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 faits 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 exceptions, 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

Remercîments

C’est avec un grand honneur et un grand plaisir que j’écris ces quelques ligne 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 encadrent 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 sacrifié.

Mouheb Mehdoui

Table de matières

Table des figures

Listes des tableaux

Listes des abréviations

Introduction générale

Tout au long de l’histoire humain, 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 aux seins 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 chines 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 des 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 projet.

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 suivis pour la réalisation de ce projet et qui compte cinq chapitres qui font le rôle d’une simple introduction au contexte générale du projet et qui s’étend sur quatre phases commençant par une introduction de l’organisme d’accueil puis l’étude de l’existant, choix de la méthode de travail et enfin l’élaboration d’un planning prévisionnel qui organisera la réalisation de ce projet.

Chapitre 1 contexte générale

## 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 accueilli pendant la réalisation de notre projet. La deuxième partie sera consacré à l’étude de l’existant au cours de laquelle nous feront 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

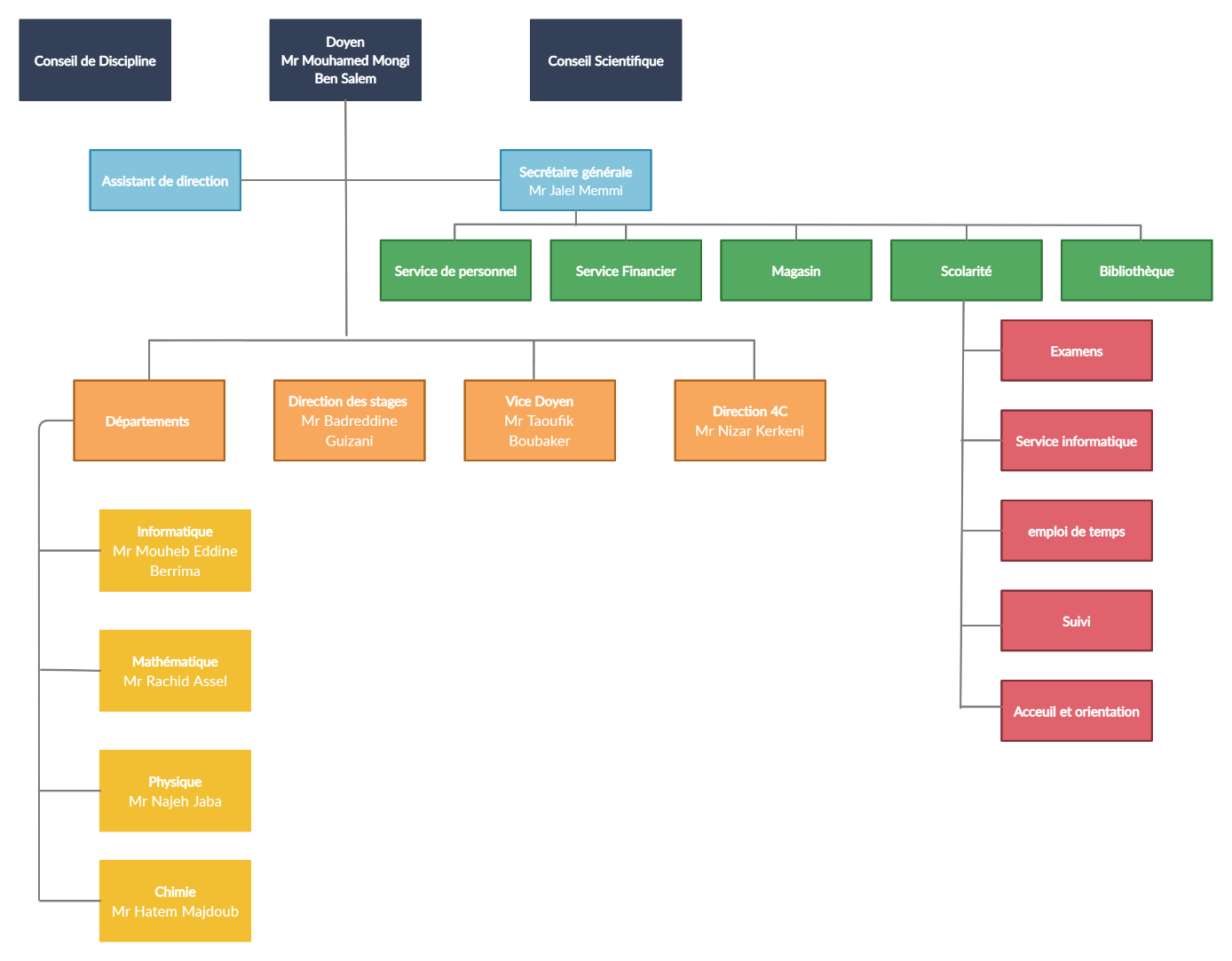
Le département des sciences de l’informatique est l’un des quatre départements qui composent la Faculté des sciences de Monastir. Ce département compte 48 enseignants-chercheurs entre professeurs, assistants, maitres assistants et enseignants contractuels qui assurent la formation d’une licence en sciences 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.



## Organigramme de l’organisme d’accueil

Un organigramme est un diagramme qui montre la structure interne d'une organisation ou d'une entreprise. Ils permettent de simplifier des structures complexes. Ce guide vous aidera à comprendre leurs rôles, leur structure et les étapes pour créer un organigramme à partir de rien ou à l'aide d'un modèle.

La figure suivante présente l’organigramme de la Faculté des sciences de Monastir qui accueilli notre organisme d’accueil.



**Figure 1 :** l’organigramme de la Faculté des sciences de Monastir

1. Étude de l’existant

## Introduction

Une bonne planification et l'étude détaillée sont insuffisant 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 projet, plusieurs entreprises gérer jusqu'à aujourd'hui leurs projets d'une façon manuelle ce qui entraîne des pertes de temps des retards et un mal-organisation des responsabilités entre les membres de l'équipe de réalisation d'un projet. Pour résoudre cela, de nombreuses solutions sont réalisées pour résoudre ces problèmes, mais à cause de la crise sanitaire depuis décembre 2019, la plupart des entreprises ont adopté la culture de travail à distance ce qui augmente le trafic sur les solutions existantes de gestion et suivi de projet. Cette migration vers les outils en ligne a conduit à changer les mesures qui font la différence entre les outils existants vers des mesures de performance, de qualité et de scalabilité face à la grande demande de ce genre de solutions.

Tous ces obstacles rendent nécessaire et urgent de l'installation d'un système de gestion et de suivi de projets agiles qui suffit les besoins mentionnés ci-dessus.

## Critique de l’existant

Lorsqu'une entreprise décide d'adapter un outil pour la gestion et le suivi de projet agiles, l'entreprise consistera 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'autre. Pour cela, dans les deux sections suivantes nous nous intéressons par présenter les outils les plus populaires de gestion de projets agiles puis nous feront une sorte de comparaison pour présenter les avantages d'un outil par rapport aux autres outils et qui nous aident aussi à introduire des fonctionnalités de correction des problèmes présentes dans les autres solutions de gestion de projets agiles.

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

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

* JIRA

Jira est une application logicielle utilisée pour le suivi des problèmes et la gestion de projet. L'outil, développé par la société de logiciels australienne Atlassian en 2002, est devenu largement utilisé par les équipes de développement agiles pour suivre les bogues, les histoires, les épopées et d'autres tâches.



**Figure 2 :** Le logo de Jira

* Trello

Trello est un outil de gestion de projet en ligne, lancé en septembre 2011, inspiré de la méthode Kanban de Toyota. Il est organisé en fonction des éléments des cartes de la liste du tableau, chaque carte représentant une tâche. Ces cartes peuvent être attribuées aux utilisateurs et peuvent être déplacées d'un tableau à l'autre, reflétant leur progression.



**Figure 3 :** Le logo de Trello

* Notion

Notion est un logiciel de gestion de projet et de prise de notes. Notion est un logiciel conçu 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.



**Figure 4 :** Le logo de Notion

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

Après la phase de rechercher des outils de gestion de projets agiles qui a conduit à relever Jira, Trello et Notion en tant qu'outils de gestion de projets les plus utilisés, 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’utiliser ce service est un facteur assez important pour décider soit de faire usage d’un tel service ou d’un tel outil soit de chercher un autre alternatif qui suffit l’équilibre entre la qualité et le coût.

* 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 demandeur 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 les fondements des méthodes agiles ?

* L’outil de planification des réunions

Un outil de planification des réunions est assez nécessaire pour une équipe pour garder des horaires fixes pour synchroniser leurs avancements dans un projet et d’échanger leurs idées afin de garder une vision commune de projet.

* Les outils de communication

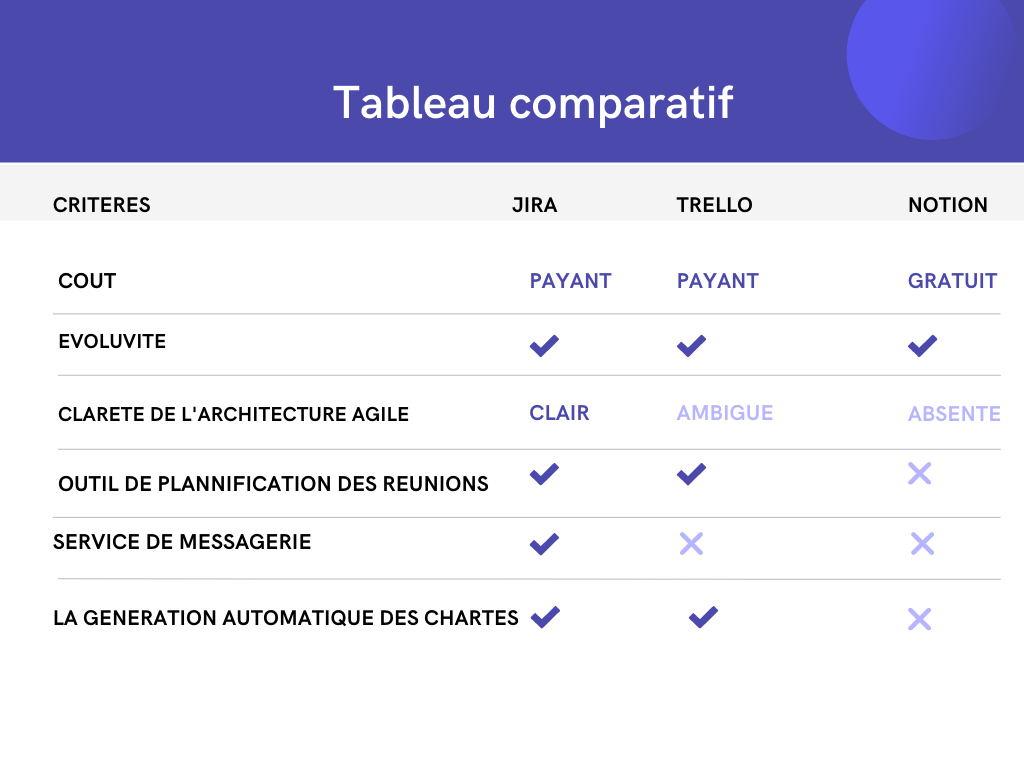
Ces outils sont des outils de base nécessaires pour garder un fil de communication entre les membres d’une équipe sans être obligé d’utiliser un service externe qui entraine la perte de temps et de budget pour l’entreprise.

* 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 tel que le coût supplémentaire, la perte de temps et un taux d'erreurs élevé puisque les données de chartes sont saisies manuellement et ne sont pas extraites directement de la source de données où ils sont stockés.

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.

Le tableau présent dans la page suivante présente une comparaison entre les différents outils de gestion de projets selon les critères mentionnées ci-dessus.



**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 tel que les clients et les fournisseurs 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 qui 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 de leurs projets 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 permettent 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 disponible à tout moment et de sauvegarder leurs informations dans un endroit sécurisé.

1. Choix de la méthodologie de travail

Notre projet portera sur l'adoption de l'approche Agile au sein d'un système de gestion et de suivi de projets. 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'adopter l'approche Agile 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.

Ce cadre de travail très populaire se distingue par un travail collectif au profit d'un but commun, des réunions quotidiennes aussi que le suivi continu de projet et qui facilite aussi la gestion des changements fréquents. Cette méthodologie assure la livraison de produits de qualité et qui satisfait les besoins et les attentes des clients.

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

* L’extraction des fonctionnalités inclus dans notre système après la réalisation d’un backlog de produit.
* La subdivision du projet en un ensemble des user stories (fonctionnalités).
* La subdivision de la réalisation de projet sur un ensemble des sprints.
* Des réunions quotidiennes avec l’encadrant pour le suivi de l’avancement.
* Une réunion chaque vendredi ou nous essayons de répondre à ces trois questions :
* Quelles sont les tâches terminées ?
* Quelles 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 mais 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.



## Diagramme de Gantt

Le diagramme de Gantt est l’un des outils les plus puissant et les plus efficace pour la représentation visuelle de l’état d’avancement des différentes activités constituant un projet. Le diagramme de Gantt est sous la forme d’un tableau dont la colonne à gauche énumère les activités à effectuer et la ligne d’en-tête représente l’unité temporelle qui s’adapte le mieux avec le projet. Chaque activité dans ce diagramme est représentée sous forme d’une barre horizontale dont sa position verticale indique la phase au quelle elle appartient et la position horizontale indique la date de début, la date de terminaison ainsi que la durée.

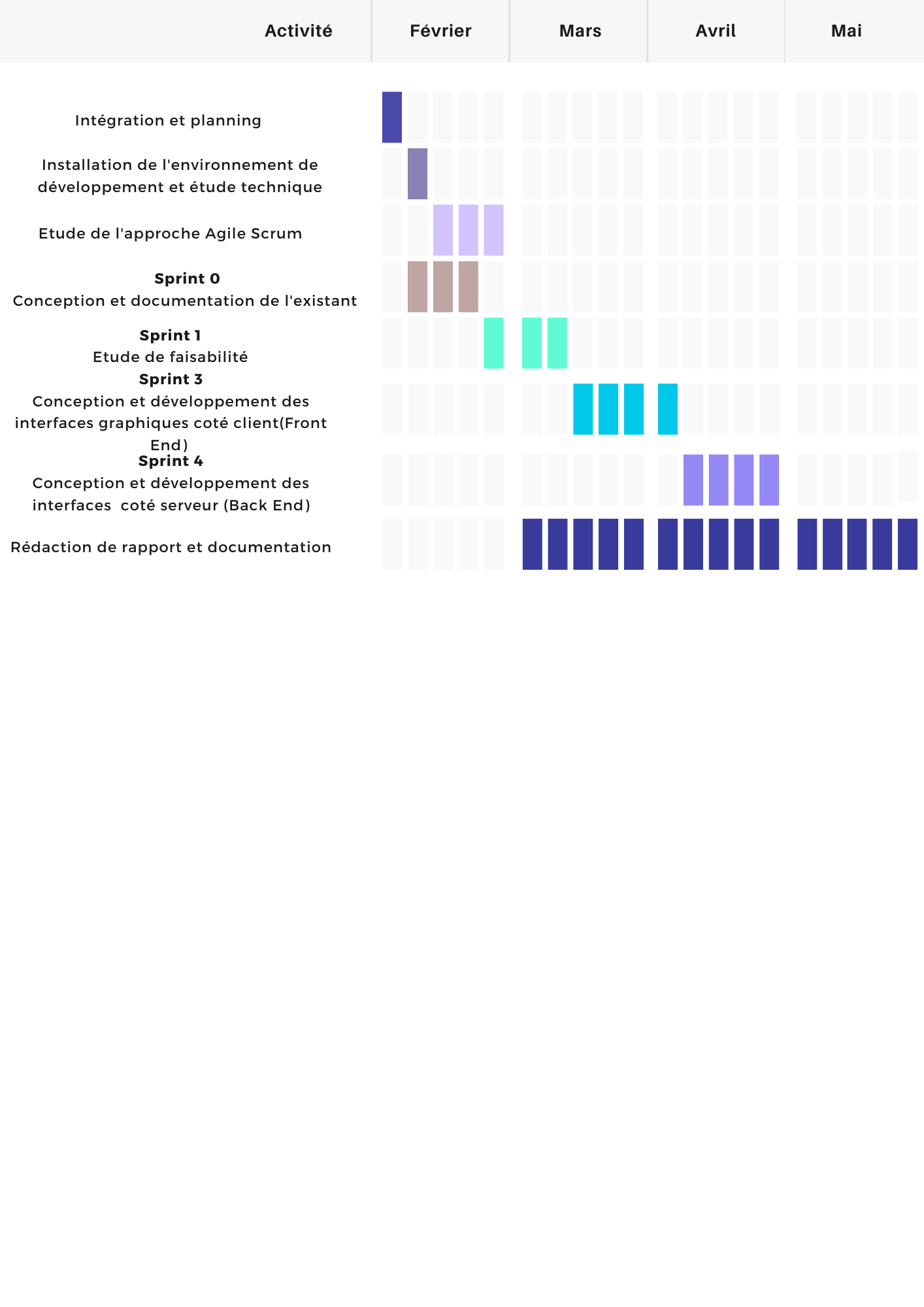
La figure présente ci-dessous représente un exemple concret de diagramme de Gantt.



**Figure 6 :** Exemple de diagramme de Gantt

## Élaboration d’un planning de réalisation

Apres l’identification des différentes phases de notre projet et l’estimation de temps nécessaires pour finaliser chacune de ces phases, le diagramme de Gantt sera comme le montre la figure dans la page suivante.



**Figure 7 :** Diagramme de Gantt du planning prévisionnel de projet

## Conclusion

Dans le présent chapitre, nous avons tout d’abord présenté l’organisme qui nous a accueilli pendant la réalisation de ce projet de fin d’études puis nous avons étudié l’existant et enfin 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.

Chapitre 2 État de l’Art

1. Autour de la culture de gestion de projet

## Projet

Un projet est une entreprise unique et transitoire, entreprise qui vise à atteindre des objectifs programmés qui peuvent être définies en termes de produits, avantages ou encore des résultats. En générale, un projet est classe comme succès s’il réussit à atteindre les objectifs spécifiés selon les critères d’acceptation dans les limites de budget, de qualité et de délai.

## Gestion et suivi de projet

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 finale conforme aux critères d’acceptation du projet. La différence fondamentale entre la gestion de projet et la gestion classique est que la gestion classique s’agit d’un processus contenu par contre la gestion de projet s’agit d’un produit final à livrer a contrainte de temps, de qualité et coût. A nos jours, un chef de projet doit présenter des différents groupes de compétences non seulement ceux qui sont techniques mais aussi des compétences en gestion des ressources humaines, de gestion de temps, de communication et une bonne connaissance des affaires.

1. Les méthodes classiques de gestion de projet

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

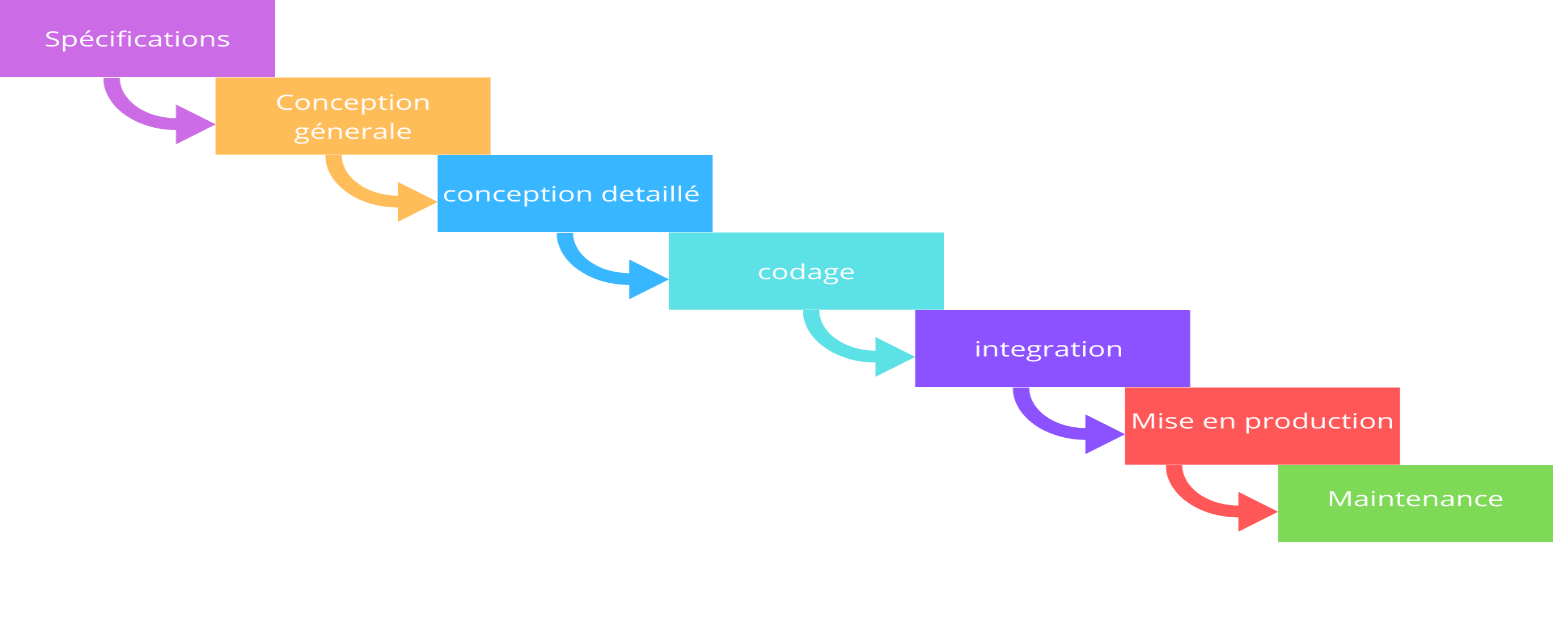
Le principe commun de toutes les méthodes de gestion de projets classique est qu’elle consiste à définir l’ensembles de fonctionnalités de projet, les spécifier de façon détaillée, les développer puis une phase pré-final de tests et enfin la mise en service de produit finale.

Lorsque 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 est inspire du vrai sens du mot cascade puisque cette méthode est basée sur une succession des phases où chacune de ses phases doit être terminé avant le passage à la phase suivant. Lors de l’utilisation d’une méthode de cycle en v, tout demande de changement dans la spécification du projet est très couteuse pour l’entreprise et même pour l’équipe responsable au développent de ce projet donc tout retour en arrière est presque impossible.

La figure ci-dessous modélise les phases de réalisation d’un projet selon la méthode de cycle en cascade.

**Figure 2 :** 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 détaillé

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 effectué.

* 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 délivrée au client.

