilisateur.**

**R2 : un utilisateur peut recevoir plusieurs messages et un message ne peut être reçu que par un et un seul utilisateur.**

**R3 : un utilisateur peut recevoir plusieurs notifications et une notification est reçue par un et un seul utilisateur**

**R4 : un utilisateur peut avoir plusieurs descriptions de compétence alors qu’une description est relative à un et un seul utilisateur**

**R5 : un projet doit avoir obligatoirement un et un seul Product Owner et un Product Owner peut avoir plusieurs projets**

**R6 : un projet doit avoir un et un Seul Scrum master et un utilisateur peut avoir plusieurs projets en tant que Scrum master**

**R7 : un projet peut avoir un ou plusieurs développeurs et un utilisateur peut avoir plusieurs projets en tant que développeur avec un rôle bien défini.**

**R8 : l'un des développeurs est désigné comme un chef technique et un utilisateur peuvent être un chef technique pour plusieurs projets.**

**R9 : un projet peut avoir des stakeholders, mais pas obligatoirement et un utilisateur peut être un stakeholder pour plusieurs projets avec un rôle bien défini.**

**R10 : un projet peut avoir des obstacles et chaque obstacle est relatif à un et un seul projet.**

**R11 : un projet peut avoir des risques et chaque risque est relatif à un et un seul projet.**

**R12 : un projet est composé par une ou plusieurs user stories et une user Story compose un et un seul projet.**

**R13 :** une user story est composée par une ou plusieurs tâches et une tâche compose une et une seule user Story.

**R14 :** un projet doit avoir au minimum un sprint et chaque sprint est relatif à un et un seul projet.

**R15 :** une réunion peut être une réunion personnelle à un utilisateur ou une réunion relative à un projet.

**R16 :** une réunion personnelle est relative à un utilisateur exactement alors qu’un utilisateur peut avoir plusieurs réunions de ce type

**R17 :** une réunion de projet est relative à un projet exactement alors qu’un projet peut avoir plusieurs réunions de ce type.

**R18 :** un projet peut avoir une persona et une persona doit obligatoirement être relative à un et un seul projet.



## Diagrammes de séquences

Durant cette sous-section nous essayons d’étendre la description de nos cas d’utilisation par quelques diagrammes de séquences afin de souligner l’ordre

chronologique des actions dans chaque cas d’utilisation.

6. 1. 1. Digramme de séquence supprimer la persona d’un projet

La figure ci-dessous représente le diagramme de séquence **supprimer la persona d’un projet.**

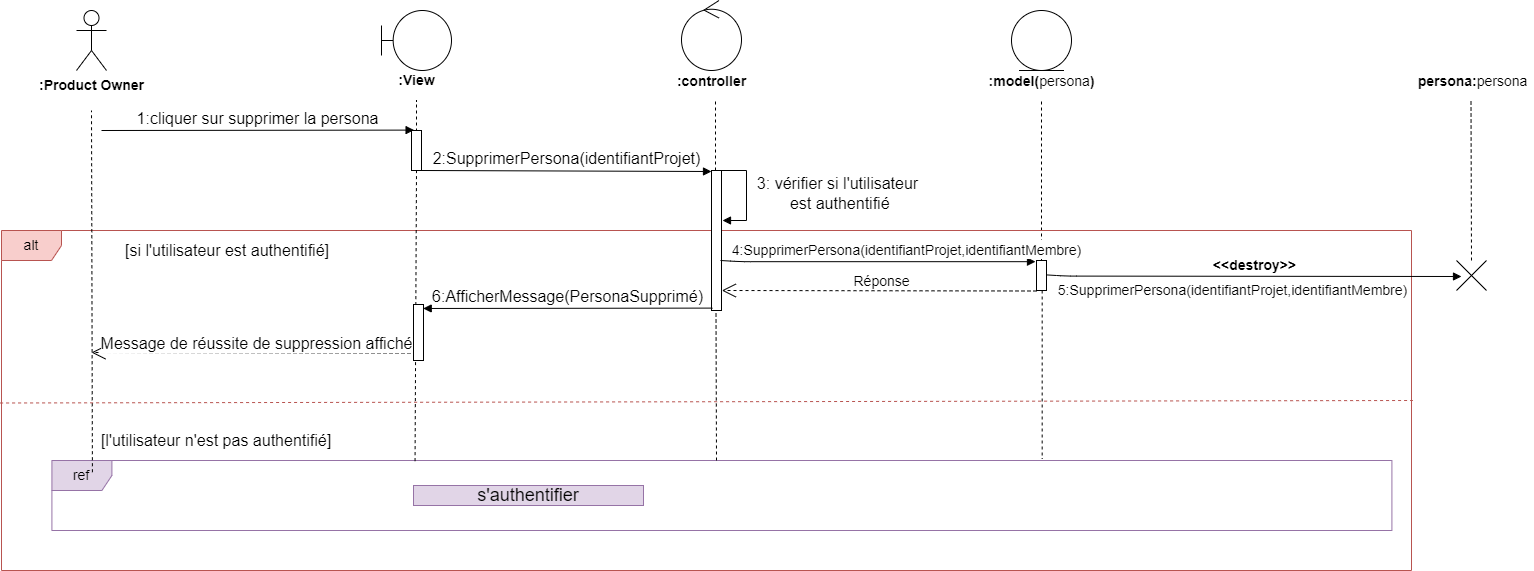


Figure 31: Diagramme de séquence supprimer la persona d’un projet

* + 1. Digramme de séquence créer la persona d’un projet

La figure ci-dessous représente le diagramme de séquence **créer la persona d’un projet.**

