

Référence	Dép	TI
	AN	2025
	N°	

Rapport de
PROJET DE FIN D'ETUDES

En vue de l'obtention de :
Licence Appliquée en Développement des Systèmes d'Information

Développement d'une application de Gestion des Appels d'Offres

Elaboré par :
Nourhen Jlassi

&

Nebras Soltani

Encadré par :

Mme Afef Ferjaoui (ISET)

Mr Wassim Mhamdi (ENTREPRISE)

Effectué à :

Entreprise : Siga

- **Adresse : 13 Rue Ibn Nafis, Les berges du Lac 3, ZI Kheireddine, Tunis**
- **Mail : contact@siga.com.tn**

Année universitaire : 2024/2025

TABLE DES MATIÈRES

Liste des Abréviations	6
Introduction Générale	7
1 Présentation du cadre du projet	9
1.1 Introduction	9
1.2 Présentation de l'organisme d'accueil	9
1.2.1 SIGA	9
1.2.2 Activité	10
1.2.3 Organigramme	11
1.3 Etude préalable	12
1.3.1 Etude de l'existant	12
1.3.2 Critique de l'existant	12
1.3.3 Solution proposée	12
1.4 Méthodologie de modélisation et de conception	13
1.5 Méthodologie de gestion du projet	14
1.5.1 Pourquoi Scrum ?	14

1.5.2	Les Trois Piliers de Notre Gestion de Projet	14
1.5.3	Les Éléments Clés de Scrum	15
1.6	Environnement de travail	16
1.6.1	Environnement matériel	16
1.6.2	Environnement logiciel	16
1.6.3	Langages de programmation	18
1.6.4	Frameworks	20
1.6.5	Système de gestion de base de données	21
1.7	Conclusion	22
2	Analyse et spécifications des besoins	23
2.1	Introduction	23
2.2	Planification de projet	23
2.2.1	Identification des acteurs	23
2.2.2	Les besoins fonctionnels	24
2.2.3	Les besoins non fonctionnels	26
2.3	Organisation et planification du projet avec Scrum	27
2.3.1	Equipe Scrum	27
2.3.2	Product Backlog	27
2.3.3	Planification des Sprints	30
2.4	Architecture logicielle adoptée	31
2.4.1	La couche de présentation	31
2.4.2	La couche applicative	33
2.4.3	La couche accès aux données	33
2.5	Conclusion	33

3	Release 1 :	34
----------	--------------------	-----------

TABLE DES FIGURES

1.1	Organigramme de la société SIGA	10
1.2	Solutions SIGA	11
1.3	Organigramme SIGA	11
1.4	Le processus de Scrum	15
2.1	Equipe Scrum	27
2.2	Planification des Sprints	30
2.3	L'architecture n-tiers	31

LISTE DES TABLEAUX

1.1	Spécifications des postes de développement	16
2.1	Product Backlog	28
2.2	Product Backlog - Partie 2	29
2.3	Product Backlog - Partie 3	30

LISTE DES ABRÉVIATIONS

SIGA :Système Informatique et Gestion Automatisée

INTRODUCTION GÉNÉRALE

La gestion des appels d'offres représente un défi stratégique pour les entreprises et les organisations, qui doivent concilier efficacité, transparence et précision dans le traitement des soumissions et des décisions. Dans un contexte où les processus manuels et dispersés engendrent des pertes de temps, des erreurs potentielles et un manque de visibilité pour les parties prenantes, la nécessité d'une solution centralisée et automatisée devient impérative.

Dans le cadre de notre projet de fin d'études réalisé au sein de l'entreprise SIGA, nous avons entrepris de concevoir et de développer une plateforme innovante dédiée à la gestion des appels d'offres, intégrant des fonctionnalités avancées telles que le reporting dynamique via JasperReports et une application mobile pour le suivi des offres. Cette solution vise à optimiser les processus, renforcer la sécurité des données et offrir une expérience utilisateur fluide et intuitive.

Ce rapport a pour objectif de présenter les différentes étapes de la réalisation de notre projet. Il s'articule autour de quatre chapitres principaux :

Dans le premier chapitre, « Cadre général du projet », nous introduisons l'organisme d'accueil, SIGA, ainsi que la problématique adressée. Nous détaillons ensuite la solution proposée et les méthodologies adoptées pour mener à bien le projet.

Dans le deuxième chapitre, « Analyse et spécifications des besoins », nous exposons les besoins fonctionnels et non fonctionnels de la plateforme, accompagnés de l'architecture logicielle conçue pour répondre à ces exigences.

Dans le troisième chapitre, « Release 1 : Développement du cœur de la plateforme », nous présentons la première phase de développement, incluant la gestion des appels d'offres, des soumissions et des utilisateurs.

Dans le quatrième chapitre, « Release 2 : Intégration avancée et reporting dynamique », nous abordons les fonctionnalités avancées, telles que l'intégration de JasperReports pour les rapports dynamiques et le développement de l'application mobile.

Pour chaque release, nous détaillons les sprints réalisés, en couvrant les phases d'analyse, de conception et de mise en œuvre. Enfin, nous concluons ce rapport par une synthèse des résultats obtenus et des perspectives d'évolution envisagées pour notre plateforme.

CHAPITRE 1

PRÉSENTATION DU CADRE DU PROJET

1.1 Introduction

Ce premier chapitre comportera plusieurs parties dont on peut citer l'organisme d'accueil, la problématique du projet, la solution proposée ainsi que notre méthodologie adoptée pour le développement de ce projet.

1.2 Présentation de l'organisme d'accueil

1.2.1 SIGA

dans l'ingénierie logicielle. Elle exerce ses activités auprès d'une clientèle variée, principalement constituée de grandes entreprises dans différents secteurs tels que les banques, les assurances, les administrations, les transports, les télécommunications, les organismes sociaux, les secteurs de la santé et l'industrie, pour n'en citer que quelques-uns.