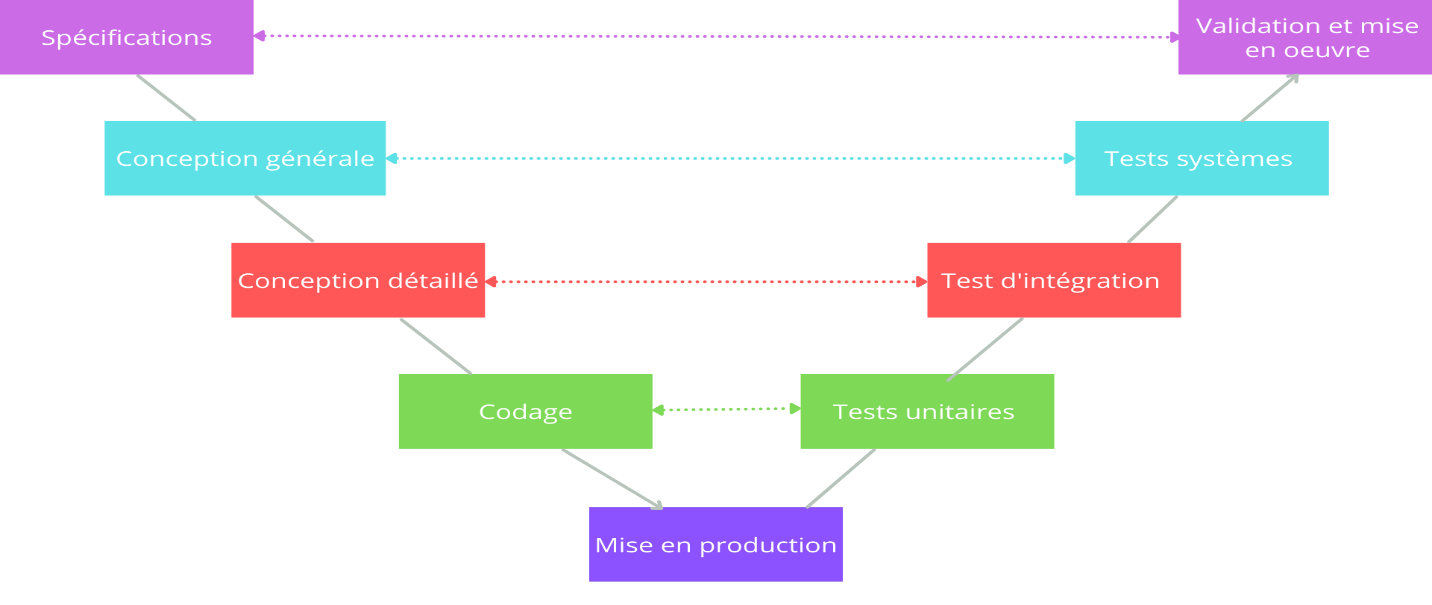
* La phase de maintenance

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

## La méthode de cycle en V

Le modèle de cycle en V est inspiré du modèle de cycle en cascade. Il reprend le même principe de succession des séquences sauf qu’on associe à chaque étape descendante (correspond à la conception de produit) une étape ascendante (correspond à la validation de produit).

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



**Figure 3 :** les phases de modèle de cycle en V

* 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 détaillé

L’équipe de projet s’attaque à la spécification formelle du projet et l’expression des besoins à travers un langage de modélisation (UML, Merise, OCL …)

* La phase de codage

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

* 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 délivrée 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 évolutifs ou correctifs au niveau du produit.

* La phase des tests unitaires

Le principe des tests unitaires consiste à isoler une partie du code et à vérifier qu’elle fonctionne parfaitement. Il s’agit de petits tests qui valident l’attitude d’un objet et la logique du code afin de dégager les vulnérabilités dans le code et de corriger les erreurs qui peuvent apparaitre.

* La phase des tests d’intégration

Après effectuer les tests unitaires nécessaires ou chaque partie du code est teste indépendamment des autres parties, l’équipe de développement construit le produit en faisant l’assemblage des parties ensemble et puis ils

(Les développeurs) effectuaient des tests pour mesure la fiabilité de produit et la compatibilité de chaque partie avec le reste des blocs.

* La phase des tests systèmes

Contrairement aux autres catégories des tests, les tests systèmes sont effectués en partenariat avec les futurs utilisateurs du produit qui doivent vérifier la conformité de produit implémente avec leurs besoins et vérifient les fonctionnalités fournies par l’équipe de développement du produit.

## Les avantages des méthodes classiques de gestion de projet

L’utilisation du mot classique pour décrire une tel méthode ne signifie pas qu’il s’agit d’une méthode faible ou inutile même il s’agit des méthodes classiques ou traditionnelles, elles présentent certains avantages entre autre :

* Une direction claire

Comme tout est planifié à l’avance, chaque membre de l’équipe de développement possède une vision claire de l’image finale du produit et il est conscient de ses responsabilités et l’exigence du projet. Cela leur permet de travailler efficacement avec un minimum de supervision.

* Un niveau de contrôle élevé

Les chefs de projet détiennent presque tout le pouvoir de décision dans une configuration traditionnelle. Cela permet d’éviter les écarts et les divergences entre le travail réalisé et le travail demandé.

* Une responsabilité centralisée

Puisque les chefs de projet détiennent tout le pouvoir de prise de décision dans la phase de développement. Ils seront naturellement responsables de la réussite ou l’échec du projet. Au lieu de contacter plusieurs personnes, le client sait toujours à qui il doit s’adresser afin d’obtenir les informations nécessaires.

* Une documentation claire et efficace

Une documentation appropriée est en fait la pierre angulaire des méthodes classiques Non seulement les documents normalisent l’ensemble du processus mais ils peuvent également servir de guide à l’avenir

## Les inconvénients des méthodes classiques de gestion de projet

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

Entre autre :

* Un cadre de travail rigide

L’approche traditionnelles suit un ensemble d’étapes séquencées qui permettent à une l’avant. Lorsque vous utilisez une des méthodes classiques, il y a peu de possibilités de changement dans le cas où les parties prenantes exigent des changements.

* Des tests trop tardifs

La phase des tests est une phase nécessaire dans tout processus de développement indépendamment de l’approche utilisé. 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 projet : les méthodes agiles

## Approches Agile plutôt que méthodes agiles

Le mot **méthode** est assez réducteur pour parler des méthodes agiles .il s’agit de plus que des méthodes, il s’agit de toute une vision, d’une approche agile, d’un paradigme agile ou encore plus loin : d’un état d’esprit agile.



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

Aujourd’hui, le but des entreprises n’est plus uniquement de rester dans la zone limitée par le triangle coût, délais et qualité mais aussi de livrer des produits de qualité.

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 est ni la date 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 lorsque on fait la livraison du produit et puis l’amélioration continu en se basant sur les retours du client.

## Définition des méthodes agiles

Les méthodes agiles caractérisent une approche de gestion de projet privilégiant le dialogue entre toutes les parties prenantes, les partenaires, les développeurs et autre 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 incrémentale et itératif par contradiction aux méthodes classiques qui utilisent un processus de développement incrémentale et itératif à la fois.

* + 1. Historique des méthodes agiles

Sans trop compliquer les choses, les premières apparitions des méthodes agiles était entre 1930 et 1940 à travers les recherches de Walter Schwehat et William Edward Deming. Ces recherches étaient introduit au développement des produits informatiques environ 1950. En fait, les méthodes agiles sont à la collaboration des nombreuses approches de conduite de projets tel que ceux de Tom Gilb, Scott Shultz, Brian Gallagher et celui de 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 importante dans l’historique des méthodes agile était le weekend de 13 et 14 février 2001, une réunion de 17 personnes expert de gestion de projets à Utah aux États Unies pour une mission de recherche d’une base communes aux différentes méthodes agiles.

Cette réunion a abouti à une documentation officielle des principes et les valeurs agiles qui devient connu aujourd’hui sous l’appellation de Manifeste Agile.

## 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 orienté produit, itérative et incrémentale et centrées sur les personnes. Le manifeste agile est une combinaison de quatre valeurs et douze principes fondamentaux.

* + - 1. Les valeurs de manifeste agile

Le manifeste agile s’étend sur quatre valeurs principaux 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
  + - 1. Les principes de manifeste agile

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

* Satisfaire le client en livrant rapidement et régulièrement des fonctionnalités a grande valeur ajoutée.
* Décomposer les travaux importants en composants plus petits rapidement réalisables.
* Reconnaitre que les meilleurs conceptions émergents d’équipes auto-organisé.
* 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

Etant être l’approche la plus adopté à nos jours pour faire la gestion de projets, les méthodes agiles présentent un certain nombre d’avantage. Entre autre nous citons :

* Moins de documentation et donc plus de productivité et celui-ci est le plus important pour le client car il préfère généralement de recevoir un produit fonctionnel au lieu d’une documentation complète
* Souplesse, élasticité et réactivité pour les demandes de changements
* Des versions incomplètes du produit sont livrées. Le client reçoit à intervalles réguliers une version fonctionnel du produit ce qui lui permet alors d’améliorer le produit en se basant sur les retours du client.

## Les limites des méthodes agiles

Certaines limites sont aussi présentes pour les méthodes agiles qui les rends parfois non applicable pour certaines catégories de projet ou encore pour certaines entreprises.

* Chronophage (charge supplémentaires) pour le client car il est parfois pleinement trop impliqué dans la procédure de développement du produit.
* Les méthodes agiles sont moins compatibles dans les entreprises à forte hiérarchie.
* La réduction de dégrée de détails de la documentation du projet conduit à une absence de guide en cas d’ambiguïté.

1. Étude de de la méthodologie adoptée : Scrum

## Définition du Framework Scrum

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

Le terme Scrum, signifie mêlée ou réunion, est inspirée du jeu de rugby, un jeu qui exige une équipe soudée et partage entre tous ses membres une direction commune. Historiquement, Scrum a été introduit au domaines de développement des produits informatique et par la suite, elle est étendu dans tout autres domaines.

## 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 produit. En 1995, Ken Schwaber et Jeff Sutherland ont présenté au monde les fondements de la méthode de Scrum a L’Oopsla à Austin au États Unies.

## Les principes Scrum

Comme tout autre méthode agile, Scrum exigent un règlement composé de six principes fondamentaux qui doivent obligatoirement impliques dans tout type de projet. Les six principes 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 4 :** 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

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

* Inspection

L’équipe de développement doit se consulter quotidiennement et à intervalle régulier afin de dégager d’éventuels écarts entre les objectifs finaux du projet et le travail réalisé. Si des divergences sont constatés, un ajustement aura lieu afin de rencontrer les objectifs exigés par le client.

* 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 au information relatives au projets. Ils partagent aussi une vision claire basées sur un langage et des définitions communs.

## L’auto-organisation

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

L’auto-organisation dans ce contexte est définie comme un processus par lequel une certaine forme d'ordre (organisation) découle des interactions entre les parties d'un système initialement désordonné. Le processus est spontané et ne nécessite aucun contrôle externe. L'organisation qui en résulte est complètement décentralisée et répartie sur tous les composants du système. En conséquence, l'organisation est généralement robuste, capable de survivre et de s'adapter aux perturbations.

## La collaboration

La communication, l’entraide et la collaboration sont des valeurs clés pour la méthode Scrum. La gestion de projet est considérée comme un processus qui exigent une communication efficace entre tout individus afin d’avoir une vision globale et commune de projet.

Dans un environnement de travail collaboratif, les dimensions de la collaboration sont les suivantes :

* Une prise de conscience

Les personnes qui travaillent ensemble doivent être conscientes du travail de l'autre.

* Articulation

Les personnes qui collaborent doivent diviser le travail en unités, diviser les unités entre les membres de l'équipe, puis une fois le travail terminé, le réintégrer.

* Appropriation

Adapter la technologie à sa propre situation ; la technologie peut

Être utilisée d'une manière complètement différente de celle prévue par les concepteurs.

## 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. Lors de la détermination des priorités, trois facteurs sont pris en compte :

* Evaluation
* Risque ou incertitude
* Dépendances

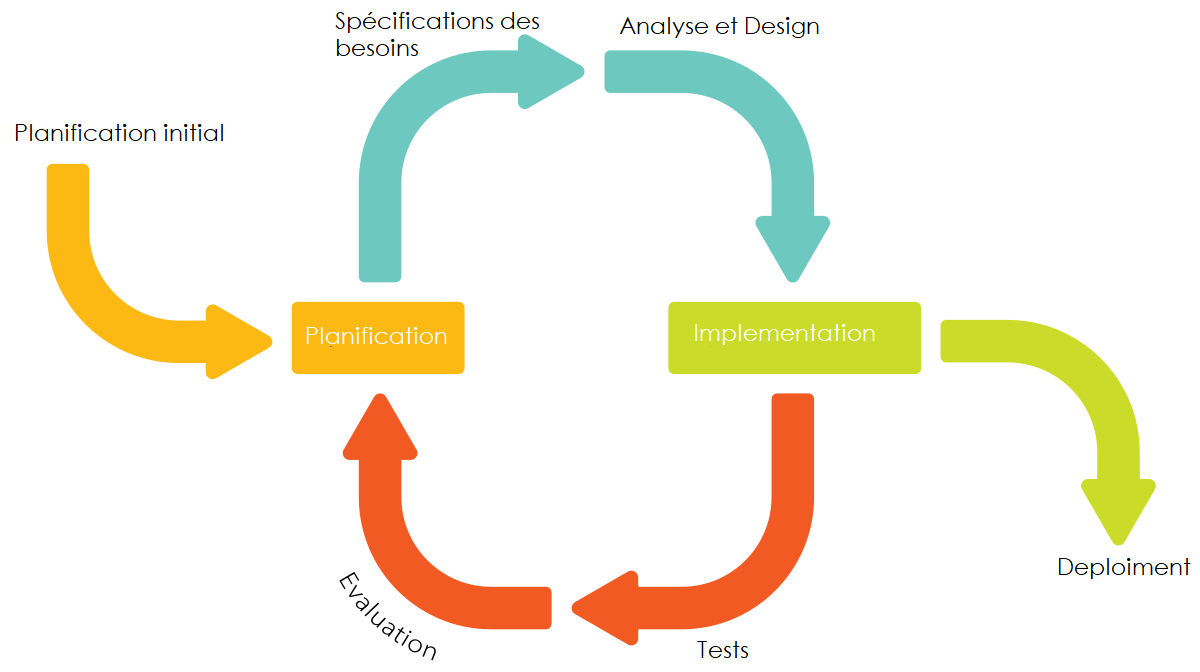
## Le Time Boxing

Comme tout autre méthode agile, le temps est considéré comme une contrainte qui doit être utilise et respecte afin de gérer l’organisation et la réalisation de projet. Dans Scrum, les sprints, les réunions et les mêlées sont tous des évènements limitées dans le temps afin d’imposer des contraintes temporelles afin de faire la livraison du maximum de valeur ajoutées dans un minimum de temps.

## 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 suivante montre les phases nécessaires lors de l’application d’une approche de développement itératif.



**Figure 5 :** Le processus de développement itératif

## Les rôles Scrum

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

* + 1. Les rôles essentiels

Les rôles essentiels sont des rôles nécessaires à la réalisation de chaque projet. 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. Il est chargé de transmettre les exigences du client à l’équipe responsable du développement de produit.

* Le Scrum Master

C’est un facilitateur qui assure que l’équipe de développement du produit possède un environnement confortable a la réussite du projet. Le Scrum Master est la personne la plus expérimenté 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 a chaque intervalle de temps. Par exemple, dans les projet informatique, l’équipe Scrum est composé par des développeurs informatiques, des designers, des testeurs etc...

* + 1. Les rôles non essentiels

Ce sont les rôles que leurs présences ne sont pas exigés. Cette catégorie inclut des individus intéressés pour le projet et qui n’ont pas un rôle défini mais qui doivent avoir un moyen pour communiquer avec l’équipe de développement mais ils n’ont aucune responsabilité de la réussite ou l’échec du projet.

* Les parties prenantes

Connu aussi sous le nom de stakeholders, c’est un terme qui regroupe les clients, les utilisateurs et les sponsors. Les parties prenantes interagissent avec l’équipe Scrum et peuvent influencer le projet.

* Le Scrum guidance Body

C’est un rôle facultatif qui peut être soit un ensemble de documents ou un groupe d’expert qui s’intéressent à la définition des mesures de qualité, aux légalisations gouvernementales, les mesures de sécurité et plusieurs autres mesures organisationnels clés.

* 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

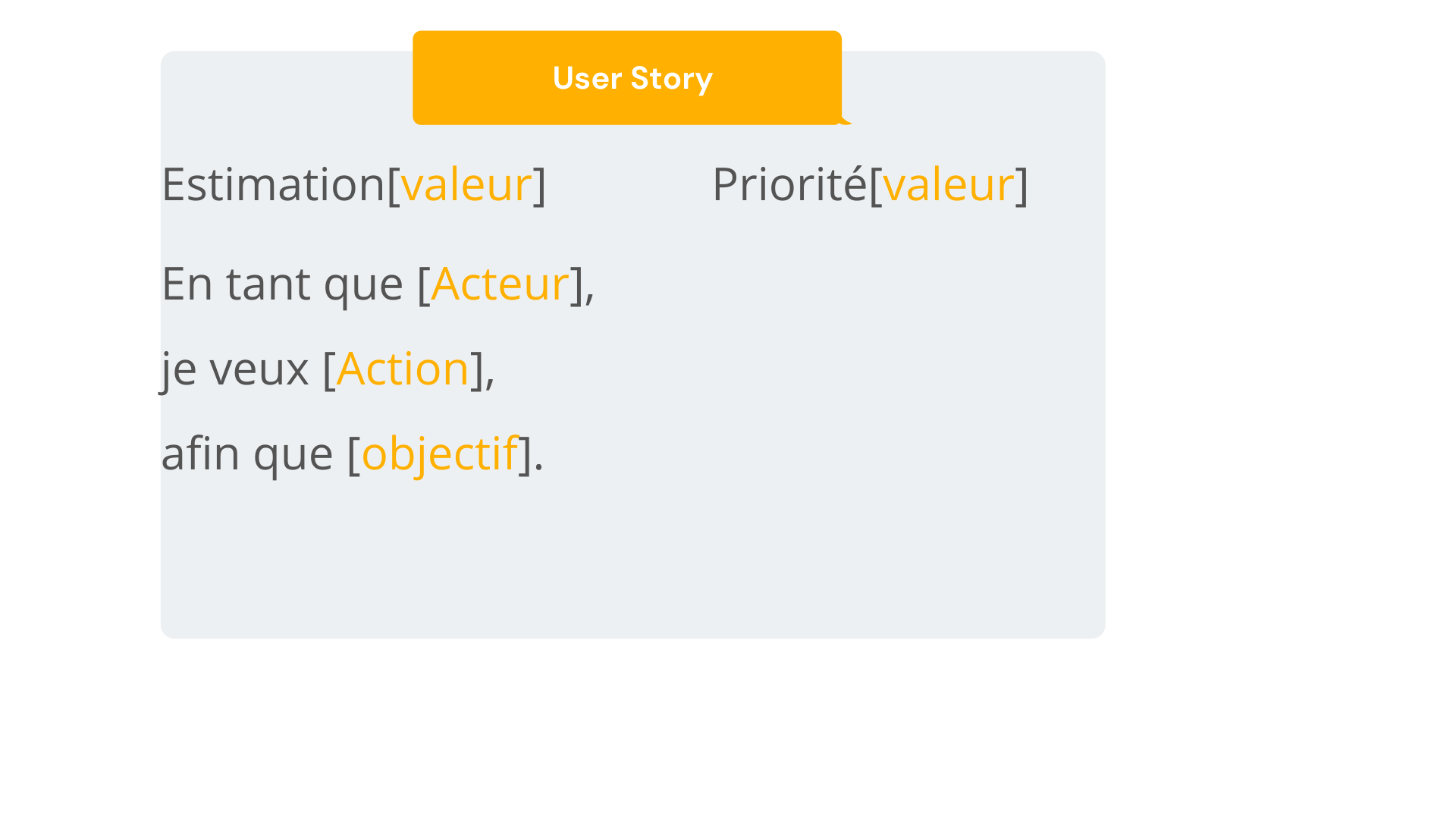
Le mot artefact dans le contexte Scrum fait référence aux composants de base permettant une transmission fluide, claire et précis d’informations et se servir comme un tableau commun consultable par tous les membres de l’équipe de développement pour tirer les informations nécessaires.

Les artefacts Scrum sont :

* User Story

Une User Story, histoire d’utilisateur en français, est une explication informelle et générale d’une fonctionnalité logicielle écrite du point de vue de l’utilisateur finale de produit. L’objectif d’une User Story est d’expliquer comment une fonctionnalité logicielle ou physique apportera de la valeur au client.

La figure ci-dessous présente le format général d’une User Story.

