

## DÉDICACES

# REMERCIEMENTS

*Je remercie*

*Je suis reconnaissant*

*J'exprime ma gratitude*

# Table des matières

<b>Introduction générale</b>	<b>1</b>
<b>I Contexte et objectif du projet</b>	<b>3</b>
INTRODUCTION . . . . .	3
I.1 CONTEXTE GENERAL . . . . .	3
I.1.1 Cadre du projet . . . . .	3
I.1.2 Présentation de l'organisme d'accueil . . . . .	3
I.1.3 Problématique du projet . . . . .	4
I.2 Étude de l'existant . . . . .	5
I.2.1 Solutions existantes . . . . .	5
I.2.2 Critique de l'existant . . . . .	7
I.2.2.1 Critères Techniques : . . . . .	7
I.2.2.2 Critères Fonctionnelles : . . . . .	8
I.3 Solution proposé . . . . .	10
I.4 Méthodologie adoptée . . . . .	10
I.4.1 Choix méthodologique . . . . .	10
I.4.2 Présentation de la méthodologie SCRUM . . . . .	10
I.4.3 Les rôles SCRUM . . . . .	12
I.4.4 Les artefacts de la méthodologie Scrum . . . . .	12
I.4.5 Les événements SCRUM . . . . .	12
<b>II Analyse et spécification des besoins</b>	<b>15</b>
INTRODUCTION . . . . .	15
II.1 Analyse et identification des besoins . . . . .	15
II.1.1 Identification des acteurs . . . . .	15
II.1.2 Identification des besoins fonctionnels . . . . .	16
II.1.3 Identification des Besoins non fonctionnelles . . . . .	17
II.2 Spécification des besoins fonctionnels . . . . .	17
II.2.1 Diagramme de cas d'utilisation général . . . . .	18
II.2.2 Diagrammes de cas d'utilisation détaillés . . . . .	18
II.2.2.1 Diagramme de cas d'utilisation détaillé sur l'acteur « Administra- teur » et «Gestionnaire» . . . . .	18

II.2.2.2	Diagramme de cas d'utilisation détaillé sur l'acteur « Technicien »	19
II.3	Backlog du produit et pilotage du projet avec Agile Scrum	20
II.3.1	Backlog du produit	20
II.3.2	Pilotage du projet avec Agile Scrum	22
II.4	Diagramme de Gantt	22
<b>III</b>	<b>Initialisation technique du projet agiles</b>	<b>24</b>
	INTRODUCTION	24
III.1	Architecture de l'application	24
III.1.1	Architecture générale de l'application	24
III.1.2	Architecture logicielle de l'application	25
III.2	Les Frameworks de développement	26
III.2.1	Framework Front-end :	26
III.2.2	Framework Backend-end :	26
III.2.2.1	Framework	26
III.2.2.2	Web service REST	27
III.3	Base de données NoSQL	27
III.4	Environnement de travail	28
III.4.1	Environnement matériel	28
III.4.2	Environnement de développement, design et modélisation	28
<b>IV</b>	<b>ÉTUDE ET RÉALISATION DU SPRINT 1</b>	<b>30</b>
	INTRODUCTION	30
IV.1	Backlog du sprint	30
IV.2	Diagramme de cas d'utilisation général	31
	<b>Conclusion générale</b>	<b>32</b>
	<b>Bibliographie</b>	<b>33</b>
	<b>Annexes</b>	<b>34</b>
	Annexe 1. Exemple d'annexe	34
	Annexe 2. Entreprise	35

# Table des figures

1	Logo du Digital MARKET . . . . .	4
2	PlanningPME . . . . .	6
3	Track-POD . . . . .	6
4	MAPO PLANNER . . . . .	7
5	Cycle de vie de la méthode SCRUM . . . . .	11
6	diagramme cas d'utilisation général. . . . .	18
7	Diagramme cas d'utilisation de l'acteur administrateur et gestionnaire. . . . .	19
8	Diagramme cas d'utilisation de l'acteur technicien. . . . .	20
9	Diagramme DE GANTT . . . . .	23
10	Architecture MERN . . . . .	25
11	Architecture MVC . . . . .	25
12	Logo ReactJS. . . . .	26
13	Logo NodeJS et ExpressJS. . . . .	27
14	Logo MongoDB. . . . .	28
	Annexe 2.1 Logo d'entreprise . . . . .	35

# Liste des tableaux

1	Informations de contact . . . . .	4
2	<b>Analyse des criteres technique</b> . . . . .	8
3	<b>Analyse des critères fonctionnales</b> . . . . .	9
4	Backlog du produit . . . . .	21
5	Sprints . . . . .	22
6	caractéristique de pc . . . . .	28
7	Outils et leurs descriptions . . . . .	28
8	Backlog du sprint 1 . . . . .	30
	Annexe 1.1 Exemple tableau dans l’annexe . . . . .	34

## LISTE DES ABRÉVIATIONS

MERN = MongoDB Express.js React.js Node.js

MVC= Model-View-Controller

---

## INTRODUCTION GÉNÉRALE

L'évolution rapide du développement web ces dernières années a profondément transformé les modes de fonctionnement des entreprises et les attentes des utilisateurs. Parmi les processus essentiels à la gestion d'une entreprise, la planification joue un rôle central. Une bonne organisation des ressources humaines et matérielles permet d'assurer un déroulement fluide des opérations, d'éviter les conflits d'agenda et d'optimiser le temps et les coûts. Dans des secteurs nécessitant une intervention rapide et efficace, comme la maintenance technique, une planification rigoureuse est cruciale pour garantir la satisfaction des clients et la performance des équipes.

Cependant, la gestion des techniciens et des véhicules de service représente un défi majeur pour les entreprises. L'absence d'un système centralisé entraîne des conflits de réservation, des pertes de temps et un manque de visibilité sur les interventions en cours et à venir. De plus, la coordination entre les équipes techniques et les gestionnaires est souvent complexe, ce qui peut provoquer des retards et une inefficacité accrue. Il est donc essentiel de mettre en place une solution optimisée pour améliorer la répartition des ressources et assurer un suivi en temps réel des opérations.

Dans ce contexte, notre projet vise à développer une solution informatique permettant la gestion des plannings des techniciens et des véhicules de service. L'objectif est de centraliser et d'optimiser la planification des interventions afin de :

- **Assurer un suivi précis des interventions** réalisées et en cours.
- **Éliminer les conflits de réservation** grâce à une vision unifiée
- **Renforcer la transparence** avec des tableaux de bord interactifs
- **Intégration dun chatbot** Améliorer la communication et l'efficacité des équipes sur le terrain.

Cette solution apportera une valeur ajoutée en facilitant la coordination entre les techniciens et les gestionnaires, tout en offrant une meilleure visibilité sur l'utilisation des ressources . De



---

plus Les bénéfices apporté pour l'entreprise comme amélioration de la productivité en réduisant les pertes de temps liées aux mauvaises planifications ,Réduction des coûts opérationnels en optimisant l'usage des véhicules et des ressources et Meilleure satisfaction client grâce à une intervention plus rapide et efficace.

Notre travail est présenté sur deux parties principales : La première partie « **Etude préalable** » commence par le chapitre « **Contexte général** » dont nous allons introduire l'idée générale du projet, présenter la société d'accueil, étudier quelques exemples d'applications existantes et présenter la méthodologie de travail adoptée.

Le deuxième chapitre « **Analyse et spécification des besoins** » permet de présenter la spécification des besoins fonctionnels et non fonctionnels ainsi une spécification semi formelle de besoins

Le troisième chapitre « **Initialisation du projet** » permet de présenter l'architecture sans serveur utilisé dans notre application, , les frameworks de développement ainsi que l'environnement matériel et logiciel.

La deuxième partie « **Mise en oeuvre avec Scrum** » est dédiée à la réalisation des sprints à savoir la spécification, l'analyse, la conception et la réalisation. nous terminons par une conclusion générale

---

## CHAPITRE I

# CONTEXTE ET OBJECTIF DU PROJET

## Introduction

Ce chapitre introductif commence par décrire le contexte général du projet. Il examine ensuite les solutions existantes et leurs critiques, avant de présenter et de comparer différentes méthodologies. Cela permet de clarifier le choix de la méthode que nous adopterons pour notre application web.