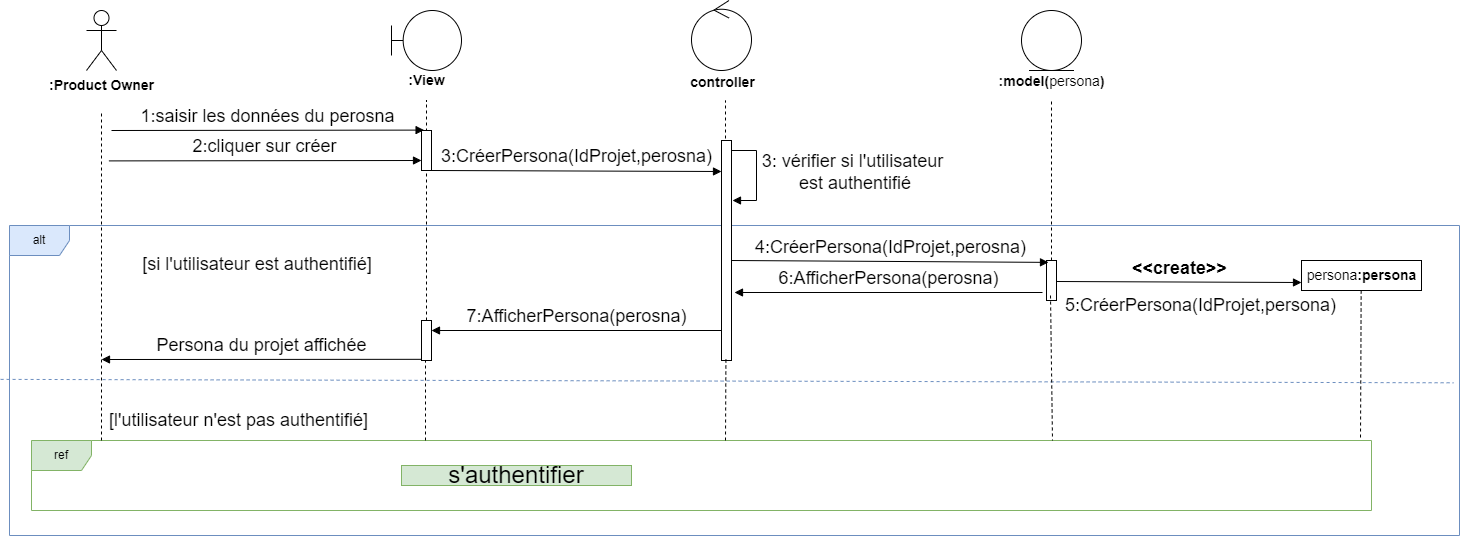
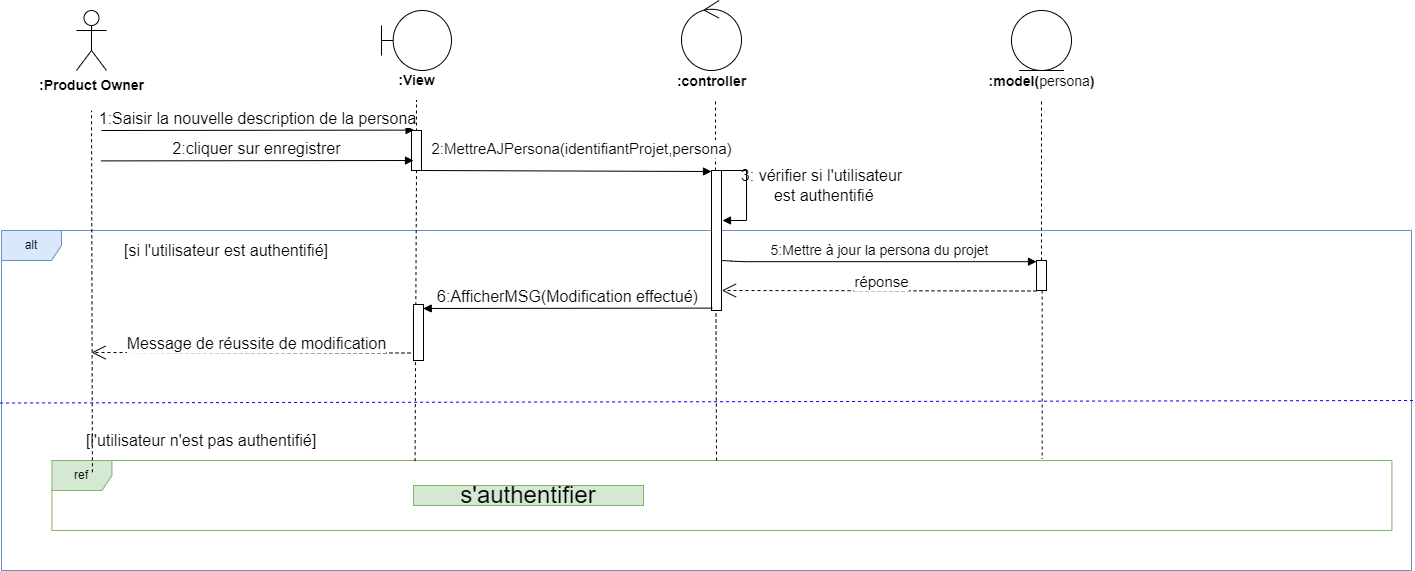


Figure 32: Diagramme de séquence créer la persona d’un projet

* + 1. Digramme de séquence modifier la persona d’un projet

La figure ci-dessous représente le diagramme de séquence **modifier la persona d’un projet.**

Figure 33: Diagramme de séquence modifier la persona d’un projet

* + 1. Digramme de séquence supprimer un membre de l’équipe de développement

La figure ci-dessous représente le diagramme de séquence **supprimer un membre de l’équipe de développement d’un projet.**

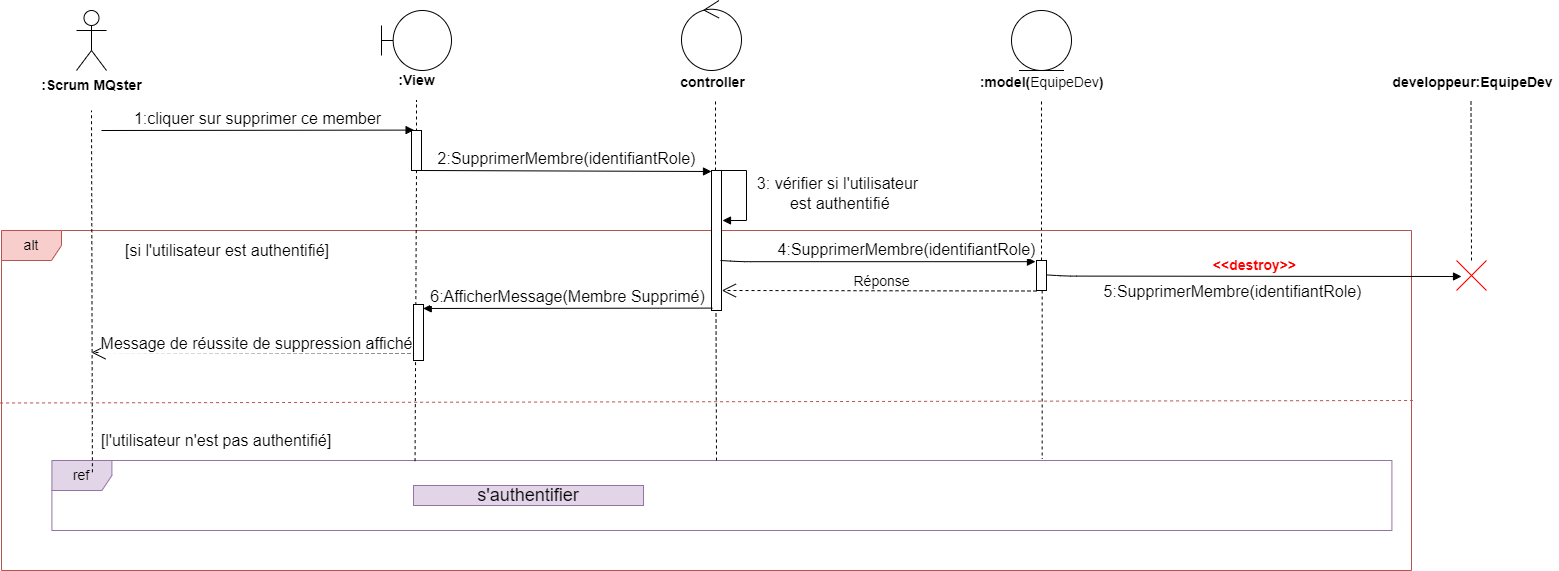


Figure 34: Diagramme de séquence supprimer un membre de l’équipe de développement d’un projet

* + 1. Digramme de séquence gestion de la persona d’un projet

La figure **35** représente le diagramme de séquence **gestion de la persona d’un projet.**