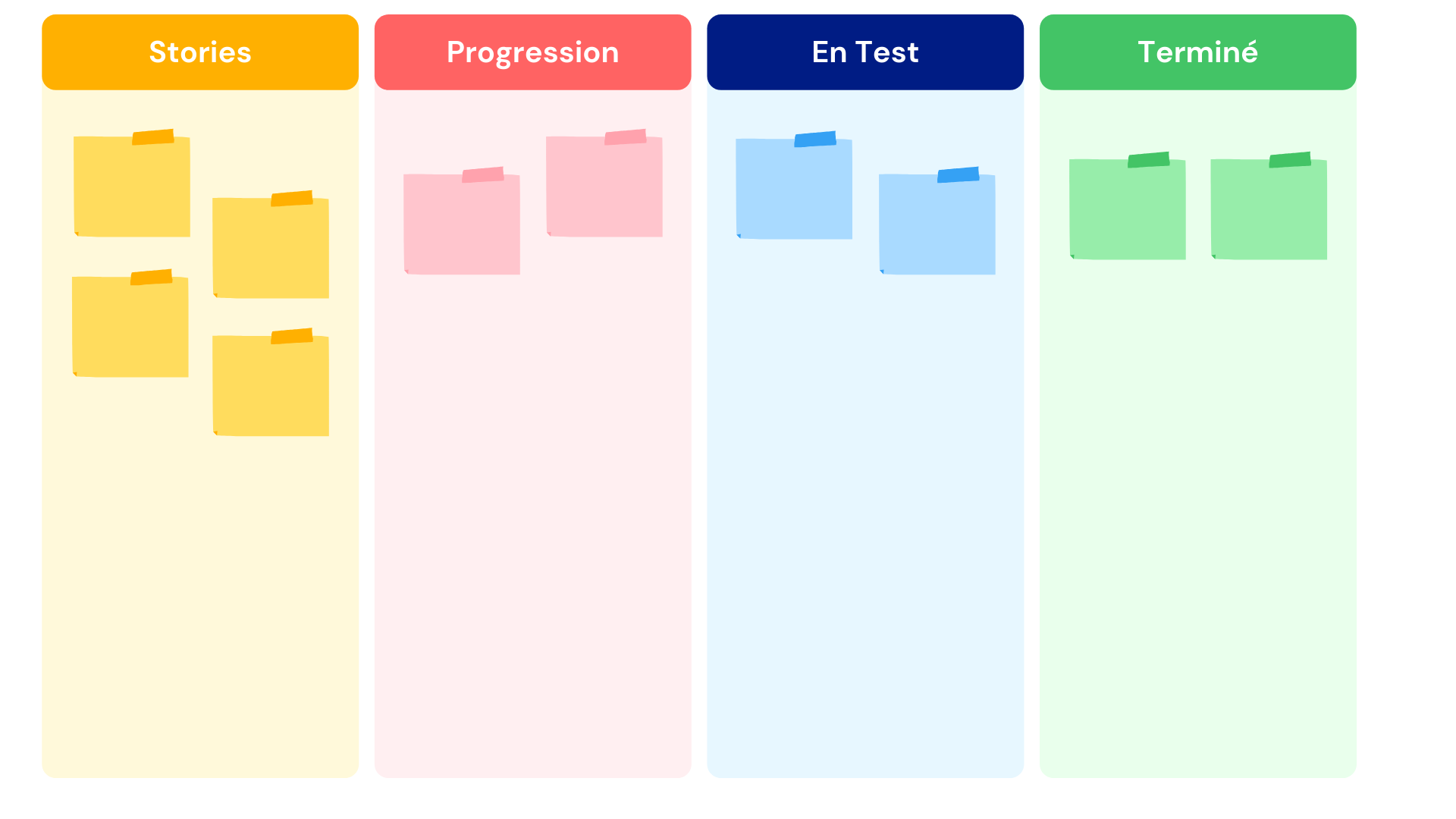


**Figure 6 :** le format général d’une user story

* 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.

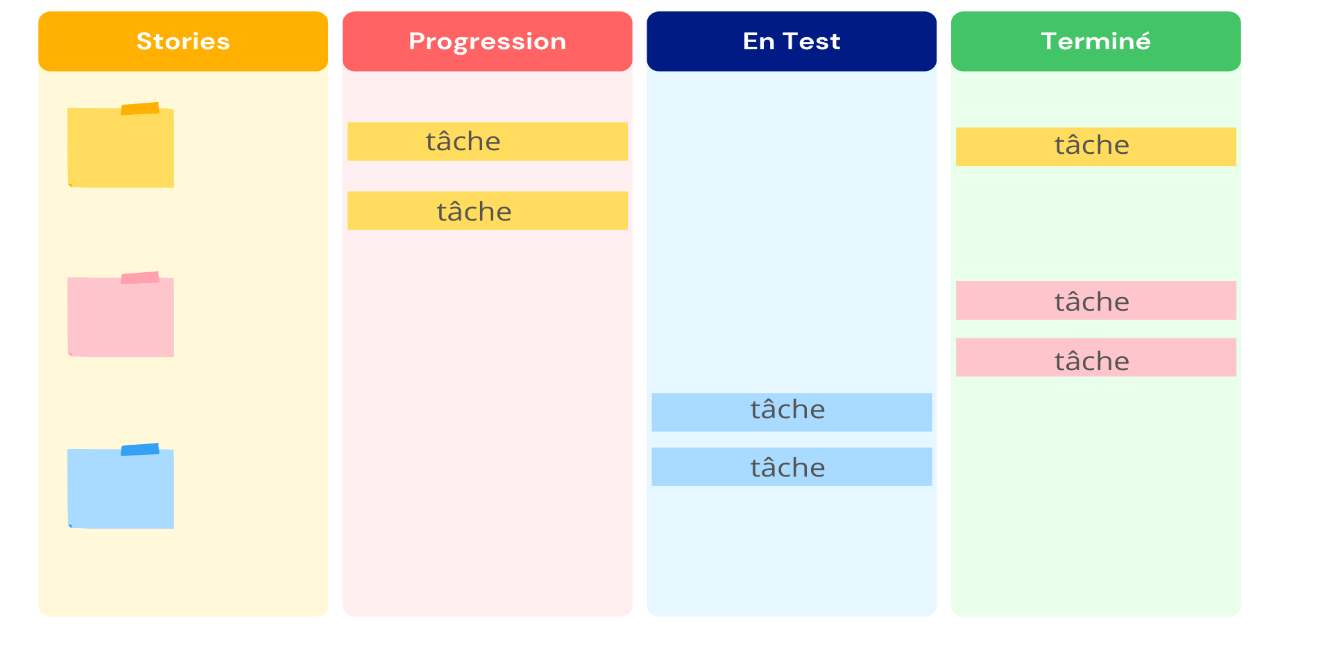
Les éléments présents dans le Backlog de produit sont ordonnés et triés selon leur priorité afin que l’équipe de développement sache ce qu’elle doit délivrer en premier.



**Figure 7 :** le format générale d’un backlog de produit

* Le Backlog de Sprint

Un backlog de sprint est l’ensemble des éléments qu’une équipé de production sélectionne dans son backlog de produit pour travailler sur le sprint à venir. En générale, les éléments présents dans le backlog de sprint représente le principal résultat de la planification de sprint.



**Figure 8 :** La forme générale d’un backlog de sprint

## Les cérémonies Scrum

En se basant au principe de Time Boxing présent dans la liste des principes de Framework 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 limité dans le temps 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.

Pour une meilleure application de Scrum, il est recommandé de limiter le temps d’un sprint à quatre semaines sauf dans le cas des projets trop complexes ou le sprint peut s’étendre jusqu’à six semaines.

* Mêlée quotidienne

La mêlée quotidienne ou le Daily meeting est une réunion quotidienne qui se limite à 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. La durée de cette réunion est calculée selon la durée du 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’équipé 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 et les fonctionnalités seront comparées aux critères d’acceptation du projet et seront classifiées soit en acceptées ou rejetées. La durée de cette réunion est calculée en fonction de la durée de sprint. 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. Cette réunion 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 dont l’objectif est d’améliorer les sprints à venir et de booster la productivité de l’équipe et aide à la création d’un environnement de travail plus confortable en se basant sur les retours de l’équipe de développement.

## Fonctionnement de Framework Scrum

La première étape d’un projet avec Scrum se concentre sur la collecte des informations autour de projet et nécessaire pour le développement.

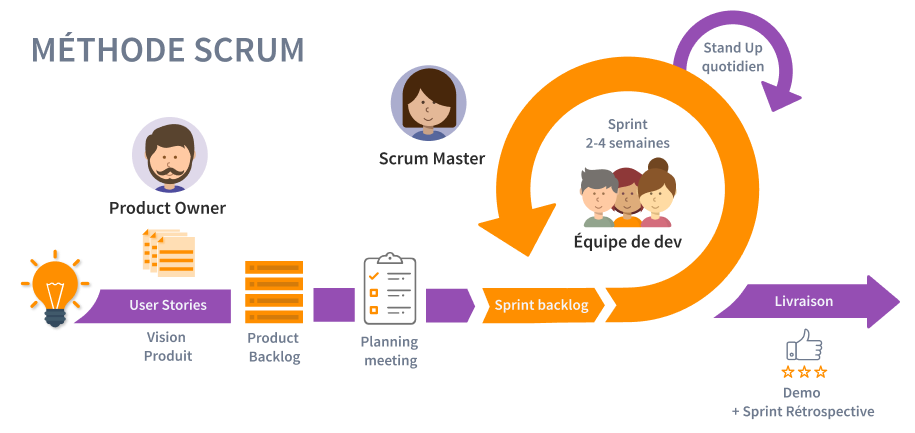
Cette mission est sous la responsabilité du Product Owner. Il doit contacter le client afin d’avoir sa vision de futur produit. Cette tâche est la brique de base pour l’implémentation des user stories.

Une fois les user stories sont collectés, elles seront classées par ordre de priorité dans ce qu’on appelle le backlog de produit qui représente le document principal de chaque projet Scrum.

Dès que le backlog de produit est priorisé, les user stories seront subdivisés en un ensemble de tâches sur plusieurs sprints lors de la réunion de planification de sprint qui conduit à la création d’un backlog de sprint.

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 de Scrum quotidiens ,des réunions d’équipe où tous les problèmes ,les obstacles, les demandes de changements sont discutées .Une fois le sprint est terminé ,tous les membres de l’équipe Scrum participent au revue de sprint et aux rétrospectives de sprint.

La phase finale de chaque cycle de sprint est la phase d’estimation du produit. Le produit est évalué par rapport à la définition de ‘Terminé’ de l’équipe et présenté au client pour recevoir ses commentaires pour des améliorations au futur. Ensuite un nouveau sprint commence et le cycle se répète.

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 9 :** Le cycle de réalisation d’un projet selon l’approche Scrum

Chapitre 3 Analyse et spécification des besoins

## 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 concernés de notre système. Puis nous élaborons l’étude des besoins fonctionnels et les besoins non fonctionnels ainsi que l’identification des acteurs qui vont interagir avec notre système.

Ces besoins seront modélisés par la suite sous forme des diagrammes de cas d’utilisation qui permettent de détailler les scenarios possibles qui peuvent être réalisées par les intervenants du système.

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 catégorie 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 et qui permettent la génération des cas d’utilisation.

* La gestion de profil

La gestion de profil est pilotée par tous les utilisateurs de notre application. Cette tâche consistera essentiellement à permettre à l’utilisateur de :

* Consulter son profil et ses informations
* Éditer les informations de son profil tel que ses informations personnelles, les informations de son parcours scolaires et la listes de ses compétences.
* La gestion des messages

La gestion des messages est une tâche essentielle de notre application.

Cette tâche consiste essentiellement à permettre aux utilisateurs de :

* 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

La gestion des réunions est une tâche essentielle qui donne aux utilisateurs le pouvoir de :

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

Comme tout autre système interactif, la gestion des notifications est une tâche de base permettant aux utilisateurs de :

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

C’est la brique de base de notre système. Cette tâche peut être effectué par le Product Owner, le Scrum Master et les développeurs.

* Le Product Owner
* Créer un nouveau projet.
* Supprimer un projet.
* Modifier un projet.
* Terminer un projet.
* Consulter la liste de ses projets.
* Le Scrum Master
* Consulter la liste de ses projets.
* Les développeurs
* Consulter la liste de ses projets.
* La gestion des tâches

Cette tâche peut être effectué par le développeur dans chaque projet dont il est membre de l’équipe de développement. Le développeur doit être capable de :

* 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

Le journal d’obstacle est un outil efficace pour ajuster l’environnement de travail et à faciliter le processus de développement en éliminant tous les obstacles rencontrés par l’équipe de développement. Le Product Owner, le Scrum Master et les développeurs sont capable de :

* 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

Le journal des risques est un journal où l’équipe d’un projet composé du Product Owner, le Scrum Master et les développeurs auront le pouvoir de :

* Consulter les risques déclarés d’un projet.
* Modifier un risque déclaré.
* Supprimer un risque.

Le but de journal des risques est de permettre le suivi de projet et d’ajuster le plan de réalisation afin d’éviter les risques qui menace la réussite

d’un projet.

* La gestion des user stories

La gestion des user stories permet au Product Owner de faire une représentation détaillée des exigences de projet. Cette tâche est réalisable par le Product Owner, le Scrum Master et les développeurs.

* Le Product Owner
* 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.
* Le Scrum Master
* Consulter la liste des user stories d’un projet.
* Les développeurs
* Consulter la liste des user stories d’un projet.
* La gestion de l’équipe de développement

La gestion de l’équipe de développement est pilotée par le Scrum Master. Cette tâche consistera essentiellement en :

* 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 peut être effectue par le Product Owner, le Scrum Master et les développeurs.

* Le Product Owner
* Consulter l’historique des sprints d’un projet.
* Consulter les charts de sprint en cours.
* Annuler un sprint.
* Le Scrum Master
* Consulter l’historique des sprints d’un projet.
* Consulter les charts de sprint en cours.
* Créer un nouveau sprint.
* Déclencher un nouveau sprint.
* Les développeurs
* Consulter l’historique des sprints d’un projet.
* Consulter les charts de sprint en cours.
* La gestion des tâches relatives aux user stories

La gestion des tâches est effectuée par deux types d’acteurs :

* Le Scrum Master
* Consulter la liste des tâches relatives aux user stories.
* Créer une nouvelle tâche.
* Supprimer une tâche.
* Modifier une tâche.
* Affecter une tâche a un membre.
* Les Développeurs
* Consulter la liste des tâches relatives aux user stories.
* Créer une nouvelle tâche.
* Supprimer une tâche.
* Modifier une tâche.
* Affecter une tâche a un membre.
* La gestion des stakeholders

La gestion des stakeholders d’un projet est effectué par deux types d’acteurs :

* Le Product Owner
* Consulter la liste des stakeholders d’un projet.
* Désigner un nouveau Stakeholder.
* Supprimer un Stakeholder.
* Modifier le rôle d’un Stakeholder.
* Le Scrum Master
* 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

La persona d’un projet est un outil qui permet à l’équipe de projet de modéliser l’utilisateur finale d’un produit. Au sein de notre projet. Cette fonctionnalité est sous la responsabilité de de trois acteurs qui sont :

* Le Product Owner
* Consulter la persona d’un projet.
* Créer la persona d’un projet.
* Supprimer la persona d’un projet.
* Modifier la persona d’un projet.
* Le Scrum Master
* Consulter la persona d’un projet.
* Les développeurs
* Consulter la persona d’un projet.

## Les besoins non fonctionnels

Les besoins non fonctionnels sont des besoins qui caractérisent le système. Ce sont mesures de performance, de type de matériel ou aussi le type de conception. Ces besoins 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 ultérieurs

* La fiabilité

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

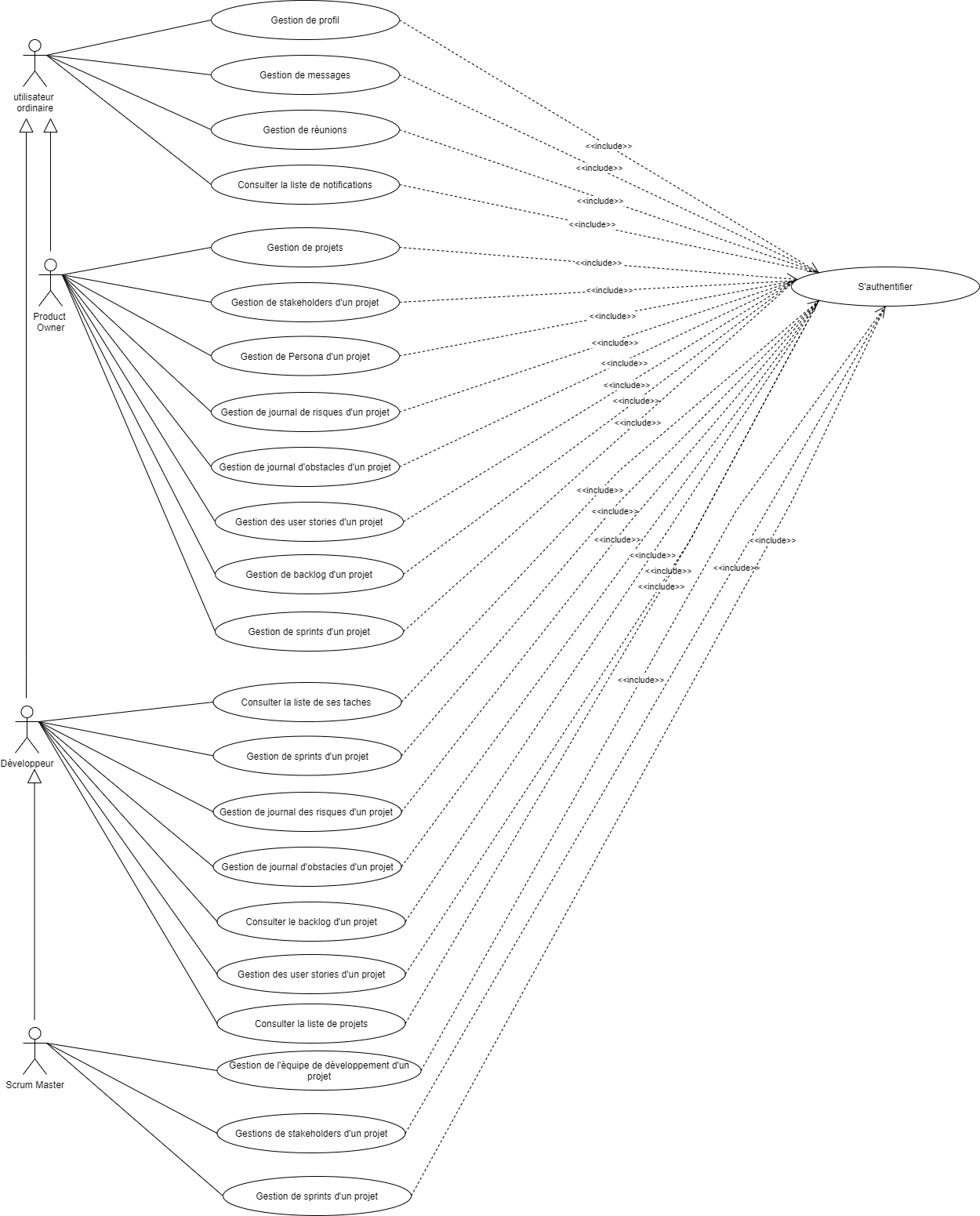
* La scalabilité ou l’évolutivité

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

## Diagramme de cas d’utilisation globale

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

dégager plusieurs cas d'utilisation.

La figure ci-dessous présente toutes les cas d'utilisation de base pour le but de fournir une vue globale de fonctionnement de notre application et de modéliser toutes les relations qui pourraient survenir d'après.

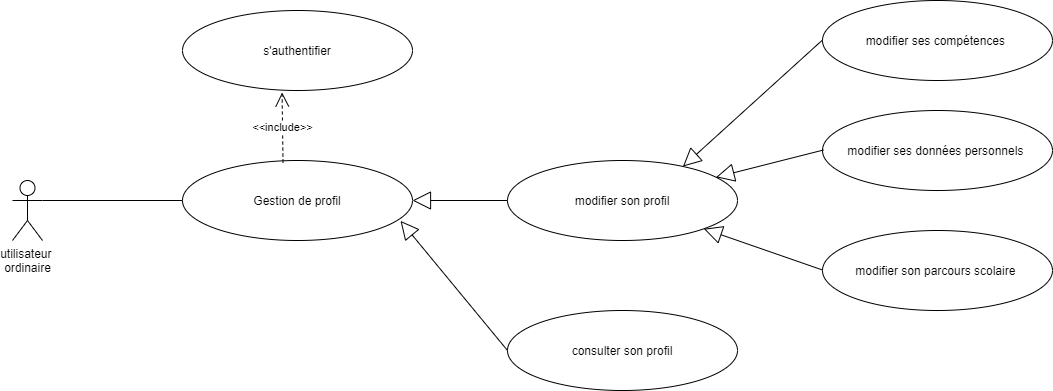
**Figure 10 :** Diagramme de cas d’utilisation globale

4. Raffinement des cas d’utilisation

Dans cette partie, nous allons présenter les diagrammes de cas d’utilisation raffinés ainsi que la description textuelle de chaque diagramme.

## Cas d’utilisation « Gestion de profil »

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



**Figure 11 :** Diagramme de cas d’utilisation Gestion de profil

Dans ce qui suit, nous allons détailler les différents cas d’utilisation présents dans le diagramme de cas d’utilisation Gestion de profil.

* Modifier son profil

Après être authentifié, l’utilisateur peut modifier les données de son profil comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier son profil |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer les données de l’utilisateur. |
| **Post condition** | Les données de l’utilisateur mis à jour. |
| **Scénario de base** | L’utilisateur saisie ses nouvelles données  L’utilisateur clique sur mettre à jour.  Le système met à jour les données de l’utilisateur. |
| **Exception** | Si un champ est vide ou invalide ,les système doit afficher un message d’erreur . |

* Consulter son profil

Après le processus d’authentification, l’utilisateur peut également consulter son profil comme le présente le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter son profil |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer les données de l’utilisateur. |
| **Post condition** | Les données de l’utilisateur sont affichées |
| **Scénario de base** | L’utilisateur clique sur Mon profil.  Le système affiche les données de l’utilisateur. |
| **Exception** | Si le système ne réussit pas à récupérer les données de l’utilisateur ,il doit afficher une page d’erreur. |



## Cas d’utilisation « Gestion de messages »

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



**Figure 12 :** Diagramme de cas d’utilisation Gestion de messages

Dans ce qui suit nous allons fournir une description textuelle de déroulement de chaque cas d’utilisation présente dans le diagramme ci-dessus.