## I.1 CONTEXTE GENERAL

### I.1.1 Cadre du projet

Le travail présenté s'inscrit dans le cadre du projet de fin d'études en vue de l'obtention du diplôme de licence en sciences de l'informatique de l'Institut Supérieur d'Informatique de Mahdia (ISIMA).

### I.1.2 Présentation de l'organisme d'accueil

#### Présentation de Digital Market

Le stage se déroule chez **Digital Market**, une agence web spécialisée dans le développement et le marketing digital.

Cette entreprise se concentre sur les services informatiques et numériques, avec une expertise dans trois domaines principaux : l'e-commerce, la gestion d'entreprise intégrée et la business intelligence.

Digital Market propose une variété de services, notamment :

- **Création Web et Mobile** : L'équipe conçoit des sites web et des applications mobiles sur mesure, adaptés à différents domaines d'activité et répondant aux exigences spécifiques de chaque client.
- **Optimisation SEO** : Les experts en référencement optimisent les sites web pour générer un trafic qualifié et augmenter les revenus des clients.
- **Stratégie e-commerce** : L'entreprise crée des boutiques en ligne performantes pour l'achat en ligne et accompagne les clients vers le succès.
- **Design UI/UX** : Création d'interfaces utilisateur et d'expériences utilisateur optimisées.
- **Design graphique** : Conception graphique pour des supports variés.



FIGURE 1 – Logo du Digital MARKET

Pour toute demande ou question concernant l'entreprise, veuillez les contacter directement aux coordonnées suivantes :

Information	Détail
Email	contact@digital-market.fr
Téléphone	+216 55 619 993
Adresse	Sahline, Monastir, Tunisie

TABLE 1 – Informations de contact

### I.1.3 Problématique du projet

L'entreprise souhaite mettre en place une solution informatique pour optimiser la gestion des plannings des techniciens et des véhicules de service. Actuellement, elle fait face à plusieurs défis majeurs en raison de l'absence d'un système intégré de gestion des plannings.

Ces défis incluent :

- **Conflits de réservation** : L'absence d'un système centralisé entraîne des chevauchements d'interventions fréquents, provoquant des retards et une insatisfaction client.

- **Répartition inégale de la charge de travail** : Certains techniciens sont surchargés, tandis que d'autres sont peu utilisés, ce qui affecte leur productivité et leur motivation.
- **Difficulté de coordination** : La planification manuelle des interventions complique la coordination et nécessite beaucoup de temps, surtout lorsque plusieurs techniciens et véhicules sont impliqués.
- **Manque de réactivité** : Le manque de visibilité en temps réel sur les disponibilités et les plannings limite la capacité de l'entreprise à répondre rapidement aux demandes urgentes des clients.
- **Absence de rapports détaillés** : Sans système automatisé, la génération de rapports sur les interventions, l'utilisation des véhicules et la performance des techniciens est laborieuse et peu fiable, ce qui complique l'analyse et la prise de décision.

Ces problèmes entravent la coordination des interventions, réduisent la réactivité de l'entreprise face aux demandes des clients et nuisent à son efficacité opérationnelle globale. Il est donc essentiel de mettre en place une solution informatique dédiée pour optimiser la gestion des plannings, améliorer l'efficacité des interventions et assurer un suivi précis des activités.

## I.2 Étude de l'existant

Avant de proposer une solution, il est essentiel d'analyser l'existant, puis de procéder à une critique afin d'identifier les contraintes à respecter.

### I.2.1 Solutions existantes

Dans ce cadre, une étude des solutions existantes sur le marché a été réalisée afin de mieux définir l'objectif du projet. Suite à une analyse approfondie, la section suivante présentera quelques exemples d'applications existantes, permettant ainsi d'identifier les fonctionnalités à intégrer pour optimiser la solution proposée.

#### • Plateforme 1 : PlanningPME

PlanningPME [1] est une solution de gestion des plannings et des ressources, conçue pour aider les entreprises à organiser et optimiser l'emploi du temps de leurs équipes, techniciens et véhicules. Elle permet de créer et suivre des plannings en temps réel, tout en facilitant l'affectation des ressources et la gestion des interventions. Les utilisateurs peuvent planifier, ajuster et visualiser facilement les tâches, tout en générant des rapports détaillés pour un suivi précis. Le système inclut également des alertes et notifications pour garantir une communication fluide et en temps réel. PlanningPME est idéale pour les entreprises nécessitant une gestion simple, claire et flexible des plannings, et s'adapte à divers secteurs d'activité.

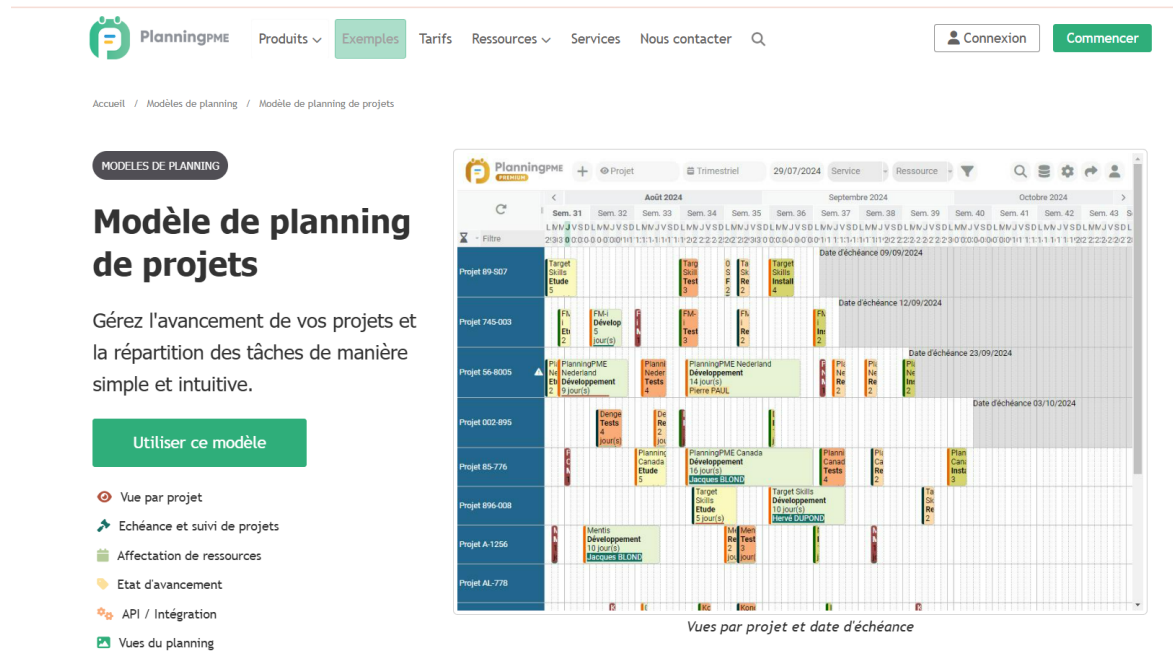


FIGURE 2 – PlanningPME

### •Plateforme 2 :Paraxedo

Praxedo [2] est une plateforme de gestion des interventions sur le terrain, qui optimise la planification, la gestion des équipes et le suivi des tâches. Elle permet de planifier les interventions, de suivre les techniciens en temps réel et d'automatiser le dispatching des tâches. Praxedo facilite également la collecte de données sur le terrain, telles que des rapports, des photos et des signatures électroniques, pour un suivi détaillé. La plateforme offre une gestion centralisée des ressources et inclut des alertes et notifications pour une gestion optimale. Elle se distingue de PlanningPME par son focus sur l'automatisation des affectations et la collecte de données sur le terrain.