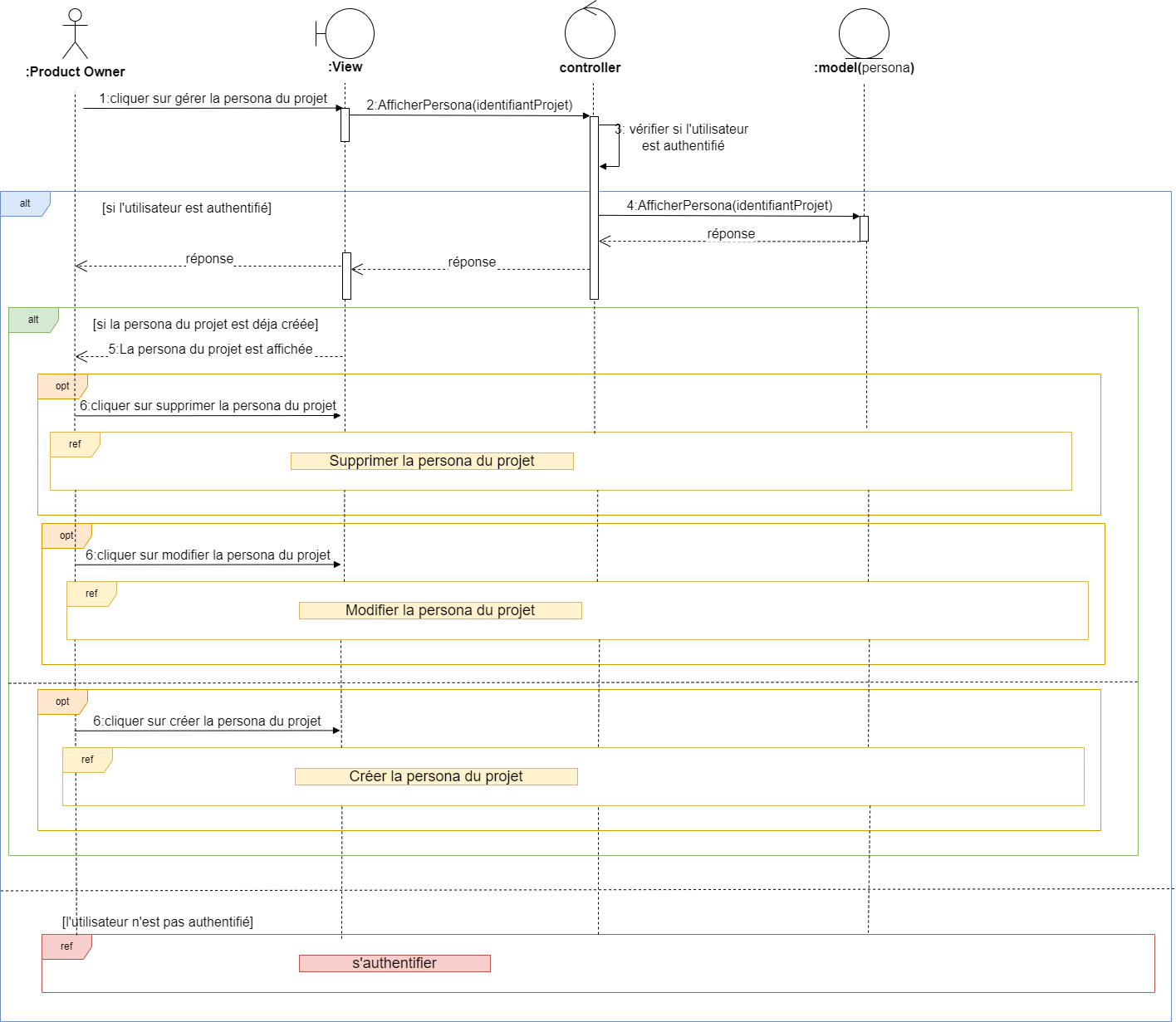


Figure 35: Diagramme de séquence gestion de la persona d’un projet

Conclusion

Dans le présent chapitre, nous avons essayé de mettre l’accent tout d’abord sur les différents acteurs qui interviennent au sein de notre système. Ensuite nous avons identifié les besoins fonctionnels et les besoins non fonctionnels du système et enfin nous avons modélisé avec UML les différents cas d’utilisation présents dans le diagramme de cas d’utilisation globale ainsi que le diagramme de classes et les diagrammes de séquence .Dans le chapitre qui suit, nous nous intéressons par la phase de l’implémentation réelle de notre projet.

Introduction

Dans les chapitres précédents, nous avons essayé d’établir un enchainement logique qui nous guide pour développer notre application. Dans le présent chapitre, nous entamons la phase de développement. Pour cela nous présentons tout d’abord l’environnement de travail, la configuration logicielle utilisée, les outils utilisés et l’exposition de choix technologique. Ensuite, nous proposons quelques imprimés-écrans de produit final.

1. Environnement de travail

## Environnement matériel

Pour la réalisation de notre projet dans des conditions favorables, il faut disposer d’une configuration matérielle minimale. Pour assurer que l’application soit fonctionnelle confortablement, il est préférable que notre solution soit déployée sur l’architecture serveur présentée par le tableau ci-dessous.

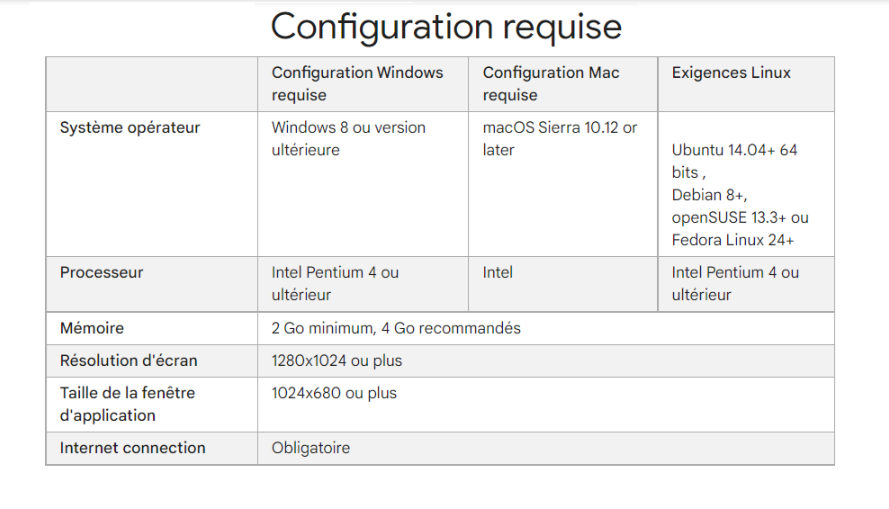
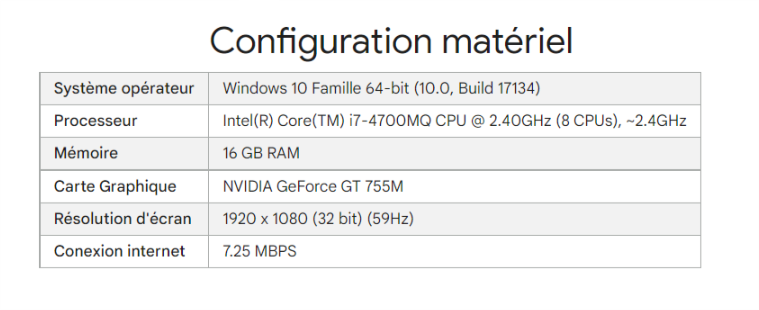


Tableau 2: La configuration minimale requise [7]

Durant le déroulement de projet, nous faisons usage d’une machine ayant la configuration matérielle exposée dans le tableau ci-dessous.

Tableau 3 : La configuration matériel utilisé pour le développement

## Environnement logiciel

Cette section est consacrée à l’exposition des différents outils logiciels et technologies utilisées pour le développement de notre application.

* + 1. Les outils logiciels

Les outils logiciels utilisés pour le développement de notre application sont :

**Draw.io** est un logiciel de dessin graphique multiplateforme utilisée pour créer des diagrammes tels que des organigrammes, des structures filaires…

**Visual Studio Code** est un éditeur de code simplifié qui prend en charge les taches de développement telles que le débogage, l'exécution de tâches et le contrôle de version.