* Consulter la liste des messages

Après être authentifié, l’utilisateur aura la possibilité de consulter la liste de ses messages.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de messages |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer les messages de l’utilisateur. |
| **Post condition** | Les messages de l’utilisateur sont affichées |
| **Scénario de base** | L’utilisateur clique sur ‘Chat’.  Le système met à jour les données de l’utilisateur. |

* Rédiger un nouveau message

Après être authentifié, l’utilisateur peut rédiger un nouveau message.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Rédiger un nouveau message |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer la liste des destinataires. |
| **Post condition** | Un nouveau message est envoyé. |
| **Scénario de base** | L’utilisateur saisie le contenu, l’objet et sélectionne le destinataire du message.  L’utilisateur clique ‘envoyer’.  Le système envoi un nouveau message. |
| **Exception** | Si un champ est vide ou invalide ,les système doit afficher un message d’erreur . |

* **Supprimer un message**

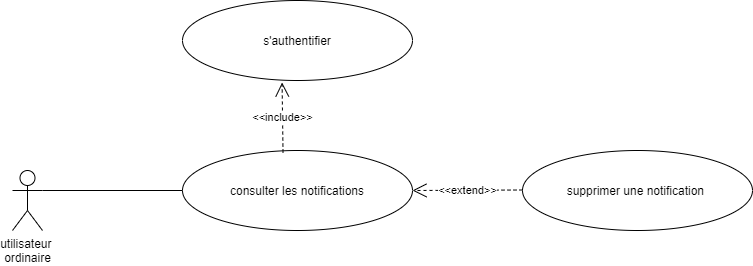
**Après être authentifié et l’affichage de la liste des messages, l’utilisateur peut également supprimer l’un de ses messages comme l’illustre le tableau suivant.**

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer un message |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer la liste de messages de l’utilisateur. |
| **Post condition** | Le message sélectionné est supprimé |
| **Scénario de base** | L’utilisateur sélectionne le message à supprimer et clique sur ‘supprimer’.  Le système supprime le message sélectionné. |
| **Exception** | Si le système ne réussit pas à supprimer le message ,il doit afficher une page contenant un message d’erreur. |



## Cas d’utilisation « consulter les notifications »

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



**Figure 13 :** Diagramme de cas d’utilisation consulter les notifications

* ****Consulter les notifications****

Après le processus d’authentification, l’utilisateur peut consulter la liste de ses notifications comme le décrit le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de notifications |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer les notifications de l’utilisateur. |
| **Post condition** | Les notifications de l’utilisateur sont affichées. |
| **Scénario de base** | L’utilisateur clique sur ‘notifications’.  Le système affiche les notifications de l’utilisateur. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

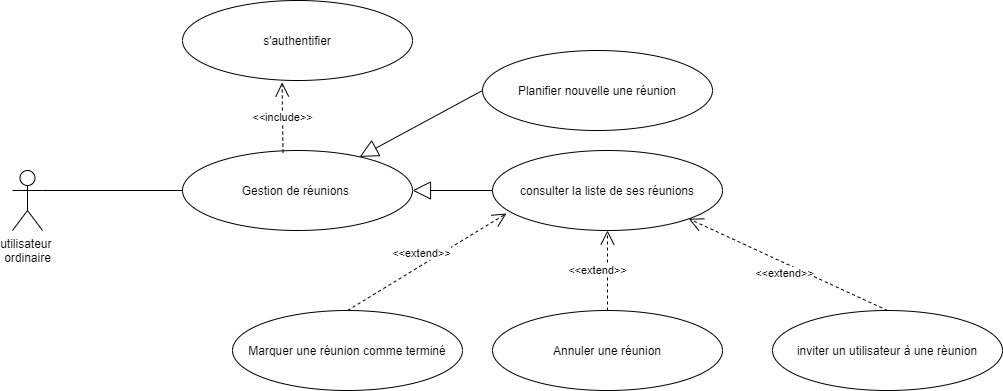
* **Supprimer une notification**

Après la consultation de la liste de notifications, l’utilisateur peut choisir de supprimer une notification comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer une notification |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer la liste de notifications de l’utilisateur. |
| **Post condition** | La notification sélectionné est supprimé. |
| **Scénario de base** | L’utilisateur sélectionne la notification à supprimer et clique sur ‘supprimer’.  Le système supprime la notification sélectionnée. |
| **Exception** | En cas d’erreur , le système doit afficher une page d’erreur . |

## Cas d’utilisation « gestion de réunion »

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

****

**Figure 11 :** Diagramme de cas d’utilisation Gestion de réunions

Dans ce qui suit, nous allons détailler les différents cas d’utilisation présents dans le diagramme de cas d’utilisation Gestion de réunion.

* Planifier une nouvelle réunion

Après être authentifié, l’utilisateur peut choisir de planifier une nouvelle réunion comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Planifier une nouvelle réunion |
| **Acteur** | Utilisateur ordinaire |
| **Post condition** | Une nouvelle réunion est planifiée. |
| **Scénario de base** | L’utilisateur saisie les données de la réunion tels que la date, la description et le type de réunion et clique sur ‘créer’.  Le système créer une nouvelle réunion. |
| **Exception** | En cas d’erreur , le système doit afficher une page d’erreur . |

* Consulter la liste de réunions

L’utilisateur peut consulter la liste de ses réunions comme il est expliqué dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de réunions |
| **Acteur** | Utilisateur ordinaire |
| **Précondition** | Le système réussit à récupérer les réunions de l’utilisateur. |
| **Post condition** | Les réunions de l’utilisateur sont affichées. |
| **Scénario de base** | L’utilisateur clique sur ‘réunion’  Le système affiche les réunions de l’utilisateur. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Marquer une réunion comme terminée

L’utilisateur peut mettre à jour l’état d’une réunion de ‘en attente’ à ‘terminée’ comme il expliqué dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Marquer une réunion comme terminée |
| **Acteur** | Utilisateur ordinaire |
| **Post condition** | La réunion sélectionnée est marquée comme terminée |
| **Scénario de base** | L’utilisateur sélectionne une réunion et clique sur ‘terminer’  Le système met à jour la réunion sélectionnée. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Annuler une réunion

L’utilisateur peut annuler une réunion comme il expliqué dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Annuler une réunion |
| **Acteur** | Utilisateur ordinaire |
| **Post condition** | La réunion sélectionnée est marquée comme annulée |
| **Scénario de base** | L’utilisateur sélectionne une réunion et clique sur ‘Annuler’  Le système met à jour l’état de la réunion sélectionnée. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

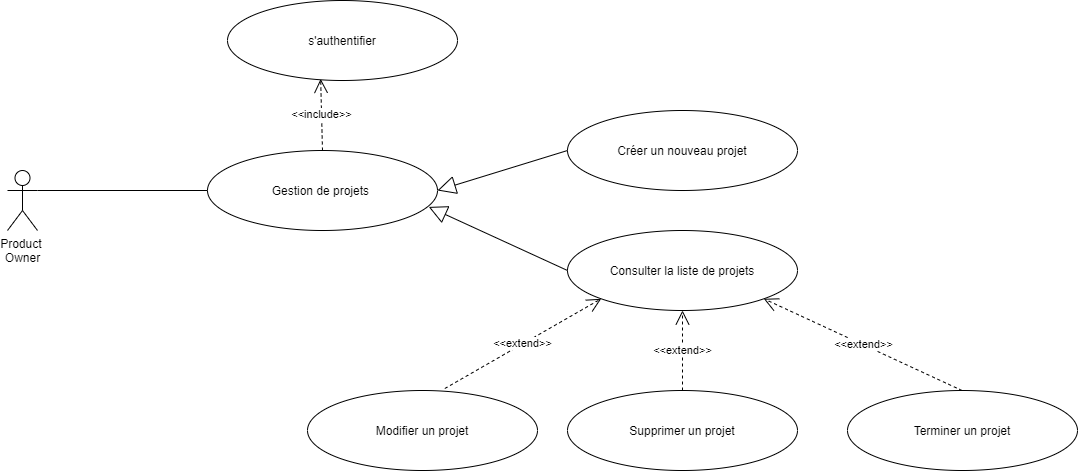
* Inviter un utilisateur à une réunion

L’utilisateur peut également envoyer une invitation à un autre utilisateur de système pour assister une réunion comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Inviter un utilisateur à une réunion |
| **Acteur** | Utilisateur ordinaire |
| **Post condition** | L’utilisateur sélectionné reçoit une invitation pour la réunion sélectionnée. |
| **Scénario de base** | L’utilisateur sélectionne une réunion et clique sur ‘Inviter un utilisateur’.  L’utilisateur sélectionne le destinataire de l’invitation et clique sur ‘Envoyer’.  Le système envoie l’invitation sous forme d’une notification. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « Gestion de projets »

La figure présenté ci-dessous représente le diagramme de cas d’utilisation gestion de projets.



**Figure 13 :** Diagramme de cas d’utilisation Gestion de projets

Dans ce qui suit, nous allons détailler les différents cas d’utilisation présents dans le diagramme de cas d’utilisation Gestion de projets.

* Créer un nouveau projet

Après la terminaison de processus d’authentification, le Product Owner peut créer un nouveau projet comme illustré dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Créer un nouveau projet |
| **Acteur** | Product Owner |
| **Post condition** | Un nouveau projet est créé. |
| **Scénario de base** | L’utilisateur saisie les données du nouveau projet tels que le titre, la description, le domaine ...  L’utilisateur clique sur ‘Créer’.  Le système crée un nouveau projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter la liste de projets

Le Product Owner, le Scrum Master et les développeurs sont capables également de consulter la liste de projets comme le décrit le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de projets |
| **Acteur** | Product Owner, Scrum Master ,Développeur |
| **Précondition** | Le système réussit à trouver la liste de projets de l’utilisateur. |
| **Post condition** | La liste de projets est affiché |
| **Scénario de base** | L’utilisateur clique sur ‘Projets’  Le système affiche la liste de projets de l’utilisateur |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier un projet

Autre que la création d’un nouveau projet, le Product Owner est capable aussi de modifier un projet comme expliqué dans le tableau ci-dessous.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier un projet |
| **Acteur** | Product Owner |
| **Post condition** | Les données du projet sont modifiées |
| **Scénario de base** | L’utilisateur saisie les nouvelles données relatives au projet.  L’utilisateur clique sur ‘Mettre à jour’.  Le système modifie les données du projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Terminer un projet

Après être authentifié, le Product Owner peut également terminer un projet comme l’illustre le tableau ci-dessous.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Terminer un projet |
| **Acteur** | Product Owner |
| **Post condition** | La projet sélectionné est marqué comme terminé |
| **Scénario de base** | L’utilisateur sélectionne un projet et clique sur ‘terminer’.  Le système marque le projet sélectionné comme terminé. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

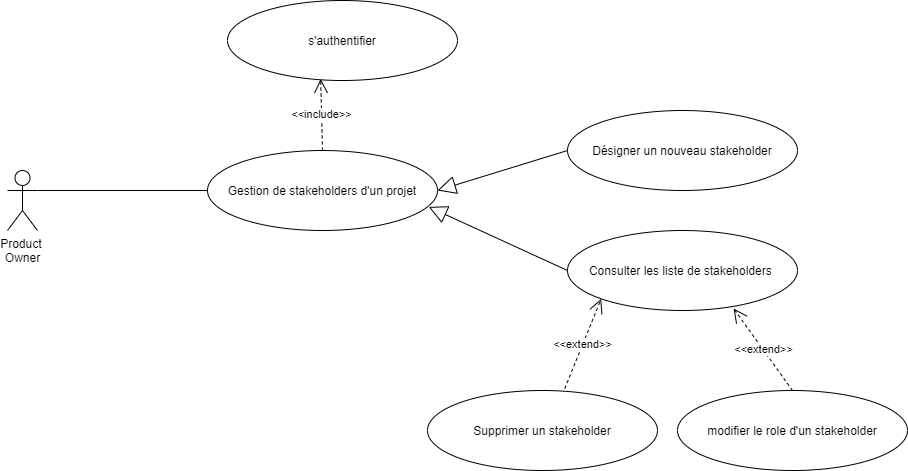
* Supprimer un projet

Après être authentifié, le Product Owner peut supprimer un projet comme l’illustre le tableau ci-dessous.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer un projet |
| **Acteur** | Product Owner |
| **Post condition** | Le projet sélectionné est supprimé |
| **Scénario de base** | L’utilisateur sélectionne un projet et clique sur ‘Supprimer’.  Le système supprime le projet sélectionné. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation «Gestion de stakeholders d’un projet»

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



**Figure 14:** Diagramme de cas d’utilisation Gestion de stakeholders d’un projet

Dans ce qui suit, nous allons détailler les cas d’utilisation présents dans le diagramme de cas d’utilisation Gestion de stakeholders d’un projet.

* Designer un nouveau stakeholder

Après être authentifié, les Product Owner et le Scrum Master peuvent désigner un nouveau stakeholder pour un projet comme l’illustre le tableau ci-dessous.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Designer un nouveau stakeholder |
| **Acteur** | Product Owner, Scrum Master |
| **Précondition** | Le système réussit à sélectionner tous les personnels présents dans la base de données.  Le stakeholder sélectionné n’est pas présent dans la liste des stakeholder . |
| **Post condition** | Un nouveau stakeholder est désigné pour le projet sélectionné |
| **Scénario de base** | L’utilisateur sélectionne un stakeholder parmi la liste des utilisateurs.  Le système enregistre l’utilisateur sélectionné comme un nouveau stakeholder du projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |
| **Extensions** | Le Product Owner et le Scrum Master peuvent faire une recherche par nom et prénom pour trouver le profil cherché.  **Enchainement :**   1. Le PO ou le SM saisie le nom et le prénom du profil cherché.   Le PO ou le SM clique sur ‘Rechercher’.  Le système affiche tous les utilisateurs dont le nom et le prénom incluent l’expression saisie. |

* Consulter la liste de stakeholders

Après être authentifié, le Product Owner et le Scrum Master peuvent consulter la liste de stakeholders d’un projet comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de stakeholders d’un projet |
| **Acteur** | Product Owner, Scrum Master |
| **Précondition** | Le système réussit à trouver la liste de stakeholders d’un projet. |
| **Post condition** | La liste de stakeholders est affiché |
| **Scénario de base** | L’utilisateur clique sur ‘Stakeholders’.  Le système affiche la liste de stakeholders d’un projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier le rôle d’un stakeholder

Après la consultation de la liste de stakeholders d’un projet, le Product Owner et le Scrum master peuvent modifier le rôle d’un stakeholder comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier le rôle d’un stakeholder |
| **Acteur** | Product Owner ,Scrum Master |
| **Post condition** | Le rôle du stakeholder est modifié. |
| **Scénario de base** | L’utilisateur saisie le nouveau rôle du stakeholder.  L’utilisateur clique sur ‘Mettre à jour’.  Le système modifie le rôle du stakeholder. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

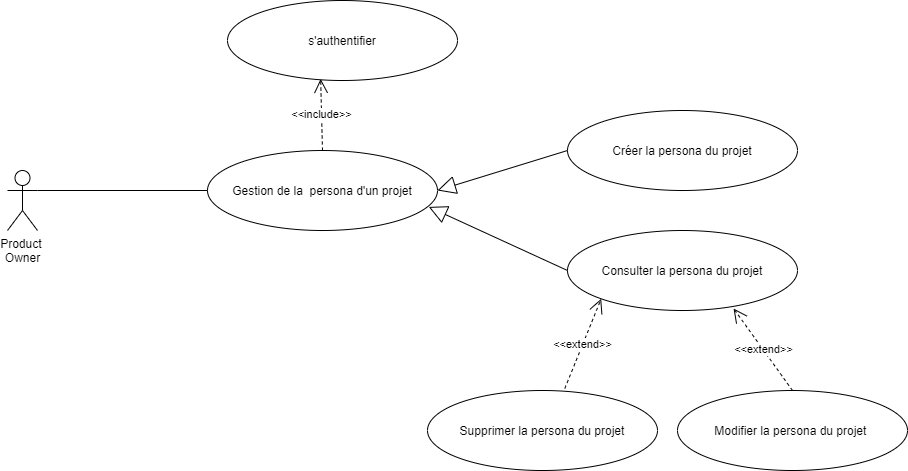
* Supprimer un stakeholder

Autre que la modification du rôle d’un stakeholder, le Product Owner et le Scrum Master sont capables de supprimer un stakeholder de la liste de stakeholders d’un projet.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer un stakeholder |
| **Acteur** | Product Owner, Scrum Master |
| **Post condition** | Le stakeholder sélectionné est supprimé |
| **Scénario de base** | L’utilisateur sélectionne un stakeholder et clique sur ‘Supprimer’.  Le système supprime le stakeholder sélectionné. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « gestion de la Persona d’un projet »

La figurer ci-dessous représente le diagramme de cas d’utilisation Gestion du Persona d’un projet.



**Figure 14:** Diagramme de cas d’utilisation Gestion du persona d’un projet

Dans ce qui suit, nous allons détailler tous les cas d’utilisation présentes dans le diagramme de cas d’utilisation Gestion du persona d’un projet.

* Créer la persona d’un projet

Après être authentifié, le Product Owner peut créer la persona d’un projet si elle n’est pas encore créée comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Créer la persona du projet |
| **Acteur** | Product Owner |
| **Précondition** | La persona du projet n’est pas encore crée. |
| **Post condition** | La persona du projet est créée . |
| **Scénario de base** | L’utilisateur saisie les données de la persona du projet tels que le nom, l’âge, le sexe ...  L’utilisateur clique sur ‘Créer’.  Le système crée la persona du projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter la persona du projet

Une fois la persona du projet est créée le Product Owner, le Scrum Master et les développeurs peuvent la consulter selon la description présente dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la persona du projet |
| **Acteur** | Product Owner, Scrum Master, développeur |
| **Précondition** | La persona du projet est déjà encore crée. |
| **Post condition** | La persona du projet est affiché . |
| **Scénario de base** | L’utilisateur clique sur afficher la persona du projet.  Le système rechercher la persona du projet et l’affiche dans l’interface adéquate . |
| **Exception** | En cas d’erreur ou lorsque la persona du projet n’est pas encore crée , le système doit afficher une page d’erreur . |

* Modifier la persona du projet

Après la consultation de la persona de projet, le Product Owner peut modifier la persona de projet comme l’explique le tableau suivant.

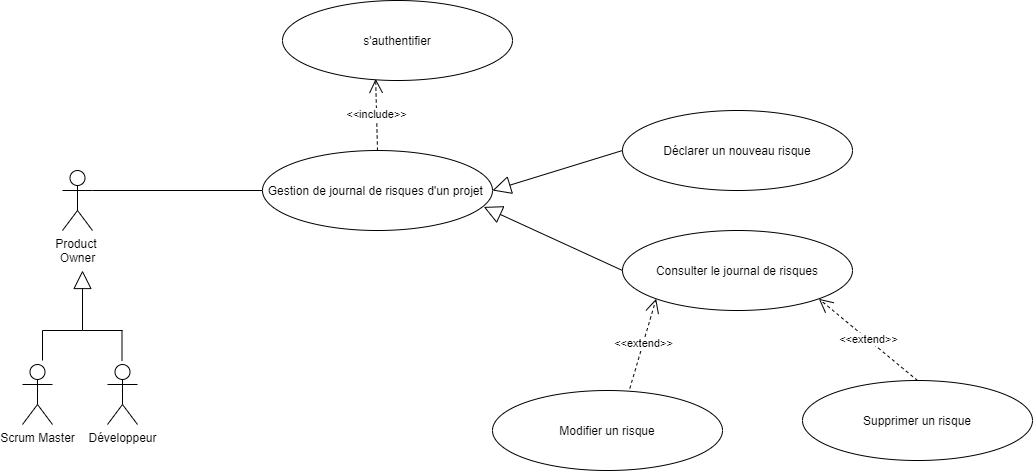
|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier la persona du projet |
| **Acteur** | Product Owner |
| **Post condition** | La persona du projet est modifiée. |
| **Scénario de base** | L’utilisateur saisie les nouvelles données de la persona.  L’utilisateur clique sur ‘Mettre à jour’.  Le système modifie la description de la persona du projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Supprimer la persona du projet

Après être authentifié et autre que la modification de la persona du projet, le Product Owner est capable de supprimer la persona du projet comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer la persona du projet |
| **Acteur** | Product Owner |
| **Précondition** | La persona du projet est existante. |
| **Post condition** | La persona du projet est supprimé |
| **Scénario de base** | L’utilisateur clique sur ‘Supprimer la persona’.  Le système supprime la persona du projet . |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « gestion de journal de risques »

La figure ci-dessous représente le diagramme de cas d’utilisation Gestion de journal de risques.

**Figure 15:** Diagramme de cas d’utilisation Gestion de journal de risques

**Dans ce qui suit,** nous allons détailler tous les cas d’utilisation présents dans le diagramme de cas d’utilisation Gestion de journal de risques.

* Déclarer un nouveau risque

Après être authentifié, le Product Owner, le Scrum Master et les développeurs peuvent déclarer des nouveaux risques comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Décaler un nouveau obstacle |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Post condition** | Le nouveau risque est déclarer. |
| **Scénario de base** | L’utilisateur saisie les données du nouveau risque.  L’utilisateur clique sur ‘déclarer’.  Le système enregistre le nouveau risque. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter le journal de risques

Une fois authentifié, le Product Owner, le Scrum Master et les développeurs peuvent consulter le journal de risques d’un projet comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter le journal de risques |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | Le système réussit a sélectionner tous les risques déclarés relatives à un projet. |
| **Post condition** | La liste de risques déclarés est affiché. |
| **Scénario de base** | L’utilisateur clique sur ‘journal de risques’  Le système affiche la liste de risques. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier un risque

Une fois l’un de trois acteurs déclare un risque, il peut à tout moment le modifier comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier un risque |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | L’utilisateur est le propriétaire de la déclaration du risque. |
| **Post condition** | Le risque sélectionné est modifié. |
| **Scénario de base** | L’utilisateur saisie les nouvelles données du risque.  L’utilisateur clique sur ‘mettre à jour’.  Le système enregistre les nouvelles données du risque. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

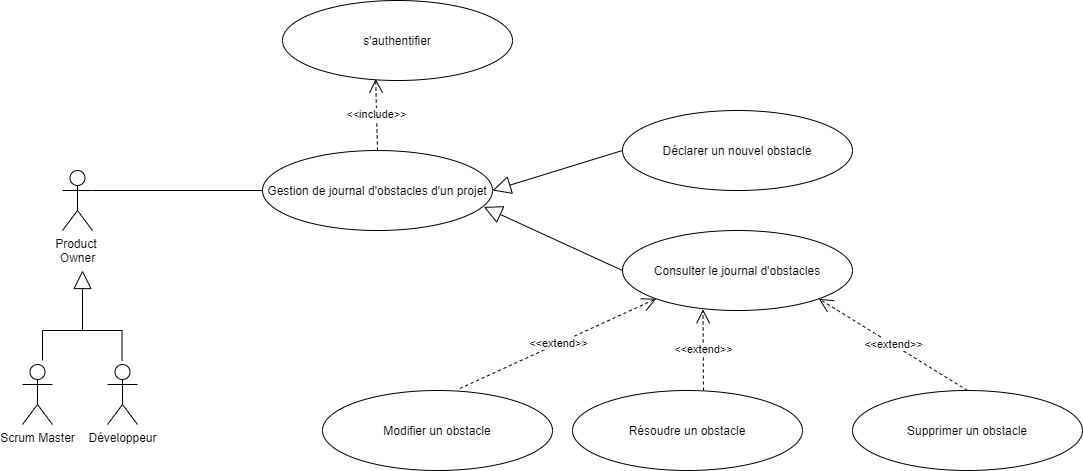
* Supprimer un risque

Une fois l’un de trois acteurs déclare un risque, il peut à tout moment le supprimer comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer un risque |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | L’utilisateur est le propriétaire de la déclaration du risque. |
| **Post condition** | Le risque sélectionné est supprimé. |
| **Scénario de base** | L’utilisateur clique sur ‘supprimer’.  Le système supprime le risque sélectionné. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « gestion de journal d’obstacles d’un projet »

La figure présente ci-dessous représente le diagramme de cas d’utilisation Gestion de journal d’obstacles d’un projet.



