

# Table des matières

<b>Introduction générale</b>	<b>1</b>
<b>1 Contexte général et étude préalable</b>	<b>2</b>
1.1 Introduction . . . . .	3
1.2 Présentation de l'organisme d'accueil . . . . .	3
1.3 Présentation du projet . . . . .	3
1.3.1 Contexte du Projet . . . . .	3
1.3.2 Objectifs du Projet . . . . .	4
1.3.3 Impact Attendu . . . . .	4
1.3.4 Étude de l'existant . . . . .	4
1.3.4.1 Solutions existantes . . . . .	4
1.3.5 Critique de l'existant . . . . .	7
1.4 Choix méthodologique et conceptuel . . . . .	10
1.4.1 Méthodologie de conception . . . . .	10
1.5 Méthodologie de travail . . . . .	10
1.5.1 Méthode 2TUP : Two Tracks Unified . . . . .	10
1.5.2 Méthodologie Agile . . . . .	11
1.5.2.1 Méthode SCRUM . . . . .	12
1.5.2.2 Déroulement de la méthode SCRUM . . . . .	12
1.5.3 Méthodologie adoptée . . . . .	13
1.6 Conclusion . . . . .	14

## TABLE DES MATIÈRES

---

<b>2</b>	<b>Analyse et spécification des besoins</b>	<b>15</b>
2.1	Introduction . . . . .	16
2.2	Analyse et identification des besoins . . . . .	16
2.2.1	Identification des acteurs . . . . .	16
2.2.2	Identification des besoins fonctionnel . . . . .	17
2.2.2.1	Besoins fonctionnels de l'internaute . . . . .	17
2.2.2.2	Besoins fonctionnels du client . . . . .	18
2.2.2.3	Besoins fonctionnels de l'architecte . . . . .	18
2.2.2.4	Besoins fonctionnels de l'administrateur de ArchiMatch	19
2.2.3	Identification des besoins non fonctionnels . . . . .	20
2.3	Conception . . . . .	20
2.3.1	Spécification des besoins fonctionnels . . . . .	20
2.3.1.1	Diagramme de cas d'utilisation général . . . . .	21
2.4	Backlog de produit . . . . .	23
2.5	Architecture physique . . . . .	30
2.6	Patron d'architecture . . . . .	31
2.6.1	Partie Backend . . . . .	31
2.6.1.1	Les avantages de l'architecture MVT : . . . . .	32
2.6.2	Partie Frontend . . . . .	32
2.6.3	Environnement logiciel . . . . .	33
2.7	Conclusion . . . . .	35
<b>3</b>	<b>Réalisation du Release 1</b>	<b>36</b>
3.1	Introduction . . . . .	37
3.2	Sprint 1 . . . . .	37
3.2.1	Backlog du sprint 1 . . . . .	37
3.2.2	Analyse des besoins . . . . .	38
3.2.2.1	Diagramme de cas d'utilisation détaillé . . . . .	38
3.2.2.2	Diagramme de classe détaillé . . . . .	39
3.2.2.3	Diagramme de séquence du scénario « Demande de création de compte architecte » . . . . .	39

## TABLE DES MATIÈRES

---

3.2.2.4	Diagramme de séquence du scénario « Première connexion architecte » . . . . .	40
3.3	Sprint 2 . . . . .	42
3.3.1	Backlog du sprint 2 . . . . .	42
3.3.2	Analyse des besoins . . . . .	43
3.3.2.1	Diagramme de cas d'utilisation détaillé . . . . .	43
3.3.2.2	Diagramme de classe détaillé . . . . .	43
3.3.2.3	Diagramme de séquence du scénario « Gestion des Admin et permission » . . . . .	44
3.3.2.4	Diagramme de séquence du scénario « Gestion des réunions » . . . . .	45
3.4	Sprint 3 . . . . .	46
3.4.1	Backlog du sprint 3 . . . . .	46
3.4.2	Analyse des besoins . . . . .	47
3.4.2.1	Diagramme de cas d'utilisation détaillé . . . . .	47
3.4.2.2	Diagramme de classe détaillé . . . . .	48
3.4.2.3	Diagramme de séquence du scénario « Gestions des demandes de création de compte » . . . . .	49
3.4.2.4	Diagramme de séquence du scénario « Gestions des abonnement coté Admin » . . . . .	50
	<b>Conclusion</b>	<b>52</b>