**Robot 3T** est une ressource populaire pour les déploiements gérés de Mongo DB. Il fournit une interface utilisateur graphique (GUI) pour interagir avec des blocs de données.

**Adobe Illustrator** est l'application de conception standard du secteur qui vous permet de concrétiser votre inspiration créative à l'aide de formes, de couleurs, d'effets et de typographie.

**Adobe XD** est un outil pour l'élaboration d'expérience usager vectorielle à les applications web et mobiles évolué et publié dans Adobe inc.



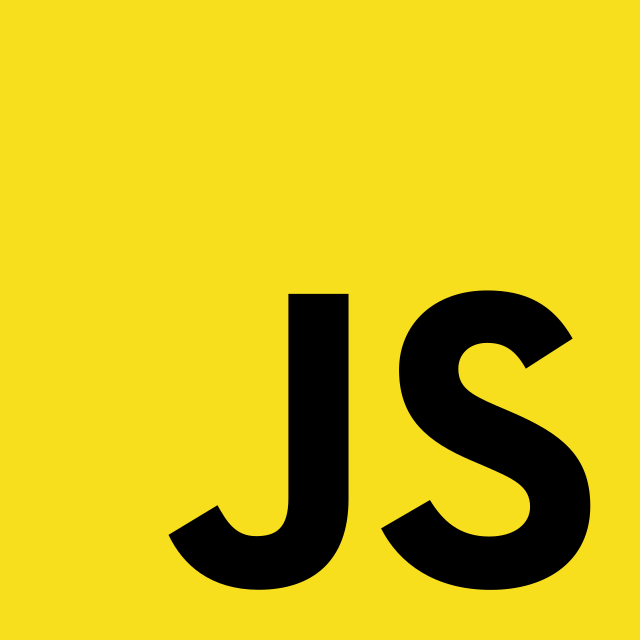
## Choix technologique

Les technologies que nous avons permis d’implémenter l’application sont les suivants :

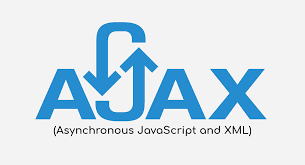
**HTML** qui est un langage de balisage standard pour la création des pages web. [8]



**CSS** est un langage qui permet de définir comment les éléments HTML doivent être imprimés sur l’écran, sur des supports papier ou d’autres supports. [9]

**JavaScript** est un langage de programmation et de Scripting orienté objet caractérisé par sa capacité de rendre le navigateur capable d’exécuter les requêtes. [10]

**JSON** fait référence à un langage léger qui garantit l’échange de données textuelles entre les côtés client et serveur. [11]

**AJAX** est une technologie très populaire dans le domaine de développement des applications web ou il permet d’apporter de l’interactivité au sein des pages web. [12]

**NodeJS** est une plateforme logicielle JavaScript orientée pour les applications utilisant le réseau pour la communication. [13]

**Bootstrap** est un Framework open source qui fournit un outil de design permettant l’optimisation de code source, d’effort, mais aussi de fournir un design responsif. [14]

**Mongo DB** est un système de gestion de bases de données NOSQL orienté document

Logo Heroku.svg**Heroku** est un fournisseur cloud qui offre une plateforme en tant que service qui supporte plusieurs langages de programmation.

**Mongo** **DB** **Atlas** est une plate-forme de données d'application multicloud qui permet d’héberger les bases de données des applications modernes.

1. Travail réalisé

## Étape de construction de l’application

Au cours de cette section, nous voulons mettre l’accent sur l’ensemble d’étapes nécessaires pour la réalisation de ce projet. La figure **36** visualise le cycle de vie

de notre projet.

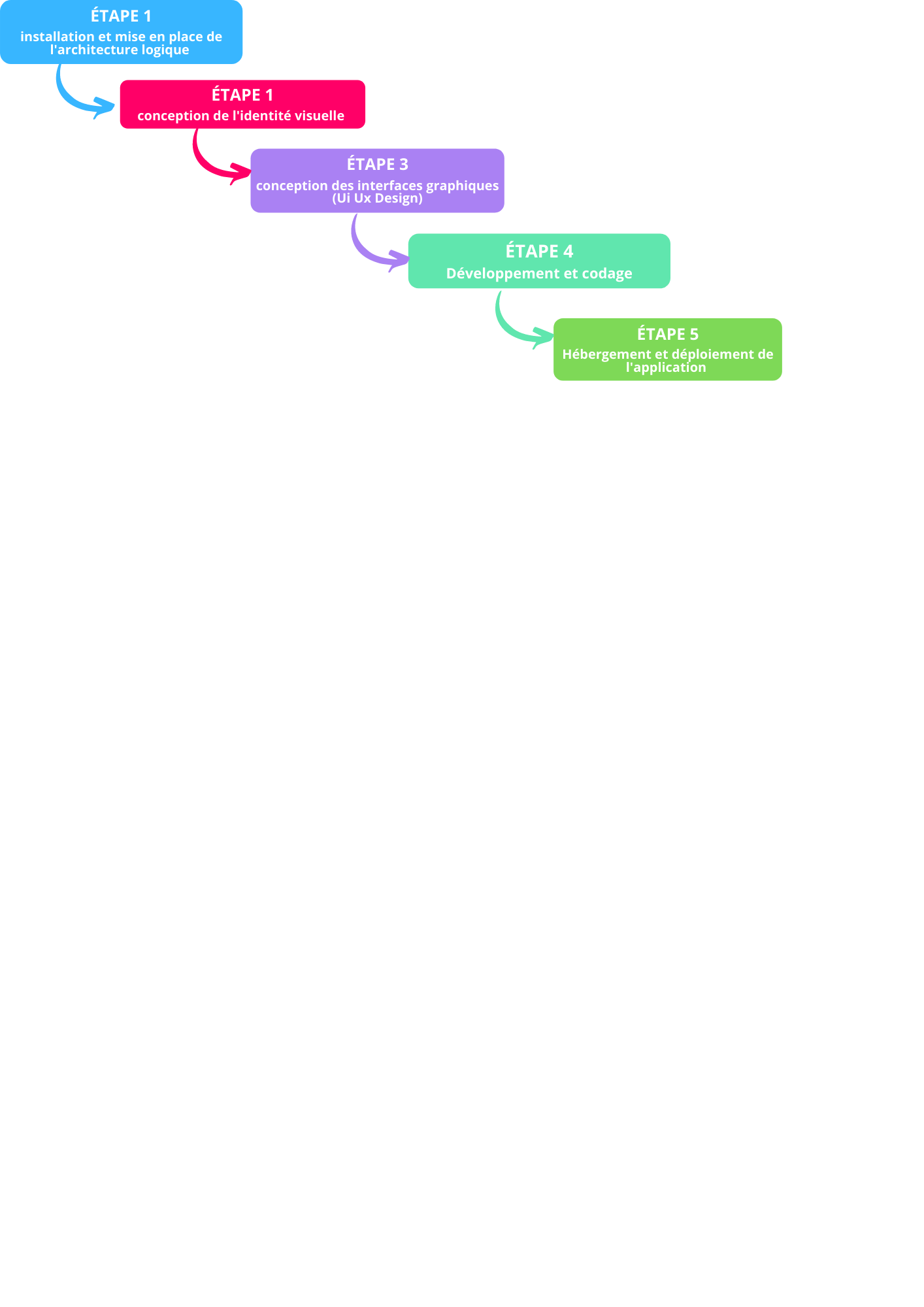


Figure 36: le schéma des étapes de réalisation de projet

* + 1. Installation et mise en place de l’architecture logique

Cette étape est consacrée à la justification de l’architecture logique déployée pour la réalisation de ce projet. Au cours de ce projet, nous faisons usage de

l’architecture logique MVC.