**Figure 15:** Diagramme de cas d’utilisation Gestion de journal d’obstacles

**Dans ce qui suit,** nous allons détailler tous les cas d’utilisation présents dans le diagramme de cas d’utilisation Gestion de journal d’obstacles

* Déclarer un nouvel obstacle

Après être authentifié, le Product Owner, le Scrum Master et les développeurs peuvent déclarer des nouveaux obstacles comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Décaler un nouvel obstacle |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Post condition** | Le nouveau obstacle est déclarer. |
| **Scénario de base** | L’utilisateur saisie les données du nouvel obstacle.  L’utilisateur clique sur ‘déclarer’.  Le système enregistre le nouveau obstacle. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter le journal d’obstacles

Une fois authentifié, le Product Owner, le Scrum Master et les développeurs peuvent consulter le journal d’obstacles d’un projet comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter le journal de obstacles |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | Le système réussit à sélectionner tous les obstacles déclarés relatives à un projet. |
| **Post condition** | La liste de obstacles déclarés est affiché. |
| **Scénario de base** | L’utilisateur clique sur ‘journal d’obstacles’  Le système affiche la liste d’obstacles. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier un obstacle

Une fois l’un de trois acteurs déclare un obstacle, il peut à tout moment le modifier comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier un obstacle |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | L’utilisateur est le propriétaire de la déclaration de l’obstacle. |
| **Post condition** | L’obstacle sélectionné est modifié. |
| **Scénario de base** | L’utilisateur saisie les nouvelles données de l’obstacle.  L’utilisateur clique sur ‘mettre à jour’.  Le système enregistre les nouvelles données de l’obstacle. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Supprimer un obstacle

Une fois l’un de trois acteurs déclare un obstacle, il peut à tout moment le supprimer comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer un obstacle |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | L’utilisateur est le propriétaire de la déclaration de l’obstacle. |
| **Post condition** | L’obstacle sélectionné est supprimé. |
| **Scénario de base** | L’utilisateur clique sur ‘supprimer’.  Le système supprime l’obstacle sélectionné. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

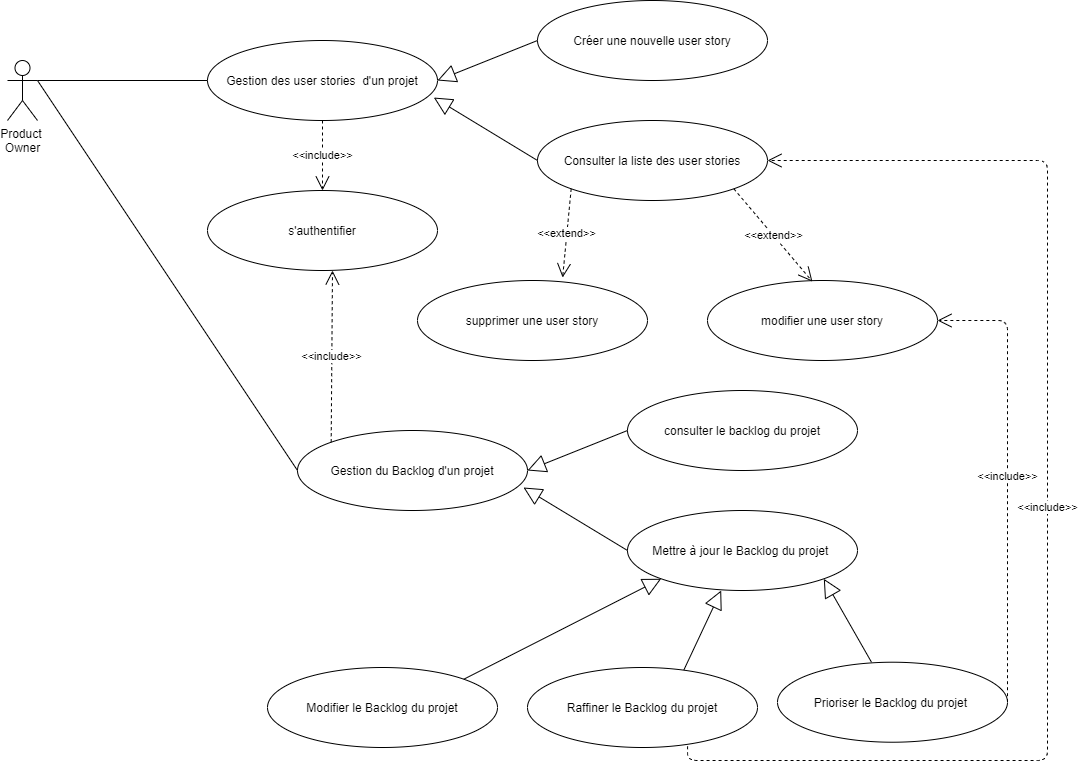
* Résoudre un obstacle

Le Product Owner, le Scrum Master et les développeurs peuvent à tout moment résoudre des obstacles déclarés par les autre membre d’équipe d’un projet comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Résoudre un obstacle |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Post condition** | L’obstacle sélectionné est résolu. |
| **Scénario de base** | L’utilisateur sélectionne l’obstacle à résoudre.  L’utilisateur clique sur ‘Résoudre’.  Le système enregistre la nouvelle statut de l’obstacle. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## 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 15:** Diagramme des cas d’utilisation Gestion des user stories d’un projet et gestion du backlog d’un projet

Dans ce qui suit, nous allons détailler tous les cas d’utilisation présentes dans le diagramme de cas d’utilisation ci-dessus.

* Créer une nouvelle user story

Après être authentifié, le Product Owner est capable de créer des nouvelles user stores comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Créer une nouvelle user story |
| **Acteur** | Product Owner |
| **Post condition** | Une nouvelle user story est créée . |
| **Scénario de base** | L’utilisateur saisie les données de la nouvelle user story.  L’utilisateur clique sur ‘créer ’.  Le système enregistre la nouvelle user story |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter la liste des user stories

Une fois les user stories sont créées, le Product Owner, le Scrum master et les développeurs peuvent consulter la liste des user stories du projet comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste des user stories |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | Le système réussit à sélectionner tous les user stories relatives à un projet. |
| **Post condition** | La liste des user stories est affichée |
| **Scénario de base** | L’utilisateur clique sur ‘User Stories’  Le système affiche la liste des user stories du projet classé selon leurs priorités . |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier une user story

Une fois authentifié, le Product Owner peut à tout moment le modifier comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier une user story |
| **Acteur** | Product Owner |
| **Post condition** | L’user story sélectionné est modifié. |
| **Scénario de base** | L’utilisateur saisie les nouvelles données de l’user story.  L’utilisateur clique sur ‘mettre à jour’.  Le système enregistre les nouvelles données de l’user story. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Supprimer une user story

Une fois une user story est créée, elle peut être supprimée comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer une user story |
| **Acteur** | Product Owner |
| **Post condition** | L’user story sélectionné est supprimé. |
| **Scénario de base** | L’utilisateur clique sur ‘supprimer’.  Le système supprime l’user story sélectionnée. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter le backlog du projet

Après être authentifié, le Product Owner, le Scrum Master et les développeurs peuvent consulter le backlog du projet comme le montre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter le backlog du projet |
| **Acteur** | Product Owner, Scrum Master ,développeur |
| **Précondition** | Le système réussit à sélectionner tous les user stories ainsi que leurs statuts . |
| **Post condition** | La liste des user stories et leurs statuts(en attente, en cours, en progression…)sont affichées . |
| **Scénario de base** | L’utilisateur clique sur ‘Backlog du projet’  Le système affiche la liste des user stories du projet classées selon leurs statuts. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Raffiner le backlog du projet

Une fois authentifié, le Product Owner peut raffiner le backlog du projet comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Raffiner le backlog du projet |
| **Acteur** | Product Owner |
| **Post condition** | La liste des user stories et leurs statuts(en attente, en cours, en progression…)sont mis à jour . |
| **Scénario de base** | Le système affiche la liste des user stories du projet.  L’utilisateur modifie les user stories ou il peut supprimer et modifier les user stories.  Le système enregistre les nouvelles user stories du projet |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Prioriser le backlog du projet

Une fois authentifié, le Product Owner peut prioriser le backlog du projet comme l’illustre le tableau suivant.

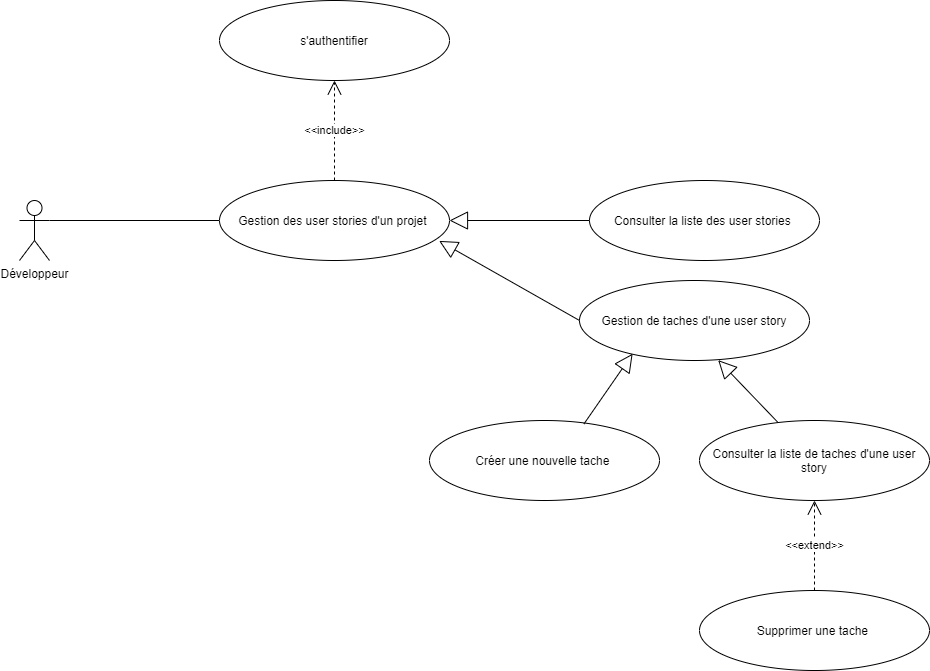
|  |  |
| --- | --- |
| **Cas d’utilisation** | Prioriser le backlog du projet |
| **Acteur** | Product Owner |
| **Post condition** | La liste des user stories et leurs priorités sont mis à jour . |
| **Scénario de base** | L’utilisateur modifie uniquement la priorité de chaque user story  Le système enregistre les nouvelles priorités des user stories du projet |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier le backlog du projet

Une fois authentifié, le Product Owner peut modifier le backlog du projet c’est-à-dire de mettre à jour les statuts des tâches comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier le backlog du projet |
| **Acteur** | Product Owner |
| **Post condition** | La liste des user stories et leurs priorités sont mis à jour . |
| **Scénario de base** | L’utilisateur glisse et dépose les user stories à modifier de la colonne indiquant leurs statuts originaux vers la colonne qui indique leurs nouveaux statuts.  A chaque fois ou l’utilisateur dépose une user story, le système doit mettre à jour le statut de l’user story concernée par la modification. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

L’acteur **développeur** aussi est capable de faire la gestion des user stories d’un projet comme l’illustre la figure suivante.



**Figure 15:** Diagramme de cas d’utilisation Gestion des user stories de l’acteur ‘développeur’

* Créer une nouvelle tâche

Après être authentifié le développeur est capable de créer une nouvelle tâche relative à une user story comme mentionne le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Créer une nouvelle tâche |
| **Acteur** | Développeur |
| **Post condition** | Une nouvelle tâche est créée |
| **Scénario de base** | L’utilisateur saisie les données de la nouvelle tâche.  L’utilisateur clique sur ‘Créer’  Le système créer une nouvelle tâche. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter la liste de tâches d’une user story

Après être authentifié, le développeur, le PO et le SM peuvent consulter la liste de tâches d’une user story comme mentionne le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de tâches d’une user story |
| **Acteur** | Product Owner, Scrum Master, développeurs. |
| **Post Condition** | La liste de tâches d’une user story est affichée. |
| **Scénario de base** | L’utilisateur clique sur ‘Consulter la liste de tâches’.  Le système affiche la liste de tâches d’une user story. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

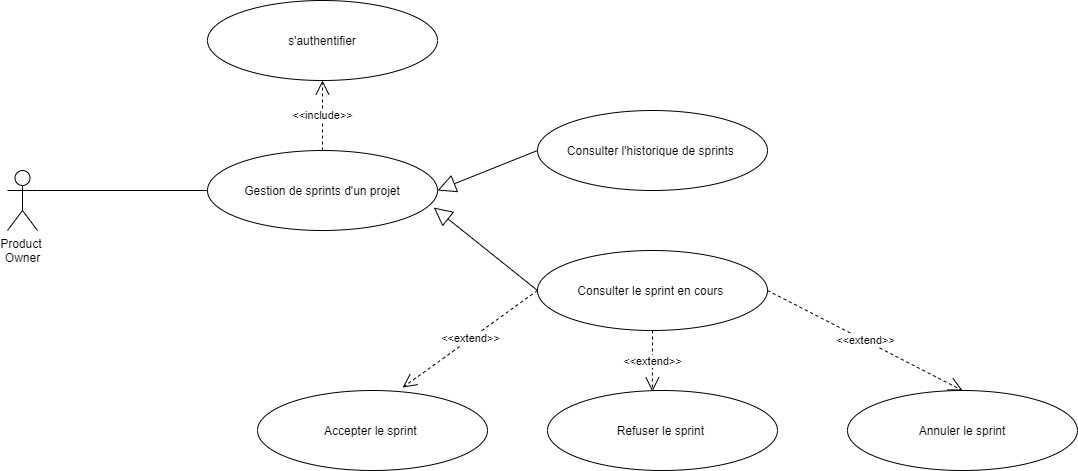
* Supprimer une tâche

Une fois une tache est créée, elle peut être supprimer par le développeur selon la description fournie dans le tableau suivant.

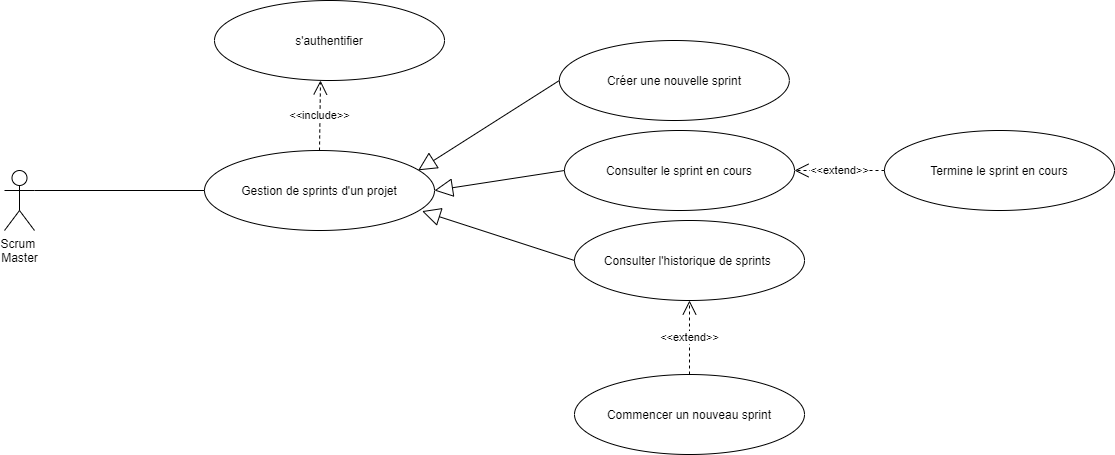
|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer une tâche |
| **Acteur** | développeurs. |
| **Post Condition** | La liste de tâches d’une user story est n’est pas vide . |
| **Scénario de base** | L’utilisateur clique sur ‘Supprimer cette tâche’.  Le système supprime la tache sélectionné. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « gestion de sprints d’un projet »

Les deux figures ci-dessous représentent les diagrammes de cas d’utilisation Gestion de sprints d’un projet relatifs successivement aux deux acteurs Product Owner et Scrum et Scrum Master.



**Figure 15:** Diagramme de cas d’utilisation Gestion de sprints d’un projet de l’acteur Product Owner



**Figure 16:** Diagramme de cas d’utilisation Gestion de sprints d’un projet de l’acteur Scrum Master

Dans ce qui suit nous allons détailler les cas d’utilisation présentes dans les diagrammes de cas d’utilisation présenté ci-dessus.

* Consulter l’historique de sprints

Une fois authentifié, le Product Owner, le Scrum Master et les développeurs peuvent consulter l’historique de sprints du projet courant comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter l’historique de sprints |
| **Acteur** | Product Owner, Scrum Master, développeurs. |
| **Post Condition** | La liste de sprints est affichée. |
| **Scénario de base** | L’utilisateur clique sur ‘Historique de sprints’.  Le système affiche la liste de sprint du projets ainsi que leurs statuts(terminé, en cours ,annulé ,refusé ,accepté) |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Consulter le sprint en cours

Une fois authentifié, le Product Owner, le Scrum Master et les développeurs peuvent consulter la description du sprint en cours comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter le sprint en cours |
| **Acteur** | Product Owner, Scrum Master, développeurs. |
| **Post Condition** | Le sprint en cours est affiché ainsi que ses chartes et ses statistiques. |
| **Scénario de base** | Le système affiche le sprint en cours ainsi que ses statistiques et ses charts(sprint burn down chart, sprint burn up chart). |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Accepter un sprint

Une fois authentifié, le Product Owner peut accepter le sprint en cours selon la démonstration présenté dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Accepter un sprint |
| **Acteur** | Product Owner |
| **Post condition** | Le sprint en cours est accepté. |
| **Scénario de base** | L’utilisateur clique sur ‘Accepter ce sprint’  Le système met à jour le statut du sprint en cours en ‘Accepté’ |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Refuser un sprint

Une fois authentifié, le Product Owner peut refuser le sprint en cours selon la démonstration présenté dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Refuser un sprint |
| **Acteur** | Product Owner |
| **Post condition** | Le sprint en cours est refusé. |
| **Scénario de base** | L’utilisateur clique sur ‘Refuser ce sprint’  Le système met à jour le statut du sprint en cours en ‘refusé’ |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Annuler un sprint

Une fois authentifié, le Product Owner peut annuler le sprint en cours selon la démonstration présenté dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Annuler un sprint |
| **Acteur** | Product Owner |
| **Post condition** | Le sprint en cours est annulé. |
| **Scénario de base** | L’utilisateur clique sur ‘Annulé ce sprint’  Le système met à jour le statut du sprint en cours en ‘Annulé’ |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Créer un nouveau sprint

Une fois authentifié, le Scrum Master est capable de créer un nouveau sprint comme mentionné dans le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Créer un nouveau sprint |
| **Acteur** | Scrum Master |
| **Post condition** | Un nouveau sprint est créer |
| **Scénario de base** | L’utilisateur saisie les données du nouveau sprint.  L’utilisateur clique sur ‘Créer’  Le système créer un nouveau sprint. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Commencer un nouveau sprint

Une fois authentifié, le Scrum Master peut commencer un nouveau sprint comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Commencer un nouveau sprint |
| **Acteur** | Scrum Master |
| **Précondition** | Il n’existe aucun autre sprint en cours. |
| **Post condition** | Un nouveau sprint est commencé.  Les développeurs concernés reçoivent une notification informative. |
| **Scénario de base** | L’utilisateur clique sur ‘Déclencher’  Le système met à jour le statut du sprint en ‘En cours’ |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Terminer un sprint

Une fois authentifié, le Scrum Master est capable de terminer le sprint en cours comme le mentionne le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Commencer un nouveau sprint |
| **Acteur** | Scrum Master |
| **Précondition** | Il existe un sprint en cours. |
| **Post condition** | Le sprint en cours est terminé. |
| **Scénario de base** | L’utilisateur clique sur ‘Terminer ce sprint’.  Le système met à jour le statut du sprint en ‘terminé |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « consulter la liste de ses tâches »

Le figure suivant représente le diagramme de cas d’utilisation Gestion de tâches.

Dans ce qui suit, nous allons détailler les cas d’utilisation présents dans le diagramme ci-dessus.

* Consulter la liste de ses tâches

Une fois authentifié, le développeur peut consulter la liste de ses tâches comme le montre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de ses tâches |
| **Acteur** | Développeur |
| **Post condition** | La liste de tâches affectées à l’utilisateur est affichée. |
| **Scénario de base** | L’utilisateur clique sur ‘mes tâches’.  Le système affiche les tâches affectées à l’utilisateur. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Prendre en charge une tâche

Une fois authentifié, le développeur peut prendre en charge une des tâches en attente comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Prendre en charge une tâche |
| **Acteur** | Développeur |
| **Post condition** | Le statut de la tâche sélectionnée devient ‘En cours ’ |
| **Scénario de base** | L’utilisateur sélectionne une tâche et clique sur ‘commencer’.  Le système met à jour le statut de la tâche sélectionnée. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Terminer une tâche

Une fois authentifié, le développeur peut terminer une des tâches en cours comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Terminer une tâche |
| **Acteur** | Développeur |
| **Post condition** | Le statut de la tâche sélectionnée devient ‘Terminé ’ |
| **Scénario de base** | L’utilisateur sélectionne une tâche et clique sur ‘Terminé’.  Le système met à jour le statut de la tâche sélectionnée. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

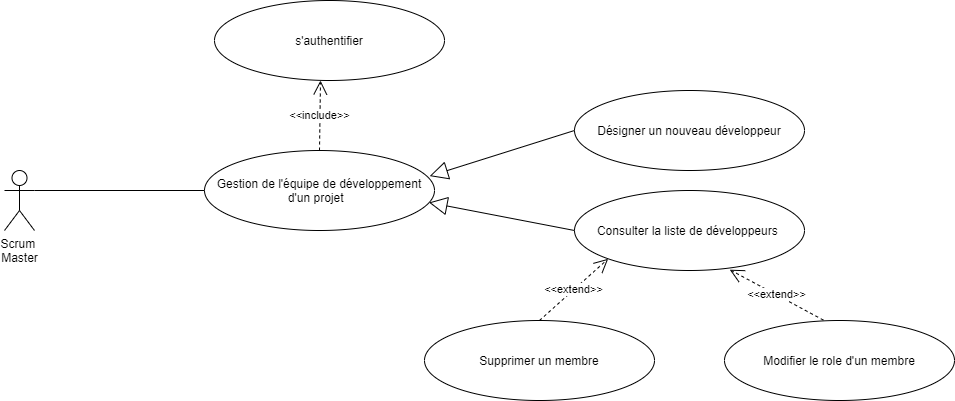
* Abandonner une tâche

Une fois authentifié, le développeur peut abandonner une de ses tâches comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Abandonner une tâche |
| **Acteur** | Développeur |
| **Post condition** | Le statut de la tâche sélectionnée devient ‘Abandonné’ |
| **Scénario de base** | L’utilisateur sélectionne une tâche et clique sur ‘Abandonné’.  Le système met à jour le statut de la tâche sélectionnée. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Cas d’utilisation « gestion de l’équipe de développement d’un projet »

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



**Figure 16:** Diagramme de cas d’utilisation Gestion de l’équipe de développement d’un projet

Dans ce qui suit, nous allons détailler les cas d’utilisation présents dans le diagramme ci-dessus.