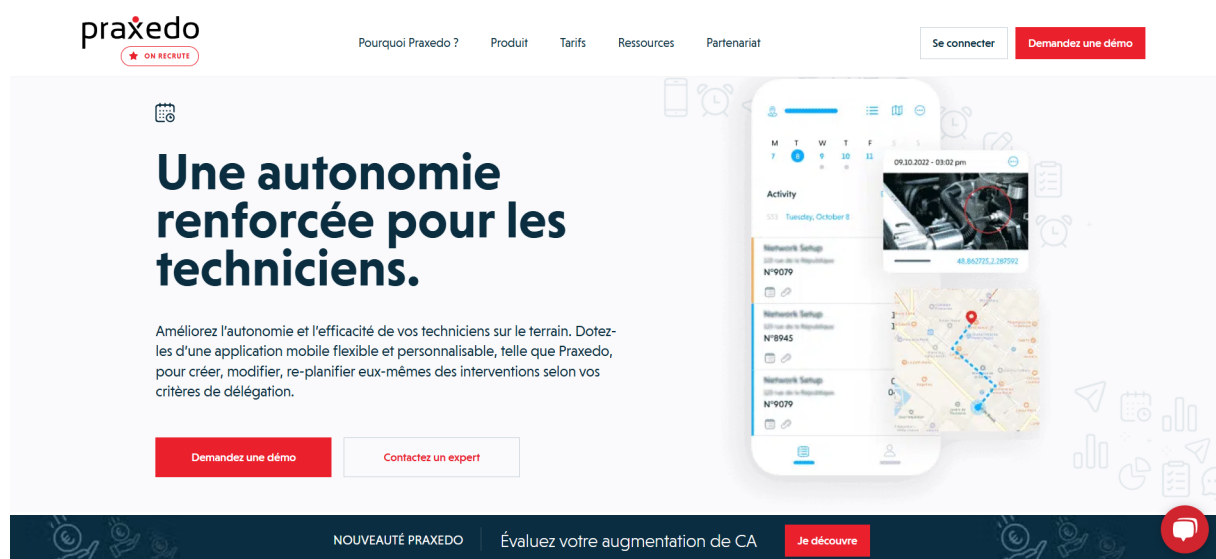


FIGURE 3 – Track-POD

### •Plateforme 3 :MAPO PLANNER

MAPO Planner [3] est une plateforme de gestion des plannings et des ressources, optimisant l'affectation des techniciens, véhicules et autres ressources en fonction de la disponibilité, des compétences et de la localisation. Elle offre un suivi en temps réel, des notifications automatiques et des rapports détaillés via une interface intuitive et une application mobile. Contrairement à PlanningPME, qui se concentre sur la gestion des plannings, MAPO Planner optimise l'affectation des ressources en prenant en compte plusieurs critères. Par rapport à Praxedo, qui se distingue par l'automatisation du dispatching, MAPO Planner offre une gestion plus globale des ressources pour une efficacité maximale.

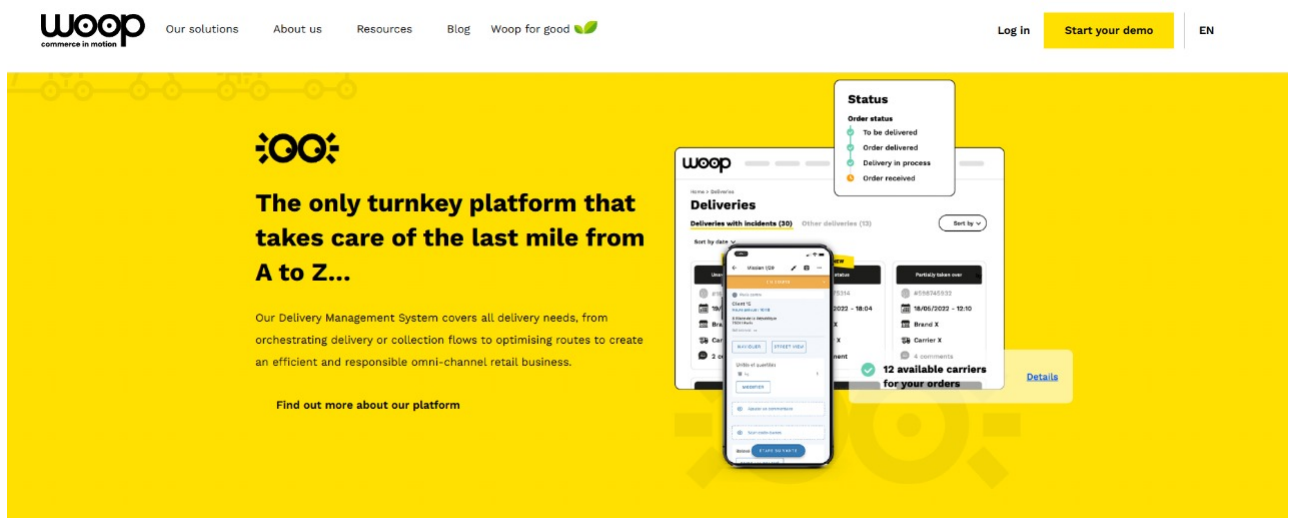


FIGURE 4 – MAPO PLANNER

## I.2.2 Critique de l'existant

L'évaluation des solutions existantes, mentionnées précédemment, repose sur deux critères essentiels, déterminants dans l'analyse de ces applications.

### I.2.2.1 Critères Techniques :

Afin d'assurer des résultats objectifs et comparables, GTmetrix 6, un outil de référence pour l'évaluation des performances web, a été utilisé. Cet outil permet l'analyse de plusieurs indicateurs clés, notamment :

—**Performance** : Ce critère mesure la rapidité globale du site en fonction de son score de performance. Une note entre 90% et 100% indique une excellente performance, entre 80% et 89% une bonne performance, entre 70 % et 79% une performance moyenne, tandis qu'un score inférieur à 70% suggère des améliorations nécessaires.

—**Structure** : Ce critère évalue la qualité du code et l'organisation du site. Un score supérieur ou égal à 90 % indique une plateforme bien optimisée, entre 80% et 89% une bonne qualité, tandis qu'un score inférieur à 80% révèle des axes d'amélioration.

— **Temps de chargement** : Ce critère, mesuré par LCP, évalue le temps nécessaire pour afficher l'élément principal visible à l'écran. Une valeur inférieure à 1,5 seconde est considérée comme excellente, comprise entre 1,5 et 2,5 secondes, elle est satisfaisante, tandis qu'un temps supérieur à 2,5 secondes est jugé lent et nécessite une optimisation

—**Stabilité visuelle** : Ce critère est évalué en mesurant les décalages inattendus des éléments de la page. Un score de CLS égal à 0 reflète une stabilité parfaite, entre 0,1 et 0,25, il reste tolérable, tandis qu'un score supérieur à 0,25 indique des décalages pouvant nuire à l'expérience utilisateur.

Le tableau I.2 présente une étude comparative technique de solutions précédemment

Critère	PlanningPME	Paraxedo	MAPO PLANNER
Performance	85%	62%	57%
Structure	84%	69%	83%
Temps de chargement	1.3s	3.7s	1s
Stabilité visuelle	0.03	0.04	0.08

TABLE 2 – Analyse des critères technique

Sur le plan technique, PlanningPME se distingue par son excellente performance, sa structure optimisée et sa grande stabilité visuelle, tandis que MAPO Planner s'illustre par le temps de chargement le plus rapide. Praxedo, bien que moins performant sur ces indicateurs, reste correct mais nécessite des optimisations, notamment en vitesse de chargement et en structure.

### I.2.2.2 Critères Fonctionnelles :

Des critères ont été identifiés afin de comparer et d'évaluer les applications présentées dans le tableau I.3. Cette analyse permet de déterminer les points forts et les limites des solutions existantes, facilitant ainsi le choix des fonctionnalités à intégrer pour améliorer l'application en développement. Ces critères sont :

#### 1- Rapport et Historique :

Générer des rapports sur les activités des techniciens, l'utilisation des véhicules et les interventions réalisées.

#### 2-Suivi en temps réel :

Visualiser les plannings et suivre l'avancement des tâches.

#### 4-Assistant virtuel :

L'utilisateur a un accès à un assistant virtuel (chatbot) avec laquelle la communication se fait en temps réel.

**5-Payant :**

pour pouvoir utiliser et accéder aux fonctionnalités de l'application un prix est payé.

**8-Notifications et Alertes :**

Alertes pour les missions à venir ou les changements de planning.