FIGURE 1.1 – Organigramme de la société SIGA

1.2.2 Activité

SIGA se positionne principalement sur deux domaines d'activités stratégiques :

- L'édition des progiciels de gestion performants et évolutifs.
- La conception et la réalisation de systèmes informatiques spécifiques.

SIGA propose une gamme variée de solutions logicielles intégrées :

- **SIGA'ERP** : Intégrant les différents domaines d'une entreprise, SIGA rassemble ces derniers au sein d'une même base de données relationnelle.
- **SIGA'GTC** : SIGA assure la gestion technique et l'analyse de la situation des caisses de sécurité sociale.
- **SIGA'AM** : Facilite les démarches et procédures d'assurance maladie.
- **SIGA'CRM** : Assure la gestion commerciale et la gestion des relations clients.
- **SIGA'GMAO** : Gère la maintenance assistée par ordinateur.
- **SIGA'GPAO** : Garantit la production assistée par ordinateur.
- **SIGA'GED** : Numérise et archive les documents.
- **Business Intelligence** : Assure l'aide à la prise de décision.
- **Workflow** : Gère le travail collaboratif.



FIGURE 1.2 – Solutions SIGA

1.2.3 Organigramme

Nous effectuons notre stage au sein de la direction technique, plus spécifiquement au sein de la division Recherche et Développement des services.

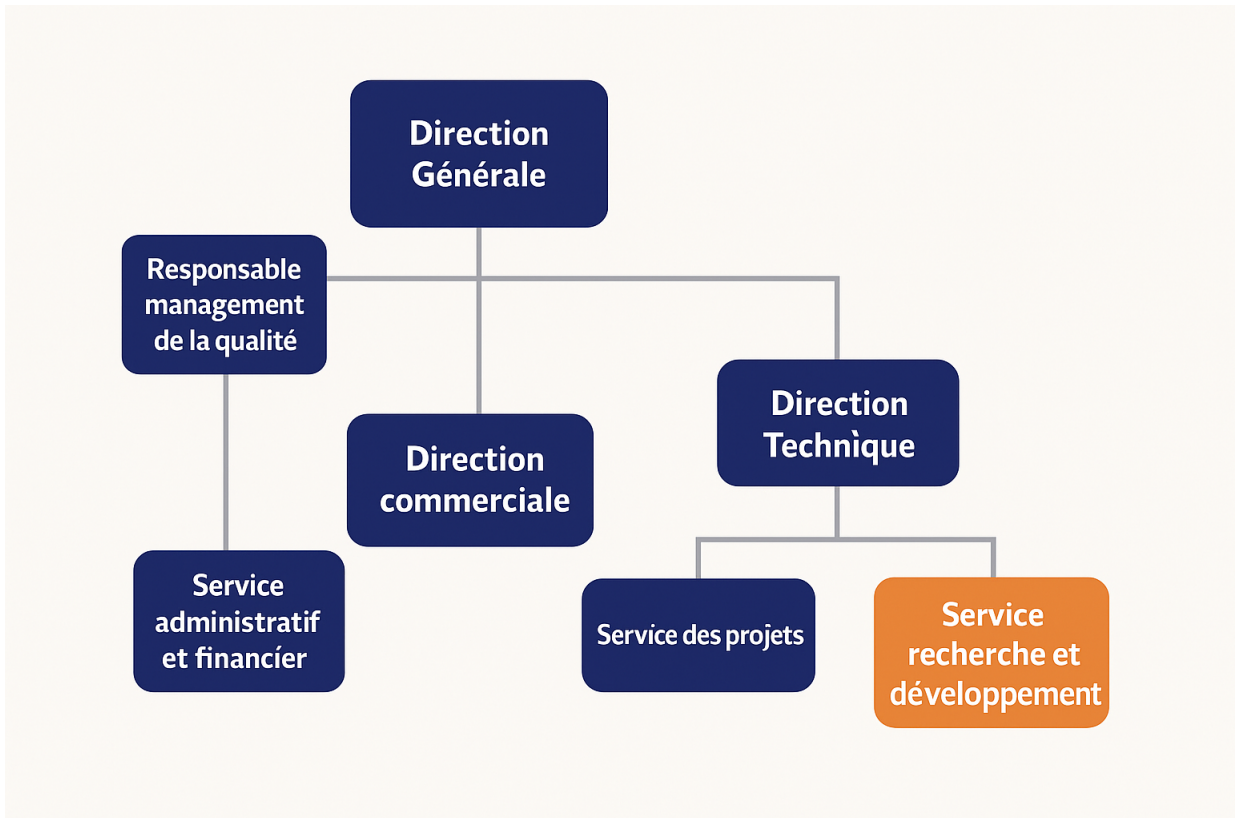


FIGURE 1.3 – Organigramme SIGA





1.3 Etude préalable

1.3.1 Etude de l'existant

La gestion des appels d'offres chez SIGA s'effectue actuellement de manière manuelle, en s'appuyant sur des outils bureautiques classiques comme les tableurs, documents texte et courriels. Chaque étape — de la rédaction à l'analyse des offres — est prise en charge par les services concernés selon des procédures internes. Les offres reçues sont traitées et évaluées manuellement à l'aide de grilles internes, puis archivées sous format numérique ou papier. Le suivi des soumissions et la communication avec les soumissionnaires se font par e-mail ou téléphone, en fonction des usages en vigueur. Ce fonctionnement repose sur une organisation en suivant des règles internes, avec un traitement structuré des dossiers d'appel d'offres tout au long du processus.

1.3.2 Critique de l'existant

le système actuel de gestion des appels d'offres chez SIGA présente plusieurs limites qui impactent son efficacité et sa performance globale :

-  **Risque d'erreurs et lenteur des opérations** : L'utilisation de méthodes manuelles pour la saisie des données rend les tâches chronophages et multiplie les risques d'erreurs humaines, ce qui nuit à la fluidité et à la fiabilité du processus.
-  **Manque de visibilité décisionnelle** : L'absence d'outils d'analyse ou de reporting rend difficile l'extraction d'informations claires et structurées, limitant ainsi la capacité des décideurs à agir efficacement.
-  **Traçabilité limitée** : Il est complexe, voire impossible, de suivre avec précision l'historique des actions effectuées (modifications, dépôts d'offres, validations...), ce qui nuit à la transparence et à l'auditabilité du système.
-  **Expérience utilisateur peu engageante** : L'absence d'interface ergonomique et intuitive pénalise aussi bien les soumissionnaires que les gestionnaires, rendant l'interaction avec le système peu agréable et parfois confuse.