3. 1. 2. 1. L’architecture MVC

Le MVC est un modèle architectural qui fait la séparation d'une application en trois composants logiques principaux qui sont le modèle, la vue et le contrôleur.

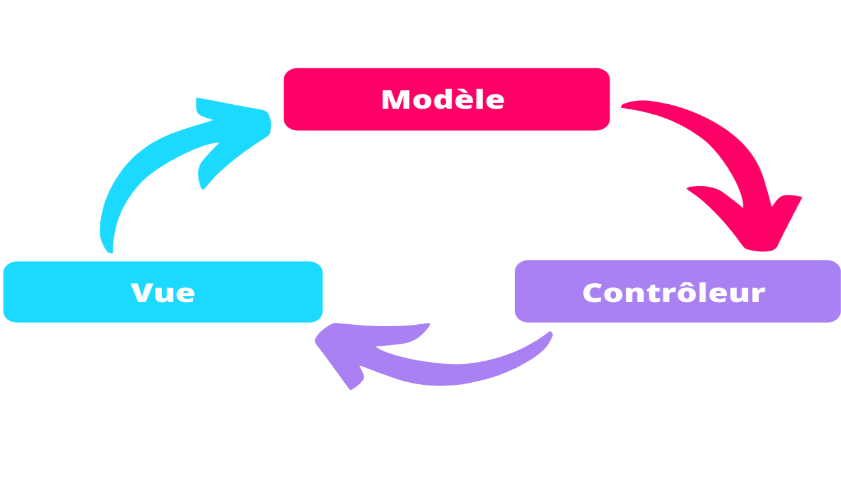


Figure 37: l’architecture de modèle MVC

* La couche vue

La vue est l’interface utilisateur qui permet de visualiser les données à l’aide d’un modèle qui lui permet également de modifier ces données.

* La couche modèle

Un modèle contient les données utilisées par un programme. Il peut s'agir d'une base de données, d'un fichier ou d'un simple objet.

* La couche contrôleur

Un contrôleur agit comme une interface entre le modèle et la vue pour traiter la logique métier et les requêtes externes, manipuler les données ....

* + 1. Conception de l’identité visuelle

L'identité visuelle d'une marque représente sa propre carte d'identité, qui permet de la reconnaître sans même chanter son nom. L'identité visuelle passera par le nom, le stylisme du logo, le logo lui-même, l’image de marque, les couleurs, la typographie, les signes…

* + 1. Conception des interfaces graphiques

Cette phase de conception des interfaces graphiques est consacrée à faire le design graphique des interfaces utilisateur de l’application ainsi que le design de l’expérience utilisateur.

* + 1. Développement et codage

La phase de développement et de codage est la phase durant laquelle nous faisons la construction effective de notre application ainsi que l'implémentation réelle des fonctionnalités du système. Durant cette phase, les conceptions réalisées précédemment seront transformées en éléments interactifs.

Pour que les codes rédigés soient lisibles, extensibles et faciles à maintenir, nous faisons usage des principes SOLID.

* + 1. Le déploiement de l’application

La figure ci-dessous représente le diagramme de déploiement de notre application.

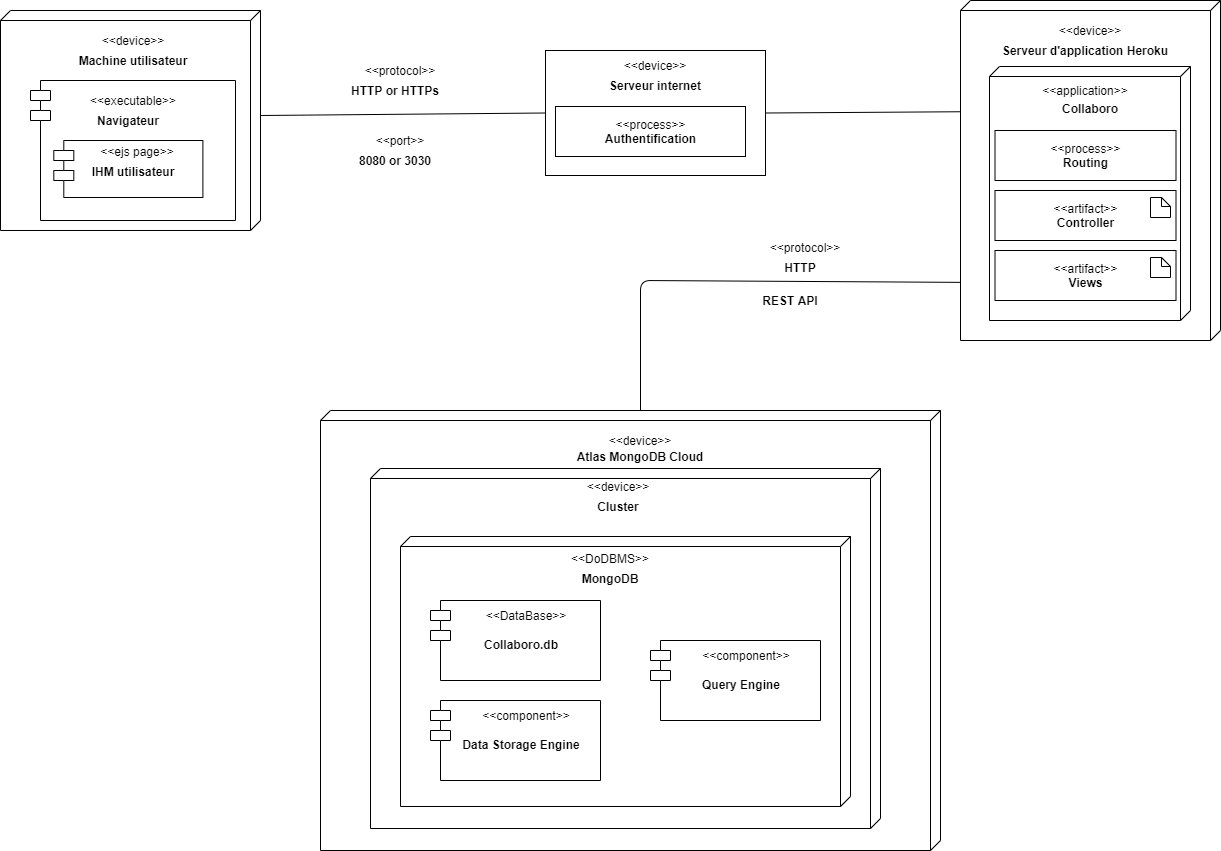


Figure 38: Diagramme de déploiement

Comme il est présenté dans le diagramme de déploiement de notre application est déployée dans une architecture classique composée de trois nœuds : un nœud qui représente la machine utilisateur, un nœud représentant le serveur d’application et un nœud représentant le serveur de base de données.

Outre que la base de données, le cluster de notre projet contient aussi :

* Moteur de stockage de données

Le moteur de stockage est le composant de la base de données qui est responsable de la gestion du stockage des données, à la fois en mémoire et sur disque.