Critère	PlanningPME	Paraxedo	MAPO PLANNER	Notre Application
Rapport et Historique	Oui	Oui	Oui	Oui
Suivi en temps réel	Oui	Oui	Oui	Oui
Assistant virtuel	Non	Non	Non	Oui
Payant	Oui	Oui	Oui	Non
Intégration avec Google Maps	Non	Oui	Oui	Oui
Notifications et Alertes	Oui	Oui	Oui	Oui

TABLE 3 – Analyse des critères fonctionnelles



Sur le plan fonctionnel, les trois plateformes offrent des fonctionnalités essentielles (rapports, suivi en temps réel, notifications). Cependant, l'intégration de Google Maps n'est présente que chez Praxedo et MAPO Planner, tandis qu'aucune ne propose d'assistant virtuel intégré. Toutes les solutions sont payantes et visent une clientèle professionnelle prête à investir pour la gestion et l'optimisation de ses plannings

### **I.3 Solution proposé**

Notre application se distingue par son intégration de toutes les fonctionnalités requises, notamment un assistant virtuel et l'intégration de Google Maps, lui conférant ainsi un avantage concurrentiel significatif. Elle propose donc une solution plus complète et optimisée pour la gestion des plannings des techniciens et des véhicules de service.

### **I.4 Méthodologie adoptée**

Avant de commencer un projet informatique, il est crucial de choisir une méthodologie de travail et un processus de suivi appropriés pour garantir la création d'un logiciel fiable. Cette méthodologie définit un cadre qui structure les étapes initiales du développement d'un système pour répondre de manière optimale aux besoins des clients.

#### **I.4.1 Choix méthodologique**

Le choix de la méthodologie de gestion de projet est crucial et dépend de plusieurs facteurs, tels que la nature et la taille du projet.

Le modèle en cascade est une méthode de développement de logiciels qui suit une séquence étape par étape, adaptée aux projets de petite taille et bien définis. En revanche, le modèle en V est généralement utilisé pour des projets où les exigences sont clairement établies dès le départ et où les modifications ultérieures sont limitées.

La méthode Agile favorise une meilleure communication avec le client ainsi qu'une transparence accrue du produit final. Elle permet une gestion continue de la qualité et une détection précoce des problèmes, ce qui contribue à réduire les retards et les coûts. Par conséquent, nous avons décidé d'adopter la méthodologie Scrum pour la conception et le développement, afin de garantir le succès et une gestion efficace de notre projet à chaque étape.

#### **I.4.2 Présentation de la méthodologie SCRUM**

La méthodologie SCRUM est une approche agile qui privilégie un développement logiciel itératif et incrémental. Elle repose sur une liste transparente et priorisée des demandes d'évolution

ou de corrections, appelée "backlog produit". Des livraisons fréquentes, généralement toutes les deux à quatre semaines, permettent au client de recevoir des versions fonctionnelles du logiciel de manière régulière. À chaque itération, le logiciel s'enrichit de nouvelles fonctionnalités, s'adaptant ainsi aux besoins évolutifs du client. Cette approche favorise une grande flexibilité et une adaptation continue aux changements, tout en garantissant une progression constante et mesurable.

La figure 1.5 ci-dessous illustre le cycle de vie de la méthode SCRUM.

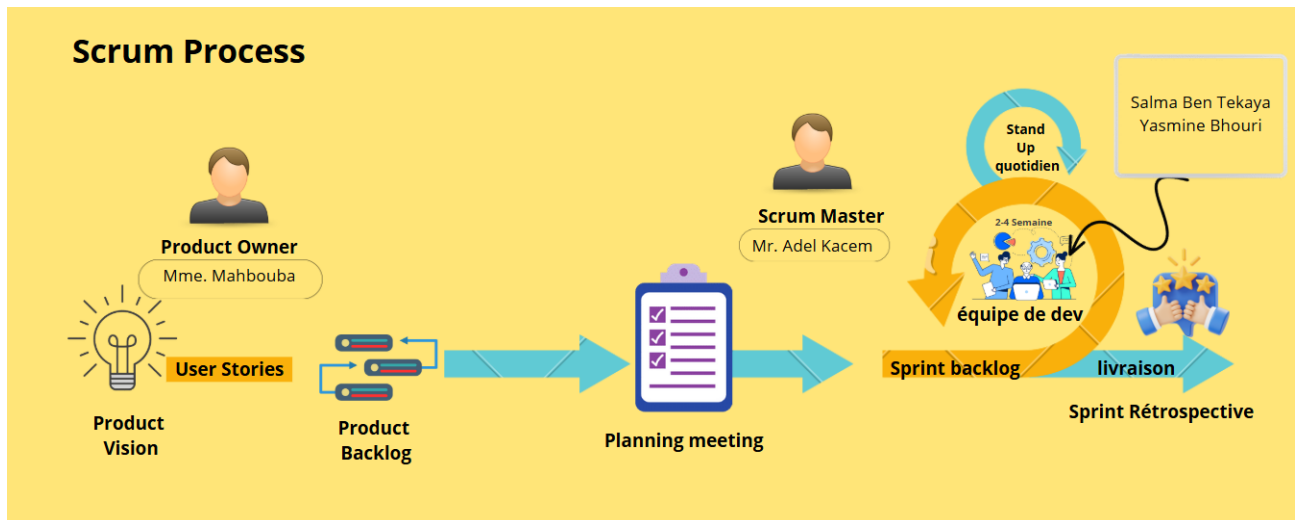


FIGURE 5 – Cycle de vie de la méthode SCRUM

SCRUM est un cadre de travail structuré, particulièrement adapté à la gestion de projets complexes. Il se caractérise par des cycles de développement courts et répétitifs, appelés "sprints", d'une durée typique de deux à quatre semaines. L'objectif central de cette approche est de maximiser la satisfaction du client en livrant rapidement des fonctionnalités opérationnelles. Pour garantir une coordination efficace, des réunions quotidiennes, ou "daily scrums", sont organisées. Ces réunions permettent à l'équipe de faire le point sur l'avancement des tâches, d'identifier les obstacles et de planifier les actions du jour.

Avant de lancer un nouveau sprint, l'équipe doit finaliser les tâches du sprint en cours. Une démonstration des fonctionnalités développées est ensuite réalisée pour recueillir les retours du client et s'assurer que les objectifs sont atteints. Cette étape, appelée "revue de sprint", est essentielle pour valider les livrables et ajuster les priorités.

Comme le montre la figure, la mise en œuvre de SCRUM commence par la définition des fonctionnalités de l'application, qui constituent le backlog produit. Ensuite, lors de la planification du sprint, l'équipe sélectionne les tâches à réaliser et établit un plan détaillé pour l'itération à venir.

À la fin du sprint, une vérification est effectuée pour s'assurer que le produit partiel est conforme aux objectifs fixés. Cette étape est suivie d'une rétrospective, où l'équipe analyse ce qui a bien

fonctionné et ce qui peut être amélioré pour optimiser les processus lors des prochains sprints. Pendant les sprints, des réunions quotidiennes permettent de maintenir une communication fluide entre les membres de l'équipe. Ces échanges facilitent la résolution rapide des problèmes et assurent une progression constante. Une fois le sprint terminé, la conformité du produit est validée, et les enseignements tirés de la rétrospective sont intégrés pour améliorer les itérations futures.

### I.4.3 Les rôles SCRUM

Scrum est une méthode de gestion de projet agile qui favorise le travail itératif et incrémental au sein de petites équipes. Tout au long de notre projet, nous avons constamment été inspirés par cette approche. En effet, Scrum identifie trois rôles clés dans le processus :

- ✓ **Product Owner** : Il représente les clients et les parties prenantes du projet. Il est responsable de définir les spécifications fonctionnelles et de communiquer la vision globale du produit à l'équipe. Son rôle inclut également la priorisation des fonctionnalités à développer ou à améliorer, ainsi que la validation de celles déjà mises en œuvre.
- ✓ **Scrum Master** : Pilier de la méthodologie Scrum, il accompagne l'équipe en veillant au respect des principes et des bonnes pratiques de Scrum. Il facilite les réunions (planification, revue, rétrospective), élimine les obstacles à la progression et favorise une collaboration fluide entre les membres de l'équipe.
- ✓ **Équipe de développement** : Les développeurs chargés de transformer les besoins définis par le Product Owner en fonctionnalités utilisables.

### I.4.4 Les artefacts de la méthodologie Scrum

- ✓ **Le backlog de produit** : Le backlog contient une liste qui englobe les exigences imposées par le client, et les fonctionnalités à implémenter.
- ✓ **Le backlog de Sprint** : Il contient la liste des user-stories présentes dans le Backlog de produit.