Dans ce contexte, il est nécessaire de trouver une solution sécurisée, fiable et ergonomique pour la gestion de réorientation universitaire

1.3.3 Solution proposée

Pour répondre efficacement aux limites identifiées, nous proposons le développement d'une plateforme intégrée, accessible via le web et le mobile, offrant une expérience fluide et complète

pour tous les utilisateurs.

- Côté backend (Spring Boot), le système assurera une gestion centralisée des appels d'offres, incluant la création, la modification, la validation ainsi que le suivi des soumissions. Chaque offre sera horodatée et son statut mis à jour automatiquement (en attente, acceptée, refusée). La sécurité sera un pilier central, avec une authentification basée sur JWT et une gestion fine des rôles et permissions.
- Sur l'interface web (Angular), un tableau de bord interactif permettra aux administrateurs de visualiser en temps réel l'ensemble des activités. Des filtres de recherche avancés faciliteront l'accès rapide aux appels d'offres pertinents.
- L'application mobile (Ionic/Angular) offrira un espace dédié aux clients, leur permettant de déposer leurs offres et de suivre leur évolution à tout moment, depuis leur smartphone.
- Enfin, un module de reporting dynamique, basé sur JasperReports, permettra la génération automatique de rapports détaillés (statistiques, tendances, etc.), avec la possibilité d'exporter les données en formats PDF ou Excel pour des besoins d'analyse stratégique.

1.4 Méthodologie de modélisation et de conception

Une méthode de modélisation et de conception est un processus qui permet de formaliser les étapes préliminaires du développement d'un système afin de rendre ce développement plus fidèle aux besoins du client lors de la réalisation d'un projet informatique. Pour cette raison, nous avons opté pour UML pour la conception de notre projet.

UML (Unified Modeling Language) est un langage de modélisation graphique à base de pictogrammes conçu comme une méthode normalisée de visualisation dans les domaines du développement logiciel et en conception orientée objet [1].



1.5 Méthodologie de gestion du projet

Pour garantir la réussite de notre projet de **Plateforme de Gestion des Appels d’Offres**, nous avons opté pour une approche agile basée sur **Scrum**. Ce choix s’impose naturellement car il permet de concilier **flexibilité**, **collaboration** et **livraison rapide de valeur**, tout en plaçant le client au cœur du processus de développement.

1.5.1 Pourquoi Scrum ?

Scrum est bien plus qu’une simple méthodologie – c’est un **état d’esprit**. Inspiré du rugby (où “scrum” signifie “mêlée”), il met l’accent sur :

- **Des livraisons fréquentes** (tous les 2 à 4 semaines) pour obtenir rapidement un produit fonctionnel
- **Une adaptation continue** aux besoins changeants du client
- **Une transparence totale** entre toutes les parties prenantes

1.5.2 Les Trois Piliers de Notre Gestion de Projet

Afin d’assurer une gestion efficace et agile de notre projet, nous nous sommes appuyés sur trois piliers fondamentaux : la transparence, l’inspection et l’adaptation. Ces principes nous ont permis de garantir une organisation claire, une amélioration continue et une réactivité face aux besoins évolutifs du client.

1. **Transparence** La transparence favorise une communication fluide au sein de l’équipe et avec les parties prenantes. Pour cela, nous avons utilisé Jira pour le suivi des tâches, un Product Backlog partagé et mis à jour régulièrement, ainsi que des tableaux de bord visuels pour suivre l’avancement des sprints.
2. **Inspection** Chaque sprint se termine par une revue permettant de valider les livrables avec le client, suivie d’une rétrospective d’équipe afin d’identifier les points à améliorer et d’optimiser notre méthode de travail.
3. **Adaptation** Grâce à une réévaluation régulière des priorités entre les sprints et une grande flexibilité face aux retours clients, nous avons pu adapter efficacement notre plan d’action tout au long du projet

1.5.3 Les Éléments Clés de Scrum

La méthodologie Scrum repose sur trois piliers essentiels : les artefacts, les cérémonies, et les rôles, qui garantissent une organisation agile, itérative et centrée sur la valeur ajoutée.

Les Artefacts

- ✓ **Product Backlog** : Liste évolutive et priorisée des fonctionnalités à développer (ex. : Créer un formulaire de soumission d'offres).
- ✓ **Sprint Backlog** : Ensemble des tâches sélectionnées pour le sprint en cours.
- ✓ **Incrément Produit** : Version livrable du produit à la fin de chaque sprint.

Les Cérémonies

- ✓ **Sprint Planning** : Définition des objectifs et des tâches du sprint.
- ✓ **Daily Scrum (15 min)** : Réunion quotidienne centrée sur trois questions clés : Ce que j'ai fait hier ? Ce que je vais faire aujourd'hui ? Quels obstacles je rencontre ?
- ✓ **Sprint Review** : Présentation du livrable au client et recueil de feedback.
- ✓ **Sprint Retrospective** : Analyse de l'organisation et identification d'axes d'amélioration.

Les Rôles

- ✓ **Product Owner** : Représente les besoins du client et gère le backlog.
- ✓ **Scrum Master** : Facilite l'application de Scrum et aide l'équipe à surmonter les obstacles.
- ✓ **Équipe de Développement** : Pluridisciplinaire et auto-organisée, elle assure la réalisation des tâches du sprint.