* Moteur de requêtes

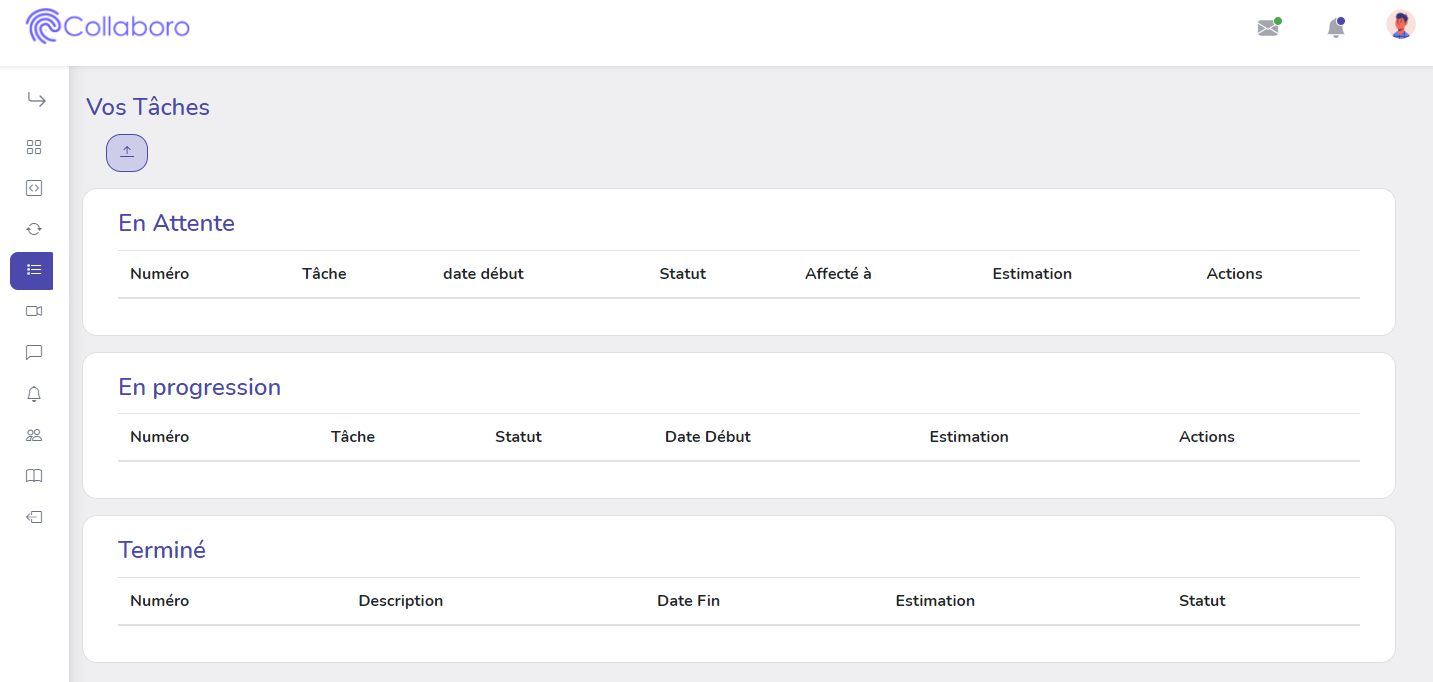
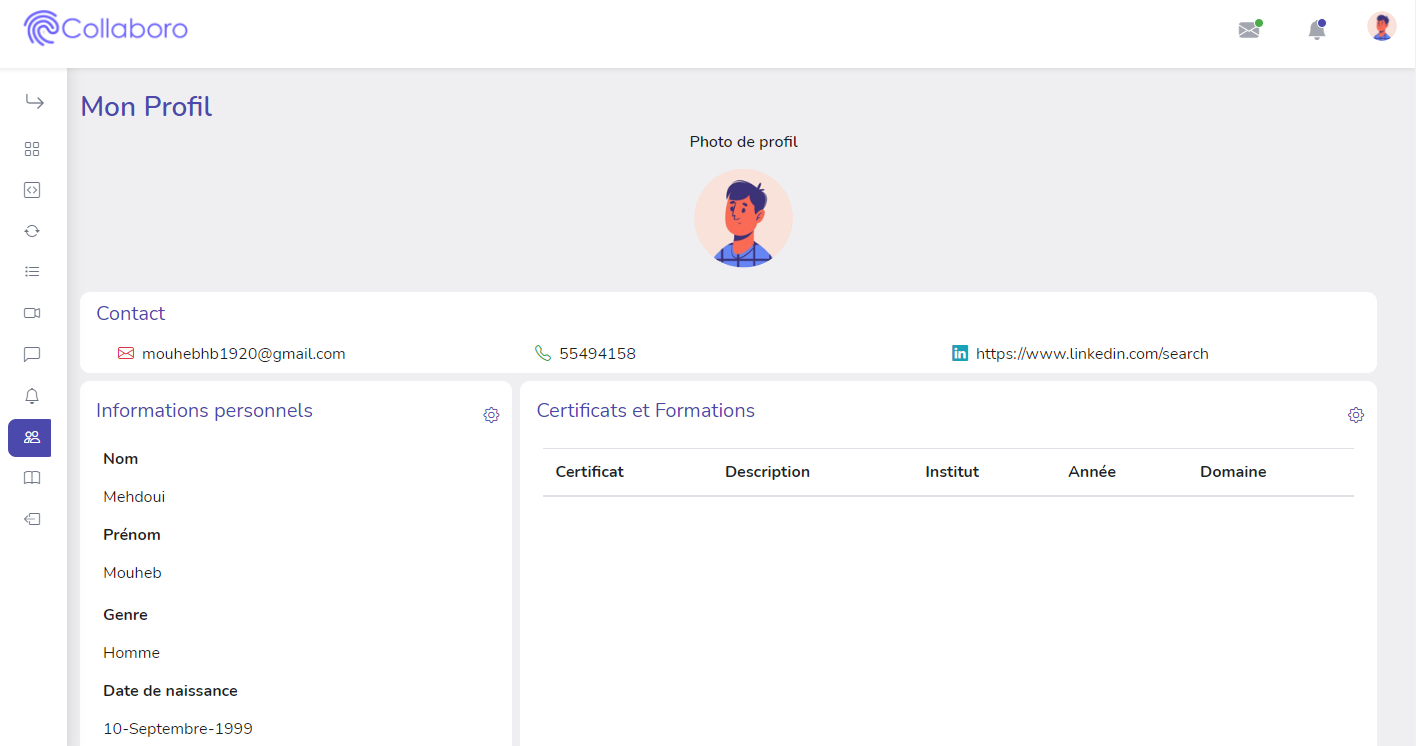
Le moteur de requêtes est le composant logiciel sous-jacent qu'un système de gestion de bases de données (SGBD) utilise pour créer, lire, mettre à jour et supprimer (CRUD) des données dans une base de données.