### I.4.5 Les événements SCRUM

- ✓ **Le Sprint** : C'est le cœur de Scrum. L'itération dure entre deux et quatre semaines. À la fin d'un sprint, une version du produit utilisable est livrée.
- ✓ **Le Sprint planning** : C'est un événement qui précède le début du sprint et permet de fixer les objectifs de ce dernier.

- ✓ **la mêlée quotidienne (Daily Scrum)** : C'est une réunion courte d'environ quinze minutes qui permet d'avoir une vue d'ensemble de l'avancement du projet.
- ✓ **Revue du Sprint** : C'est une réunion faite à chaque sprint, au cours de laquelle on clôture ce dernier en faisant un bilan détaillé, et on en entame un nouveau.
- ✓ **Rétrospective de Sprint** : Une réunion de 45 minutes par semaine de sprint. Le Scrum Master coordonne cette réunion. Elle a pour but de mettre en valeur les idées de chacun et de soulever d'éventuels obstacles rencontrés.

Scrum est la méthode agile la plus utilisée aujourd'hui car elle est plus efficace et moins rigide que les méthodes classiques, en fait elle place les besoins du client au centre des priorités du projet. Elle offre une plus grande flexibilité et une meilleure visibilité dans la gestion du projet, ce qui permet à l'équipe d'être plus réactive aux attentes du client.

## **Conclusion**

Ce chapitre présente le contexte général du projet, analyse les solutions existantes et évalue leurs limites. Une comparaison des différentes méthodologies de développement a également été effectuée, aboutissant au choix de l'approche la plus adaptée. Le chapitre suivant sera consacré à l'analyse et à la spécification des besoins, étape essentielle pour définir précisément les exigences fonctionnelles et techniques de la solution.

---

## CHAPITRE II

# ANALYSE ET SPÉCIFICATION DES BESOINS

### Introduction

Ce chapitre a permis de présenter le contexte général du projet, d'analyser les solutions existantes et d'évaluer leurs limites. Une comparaison des différentes méthodologies de développement a également été effectuée, conduisant au choix de l'approche la plus appropriée. Le chapitre suivant sera dédié à l'analyse et à la spécification des besoins, une étape essentielle pour définir précisément les exigences fonctionnelles et techniques de la solution.

### II.1 Analyse et identification des besoins

Cette partie concerne l'identification des acteurs et leurs rôles ainsi que les besoins fonctionnels et non fonctionnels qu'on cherche à adapter dans notre application.

#### II.1.1 Identification des acteurs

Dans cette partie nous allons identifier les acteurs du système à réaliser en prenant compte de ceux du métier qui toucheront de façon directe ce système. Les acteurs du métier qui sont impliqués dans le système informatique sont :

- **Administrateur** : C'est l'acteur principal de l'application. Cette personne doit avoir accès à toutes les informations du système concernant chaque employé, quel que soit leur statut. Elle garantit le bon fonctionnement de l'application. Elle dispose de fonctionnalités pour valider les inscriptions, gérer les techniciens, les gestionnaires et les rapports. En plus il peut suivre et modifier les plannings.

- **Gestionnaire** : C'est un employé de la société qui a un grade inférieur à celui d'un administrateur et qui s'occupe de la plupart des tâches de l'application, telles que la gestion des tâches, des voitures et du calendrier.
- **Technicien** : L'employé de la société dispose d'un profil dans l'application, contenant des informations sur son planning. Il a la possibilité de publier des réclamations et de soumettre des rapports. Par ailleurs, il devrait recevoir des notifications en temps réel pour être informé des missions à venir ou des changements de planning.

=>Maintenant on va préciser les besoins fonctionnels et non fonctionnels de ces acteurs.

### II.1.2 Identification des besoins fonctionnels

Il s'agit des fonctionnalités du système. Ce sont les besoins spécifiant un comportement d'entrée / sortie du système.

L'application doit permettre au **Administrateur** :

- **S'authentifier** : le responsable saisit son identifiant et son mot de passe pour accéder à l'espace d'administration.
- **Gérer techniciens** : le responsable peut ajouter, archiver et mettre à jour les techniciens.
- **Gérer gestionnaires** : le responsable peut ajouter, archiver et mettre à jour les gestionnaires.
- **Gérer les rapports** : le responsable peut soumettre un rapport sur la suivie des techniciens et l'utilisation des voitures .
- **Consulter le planning** : le responsable peut consulter les planning et le modifier .
- **Vailder l'inscription d'un technicien** : l'administrateur décide d'approuver ou de refuser demande d'inscription d'un technicien.

L'application doit permettre au **Gestionnaire** de :

- **S'authentifier** : le gestionnaire saisit son identifiant et son mot de passe pour accéder à son espace personnel.
- **Gérer les voitures** : le gestionnaire peut ajouter des informations sur les voitures et les supprimer.
- **planifier les missions** : le gestionnaire peut ajouter, supprimer ou modifier des missions,associer chaque technicien à une voiture et gérer le calendrier.

L'application doit permettre au **technicien** :

- **S'authentifier** : le technicien saisit son identifiant et son mot de passe pour accéder à son espace personnel.

- **s’inscrit** : le technicien doit s’inscrire et attend l’approbation de admin.
- **soumettre des rapports** : le technicien peut soumettre un rapport inclus description ,les problemes racontée sur sa missions.
- **Consulter planning** : le technicien consulte sa planning et ajouter le statut de la mission.

### II.1.3 Identification des Besoins non fonctionnelles

Il s’agit des besoins qui caractérisent le système. Ce sont des besoins en matière de performance, de type de matériel ou le type de conception. Ces besoins peuvent concerner les contraintes d’implémentation.

**Sécurité** : les contraintes sécuritaires à prendre en compte lors de la réalisation de notre application sont l’attribution des rôles ainsi que des informations d’accès à chaque groupe d’utilisateurs pour protéger l’accès aux différents espaces.

**Ergonomie** : tous les standards d’ergonomie doivent être respectés dont la convivialité et la compréhensibilité des interfaces graphiques, le choix des couleurs, la densité et l’organisation des éléments sur l’écran doivent être lisible et facile à lire, l’application doit être simple à utiliser même par des non experts.

**Disponibilité** : l’application doit être accessible à tout moment pendant tous les jours de la semaine.

## II.2 Spécification des besoins fonctionnels

Cette partie consiste à comprendre le contexte du système. Il s’agit de déterminer les fonctionnalités et les acteurs les plus pertinents, de préciser les risques les plus critiques et d’identifier les cas d’utilisation initiaux. Pour cela nous allons utiliser les cas d’utilisations et les user stories d’une façon complémentaire. Comment ? On va commencer par le diagramme de cas d’utilisation générale puis les diagrammes cas d’utilisation de chaque acteur.