* Designer un nouveau développeur

Après être authentifié, les Product Owner et le Scrum Master peuvent désigner un nouveau développeur pour un projet comme l’illustre le tableau ci-dessous.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Designer un nouveau développeur |
| **Acteur** | Product Owner, Scrum Master |
| **Précondition** | Le système réussit à sélectionner tous les personnels présents dans la base de données.  Le développeur sélectionné n’est pas présent dans la liste des développeur . |
| **Post condition** | Un nouveau développeur est désigné pour le projet sélectionné |
| **Scénario de base** | L’utilisateur sélectionne un développeur parmi la liste des utilisateurs.  Le système enregistre l’utilisateur sélectionné comme un nouveau développeur du projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |
| **Extensions** | Le Product Owner et le Scrum Master peuvent faire une recherche par nom et prénom pour trouver le profil cherché.  **Enchainement :**   1. Le PO ou le SM saisie le nom et le prénom du profil cherché.   Le PO ou le SM clique sur ‘Rechercher’.  Le système affiche tous les utilisateurs dont le nom et le prénom incluent l’expression saisie. |

* Consulter la liste de développeurs

Après être authentifié, le Product Owner et le Scrum Master peuvent consulter la liste de développeurs d’un projet comme l’illustre le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Consulter la liste de développeurs d’un projet |
| **Acteur** | Product Owner, Scrum Master |
| **Précondition** | Le système réussit à trouver la liste de développeurs d’un projet. |
| **Post condition** | La liste de développeurs est affiché |
| **Scénario de base** | L’utilisateur clique sur ‘Développeurs’.  Le système affiche la liste de développeurs d’un projet. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Modifier le rôle d’un développeur

Après la consultation de la liste de développeurs d’un projet, le Product Owner et le Scrum master peuvent modifier le rôle d’un développeur comme l’explique le tableau suivant.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Modifier le rôle d’un développeur |
| **Acteur** | Product Owner ,Scrum Master |
| **Post condition** | Le rôle du développeur est modifié. |
| **Scénario de base** | L’utilisateur saisie le nouveau rôle du développeur.  L’utilisateur clique sur ‘Mettre à jour’.  Le système modifie le rôle du développeur. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

* Supprimer un développeur

Autre que la modification du rôle d’un développeur, le Product Owner et le Scrum Master sont capables de supprimer un développeur de la liste de développeurs d’un projet.

|  |  |
| --- | --- |
| **Cas d’utilisation** | Supprimer un développeur |
| **Acteur** | Product Owner, Scrum Master |
| **Post condition** | Le développeur sélectionné est supprimé |
| **Scénario de base** | L’utilisateur sélectionne un développeur et clique sur ‘Supprimer’.  Le système supprime le développeur sélectionné. |
| **Exception** | En cas d’erreur, le système doit afficher une page d’erreur . |

## Conclusion

Dans le présent chapitre, nous avons essayé de mettre l’accent tout d’abord sur les différents acteurs qui intervient 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.

Chapitre 4 : Conception

Introduction

Après avoir dégager les besoins de notre système, nous entamons, dans ce chapitre, la phase de conception. Ainsi, nous nous intéressons à la conception d’une structuration adéquate pour notre application qui va faciliter à la suite la phase de l’implémentation réel de l’application. Pour cela, nous commençons le présent chapitre par la conception globale de notre projet. Ensuite nous détaillons la conception à travers les diagrammes UML appropriées avant de finir avec une présentation de la conception logique de données.

1. Architecture globale

Notre projet consiste à développer une application web qui offre des services qui sont accessible uniquement à travers le moyen de l’internet. Il s’agit donc d’une application à trois couche qui sont :

* Un client

C’est la machine de ressources équipé d’une interface Homme-Machine qui permet l’accès aux différents services de l’application qui est (l’interface Homme-Machine) généralement un navigateur web.

* Un service

C’est la couche chargé de fournir les ressources nécessaires en faisant l’appel à d’autres serveurs.

* Les données

Ce sont les données gérées par les utilisateurs et stockées sous forme de documents dans une base de données NOSQL.

1. Conception détaillée

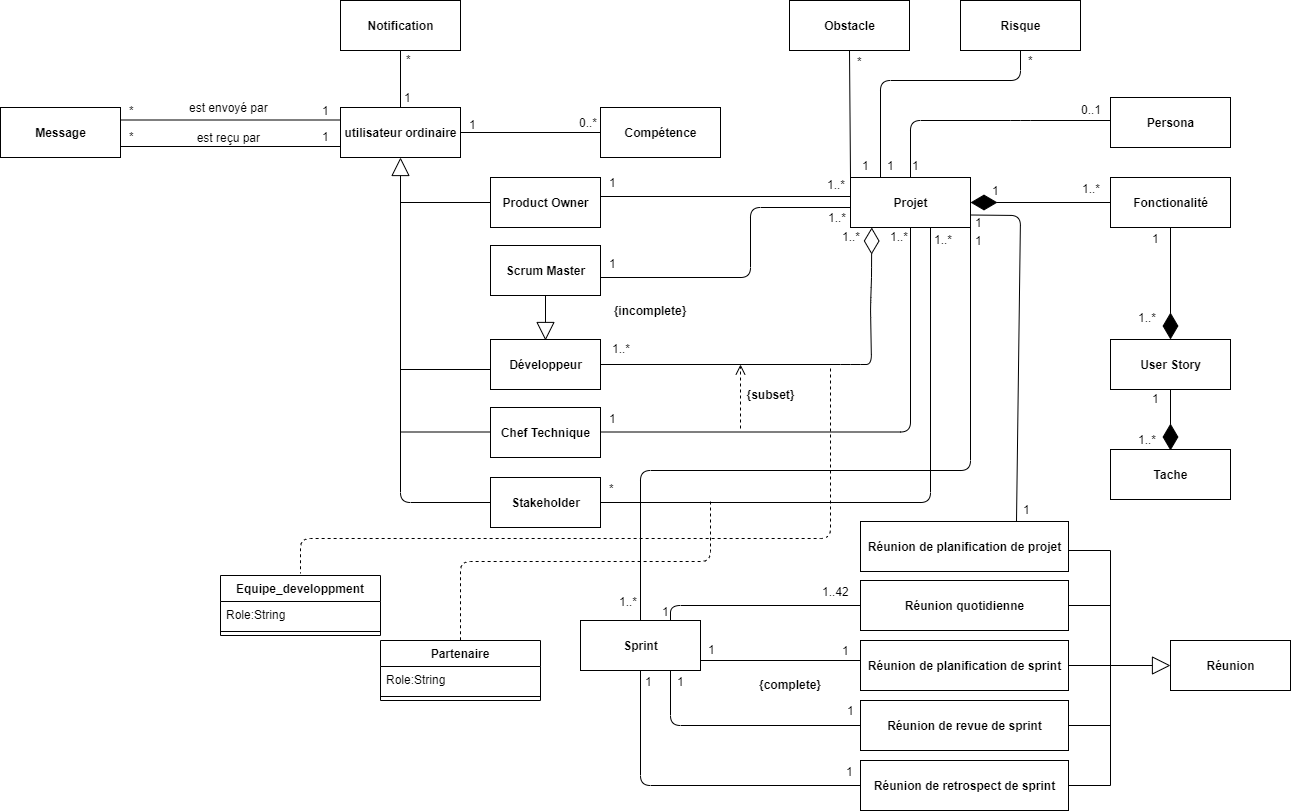
A ce stade, nous allons entamer la partie conception au cours de laquelle nous faisons la présentation de diagramme de classes et des diagrammes de séquences.

## Diagramme de classes

* + 1. Présentation de diagramme de classes

Le diagramme de classes, qui fait partie de la partie statique d’UML, est un schéma utilisé dans le domaine de génie logiciel pour visualiser les classes, les objets, les interfaces qui interagissent dans le système et qui permet aussi de viser les relations entre les différents composants présents dans le système.

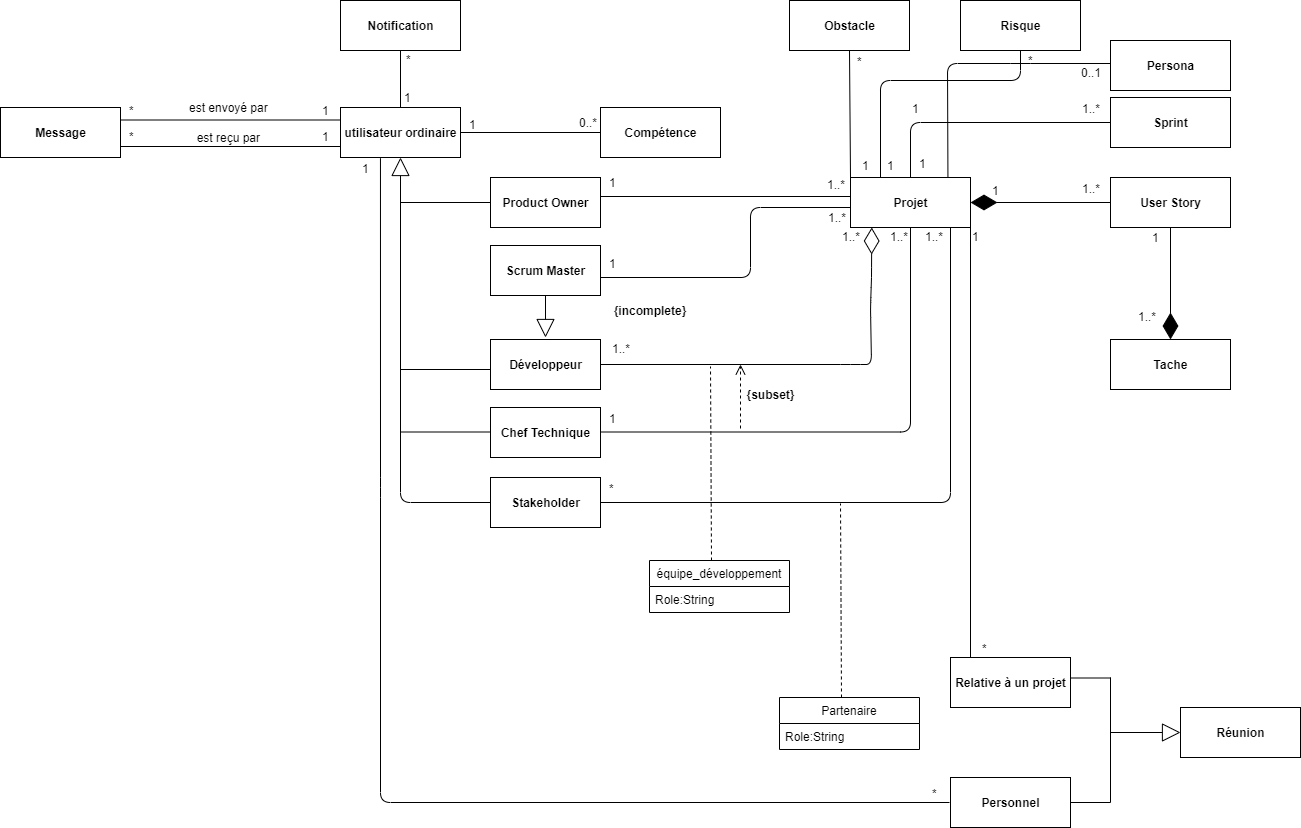
Dans notre cas, Scrum offre une très grande diversité d’acteurs et de composants de telle façon qu’il devient très compliqué d’implémenter tous les détails de Scrum dans une période très limitée dans le temps et rend très difficile aux utilisateurs d’utiliser l’application surtout pour ceux qui ont une simple connaissance d'Agile Scrum. Pour cela nous allons présenter deux diagrammes de classes le premier représente le diagramme général d’une application de gestion et de suivi de projets Agile Scrum et le deuxième illustre le diagramme de classes de l’application que nous avons implémenté au cours de ce projet.



**Figure 17:** Diagramme de classes général

La figure **17** représente le diagramme de classes général d’une application de gestion et de suivi de projets Agile Scrum. Pour simplifier l’utilisation de notre application, et en se référant au guide officiel de Scrum SBOK [], nous

Avons effectué quelques modifications et qui garde la même qualité de projet et qui simplifie l’utilisation de notre application qui sont remarquables dans la figure **18**.



**Figure 18:** Diagramme de classes implémenté

* + 1. Descriptions des classes

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 relative à un et un seul projet.**

**R11 : un projet peut avoir des risques et chaque risque est relative 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 relative 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

La description textuelle des cas d'utilisation présentés précédemment rend la communication simple et précise avec l'utilisateur. Par contre, cette description textuelle a certaines restrictions car il devient difficile d'afficher l'enchaînement et la chronologie des séquences. Nous essayons donc d’étendre la description textuelle de nos cas d’utilisation par quelques diagrammes de séquences afin de souligner l’ordre chronologique des actions dans chaque cas d’utilisation.

1. Conception logique de données

## Conclusion

Dans le présent chapitre, nous faisons la conception de notre système ou nous avons commencé par l’élaboration de diagrammes de classes ensuite la présentation de quelques diagrammes de séquences et enfin la conception logique de données. Dans le chapitre qui suit nous allons présenter des différentes étapes de construction de notre application.

Chapitre 5 : Réalisation

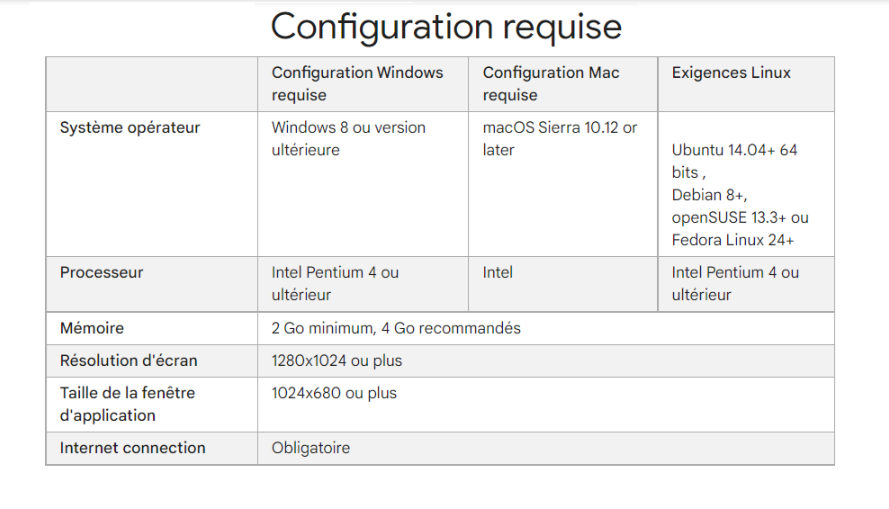
## Introduction

Dans les chapitres précédents, nous avons essayés d’établir un enchainement logique qui nous guide pour développer notre application. Dans le présent chapitre nous entamons la phase de développement qui représente la phase d’achèvement et d’aboutissement du projet. Pour cela nous présentons tout d’abord l’environnement de travail, la configuration logiciel utilisé, les outils utilisés et l’exposition de choix technologique. Ensuite, nous proposons quelques imprimes écran de produit finale et nous finissons par la phase des tests fonctionnels de notre système.

1. Environnement de travail

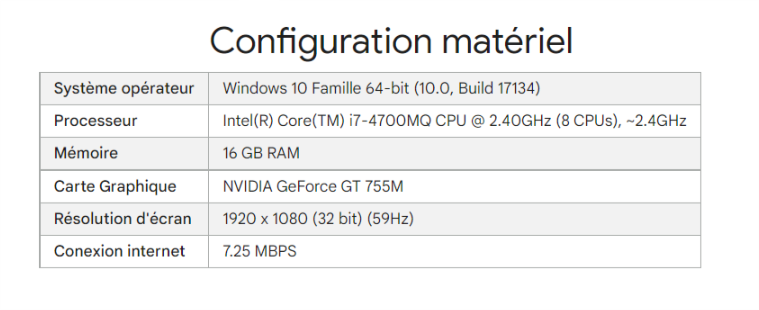
## Environnement matériel

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 déployé sur l’architecture serveur présentée par le tableau ci-dessous.



**Tableau 2 :** La configuration minimale requise

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



**Tableau 3 :** La configuration matériel utilisée 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 utilisées pour le développement de notre application sont :

* Draw.io

Diagrams.net ou draw.io est un logiciel de dessin graphique multiplateforme, gratuit et open source développé à l'aide de HTML5 et JavaScript. Son interface peut être utilisée pour créer des diagrammes tels que des organigrammes, des structures filaires, des diagrammes UML, des organigrammes et des diagrammes de réseau.



**Figure 12 :** le logo de draw.io

* Visual Studio Code

Visual Studio Code est un éditeur de code simplifié qui prend en charge les opérations de développement telles que le débogage, l'exécution de tâches et le contrôle de version. Il est conçu pour donner aux développeurs les outils dont ils ont besoin pour des cycles de création et de débogage de code rapides, et laisser les flux de travail plus complexes à des IDE plus complets, tels que l'IDE Visual Studio.



**Figure 13 :** le logo de Visual studio code

* Robo 3t

Robo 3T, anciennement connu sous le nom de RoboMongo, 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 via des indicateurs visuels plutôt qu'une interface textuelle. C'est gratuit et léger.



**Figure 14 :** le logo de Robo 3T

* Adobe Photoshop

Adobe Photoshop est un logiciel largement utilisé pour l'édition d'images, la conception graphique et l'art numérique. Il utilise la superposition pour donner de la profondeur et de la flexibilité au processus de conception et d'édition, et fournit de puissants outils d'édition qui, lorsqu'ils sont combinés, peuvent tout faire.



**Figure 16 :** le logo d’adobe Photoshop

* Adobe illustrator

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 Illustrator permet aux utilisateurs de travailler sur des postes de travail et des terminaux mobiles et de créer rapidement de superbes créations pouvant être utilisées partout. Impression, web et applications, vidéo et animations, etc.



**Figure 17 :** le logo d’adobe Illustrator

* Canva.com

Canva est une plate-forme de conception graphique australienne, utilisée pour créer des graphiques pour les médias sociaux, des présentations, des documents, des affiches et d`autres contenus visuels.



**Figure 18 :** le logo de Canva.com

* Adobe XD

Adobe XD (également réputé sur le patronyme d'Adobe Expérience Design) est un outil pour l'élaboration d'expérience usager vectorielle à les applications web et mobiles évolué et publié dans Adobe Inc. Il est ouvert à MacOs et Windows et il existe des versions mobiles pour iOS et Android. Adobe XD permet de réaliser des prototypes pour les sites web filaires et pour concevoir des prototypes cliquables.



**Figure 19 :** le logo d’Adobe XD

* + 1. Choix technologique

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

* HTML5

HyperText Markup Language qui est un langage de balisage standard pour la création des page web .HTML est utilisé au sein de notre projet pour la définition de la structure d’une page web.



**Figure 20 :** le logo de HTML5

* CSS3

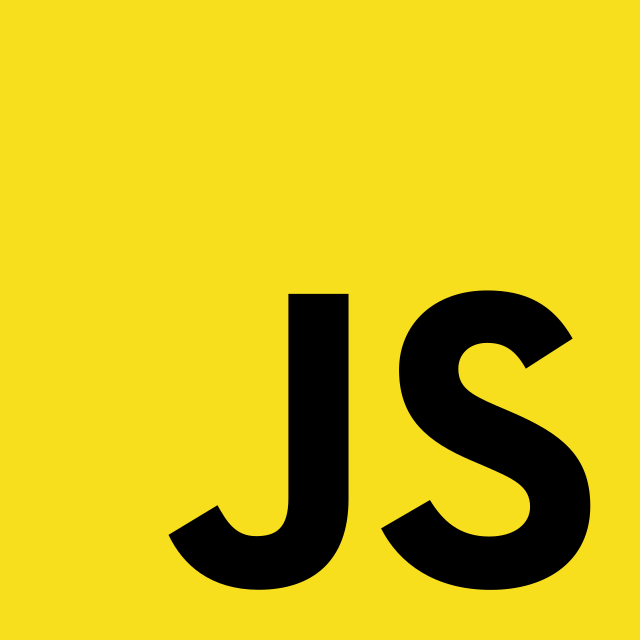
CSS est l’abréviation de Cascade Style Sheets. C’est un langage qui permet de définir comment les éléments HTML doivent être imprimé sur l’écran, sur des support papier ou d’Autre supports.CSS permet d’éditer la mise en page des pages web.



**Figure 21 :** le logo de CSS3

* JavaScript

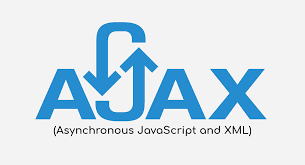
JavaScript est un langage de programmation et plus précisément un langage de Scripting oriente objet. JavaScript est le langage le plus utilisé dans le développement des application web. JavaScript se distingue des autres langages de programmation web par sa capacité de rendre le navigateur lui-même capable d’exécuter les requêtes et les tâches.



**Figure 22 :** le logo de JavaScript

* AJAX

AJAX est l’abréviation de Asynchronous JavaScript And XML. C’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 ainsi que la communication avec le server à travers un code source JavaScript.



**Figure 23:** le logo d’AJAX

* JSON

JavaScript Object Notation ou JSON, fait référence à un langage léger qui garantit l’échange de données textuelles entre les côtés client et serveur. L’avantage le plus fort pour l’utilisation de JSON est qu’il est pratique a l’écriture et à la lecture grâce à une syntaxe simple et a une structure en arborescence. JSON est le langage le plus utilisé pour la représentation des données structurées.



**Figure 24 :** le logo de JSON

* JWT

Json Web Token ou JWT est un standard ouvert qui représente un moyen compact et autonome qui garantit la transmission en toute sécurité des informations entres plusieurs parties sous format d’un objet JSON. Les jetons JWT peuvent être signées par des nombreux algorithmes de cryptage tel que l’algorithme HMAC, RSA, ECDSA …

JWT est utilisé dans la plupart des cas en tant qu’un mécanisme de sécurité qui permet de faire l’authentification d’un utilisateur au sein d’une application.