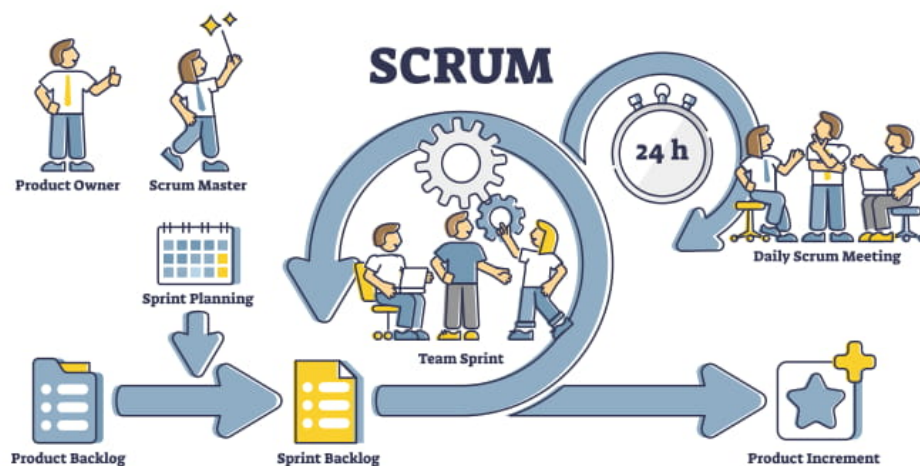


FIGURE 1.4 – Le processus de Scrum

1.6 Environnement de travail

Avant de se lancer dans l'implémentation de notre projet, nous allons d'abord décrire l'environnement et les outils de travail que nous utiliserons. Nous commencerons par définir l'environnement matériel, suivi de l'environnement logiciel. Enfin, nous présenterons les différents langages et frameworks que nous utiliserons dans le cadre de ce projet.

1.6.1 Environnement matériel

Pour mener à bien le développement de notre plateforme, nous avons utilisé deux ordinateurs aux caractéristiques bien adaptées à nos besoins. Ces machines, un Asus et un Lenovo, nous ont permis de travailler efficacement, que ce soit pour coder, tester ou compiler notre projet. Le tableau suivant détaille les spécifications techniques de ces postes de travail, offrant un aperçu clair des ressources matérielles qui ont soutenu notre travail au quotidien.

TABLE 1.1 – Spécifications des postes de développement

Composant	PC1 (Asus)	PC2 (Lenovo)
Processeur	Intel Core i5-1135G7	Intel Core i5-1135G7
Mémoire RAM	16 Go DDR4	8 Go DDR4
Stockage	237 Go SSD + 1 To HDD	476 Go SSD
Système d'exploitation	Windows 11 Pro	Windows 11 Pro
Carte graphique	NVIDIA GeForce	Intel Iris Xe

1.6.2 Environnement logiciel

La réalisation d'une application que ce soit web ou mobile nécessite l'utilisation de plusieurs logiciels. C'est pour cela, nous énumérons dans cette section les logiciels que nous avons utilisés lors du développement de notre plateforme.



Visual Studio Code

Visual Studio Code (VSCode) est un éditeur de code source et un environnement de développement intégré (IDE) de Microsoft. Il est open-source et cross-platform, c'est-à-dire qu'il fonctionne sur Windows, Linux et Mac. [2].



Postman

Postman est un logiciel permettant de créer et de tester des requêtes HTTP. Il vous permet de les personnaliser dans les plus fins détails grâce à une interface ergonomique et intuitive.[3].



Apidog Apidog est un outil complet destiné aux développeurs pour la création, la documentation, et le test d'API. Il simplifie le processus de développement d'API en offrant une interface conviviale pour concevoir les endpoints, générer automatiquement la documentation, et exécuter des tests[4].



JasperSoft Jaspersoft Studio est l'outil de développement de rapports pour JasperReports. Il simplifie le processus de création des fichiers JRXML qui décrivent les requêtes et la mise en page du rapport en présentant les fonctionnalités les plus puissantes de JasperReports. Jaspersoft Studio peut servir à développer et tester des rapports sur votre bureau local et à se connecter à JasperReports Server afin de déployer des rapports ou bien de consulter les rapports existants au sein du référentiel JasperReports Server.[5].



Miro est un espace de travail interactif qui facilite le processus de brainstorming et garantit une synchronisation totale entre les participants.ensemble.[6].



Github Desktop GitHub Desktop est une application open source gratuite qui vous permet d'utiliser des fichiers hébergés sur GitHub ou d'autres services d'hébergement Git.[7].



Overleaf Un éditeur LaTeX en ligne facile à utiliser. Pas d'installation, collaboration en temps réel, gestion des versions, des centaines de modèles de documents LaTeX, et plus encore. [8].



Draw io Draw.io est une application gratuite en ligne, accessible via son navigateur (protocole https) qui permet de dessiner des diagrammes ou des organigrammes. [9].

1.6.3 Langages de programmation

A propos des langages de programmation nous avons utilisé les langages suivants :

HTML



HTML

HTML est un langage de balisage standard pour la création d'applications Web et de sites Web. C'est un système bien standardisé utilisé pour marquer les fichiers texte afin d'obtenir des couleurs, des graphiques, des polices et des effets sur les pages Web. C'est un langage en constante évolution.[10].

CSS



CSS

CSS, appelé aussi Cascading Style Sheets, est un langage de feuille de style utilisé pour gérer la présentation d'une page Web contenant du code HTML.[11].

JS



JavaScript

JavaScript, souvent appelé JS, est un langage de programmation interprété dynamique de haut niveau qui permet aux scripts côté client de créer des applications Web et des sites Web entièrement dynamiques.[12].



Latex

LaTeX se présente sous la forme d'un langage informatique de balisage. On compose donc un document LaTeX en écrivant son code source au moyen d'un éditeur de texte. Ce code source est ensuite traité par le compilateur de LaTeX pour produire le document mis en forme.[13].



Java

Java est un langage de programmation très répandu, expressément conçu pour coder des applications et des services utilisés dans l'environnement distribué de l'internet.[14].