## II.2.1 Diagramme de cas d'utilisation général

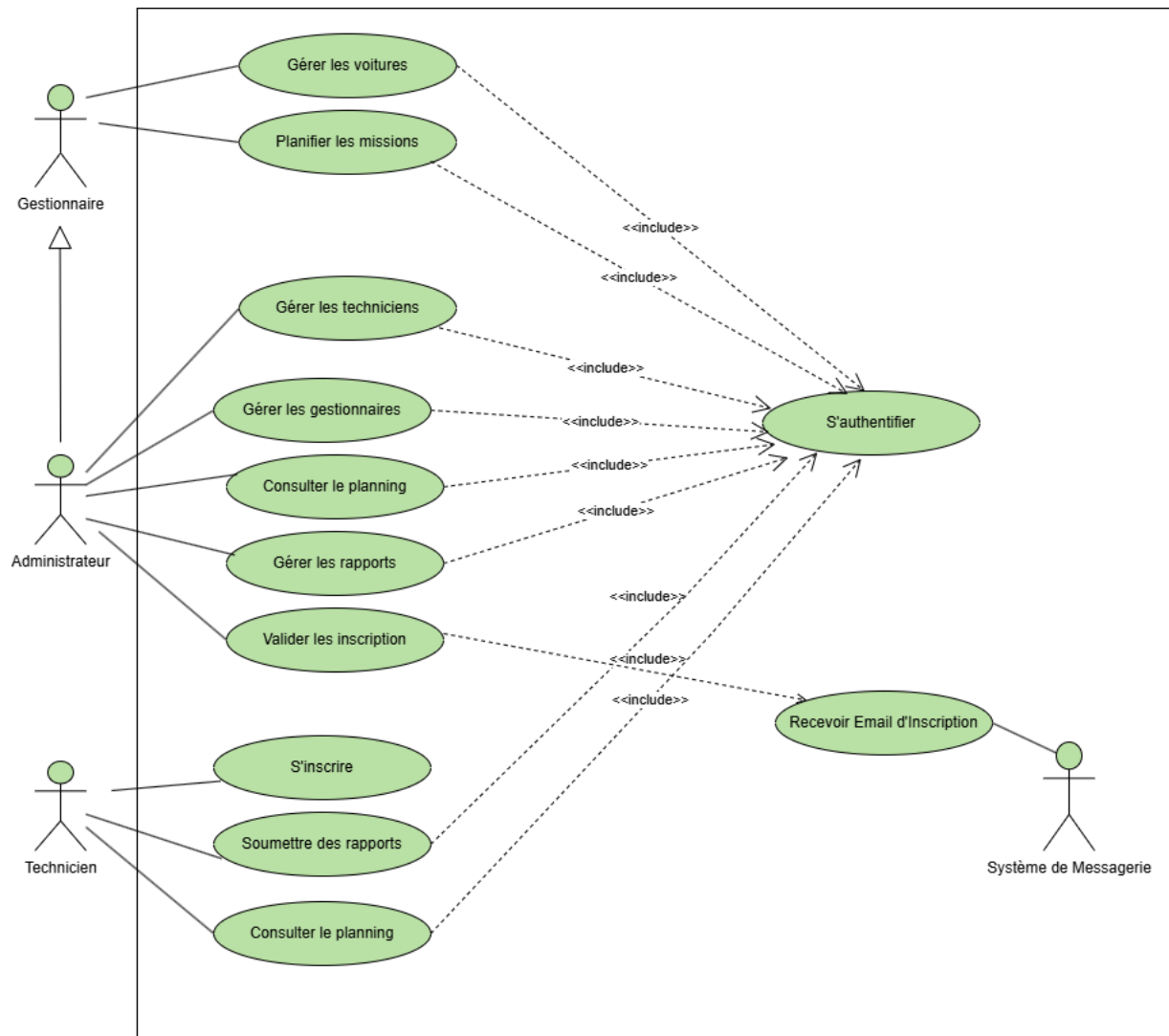


FIGURE 6 – diagramme cas d'utilisation général.

## II.2.2 Diagrammes de cas d'utilisation détaillés

### II.2.2.1 Diagramme de cas d'utilisation détaillé sur l'acteur « Administrateur » et « Gestionnaire »

Après l'authentification, le gestionnaire a accès aux fonctionnalités illustrées dans la Figure 7, dont laquelle : il peut ajouter ou supprimer des voitures, aussi il peut planifier les missions. Ainsi que les fonctionnalités d'un gestionnaire, l'admin possède d'autres privilèges qui consistent à gérer les techniciens et les gestionnaires et valider l'inscription des techniciens.



FIGURE 7 – Diagramme cas d'utilisation de l'acteur administrateur et gestionnaire.

### II.2.2.2 Diagramme de cas d'utilisation détaillé sur l'acteur « Technicien »

Après l'authentification, le technicien a accès aux fonctionnalités illustrées dans la Figure 8, dont laquelle : soumettre des rapports et consulter le planning.

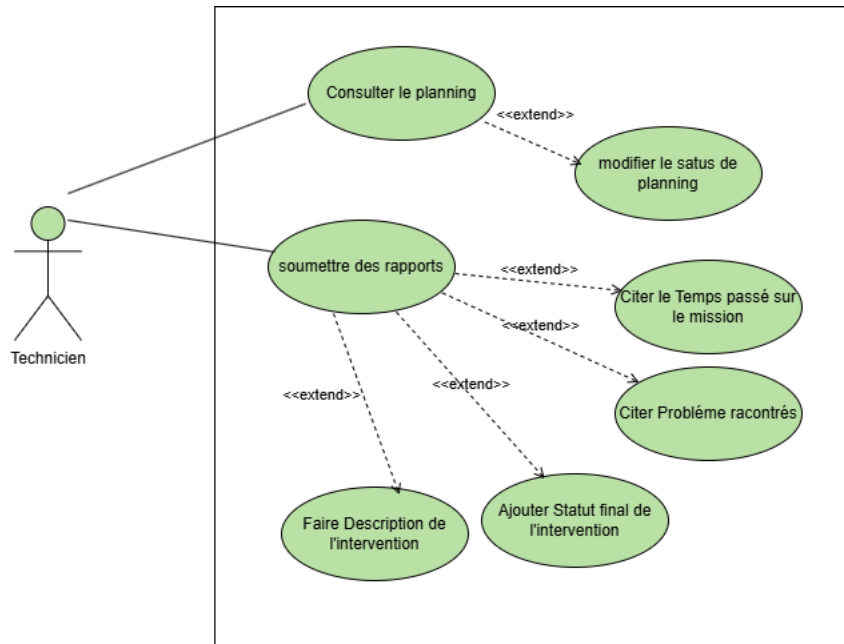


FIGURE 8 – Diagramme cas d'utilisation de l'acteur technicien.

## II.3 Backlog du produit et pilotage du projet avec Agile Scrum

Dans cette partie, l'équipe Scrum est d'abord identifiée, puis le backlog de l'application est défini. Enfin, la mise en place des sprints est réalisée

### II.3.1 Backlog du produit

Le backlog de produit est la liste des fonctionnalités attendues d'un produit, les éléments y sont classés par priorité et ils représentent une description brève de l'exigence fonctionnelle formulée en une phrase sous la forme suivante : « En tant que <acteur>, je veux <un objectif> pour que <une raison> ».

Le Backlog du produit présenté dans le tableau 4 comprend les champs suivants :

**Rang :** C'est un identifiant unique de chaque user story.

**Thème :** Représente le contexte dont existe l'user story.

**Fonctionnalité (User story) :** C'est l'histoire utilisateur décrivant un besoin fonctionnel désiré par le client.

**Priorité :** chaque user story est priorisée

La priorité d'une story peut être changée en cours de réalisation du projet.

Rang	Thème	User Story	Priorité
1	Inscription des techniciens	En tant que <b>technicien</b> , je veux m’inscrire et attend l’approbation de l’admin.	Élevée
		En tant que <b>Admin</b> , je veux valider les demandes d’inscription des techniciens pour sécuriser l’accès.	
2	Gestion des techniciens	En tant que <b>Admin</b> , je veux modifier,consulter et archiver un technicien.	Élevée
3	Gestion des gestionnaires	En tant que <b>Admin</b> , je veux ajouter,modifier,consulter et archiver un gestionnaire.	Élevée
4	Gestion des véhicules	En tant que <b>Gestionnaire</b> ,je veux ajouter,consulter et supprimer des voitures.	Élevée
5	Gestion des missions	En tant que <b>gestionnaire</b> , je veux modifier et ajouter une mission en l’associant à un technicien et un véhicule.	Élevée
		En tant que <b>gestionnaire</b> , je veux gérer le galandrier global pour éviter les conflits.	Moyenne
6	Rapports d’intervention	En tant que <b>technicien</b> , je veux soumettre un rapport après une mission pour informer l’entreprise.	Moyenne
7	Notifications et Alertes	En tant que <b>Système</b> , je veux envoyer des rappels pour les missions à venir aux techniciens.	faible
8	Rapports et historique	En tant que <b>Admin</b> , je veux générer un rapport mensuel des activités pour analyser les performances.	Moyenne
		En tant que <b>Admin</b> , je veux suivre l’état des véhicules pour planifier l’entretien.	Moyenne

TABLE 4 – Backlog du produit

### II.3.2 Pilotage du projet avec Agile Scrum

• **Sprint** : Un sprint est une itération de développement de la méthode Scrum, c'est un intervalle de temps fixe répétable pendant lequel un produit considérée "Terminé" est créé. Pour notre projet nous avons choisi de développer trois sprint illustrées par le Tableau 5.