**Figure 25 :** le logo de JWT

* NodeJS

NodeJS est la première plateforme logicielle basée JavaScript orientée pour les applications utilisant le réseau pour la communication. Dans les dernières années, NodeJS est devenue célèbre en tant que langage de développement web orienté Back End grâce à son moteur de traitement V8 Engine. NodeJS est utilisé généralement dans les applications RTA (Real Time Applications) et les applications SPA (single Page Applications).



**Figure 27 :** le logo de nodeJs

* Bootstrap

Bootstrap est un Framework open source lancé par Twitter utilisant les langages HTML, CSS et JavaScript pour fournir aux développeurs un outil de design permettant l’optimisation de code source, d’effort mais aussi de fournir un design responsif qui s’adapte à tout type d’écran. Bootstrap fournie un large choix de styles déjà mis en place pour des boutons, des interfaces, des formulaires et des conteneurs. Ce type de Framework est appelée ‘Front End Framework ’



**Figure 28 :** le logo de Bootstrap

* MongoDB

MongoDB fait référence à un système de gestion des bases de données NOSQL orienté document où les données sont stockées dans des documents ayant le format de clé/valeur (contrairement au base de données SQL ou les documents sont stockes dans des tables). MongoDB permet le stockage de quantité massifs de données grâce à son système de fichier qui permet de faire la répartition des données sur un nombre quelconque de machines.



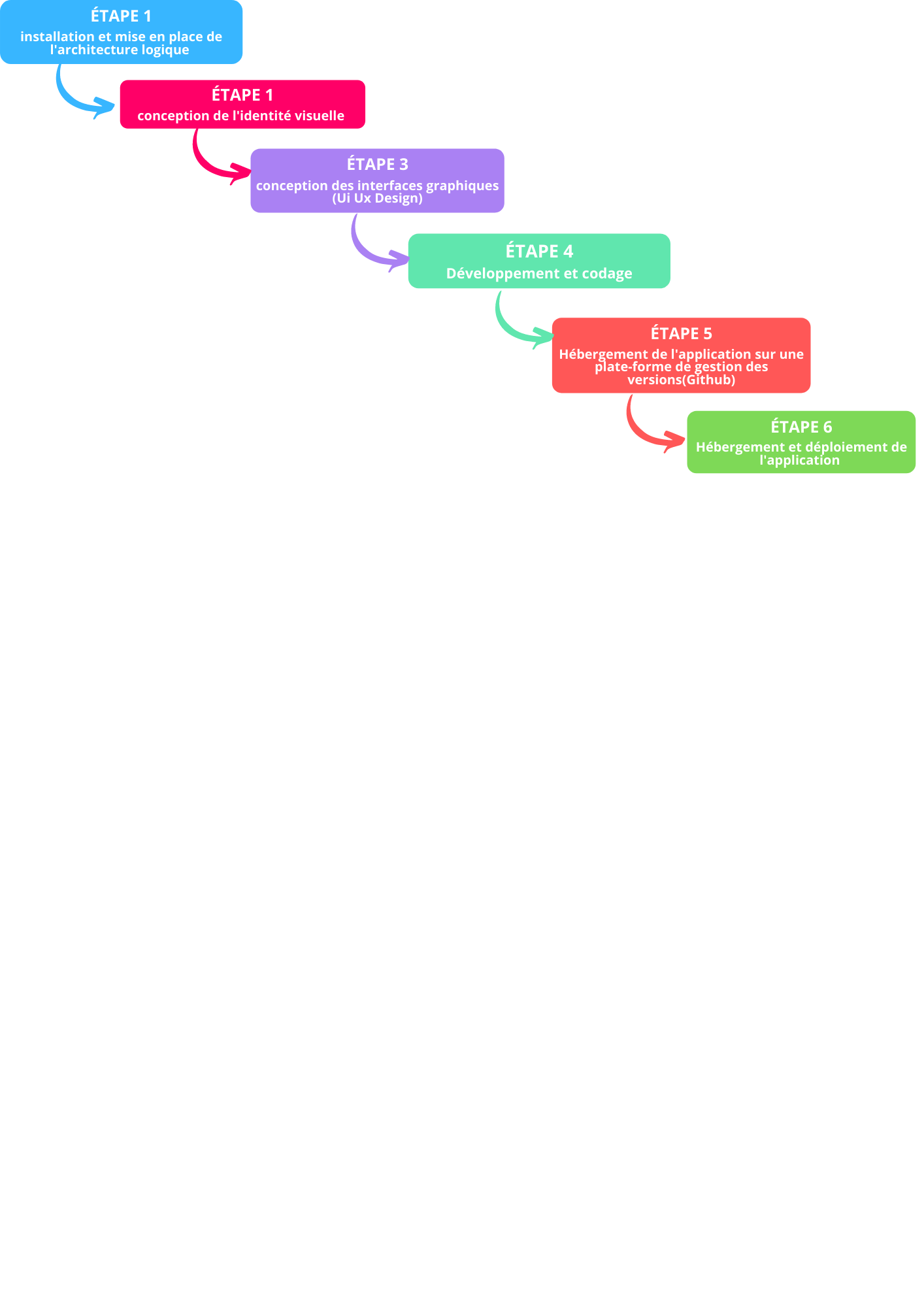
**Figure 29 :** le logo de MongoDB

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 présente ci-dessous visualise le cycle de vie de notre projet.





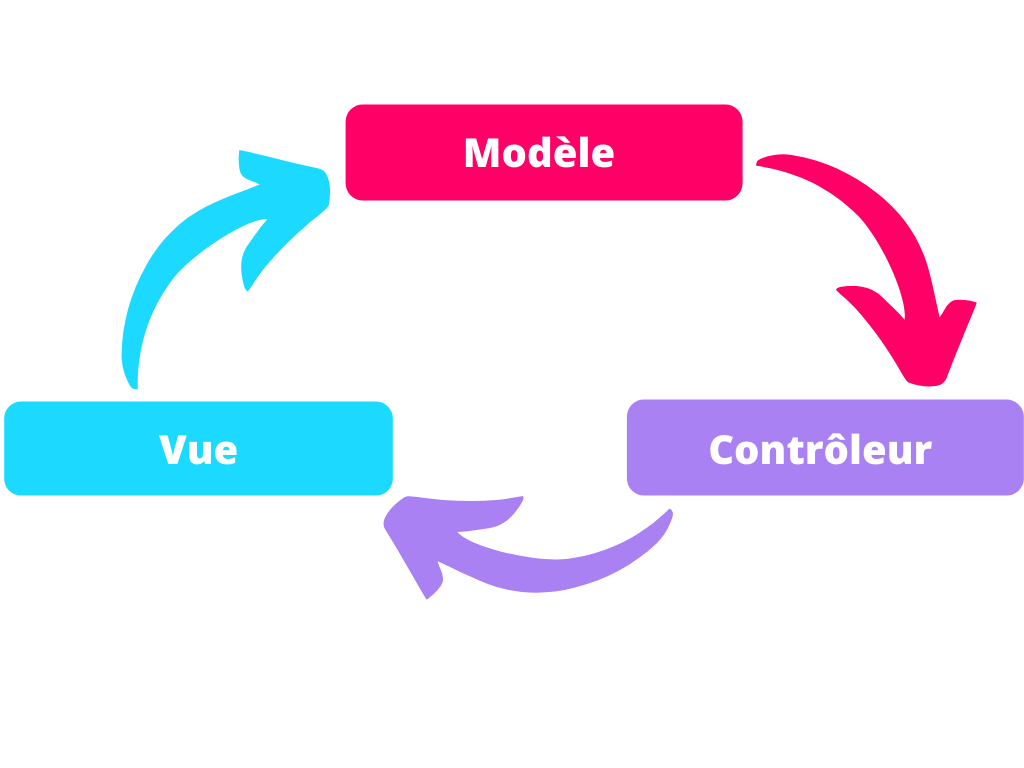
**Figure 30 :** le schéma des étapes suivis pour la réalisation de projet

* + 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. 1. 1. L’architecture MVC

Le MVC est l’abréviation Model-View-Controller. C'est un modèle architectural qui fait la séparation d'une application en trois composants logiques principales qui sont le modèle, la vue et le contrôleur. Chacun de ces composants est construit pour le but de gérer les aspects de développement relatif à une application.



**Figure 31 :** l’architecture de modèle MVC

* La couche Vue

Une vue est un moyen d’afficher des objets dans une application.

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. Par exemple un objet client récupéra les informations de la source de données, les manipuler et mettre à jour ses données dans la base de données.

* 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 à l'aide du composant modèle et interagir avec les vues pour rendre le résultat final.

2. 1. 1. 2. La mise en place de l’architecture MVC

L’arborescence des fichiers et de dossiers après la mise en place de l’architecture MVC sera comme la montre la figure ci-dessous.



**Figure 32 :** l’arborescence de fichiers de notre application avec MVC

* Le répertoire assets

Ce répertoire contient les fichiers statiques de notre projet tel que les fichiers de style CSS, les scripts standard JavaScript et les images.

* Le répertoire node\_modules

Ce répertoire contient les modules utilisés par nodeJs tels que les bibliothèques externes.

* Le répertoire server

Ce répertoire contient cinq sous-répertoires qui sont :

* + Le répertoire database

Ce répertoire contient les scriptes responsables à la connexion avec

la base de données.

* + Le répertoire Controller

Ce répertoire contient la logique de l’application tel que les requêtes d’écriture, de mise à jour et de suppression de données de la base de données.

* + Le répertoire model

Ce répertoire contient les structures générales des enregistrements stockées dans la base de données.

* + Le répertoire routes

Ce répertoire contient les scriptes de directions et les Apis de notre application.

* Le répertoire services

Ce répertoire contient les scripts standard de l’application tel que les opérations de lecture de données de la base de données.

* Le répertoire Views

Ce répertoire contient l’implémentation des interfaces graphiques de l’application.

* Le fichier config.env

Ce fichier est utilisé pour stocker des variables d’environnement tels que le cas avec les clés des apis, le schéma vers la base de données, la zone horaire etc. …

* Le fichier package\_lock.json

Ce fichier servira pour plusieurs objectifs. Dans notre cas, il est utilisé pour faire la description de l’arborescence exacte des dépendances.

* Le fichier package.json

Le fichier package. Json est utilisé pour stocker des informations relatives au projet tel que le nom de projet, la description, la version, les packages installés et les mots clés.

* Le fichier server.js

Ce le fichier qui sera lancé le premier lors de l’exécution de l’application. En effet, c’est le fichier qui permet de faire appel aux différents composants de l’application pour assurer la cohérence entre les différentes parties de projet.

4. 1. 2. Conception de l’identité visuelle

L'identité visuelle d'une marque représente sa propre carte d'identité, qui permet de la reconnaître, de temps en temps 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…

Pendant cette section, nous nous intéressons par représenter l’image de marque de notre projet, la conception de logo ainsi que le choix des couleurs de base.

* + - 1. Choix de la couleur primaire

Chaque couleur possède son propre code psychologique. Au titre d'exemple, le jaune fait référence à la jeunesse, la tromperie et la prudence. Dans notre cas, nous voulons choisir une couleur qui se réfère à la productivité, le luxe, le mystère et à la collaboration.

La couleur qui satisfait ces critères est le violet qui, grâce à une combinaison magique de la chaleur du rouge et la froideur du bleu et à ses nombreuses dégrées, rend son usage dans un design assez complexe, souple et profond.

La figure présente ci-dessous représente une palette de couleur crée en utilisant les différentes dégrées de violet.



**Figure 33 :** Palette de couleur de différentes dégrées de violet

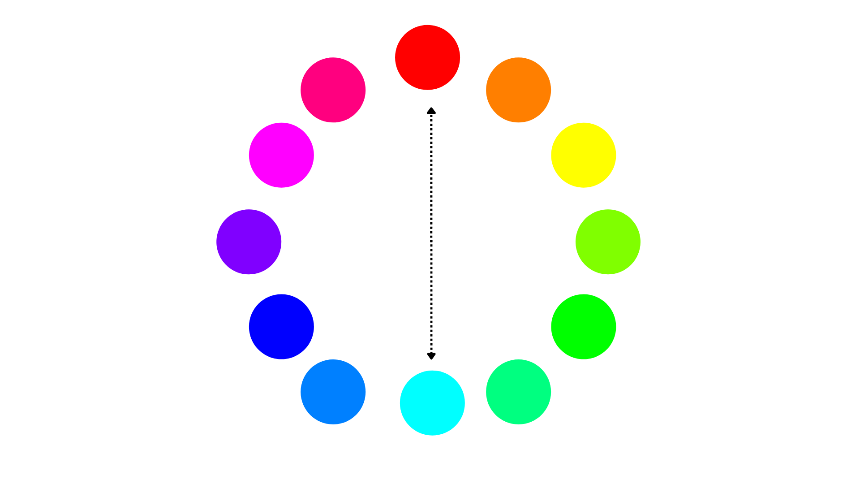
* + - 1. Choix des couleurs secondaires

La couleur étant l'élément le plus important dans la phase de design de l'identité visuelle d'un projet. L'étude de couleur est un processus qui est à la fois artistique et scientifique. Pour cela, nous faisons usage des différentes théories de couleurs afin de générer des combinaisons de couleur notre design.

Les théories des couleurs utilisées sont :

* Théorie des couleurs complémentaires

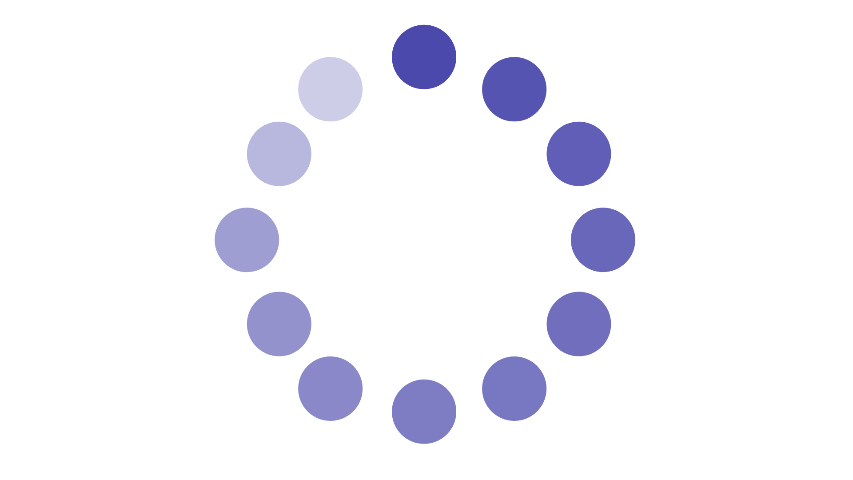
Cette théorie fait une correspondance entre un couleur et la couleur qui se trouve dans le côté opposé de la roue chromatique comme la montre la figure ci-dessous.



**Figure 34 :** Théorie des couleurs complémentaires

* Théorie des couleurs monochromatique

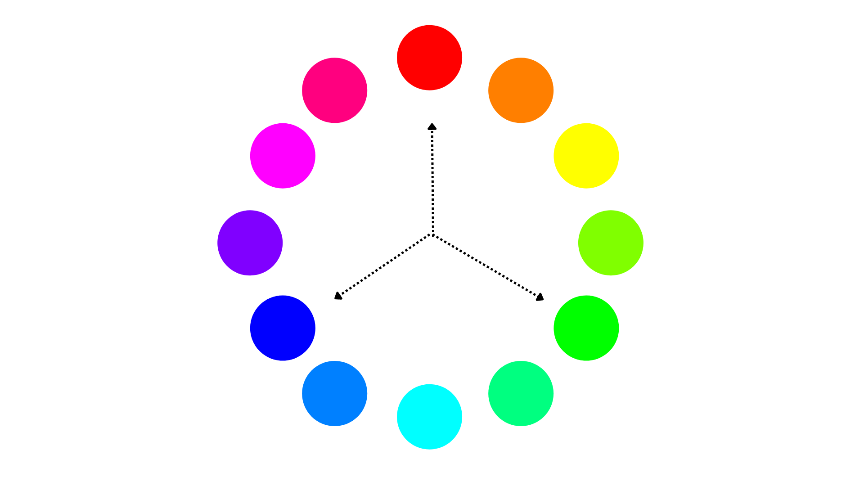
Cette théorie utilise les teintes, les tons et les nuances pour générer une combinaison des couleurs qui sont commun de la couleur de base. En ajoutant soit le blanc, soit le noir soit les deux ensembles pour former le gris, une palette de nuances de violet est générée comme la montre la figure présente ci-dessous.



**Figure 35 :** Théorie des couleurs monochromatiques

* Théorie des couleurs triadiques

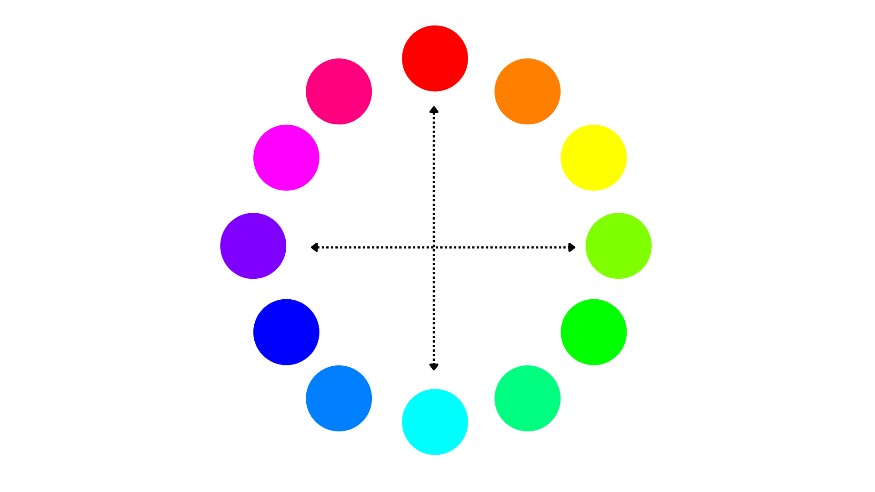
Cette théorie généré une palette des couleurs en relevant trois couleurs qui dont régulièrement espacées sur la roue chromatique comme la montre la figure présente ci-dessous.



**Figure 36 :** Théorie des couleurs triadiques

* Théorie des couleurs tétradiques

Cette théorie généré une palette compose de quatre couleurs régulièrement espacées sur la roue chromatique comme l’exemple présenté dans la figure ci-dessous.



**Figure 37 :** Théorie des couleurs tétradiques

Après l'utilisation de ces différentes théories de couleurs, notre palette de couleurs composé de dix couleurs est présentée dans la figure ci-dessous.



**Figure 38 :** Palette finale de couleurs

* + - 1. Design de l’identité visuelle

Comme nous avons mentionné ci-dessus, l’identité visuelle d’un produit ou d’une entreprise est un actif incorporel qui permet au public d’identifier et de mémoriser un produit.

La figure ci-dessous représente l’identité visuelle de notre produit.



**Figure 39 :** Identité visuelle de projet

4. 1. 2. 3. 1. Signification de logo

La forme de logo, qui est sous la forme d’une ligne droite horizontale, donne un impact de stabilité, de tranquillité, l’infini et el calme.

Le symbole de l’empreinte donne l’impression que notre produit aura un impact positif et une empreinte unique sur les projets.

* + - * 1. Signification de nom de marque Collaboro

Le nom de marque Collaboro est une combinaison de deux noms collaboration et labeur, qui signifie travail ou travailleur. Ce nom de marque est simple et mémorable qui donne impression qu'il s'agit d'un outil qui va booster la créativité et la productivité des équipes de développement.

* + 1. Conception des interfaces graphiques

Cette phase de conception des interfaces graphiques, Ui Ux Design en anglais, est consacré à faire le design graphique des interfaces utilisateur de l’application ainsi que le design de l’expérience utilisateur.

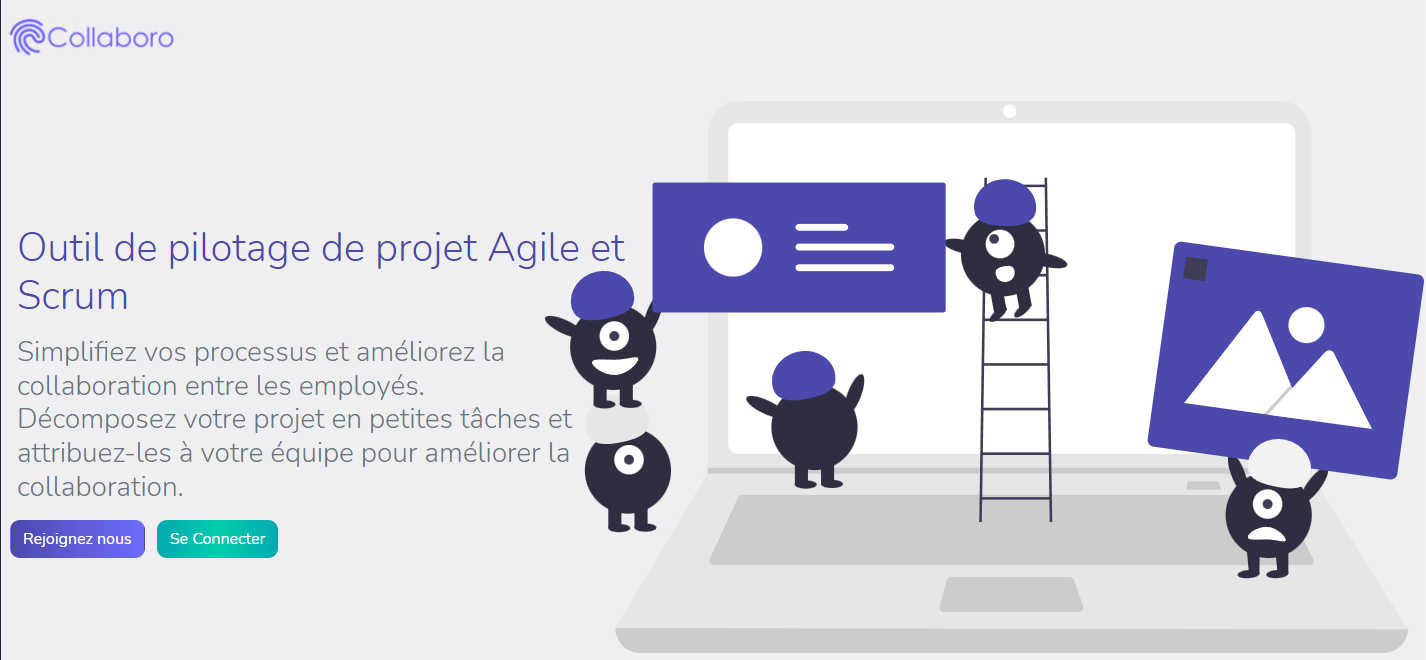
Le design de l’expérience utilisateur a un effet direct sur le design des interfaces utilisateurs car le rôle des interfaces graphique est de fournir un produit confortable à l’utilisation.