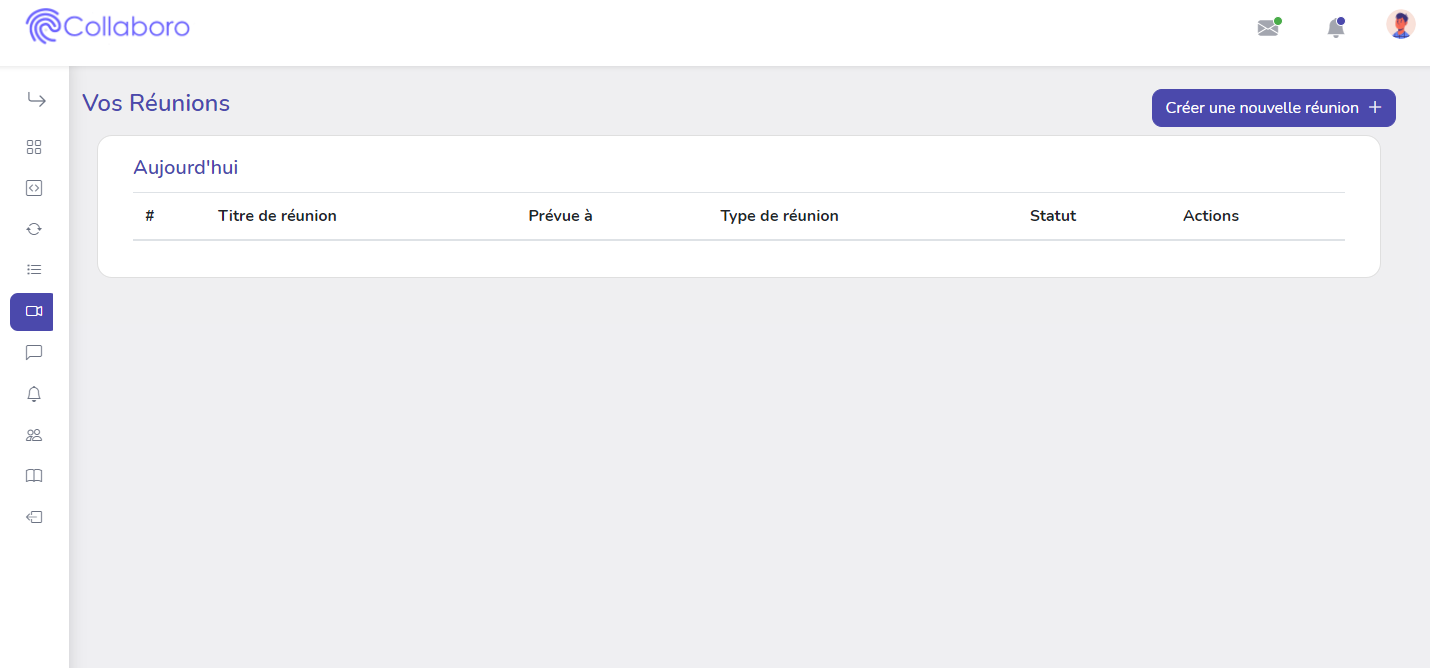
1. Solution réalisée

Durant cette sous-section, nous nous intéressons par présenter quelques captures

d’écran de la solution finale de notre projet.

Les figures présentées ci-dessous représentent quelques interfaces de notre application.

Figure 39: interface de gestion de tachesFigure 40: interface de gestion de profil

Figure 41: interface de gestion de réunions

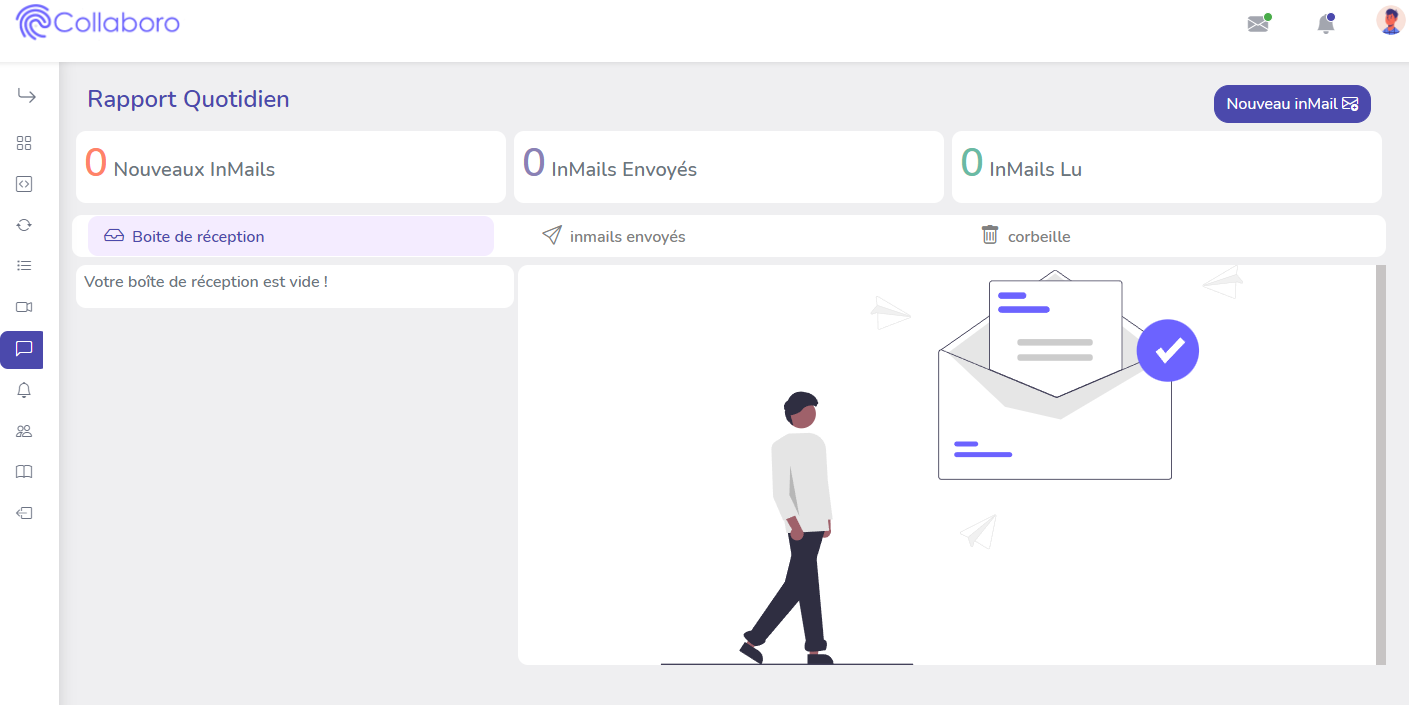
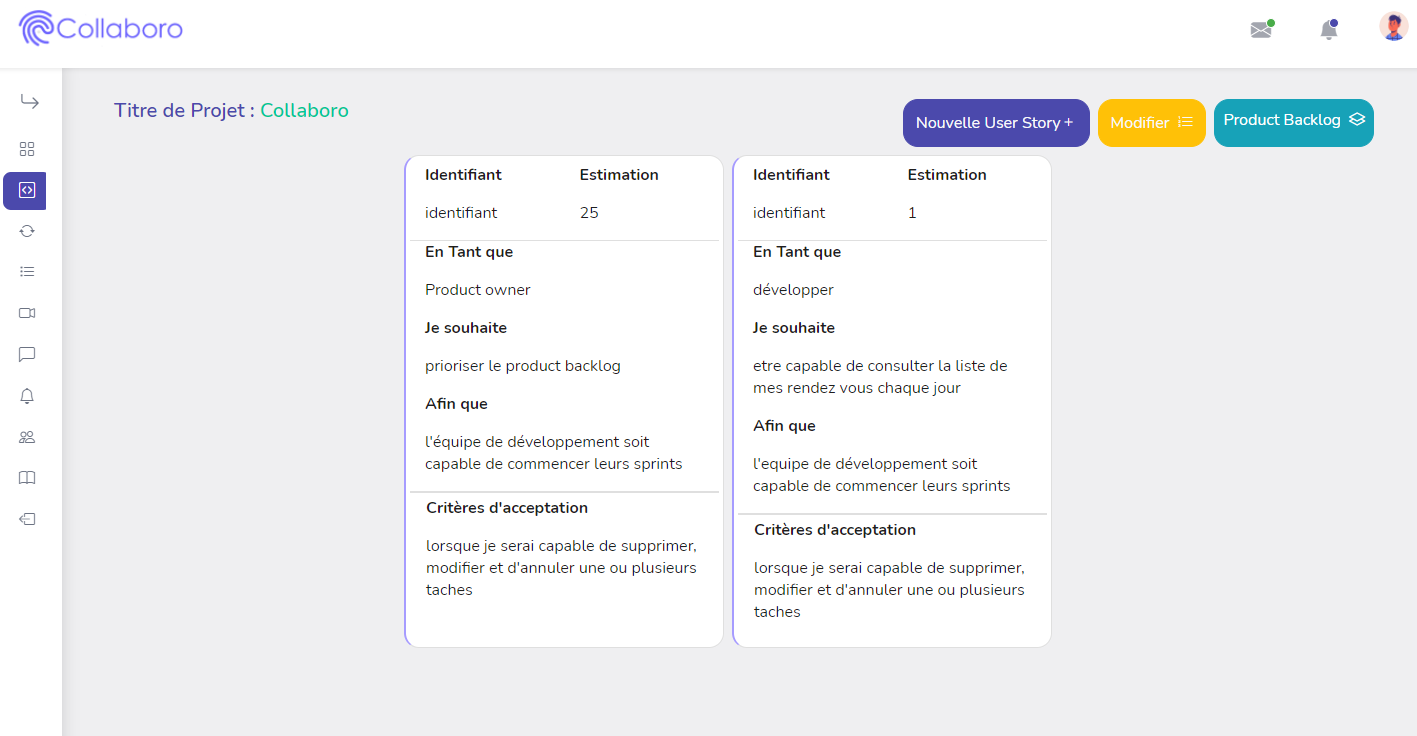
Figure 42: interface de messagerie In Mail Figure 43: interface de backlog d’un projet