Sprint	Tâche
<b>Sprint 1</b>	<ul style="list-style-type: none"> <li>— S'inscrire .</li> <li>— S'authentifier.</li> <li>— Gérer les techniciens.</li> <li>— Gérer les gestionnaires.</li> <li>— Gérer les voitures par le gestionnaire.</li> <li>— Gérer les missions par le gestionnaire.</li> </ul>
<b>Sprint 2</b>	<ul style="list-style-type: none"> <li>— consulter le planning des missions par technicien.</li> <li>— soumettre un rapport détaillé à la fin de chaque mission(technicien).</li> <li>— générer des rapports pour le suivi des techniciens, l'utilisation des voitures et les interventions réalisées (admin).</li> <li>— faire des rappels d'alertes pour les missions à venir ou les changements de planning.</li> <li>— S'il y a conflit du planning, il y a des alertes en cas de chevauchement ou de disponibilités insuffisantes.</li> </ul>
<b>Sprint 3</b>	<ul style="list-style-type: none"> <li>— Fonctionnalités offertes à l'employé via Chatbot :               <ul style="list-style-type: none"> <li>— demandes des questions par l'employé.</li> <li>— Poursuivre travail de l'employé.</li> </ul> </li> </ul>

TABLE 5 – Sprints

## II.4 Diagramme de Gantt

Le diagramme de Gantt organise notre projet sur 16 semaines, avec des phases clés telles que l'étude, le pilotage, et trois sprints de développement. Chaque sprint correspond à des fonctionnalités spécifiques à livrer, permettant de structurer le travail selon les priorités. Ce diagramme sert de cadre visuel pour suivre l'avancement des tâches, ajuster les ressources et prendre des décisions pour respecter les délais. Il facilite la gestion de projet en offrant une vue d'ensemble évolutive. Enfin, il permet de s'adapter aux imprévus et aux changements de priorités

## DIAGRAMME DE GANTT

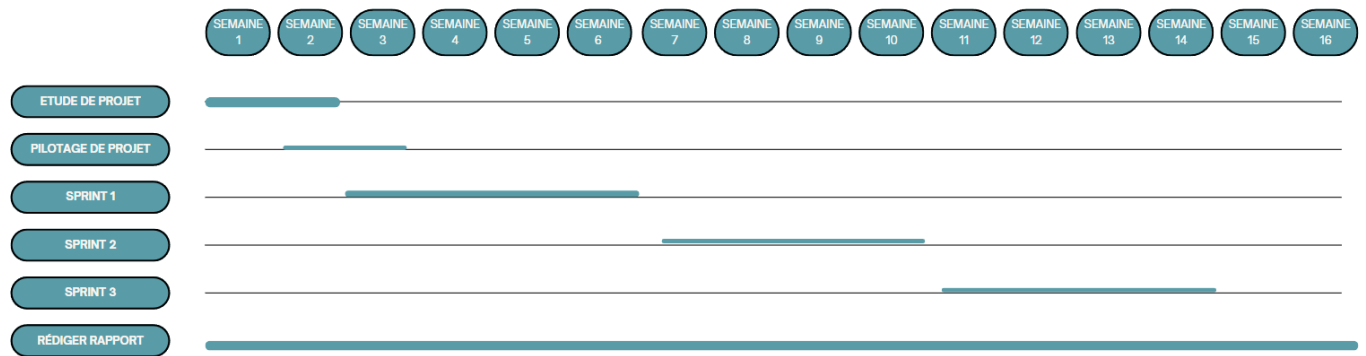


FIGURE 9 – Diagramme DE GANTT

## Conclusion

Dans ce chapitre, les acteurs, les besoins fonctionnels et non fonctionnels de l'application ont été identifiés. Le Backlog du produit ainsi que la planification des sprints et releases ont également été présentés. Enfin, les diagrammes de cas d'utilisation et les diagrammes de séquence système ont été détaillés. Le chapitre suivant sera consacré à la sprint 0 qui représente l'initialisation du projet.

---

## CHAPITRE III

# INITIALISATION TECHNIQUE DU PROJET AGILES

## Introduction

Ce chapitre décrira l'initialisation du projet. Dans un premier temps, l'architecture générale de l'application sera présentée. Ensuite, le choix de la méthodologie de base de données sera abordé. Par la suite, les frameworks de développement utilisés seront présentés. Enfin, les environnements matériels et logiciels du projet seront énumérés.

## III.1 Architecture de l'application

Cette section, présente l'architecture générale et l'architecture logicielle de notre application.

### III.1.1 Architecture générale de l'application

L'architecture offre une stack full-stack JavaScript unifiée, facilitant le développement rapide, la scalabilité et la gestion efficace des données en temps réel grâce à son approche asynchrone et orientée API. Elle permet aussi une meilleure modularité et réutilisabilité du code entre le frontend et le backend. La figure 10 illustre une représentation de l'architecture MERN.

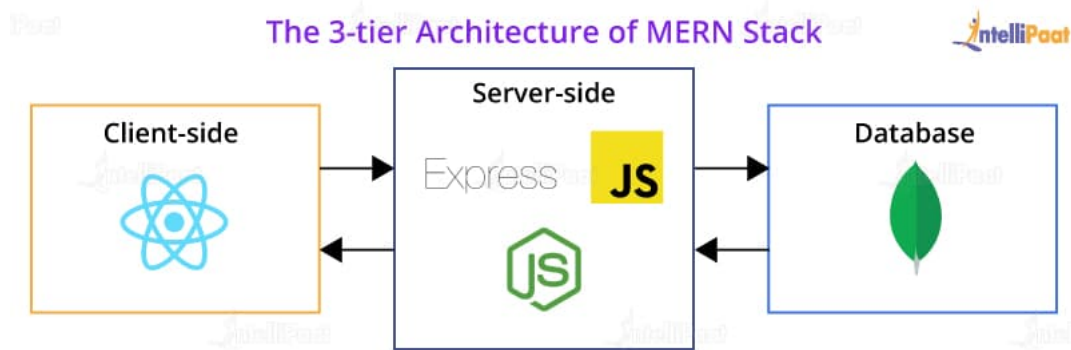


FIGURE 10 – Architecture MERN

### Fonctionnement de l'architecture MERN

- **Requête du client** : L'utilisateur interagit avec l'interface React.js dans son navigateur.
- **Requête au serveur** : React.js envoie des requêtes à l'API Express.js exécutée sur Node.js.
- **Traitement des données** : Express.js traite la requête, interagit avec la base de données MongoDB si nécessaire, et renvoie une réponse.
- **Mise à jour de l'interface** : React.js reçoit la réponse du serveur et met à jour l'interface utilisateur en conséquence.

### III.1.2 Architecture logicielle de l'application

L'architecture MVC sépare les préoccupations en trois couches distinctes, facilitant la maintenance, la réutilisabilité et l'évolutivité du code. Elle améliore aussi la collaboration entre développeurs et designers en structurant clairement le projet. La figure 11 illustre une représentation de l'architecture MVC.

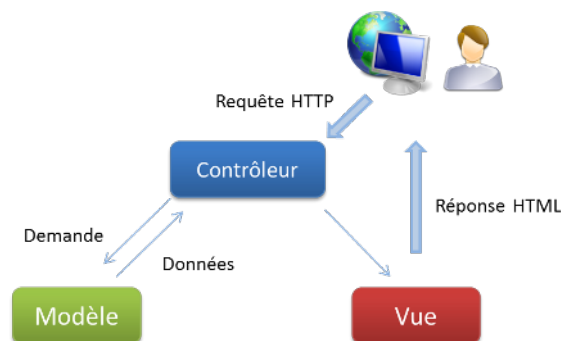


FIGURE 11 – Architecture MVC

L'architecture MVC (Model-View-Controller) est un modèle de conception logiciel qui sépare une application en trois composants principaux :



- **Modèle (Model) :** Représente la logique métier et la gestion des données. Il récupère, traite et stocke les informations (ex. : accès à une base de données).
- **Vue (View) :** Gère l'interface utilisateur et l'affichage des données fournies par le modèle. Elle se charge uniquement de la présentation sans contenir de logique métier.
- **Contrôleur (Controller) :** Sert d'intermédiaire entre le modèle et la vue. Il gère les requêtes de l'utilisateur, met à jour le modèle et sélectionne la vue à afficher.