1.6.4 Frameworks

Comme framework, nous avons utilisé :



SpringBoot

CSDN @我是一盘牛肉

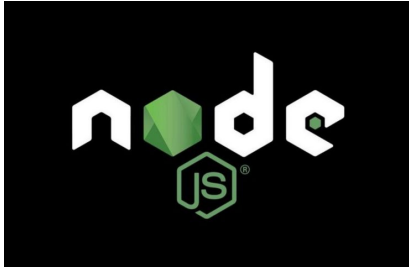
Spring Boot

Java Spring Boot (Spring Boot) est un outil qui accélère et simplifie le développement d'applications Web et de microservices avec Spring Framework .[15].



Angular

Angular est un Framework open source écrit en JavaScript qui permet la création des applications web accessibles via une page web unique qui permet de fluidifier l'expérience utilisateur et d'éviter les chargements de pages à chaque nouvelle action.[16].



Node JS

Node.js est un environnement d'exécution single-thread, open-source et multiplateforme permettant de créer des applications rapides et évolutives côté serveur et en réseau. [17].



Typescript

TypeScript est un langage de programmation développé par Microsoft en 2012. Son ambition principale est d'améliorer la productivité de développement d'applications complexes.[18].



Ionic

Ionic est un framework libre d'utilisation qui permet de créer des applications mobiles pour iOS, Android et Windows Phone, à partir d'une base de code unique. En d'autres termes, Ionic est un outil de développement mobile multiplateforme.[19].

1.6.5 Système de gestion de base de données

Le SGBD que nous avons utilisé est



mysql

MySQL est un système de gestion de base de données relationnelles (SGBDR). Il s'agit d'un SGBDR open-source développé et supporté par Oracle, le leader mondial de la base de données. MySQL est aujourd'hui un des SGBDR les plus utilisés dans le monde.[20].

1.7 Conclusion

Dans ce chapitre, nous avons posé les bases de notre projet de fin d'études en introduisant l'organisme qui nous accueille, en explorant l'étude initiale, en détaillant les approches méthodologiques choisies et en décrivant l'environnement dans lequel nous évoluons. Dans le prochain chapitre, nous allons nous plonger dans l'analyse et la définition des besoins de notre système, pour mieux en cerner les attentes et les exigences.

CHAPITRE 2

ANALYSE ET SPÉCIFICATIONS DES BESOINS

2.1 Introduction

Ce chapitre présente l'analyse et la spécification des besoins du projet, étape clé pour garantir une conception précise et alignée avec les attentes. Nous y identifions d'abord les acteurs et leurs besoins, puis détaillons les exigences fonctionnelles et non fonctionnelles. Ensuite, nous exposons le product backlog et le diagramme de cas d'utilisation, avant de conclure par une vue d'ensemble de l'architecture logicielle. Cette démarche structurée assure une base solide pour le développement.

2.2 Planification de projet

Nous procéderons tout d'abord à l'identification des acteurs du système, avant d'établir une analyse détaillée des besoins fonctionnels et non fonctionnels.

2.2.1 Identification des acteurs

Pour bien comprendre comment notre plateforme de gestion des appels d'offres va prendre vie, commençons par identifier les acteurs qui vont interagir avec elle. Un acteur, c'est une personne ou une entité qui joue un rôle dans le système, un peu comme les pièces d'un puzzle qui s'assemblent pour former une image complète. Voici les acteurs principaux de notre application :

- **L'administrateur** : Cet acteur possède tous les droits administratifs sur le système. Il est responsable de la gestion des utilisateurs (modification, suppression des comptes) et de l'attribution des rôles . Il traite également les nouveaux comptes utilisateurs et gère les permissions d'accès. Il peut générer des rapports dynamiques sur les appels d'offres en utilisant Jasper Reports.
- **Le client** : Cet acteur représente l'entité ou la personne qui initie les appels d'offres. Il utilise la plateforme pour publier des projets, consulter les soumissions reçues et suivre l'avancement des offres. Le client est au cœur du processus, car c'est lui qui définit les besoins et attend des réponses pertinentes.
- **Le soumissionnaire** : Cet acteur a pour tâche de soumettre des offres en réponse aux appels d'offres disponibles. Il peut modifier ses offres soumises, consulter les détails de ses soumissions et suivre l'état de ses offres. Il reçoit également des notifications par email lorsqu'un nouvel appel d'offres est publié dans son secteur d'activité.

2.2.2 Les besoins fonctionnels

Notre plateforme de gestion des appels d'offres est conçue pour simplifier et fluidifier les échanges entre les entreprises, les clients et les soumissionnaires, en apportant une solution moderne et centralisée. Elle vise à rendre le processus de création, de suivi et d'évaluation des appels d'offres plus transparent et efficace, tout en offrant une expérience intuitive pour tous ses utilisateurs. Dans cette section, nous allons détailler les besoins fonctionnels du système en les organisant par acteur, pour montrer comment chaque rôle interagit avec la plateforme et contribue à faire vivre ce projet.

Admin peut :

- S'authentifier
- Consulter son profil
- Générer un rapport
 - Sélectionner des critères
 - * Filtrer par status
 - * Filtrer par dates
 - Consulter la liste des utilisateurs
 - Traiter Les comptes
 - Modifier un rôle
 - Supprimer un rôle

- Consulter les appel d’offre en attente
 - Traiter un appel d’offre
 - * Approuver la publication de l’AO
 - * Refuser un appel d’offre
- Consulter la Liste des AO valider
 - Selectionner un secteur d’activité
 - * Imprimer le(s) docutment(s)

Client peut :

- S’authentifier
- Consulter son profil
- Générer un rapport
- Consulter la liste des appel d’offre
 - Filtrer par status
 - Filtrer par secteur d’activité
 - Filtrer par catégorie
- Consulter ses appels d’offres
 - Créer un AO
 - Publier un appel d’offre
 - * Imprimer le(s) docutment(s)
 - Modifier un appel d’offre
 - Supprimer un appel d’offre
 - consulter l’historique des modifications d’un AO
- Consulter les détails d’un appel d’offre
 - Consulter la liste des soumissions
 - * Traiter une soumission
 - Accepter une soumission
 - Refuser une soumission