Au cours de cette phase, nous intéressons aussi au choix de typographies, les icones, les couleurs, emplacements des composants, disposition des boutons…

Pour cela, nous faisons usage d’Adobe Experience Design pour faire le design des interfaces graphiques et de l’expérience utilisateur pour avoir des maquettes qui facilite la phase de développement de produit (codage). Les deux figures ci-dessous représentent strictement l’une de nos maquettes conçus et la figure qui le suit est l’implémentation réelle de cette maquette.



**Figure 40 :** maquette de la page d’accueil



**Figure 41 :** l’implémentation réelle de la maquette

* + 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.

Cette phase s'étend sur deux parties fondamentales : le développement de la partie BackEnd et le développement de la partie FrontEnd.

Le développement de la partie FrontEnd consiste à concevoir les éléments visuels de côté client alors que le développement de la partie BackEnd consiste à rédiger les codes pour que chaque élément exécute les fonctions nécessaires.

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

* + 1. Définition

Les principes SOLID sont des principes de conception qui nous permettent de traiter la plupart des problèmes de conception de logiciels.

Le terme SOLID est un ensemble de cinq principes de conception visant à rendre les logiciels plus faciles à comprendre, plus flexibles et plus faciles à entretenir.

Les principes SOLID sont introduits pour la première fois par Michael Feathers.

* + 1. Les principes SOLID

Le nom SOLID est l’acronyme de cinq principes qui sont :

* Principe de responsabilité unique

Ce principe exige que chaque module ou classe doit avoir la responsabilité d’une partie unique de la fonctionnalité fournie par le logiciel et cette responsabilité doit être entièrement encapsulée par cette classe.

* Principe d'ouverture et de fermeture

Les entités logicielles doivent être ouvertes pour l’extension mais fermées pour la modification. L'écriture de code devrait consister à ajouter de nouvelles fonctionnalités avec des changements minimes au code existant.

* Principe substitution de Liskov

Ce principe impose que les objets d'un programme soient remplaçables par des instances de leurs sous-types sans altérer le fonctionnement correct du programme.

* Principe de ségrégation des interfaces

De nombreuses interfaces spécifiques aux clients sont préférables à une interface générale et nous ne devrions pas non plus obliger les développeurs à implémenter des interfaces qu'ils n'utilisent pas. Au lieu de créer une interface globale, nous pouvons la décomposer en interfaces plus petites et plus spécifiques.

* Principe d'inversion de la dépendance

Les abstractions ne doivent pas dépendre des détails, alors que les détails doivent dépendre des abstractions et que les modules de haut niveau ne doivent pas dépendre des modules de bas niveau.

* + 1. Hébergement de l’application sur une plateforme de gestion des versions

La méthodologie SCRUM est un cadre de travail assez souple de telle façon que chaque équipe qui fera usage de ce framework à la possibilité d'adapter les principes de ce framework à leurs besoins.

Pour cette raison et pour d'autres raisons telles que l'amélioration continue de notre produit en se basant sur les retours des utilisateurs réels, nous décidons que notre projet subventionne la culture d'Open-Source.

Un projet Open Source est un projet dont le code source est conçu pour être accessible au public ou n'importe quel individu qui peut le consulter, le modifier de la façon qui s'adapte le mieux à ses besoins et de distribuer la version modifiée au public.

La distribution de code source d'un projet au public ne signifie pas que les droits des créateurs ne sont pas protégés. La culture d'Open Source permet aux individus d'accéder et de modifier le code source d'un projet sous contrainte de respecter les licences attribuées à ce projet.

Pour cette raison mentionnée ci-dessus, nous choisirons d'accréditer la licence CC-BY-NC à notre projet.

* Creative Commons CC

Creative Commons est une organisation internationale à but non lucratif qui offre des licences gratuites aux créateurs à utiliser lors de la mise à disposition de leurs travails au public. Ces licences permettent les individus à faire usage de l’œuvre sous certaines conditions.

* Attribution BY

Cette contrainte exige que la personne qui va modifier l’œuvre doit mentionner le créateur de l’œuvre et d’indiquer aussi si des modifications ont été apporté à l’œuvre.

* Non Commercial NC

Toutes les exploitations de l’œuvre ou des œuvres dérivées sont possibles sauf les exploitations à des fins commerciales.

Après l’enregistrement de la licence, nous devons par la suite publier notre œuvre dans un espace ou les intervenants (les experts de SCRUM et Agile, les développeurs, designers…) peuvent consulter le code source de notre solution et de la modifier.

Pour cela nous faisons usage de deux solution technologiques Git et GitHub.

* + - 1. GitHub

GitHub est une plateforme open source de gestion de versions et de la collaboration destiné aux développeurs des solutions numériques.

GitHub représente une version professionnelle de réseaux sociaux qui accueillent les développeurs où ils peuvent suivre les activités, accéder aux différentes ressources et d’évaluer le travail de leurs collègues.



Figure 42 : le logo de GitHub

* + - 1. Git

Git est un système de gestion de code source et de contrôle de version

distribué, libre et open source. Outre le contrôle de version des logiciels, Git peut-être utiliser pour d’autres fins tels que pour la gestion de contenu, la gestion de configuration, la suivi de l’historique des interventions sur le code source …



Figure 43 : le logo de Git

L’utilisation de ces deux solutions durant le développement de notre projet et pour la publication de code source présente plusieurs avantages tels que :

* L’utilisation de ceux deux solutions est gratuite à l’installation et à l’usage.
* Le grand choix de différents outils intégrés tels que les outils de de suivi de l’avancement, la localisation des erreurs, la documentation…
* Les développeurs qui collaborent sur un projet ont accès a un document où ils peuvent consulter les traces des modifications qui ont été apportées aux fichiers.
* La conservation de l’intégralité de chaque fichier dans chaque version d’où il est possible de récupérer l’ancien contenu d’un fichier.

La figure présente ci-dessous visualise notre repository (répertoire) sous GitHub qui contient l’intégralité des fichiers de notre projet ainsi que la licence du projet qui nous permet de protéger nos droits.

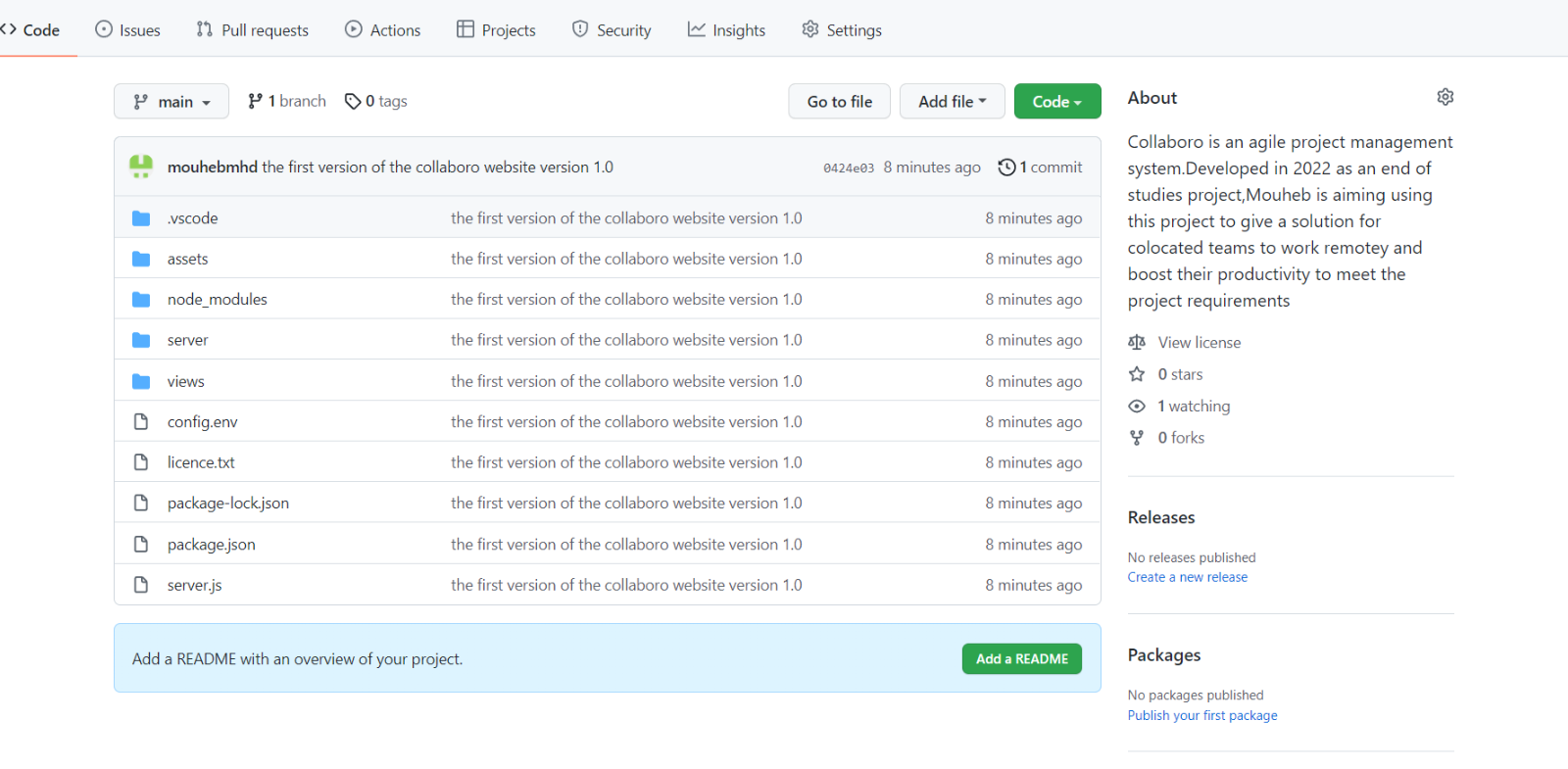


Figure 45 : Repository de projet sous GitHub

* + 1. Le déploiement de l’application

Comme il est présenté dans la figure ci-dessous, notre application est déployée dans une architecture classique composé de trois nœuds : un nœud qui représente la machine utilisateur, un nœud représentant le serveur d’application et un nœud représentant le serveur de gestion de base de données.

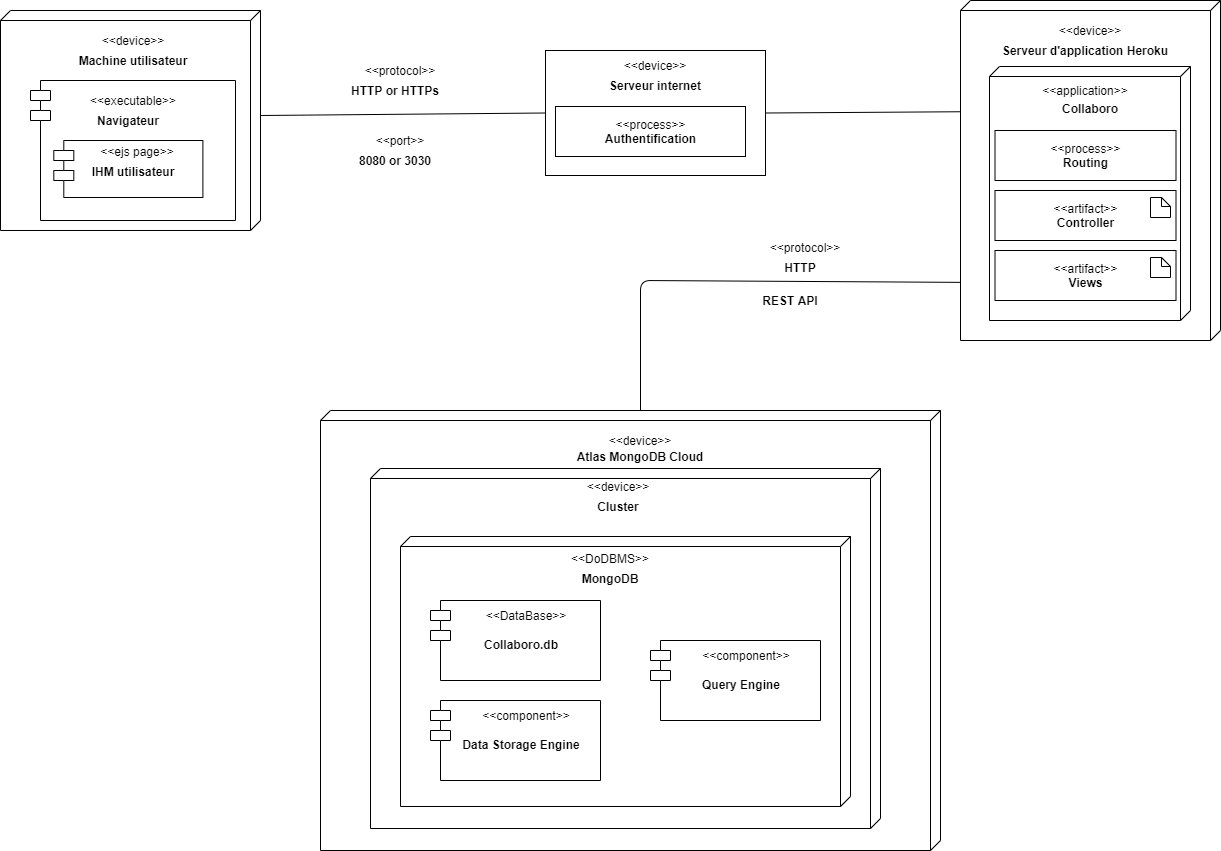


Figure 44 : Diagramme de déploiement

Les services utilisés pour le déploiement de notre application sont :

* Heroku

Heroku est un fournisseur cloud qui offre une plateforme en tant que service (Plateforme As A Service) qui supporte plusieurs langages de programmation tels que Scala, NodeJS, Ruby, PHP, GO …

Les applications hébergées par Heroku sont généralement des applications desktop ou web qui utilisent le protocole http pour établir une communication entre les deux cotés serveur et client.

Logo Heroku.svg

Figure 45 : logo de l’entreprise Heroku

* Atlas MongoDB Cloud

MongoDB Atlas est une plate-forme de données d'application multi-cloud. Ce fournisseur cloud permet d’héberger les bases de données des applications modernes telles que les applications mobiles, les applications web, les applications IOT etc…

Ce CSP permet également aux utilisateurs de créer leurs propres clusters en ligne et de faire la configuration de leurs serveurs de bases de données.



Figure 46 : logo de Atlas MongoDB Cloud

Les bases de données hébergées par Atlas sont généralement installées dans un cluster (un groupe de deux ou plusieurs ordinateurs, ou nœuds, qui fonctionnent en parallèle pour atteindre un objectif commun).

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. MongoDB prend en charge plusieurs moteurs de stockage, car différents moteurs sont plus performants pour des charges de travail spécifiques.

* Moteur de requêtes

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

1. Les tests

Durant cette dernière section, nous allons faire la conception des derniers tests d’acceptation et les implémenter afin de vérifier le bon fonctionnement de notre application.

## Les tests fonctionnels

* + 1. Test unitaire de cas d’utilisation « S’authentifier »

Le test de cas d’utilisation « S’authentifier » permet de protéger notre application contre tout type accès non autorisé.

Le tableau ci-dessous présente les informations relatives au cas de test

« S’authentifier » que nous allons concevoir.

|  |  |
| --- | --- |
| **Cas de test** | **Description** |
| **Nom de cas de test** | «S’authentifier » |
| **Auteur de cas de test** | Mouheb Mehdoui |
| **Critères d’acceptation** | Permet de vérifier si un l’utilisateur aura accès aux différentes fonctionnalités |
| **Date d’exécution** | Le 3 Février 2022 |

Tableau : Cas de test « S’authentifier »

Le tableau ci-dessous visualise la classe d’équivalence de notre cas de test

|  |  |  |
| --- | --- | --- |
| **Condition** | **Classe valide** | **Classe invalide** |
| **Nombre d’entrée** | 2 | <>2 |
| **Type d’entrée** | String String | Non spécifié |
| **Taille email** | >0 | 0 |
| **Taille mot de passe** | >0 | 0 |

Tableau : Classe d’équivalence de cas de test

Le tableau suivant représente quelques données de test que nous allons utiliser pour tester le bon fonctionnement de notre cas d’utilisation.

|  |  |  |  |
| --- | --- | --- | --- |
| **Email** | **Mot de passe** | **Retour attendu** | **conformité** |
| ‘’ | ‘’ | Login invalide | Conforme |
| a@gmail.com | ‘’ | Login invalide | Conforme |
| alexDiscard@gmail.cpm | 123 | Aucun compte avec ce email | Conforme |
| alexDiscard@gmail.cpm | Hb123456789 | Page d’accueil | Conforme |

Tableau : Résultat de test de cas d’utilisation « S’authentifier »

## Les tests fonctionnels

Le tableau ci-dessous présente les informations relatives au cas de test

« S’authentifier » que nous allons concevoir.

|  |  |
| --- | --- |
| **Cas de test** | **Description** |
| **Auteur de cas de test** | Mouheb Mehdoui |
| **Critères d’acceptation** | Permet de vérifier si les fonctionnalités répondent aux besoins spécifiés précédemment |
| **Date d’exécution** | Le 14 Avril 2022 |

Tableau : informations des cas de test

Les tests fonctionnels que nous avons effectués sur notre application sont présentés dans le tableau ci-dessous.

|  |  |  |  |
| --- | --- | --- | --- |
| **Cas de test** | **Enchainement** | **Comportement attendu** | **conformité** |
| Créer un nouveau compte | L’utilisateur doit remplir le formulaire et valide les données saisies | Un nouveau compte est créé. | Conforme |
| Gérer son profil | L’utilisateur modifie les données de son compte et valide son choix | Les données de l’utilisateur sont mis à jour. | conforme |
| Créer une nouvelle user story | Le Product Owner doit saisir les données d’une nouvelle user story puis clique sur créer | Une nouvelle user story doit être crée et affiché dans l’interface correspondante. | conforme |
| Supprimer un membre de l’équipe de développement | Le Scrum master sélectionne un membre et clique sur supprimer | Le membre sélectionné doit être retiré de l’équipe de développement de projet . | conforme |
| Modifier le rôle d’un membre de l’équipe de développement d’un projet | Le Scrum master modifie le rôle d’un membre d’une équipe et clique sur enregistrer | Le rôle du membre doit être mis à jour et affiché dans l’interface correspondante . | conforme |
| Affichage des charts du sprint en cours | Le Scrum mater et les développeurs consulte leur sprint en cours | Les charts du sprint en cours doivent être générés automatiquement en fonction de l’états des tâches | conforme |
| La déconnexion | L’utilisateur clique sur le bouton se déconnecter afin de terminer sa session. | L’utilisateur doit être redirigé vers la page de connexion | conforme |

Tableau : les résultats des tests fonctionnels

1. **Planning réel de réalisation de projet**

La figure ci-dessous présente le planning réel qui la chronologie des différentes phases de réalisation de notre projet .

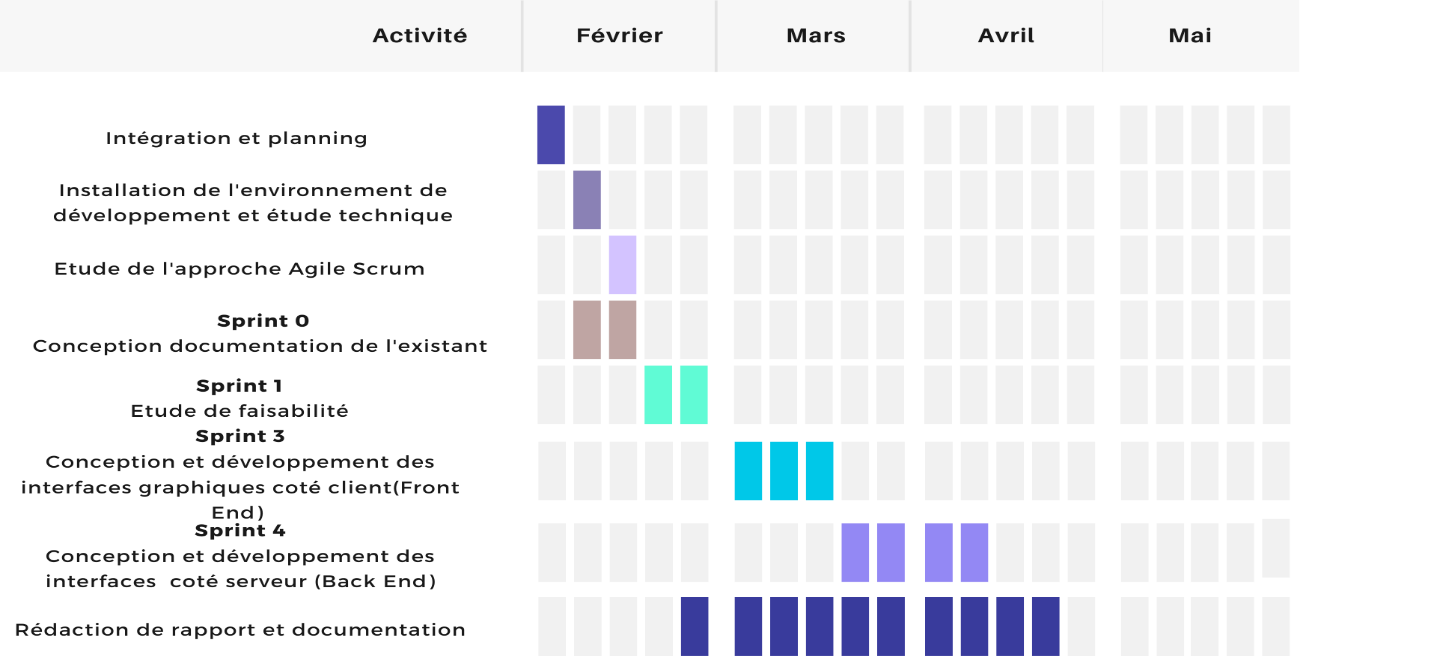


Figure 47 **:** Diagramme de Gantt du planning réel du projet

## Conclusion

Durant ce dernier chapitre, nous avons décrit les différentes phases de réalisation de notre projet, les environnement matériel et logiciel qui nous a permis de développer notre application ainsi que l’élaboration de diagramme d 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 s'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 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 demandés 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 améliorées grâce à leurs retours afin pour le but de bien mener ma mission.

Bibliographie

Annexe

La figure ci-dessous présente le résultat du rapport plagiat.



**Figure annexe 1** : Rapport plagiat