## III.2 Les Frameworks de développement

Dans La partie suivante nous présentons les frameworks front-end et back-end que nous allons utiliser pour le développement de » l'application web.

### III.2.1 Framework Front-end :

- **React.js :** React JS est l'une des options populaires pour le développement front-end. On a fait ce choix grâce à ses performances, sa réutilisabilité de composants, sa simplicité avec JSX puisqu'il est la bibliothèque JavaScript, son large écosystème et son soutien de la communauté.



FIGURE 12 – Logo ReactJS.

### III.2.2 Framework Backend-end :

#### III.2.2.1 Framework

- **NodeJS** est un environnement d'exécution JavaScript côté serveur, basé sur le moteur V8 de Google Chrome. Il permet de créer des applications réseau rapides et évolutives grâce à son modèle asynchrone et non-bloquant. Il utilise un modèle événementiel pour gérer de nombreuses connexions simultanées efficacement. Node.js est largement utilisé pour le développement de serveurs web, d'APIs RESTful, ainsi que des applications en temps réel. Il bénéficie d'un riche écosystème grâce à son gestionnaire de paquets, npm, permettant l'utilisation de milliers de bibliothèques.

- **ExpressJS** est un Framework Node.js qui permet la mise en place rapide d'un serveur de requêtes http et qui offre une gestion des routes beaucoup plus simple et facile. Ainsi, il permet de gérer les routes dynamiques d'où on peut accéder et utiliser un ensemble des paramètres dans l'URL.



FIGURE 13 – Logo NodeJS et ExpressJS.

### III.2.2.2 Web service REST

REST "REpresentational State Transfer", créé en 2000 par Roy Fielding dans sa thèse [6] est un style architectural pour les systèmes hypermédia distribués inspiré de l'architecture du web basé sur le protocole HTTP. Il est utilisé dans le développement des applications RESTful orientées ressources ou orientées données. Une API RESTful, également appelée service Web RESTful, est une approche qui permet la communication entre les différentes parties du Web. L'échange est basé sur des requêtes client (partie front-end) et serveur (partie back-end). Il utilise explicitement les méthodes HTTP comme : POST pour créer une ressource sur le serveur, GET pour la récupérer, PUT pour changer l'état d'une ressource ou la mettre à jour et DELETE pour la supprimer. REST est préférée au protocole SOAP (Simple Object Access Protocol) car il utilise HTTP qui le rend plus facile à employer. Aussi, Il autorise une plus grande variété de formats de données (HTML, XML, JSON, etc.) ce qui lui offre une meilleure compatibilité avec les clients navigateurs. De plus, il est utilisé pour des services majeurs tels que Yahoo, Ebay, Amazon et même Google.

## III.3 Base de données NoSQL

Nous avons choisi MongoDB comme solution de stockage de données pour nos applications en raison de ses nombreux avantages :

1. **Flexibilité des données** : MongoDB permet de stocker différents types de données de manière intuitive grâce à sa structure basée sur des documents JSON.
2. **Scalabilité** : Il peut facilement gérer la croissance de nos applications en répartissant les données sur plusieurs serveurs.
3. **Performances** : MongoDB offre des performances élevées, garantissant des temps de réponse rapides pour les requêtes des utilisateurs.
4. **Écosystème complet** : Il dispose d'une large gamme d'outils et d'une communauté active

pour aider au développement et à la gestion des bases de données.



FIGURE 14 – Logo MongoDB.

## III.4 Environnement de travail

Nous détaillons dans le tableau III.1, en plus des frameworks présentés précédemment, nous présentons notre environnement matériel et les outils utilisés pour la modélisation ,design et les environnement de développement de l’application.

### III.4.1 Environnement matériel


Pour la réalisation du projet, nous avons utilisé un Pc portable avec les caractéristiques suivantes :

Caractéristique	Spécification
Système d’exploitation	Windows 11 64 bits
Processeur	Intel(R) Core(TM) i7-11800H 2.30GHz
Mémoire vive	16,0 Go
Disque dur	500 Go

TABLE 6 – caractéristique de pc

### III.4.2 Environnement de développement,design et modélisation

TABLE 7: Outils et leurs descriptions

Outil	Description
	L’application gratuite Draw.io permet de créer des diagrammes professionnels pour l’architecture technique et l’application web.

Outil	Description
	Visual Studio Code : Éditeur de code extensible (Microsoft) compatible Windows, Linux et macOS.
	Postman : Environnement de développement d'API pour tester et documenter des services web.
	GitHub : Plateforme d'hébergement de code (Git) pour collaboration et gestion de projets.
	Slack : Plateforme de collaboration avec chat, partage de fichiers et appels.
	Nodemailer : Bibliothèque JavaScript pour simplifier l'envoi d'e-mails.
	HTML : Langage de balisage standard pour structurer les pages web.
	CSS : Langage de style pour la mise en forme des éléments HTML.
	JavaScript : Langage pour l'interactivité des pages web (animations, gestion d'événements).

## Conclusion

Dans ce chapitre, nous avons présenté l'architecture de l'application ainsi que les Frameworks de développement et les autres outils. Le chapitre suivant sera consacré à la mise en œuvre du premier release de notre projet.

---

## CHAPITRE IV

### ÉTUDE ET RÉALISATION DU SPRINT 1

#### Introduction

Le but de ce chapitre est de présenter en détail la première release du projet sous le nom « Technicien ET Voitures », il contient le sprint numéro 1. Ce sprint comportera les fonctionnalités suivantes : l'inscription des techniciens, l'authentification, gestion des techniciens par le super admin, gestion des voitures par le gestionnaire et la gestion de la disponibilité des techniciens. L'étude du ce sprint couvre l'analyse, la conception et la réalisation de l'application.

#### IV.1 Backlog du sprint

Le Backlog du premier sprint présenté par le tableau contient une liste des tâches identifiées qui devront être réalisées avant la fin du sprint.

ID	Histoire	Estimation
1		Élevé
2		Moyen
3		Élevé
4		Élevé
5		Moyen
6		Faible
7		Faible

TABLE 8 – Backlog du sprint 1

## **IV.2 Diagramme de cas d'utilisation général**

### **Conclusion**

Conclusion partielle ayant pour objectif de synthétiser le chapitre et d'annoncer le chapitre suivant.

---

## CONCLUSION GÉNÉRALE

Rappel du contexte et de la problématique.

Brève récapitulation du travail réalisé et de la solution proposée.

La taille de la conclusion doit être réduite, une page de texte tout au plus. Il est important de souligner que la conclusion ne comporte pas de nouveaux résultats ni de nouvelles interprétations.

Le plus souvent, la conclusion comporte :

- un résumé très rapide du corps du texte ;
- un rappel des objectifs du projet ;
- un bilan professionnel qui indique clairement les objectifs annoncés dans l'introduction et en particulier ceux qui n'ont pu être atteints. Il présente et synthétise les conclusions partielles ;
- un bilan personnel qui décrit les principales leçons que vous tirez de cette expérience sur le plan humain ;
- les limites et les perspectives du travail effectué.

---

BIBLIOGRAPHIE

- [1] <https://www.planningpme.fr/>.
- [2] <https://www.praxedo.fr/>.
- [3] <https://www.woopit.fr/>.



## Annexe 1. Exemple d'annexe

Les chapitres doivent présenter l'essentiel du travail. Certaines informations-trop détaillées ou constituant un complément d'information pour toute personne qui désire mieux comprendre ou refaire une expérience décrite dans le document- peuvent être mises au niveau des annexes. Les annexes, **placées après la bibliographie**, doivent donc être numérotées avec des titres (Annexe1, Annexe2, etc.).

Le tableau annexe 1.1 présente un exemple d'un tableau dans l'annexe.

**Tableau annexe 1.1 :** Exemple tableau dans l'annexe

0	0
1	1
2	2
3	3
4	4

---

## **Annexe 2. Entreprise**

La figure annexe 2.1 présente le logo entreprise.

**Figure annexe 2.1 :** Logo d'entreprise

---

*arabicAbstract*

: arabicAbstractKeywords

## Résumé

3mm0cm *frenchAbstract*

**Mots clés :** frenchAbstractKeywords

## Abstract

*englishAbstract*

**Keywords :** englishAbstractKeywords