Soumissionnaire peut :

- S’authentifier

- Consulter son profil
- Générer un rapport
- Consulter la liste des appel d'offre
 - Filtrer par status
 - Filtrer par secteur d'activité
 - Filtrer par catégorie
- Recevoir une notification par email d'un nouvel appel d'offre
- Participer à l'appel d'offre
 - Imprimer les fichiers de l'appel d'offre
 - Consulter l'historique des modifications d'un appel d'offre
 - Déposer une soumission
 - Proposer des questions concernant l'appel d'offre
- Consulter ses soumissions
 - Ajouter une soumission
 - Modifier une soumission
 - Supprimer une soumission

2.2.3 Les besoins non fonctionnels

Si les besoins fonctionnels décrivent ce que notre plateforme de gestion des appels d'offres doit faire, les besoins non fonctionnels se concentrent sur la manière dont elle doit le faire, en mettant l'accent sur la qualité et l'efficacité. Ils sont essentiels pour offrir une expérience fluide et sécurisée à tous les utilisateurs, qu'il s'agisse des administrateurs, des clients, des soumissionnaires ou des visiteurs. Voici les principaux aspects non fonctionnels de notre application :

- **Sécurité** : La plateforme doit protéger les données sensibles, comme les informations des utilisateurs et les détails des appels d'offres, en utilisant des méthodes de stockage sécurisées et une authentification robuste pour empêcher tout accès non autorisé.
- **Performance** : L'application doit être rapide et réactive, même avec un grand nombre d'utilisateurs connectés simultanément, pour garantir une gestion efficace des appels d'offres sans ralentissements.
- **Ergonomie** : L'interface doit être intuitive et agréable, permettant à tous les utilisateurs, peu importe leur niveau de compétence technique, de naviguer facilement et de gérer leurs tâches sans frustration.
- **Extensibilité** : Le système doit être conçu de manière modulaire, afin de pouvoir intégrer de nouvelles fonctionnalités à l'avenir, comme des outils d'analyse avancée ou des notifications supplémentaires, sans nécessiter une refonte complète.

2.3 Organisation et planification du projet avec Scrum

2.3.1 Equipe Scrum

Comme mentionné plus tôt, notre projet s'appuie sur la méthodologie Scrum pour structurer notre travail et garantir une collaboration fluide entre tous les membres de l'équipe. Scrum repose sur trois rôles clés, qui sont comme les piliers de notre organisation :

- ★ Product Owner
- ★ Scrum Master
- ★ Équipe de développement

La figure ci-dessous illustre comment ces rôles s'entrelacent pour faire avancer notre projet.

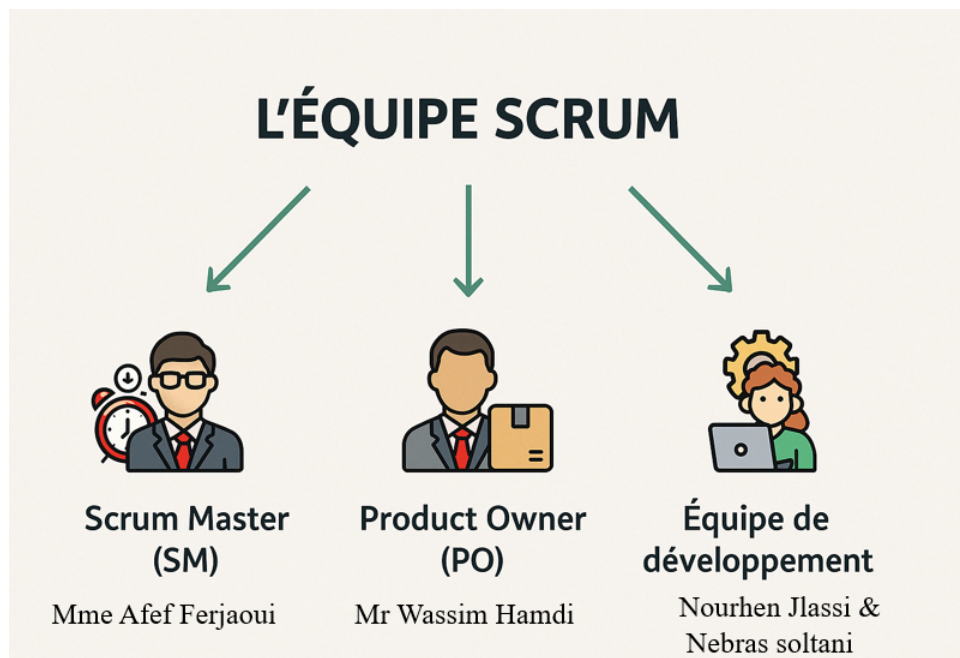


FIGURE 2.1 – Equipe Scrum

2.3.2 Product Backlog

Le product backlog est une liste ordonnée et constamment ajustée de tout ce qui est requis pour faire progresser le produit. Il évolue avec le projet, guidant l'équipe vers les priorités essentielles.

Les figures ci-dessous représente le product backlog de notre système

TABLE 2.1 – Product Backlog

ID	Thème	User Stories	Priorité	Sprint
0	Formation aux frameworks et préparation de l'environnement de développement.	En tant que développeur, je veux apprendre à créer une API REST simple avec Spring Boot.	Haute	Sprint 0
		En tant que développeur, je veux mettre en place des projets Angular et Ionic de base.	Haute	
		En tant que membre de l'équipe, je veux générer un rapport PDF avec JasperReports.	Haute	
		En tant qu'équipe, nous voulons installer et configurer tous les outils de développement nécessaires.	Haute	
		En tant que membre de l'équipe, je veux préparer un modèle LaTeX pour notre rapport de PFE.	Haute	
1	Gestion d'un appel d'offre	En tant que client, je veux créer un appel d'offre pour publier de nouvelles opportunités.	Haute	Sprint 1
		En tant que client, je veux modifier un appel d'offre pour corriger des erreurs ou mettre à jour les informations.	Haute	
		En tant que client, je veux supprimer un appel d'offre.	Haute	
		En tant que client, je veux consulter mes appels d'offres pour voir ceux que j'ai créés.	Haute	
		En tant que client, je veux consulter les détails d'un appel d'offre pour voir toutes les informations nécessaires.	Haute	
2	Traitement d'un appel d'offre	En tant qu'admin, je veux traiter un appel d'offre en l'approuvant pour publication ou en le refusant, afin de contrôler sa visibilité pour les soumissionnaires.	Haute	Sprint 1