Figure 44: interface de la page d’accueil

## Conclusion

Durant ce dernier chapitre, nous avons décrit les différentes phases de réalisation de notre projet, les environnements matériel et logiciel qui nous ont permis de développer notre application ainsi que l’élaboration de diagramme de Gantt qui nous aide à visualiser la chronologie des différentes étapes de la réalisation de ce projet.

Conclusion générale

Tout au long de ce projet, nous avons concouru à la mise en place d'une solution de gestion et de suivis de projets agiles Scrum.

Pour la réalisation de ce travail, nous avons essayé en premier de souligner le cadre général de notre projet où nous avons présenté notre organisme d'accueil ainsi qu'une étude approfondie de l'existant et le critiquer qui nous a permis de dégager tantôt d'informations et de notions.

Ensuite, nous avons effectué un état d'art qui nous a permis de nous intégrer au monde de gestion de projet où on a visé des notions critiques autour de la culture Agile Scrum qui ont fait l'objet de la brique de base pour procéder au chapitre suivant qui a été consacré à l'analyse et la spécification des besoins fonctionnels et non fonctionnels de notre application.

Cette étude approfondie de l'approche Scrum nous a servi dans la dernière phase de réalisation de ce projet où nous avons implémenté réellement la nouvelle solution que nous avons nommée Collaboro.

Grâce à ce projet, nous avons eu la chance d'appliquer les connaissances théoriques acquises pendant trois ans d'étude, d'avoir la chance d'utiliser des nouvelles technologies richement demandées dans le monde professionnel.

D'autre part, ce projet nous a permis d'utiliser la technologie de Cloud qui est une technologie tendance dans le domaine de l'informatique.

Sur le plan humain, ce stage m'a permis de faire des nouvelles connaissances avec des experts dans différents domaines (Agile, Scrum, Ui Ux design, développement informatique, designers…) avec qui j'ai partagé mes idées et les améliorer grâce à leurs retours pour le but de bien mener ma mission.

Ce projet peut être amélioré à plusieurs égards surtout avec l’intégration de modules d’intelligence artificielle capable de donner aux utilisateurs des outils puissants de prédiction et de prise de décision pour améliorer l’efficacité de Framework Scrum utilisée. Au titre d’exemple, un module d’estimation de cout d’un projet, d’estimation de la réussite ou l’échec d’un projet.

Webographie

[1] Atlassian. (s. d.). *À quoi sert Jira Software ?* Consulté le 16 février 2022, à l’adresse https://www.atlassian.com/fr/software/jira/guides/use-cases/what-is-jira-used-for

[2] Wikipedia contributors. (s. d.). *Trello*. Wikipédia. Consulté le 16 février 2022, à l’adresse https://fr.wikipedia.org/wiki/Trello

[3] Wikipedia contributors. (2022, 31 janvier). *Notion (logiciel)*. Wikipédia. Consulté le 16 février 2022, à l’adresse https://fr.wikipedia.org/wiki/Notion\_(logiciel)

[4] *Les méthodes agiles de développement*. (s. d.). Hello Pomelo. Consulté le 21 février 2022, à l’adresse https://www.hello-pomelo.com/articles/les-methodes-agiles-de-developpement#

[5] *L’Histoire de Scrum — Wiki Agile du @GroupeCESI*. (s. d.). Wiki. Consulté le 3 avril 2022, à l’adresse https://wikiagile.cesi.fr/index.php?title=L%27Histoire\_de\_Scrum

[6] Tuleap, Open Source Agile Project Management and Software Development tools. (2021, 12 novembre). *Comprendre la méthode Agile Scrum en 10 min •*. Tuleap. https://www.tuleap.org/fr/agile/comprendre-methode-agile-scrum-10-minutes

[7] *System requirements - Google Web Designer Help*. (s. d.). Google. Consulté le 24 avril 2022, à l’adresse https://support.google.com/webdesigner/answer/3232604?hl=en

[8] HTML *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

[9] CSS *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

[10] JavaScript *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

[11] JSON *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

[12] AJAX *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

[13] NodeJS w3schools.com *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

[14] Bootstrap *W3Schools Free Online Web Tutorials*. (s. d.). W3Schools. Consulté le 24 avril 2022, à l’adresse https://www.w3schools.com:443/

Bibliographie

[1] K.S. (2004). *UML Fast Track* (Vol. 173). Apress.

[2] S. (2022). *Scrum Body of Knowledge* [E-book]. Scrum.org.

Résumé

Ce projet est développé dans le but d'obtenir le diplôme de licence en génie logiciels et des systèmes d'information. Ce projet, que nous avons nommé Collaboro, est une application web libre et open source qui permet aux équipes de gérer leurs projets Scrum, leurs tâches et leurs réunions, ainsi qu'un système de messagerie intégré. Mots clé [Agile, Scrum, Projet, Equipe, développement, Kanban, User Story, AI]

Abstract

This project is being developed with the aim of obtaining a bachelor’s degree in software and information systems engineering. This project, which we have called Collaboro, is a free and open source web application that allows teams to manage their Scrum projects, tasks and meetings, as well as an integrated messaging system.

Keywords [Agile, Scrum, Project, Team, Development, Kanban, User Story, AI]

ملخص

يجري تطوير هذا المشروع بهدف الحصول على درجة البكالوريوس في هندسة البرمجيات ونظم المعلومات. هذ المشروع، الذي أطلقنا عليه اسم Collaboro، هو تطبيق ويب مجاني ومفتوح المصدر يسمح للفرق بإدارة مشاريع Scrum ومهامها واجتماعاتها، بالإضافة إلى نظام مراسلة متكامل.

الكلمات الرئيسية Agile، Scrum، Project، Team، Development، Kanban، [User Story]، AI