TABLE 2.2 – Product Backlog - Partie 2

ID	Thème	User Stories	Priorité	Sprint
3	Gestion des soumissions	En tant que soumissionnaire, je veux soumettre une offre avec tous les documents nécessaires pour répondre à un appel d'offres.	Haute	Sprint 2
		En tant que soumissionnaire, je veux modifier une offre soumise pour mettre à jour les informations.	Haute	
		En tant que soumissionnaire, je veux consulter les détails d'une offre soumise afin de suivre son avancement.	Haute	
4	Traitement des soumissions	En tant que client, je veux accepter ou refuser une soumission pour traiter les offres reçues.	Haute	Sprint 2
		En tant que client, je veux évaluer les sou FAIRISSIONS selon des critères pour faciliter la sélection.	Moyenne	
5	Gestion des utilisateurs	En tant qu'utilisateur, je veux créer un compte et m'authentifier de manière sécurisée pour accéder aux services.	Haute	Sprint 3
		En tant qu'administrateur, je veux modifier les utilisateurs pour contrôler l'accès au système.	Haute	
		En tant qu'admin, je veux traiter les nouveaux comptes utilisateurs en leur attribuant un rôle.	Haute	
		En tant qu'admin, je veux pouvoir supprimer un compte si nécessaire.	Moyenne	
6	Génération des rapports	En tant qu'Admin, je veux générer un rapport dynamique sur les appels d'offres en utilisant JasperReports.	Haute	Sprint 4
7	Système de notification	En tant que Soumissionnaire, je veux recevoir une notification par email pour nouveaux appels d'offres.	Moyenne	Sprint 4
		En tant que Soumissionnaire, je veux être notifié des changements d'état de ma soumission.	Moyenne	

TABLE 2.3 – Product Backlog - Partie 3

8	Recherche et filtrage	En tant qu'utilisateur, je veux consulter la liste des appels d'offres avec des filtres de recherche.	Faible	Sprint 5
9	Gestion documentaire	En tant qu'administrateur, je veux consulter et télécharger les documents par catégorie.	Moyenne	Sprint 5
10	Archivage automatique	En tant qu'admin, je veux que le système archive automatiquement les AO après 10 jours.	Haute	Sprint 5

2.3.3 Planification des Sprints

La planification des sprints est une étape clé dans la gestion d'un projet Scrum, car elle donne le rythme à notre travail. Pour notre projet, nous avons structuré le développement en 2 grandes releases, qui se décomposent en 10 sprints au total. La figure qui suit illustre comment ces releases se répartissent en sprints, pour mieux visualiser notre feuille de route.

**FIGURE 2.2** – Planification des Sprints

2.4 Architecture logicielle adoptée

Pour la réalisation de notre application, nous avons adopté une architecture n-tiers, qui sépare clairement les responsabilités en trois couches principales : la couche de présentation, la couche applicative et la couche d'accès aux données. Cette structure garantit une modularité, une maintenabilité et une évolutivité optimales. L'application est développée avec Spring Boot et Angular pour la partie web, ainsi qu'Ionic avec Capacitor pour la partie mobile, assurant une expérience utilisateur cohérente sur différentes plateformes.

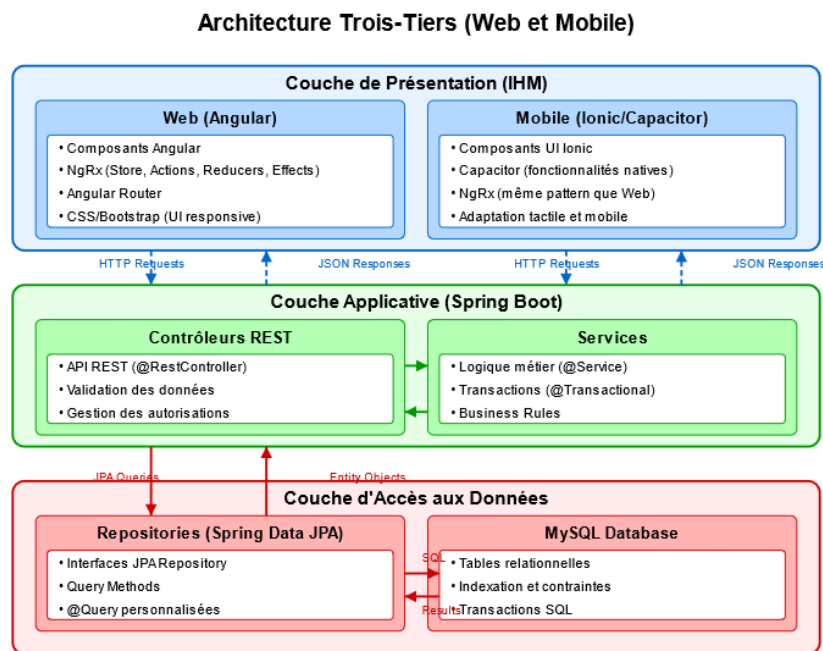


FIGURE 2.3 – L'architecture n-tiers

2.4.1 La couche de présentation

La couche de présentation, aussi appelée IHM (Interface Homme-Machine), est l'élément central qui permet à l'utilisateur d'interagir avec l'application. Elle gère tout ce qui touche aux actions de l'utilisateur, comme les clics ou les saisies au clavier, et affiche les informations de manière claire et agréable.

Partie web : Angular

Pour la version web, nous avons choisi **Angular**, un framework JavaScript robuste qui organise l'interface en composants réutilisables. Angular suit le modèle MVC (Modèle-Vue-Contrôleur), où la "vue" représente l'interface utilisateur. Voici les principales caractéristiques

de cette couche :

- **Gestion de l'état** : Nous utilisons **NgRx**, une bibliothèque inspirée de Redux, pour centraliser l'état global de l'application. Avec NgRx, toutes les données sont stockées dans une *store* unique, accessible par tous les composants, ce qui rend la gestion des interactions utilisateur simple et fluide.
 - *Store* : Contient les données globales, comme les informations sur l'utilisateur ou les réservations.
 - *Actions* : Représentent des événements, comme cliquer sur un bouton ou soumettre un formulaire.
 - *Reducers* : Fonctions qui mettent à jour l'état en fonction des actions, en créant une nouvelle version sans modifier l'original.
 - *Effects* : Gèrent les opérations asynchrones, comme les appels API vers le backend **Spring Boot**.
- **Composants réutilisables** : L'interface est construite à partir de composants Angular modulaires, stylisés avec **CSS** et **Bootstrap** pour un design moderne, adapté à tous les écrans (responsive).
- **Routage** : *Angular Router* permet une navigation fluide entre les différentes vues, comme la page d'accueil, la page de réservation ou le profil utilisateur.

Partie mobile : Ionic avec Capacitor

Pour la version mobile, nous utilisons **Ionic** combiné à **Capacitor** pour développer une application hybride compatible avec iOS et Android. Ionic réutilise les composants Angular pour garantir une apparence cohérente entre les versions web et mobile. Voici les points essentiels :

- **Composants UI Ionic** : Ils offrent une interface qui donne l'impression d'une application native, avec des éléments comme des boutons, des listes ou des fenêtres modales, tous optimisés pour une utilisation tactile.
- **Capacitor** : Cette technologie permet d'accéder aux fonctionnalités natives des smartphones, comme la caméra ou les notifications push, tout en réutilisant le code Angular de la version web.
- **Gestion de l'état** : Tout comme pour le web, *NgRx* est utilisé pour maintenir un état cohérent, synchronisé avec le backend grâce à des appels API.

Cette couche garantit une expérience utilisateur intuitive et fluide, que l'application soit utilisée sur un navigateur ou un smartphone.

2.4.2 La couche applicative

La couche applicative contient la logique métier de l'application. Elle traite les requêtes de la couche de présentation, effectue les calculs nécessaires et communique avec la couche d'accès aux données pour récupérer ou sauvegarder des informations.

Nous utilisons Spring Boot pour implémenter cette couche. Spring Boot est un framework Java puissant qui facilite le développement d'applications robustes et évolutives.

2.4.3 La couche accès aux données

La couche accès aux données regroupe tous les mécanismes qui permettent de stocker, récupérer et gérer les informations utilisées par l'application. Dans notre projet, nous avons choisi **MySQL**, une base de données relationnelle reconnue pour sa fiabilité et sa capacité à organiser les données de manière structurée.

2.5 Conclusion

Après une analyse détaillée des besoins de notre projet et la définition de son architecture, le chapitre suivant sera consacré à la mise en œuvre de notre première release.

CHAPITRE 3 _____

_____ RELEASE 1 :

BIBLIOGRAPHIE

- [1] Lucidchart. <https://www.lucidchart.com/pages/fr/language-uml>. consulté le 15/03/2025.
- [2] Visual Studio Code. <https://bility.fr/definition-visual-studio-code/>. consulté le 15/03/2025.
- [3] Postman. <https://explorweb.github.io/cours2018/cours/postman.html>. consulté le 15/03/2025.
- [4] Apidog. <https://applize.io/formations/apidog-outil-tout-en-un-pour-la-creation-et-> consulté le 15/03/2025.
- [5] Jaspersoft. <https://jaspersoft.developpez.com/tutoriels/jaspersoft-guide-demarrage-rapide/>. consulté le 15/03/2025.
- [6] Miro. <https://digitiz.fr/miro/>. consulté le 15/03/2025.
- [7] GitHub Desktop. <https://docs.github.com/fr/desktop/overview/about-github-desktop>. consulté le 15/03/2025.
- [8] Overleaf. <https://www.overleaf.com/>. consulté le 15/03/2025.
- [9] Draw.io. <https://www.tice-education.fr/tous-les-articles-et-ressources/articles-internet/819-draw-io-un-outil-pour-dessiner-des-diagrammes-en-ligne>. consulté le 15/03/2025.
- [10] HTML. <https://waytolearnx.com/2019/03/difference-entre-html-css-et-javascript.html>. consulté le 15/03/2025.
- [11] CSS. <https://waytolearnx.com/2019/03/difference-entre-html-css-et-javascript.html>. consulté le 15/03/2025.
- [12] JavaScript. <https://waytolearnx.com/2019/03/difference-entre-html-css-et-javascript.html>. consulté le 15/03/2025.
- [13] LaTeX. <https://fr.wikipedia.org/wiki/LaTeX>. consulté le 15/03/2025.
- [14] Java. <https://www.lemagit.fr/definition/Java>. consulté le 15/03/2025.

- [15] **Spring Boot.** <https://www.ibm.com/fr-fr/topics/java-spring-boot>. consulté le 15/03/2025.
- [16] **Angular.** <https://monpetitdev.fr/cest-quoi-angular-definition/>. consulté le 15/03/2025.
- [17] **Node.js.** <https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-node-js/>. consulté le 15/03/2025.
- [18] **TypeScript.** <https://blog.cellenza.com/developpement-specifique/introduction-a-typescript/>. consulté le 15/03/2025.
- [19] **Ionic.** <https://ibracilinks.com/blog/quest-ce-que-ionic-et-pourquoi-lutiliser>. consulté le 15/03/2025.
- [20] **MySQL.** <https://www.data-bird.co/blog/mysql>. consulté le 15/03/2025.