# Table des figures

1.1	Logo CleverTech . . . . .	3
1.2	Archibien Logo . . . . .	5
1.3	Archidvisor Logo . . . . .	5
1.4	Upwork Logo . . . . .	6
1.5	Architizer Logo . . . . .	7
1.6	Méthodologie 2TUP . . . . .	11
1.7	Processus SCRUM . . . . .	13
2.1	Diagramme de cas d'utilisation général . . . . .	22
2.2	Architecture physique . . . . .	31
2.3	Architecture MVT . . . . .	32
2.4	Architecture Redux . . . . .	33
3.1	Diagramme de cas d'utilisation du Sprint1 . . . . .	38
3.2	Diagramme de classe Release 1 sprint 1 . . . . .	39
3.3	Diagramme de séquence Demande de création de compte architecte . . . . .	40
3.4	Diagramme de séquence Première connexion architecte . . . . .	41
3.5	Diagramme de cas d'utilisation du Sprint2 . . . . .	43
3.6	Diagramme de classe Release 1 sprint 2 . . . . .	44
3.7	Diagramme de séquence Gestion des Admin et permission . . . . .	45
3.8	Diagramme de séquence Gestion des réunion . . . . .	46
3.9	Diagramme de cas d'utilisation du Sprint3 . . . . .	48

## TABLE DES FIGURES

---

3.10 Diagramme de classe Release 1 sprint 3 . . . . .	49
3.11 Diagramme de séquence Gestions des demandes de création de compte	50
3.12 Diagramme de séquence Gestions des abonnement coté Admin . . . . .	51

# Liste des tableaux

1.1	Tableau comparatif . . . . .	9
2.1	Tableau des acteurs du système . . . . .	17
2.2	Environnement logiciel . . . . .	35
3.1	Backlog Sprint 1 . . . . .	38
3.2	Backlog Sprint 2 . . . . .	42
3.3	Backlog Sprint 3 . . . . .	47

# Introduction général

Dans un monde en perpétuelle évolution, où la réalisation de projets architecturaux revêt une importance cruciale pour les professionnels et les particuliers, trouver l'architecte idéal est devenu un défi majeur, nécessitant du temps, des ressources et un réseau de contacts considérable. C'est dans ce contexte que notre projet de stage d'immersion en entreprise, intitulé "**Conception et réalisation d'une Plateforme Archimatch**" prend tout son sens.

**Archimatch** est une solution innovante qui vise à simplifier la recherche et la mise en relation entre les architectes de différents domaines, et les clients professionnels ou particuliers à la recherche de leurs compétences. Cette plateforme a été conçue pour répondre aux besoins pressants d'une industrie où la collaboration entre les architectes et leurs clients est essentielle à la réussite de divers projets architecturaux.

Notre mission au sein de cette entreprise a été de développer cette plateforme polyvalente interactive, composée de quatre espaces distincts, chacun dédié à une étape spécifique du processus.

Le présent rapport synthétise tout le travail que nous avons effectué tout au long de ce stage de PFE. Il est élaboré en cinq chapitres. Le premier chapitre est dédié à la présentation du contexte général ainsi l'étude préalable. Le deuxième chapitre fera l'objet de l'analyse et l'étude des besoins. Les derniers chapitres seront consacrées à la présentations des releases

# **Chapitre 1**

## **Contexte général et étude préalable**



### 1.1 Introduction

Ce chapitre est un chapitre introductif qui comprend d'abord par une présentation de l'organisation d'accueil. La deuxième partie sera consacrée à notre contexte global de projet Et pour finir, la dernière partie sera consacrée à la méthodologie de travail adoptée.

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

CleverTech [1] est une société française de conseil numérique fondée en 2014. Elle se spécialise dans la transformation et l'ingénierie des entreprises, offrant des services de consultation pour la transformation digitale, l'innovation, et le développement d'applications. CleverTech travaille avec diverses organisations pour renforcer leur croissance et leur compétitivité en optimisant leurs processus opérationnels. Mon stage de fin d'études s'est déroulé au sein de leur département d'innovation numérique, appelé StartUp Labs. La figure 1.1 représente le logo de la société CleverTech



FIGURE 1.1 – Logo CleverTech

### 1.3 Présentation du projet

#### 1.3.1 Contexte du Projet

L'univers de l'architecture est extrêmement diversifié, regroupant une multitude de compétences, de spécialisations et de domaines d'expertise. Cependant, la recherche de l'architecte parfaitement adapté à un projet particulier peut être un processus fastidieux et

chronophage. Face à cette complexité, notre projet s'inscrit dans une démarche de simplification et d'optimisation du processus de mise en relation entre les architectes et leurs clients potentiels.

### 1.3.2 Objectifs du Projet

L'objectif central de notre projet était de créer une plateforme en ligne "Archimatch", destinée à simplifier et à rationaliser la recherche et la sélection d'architectes qualifiés. Nous avons cherché à concevoir un outil convivial et performant, capable de résoudre les problèmes auxquels sont confrontés les clients à la recherche d'architectes compétents, ainsi que les architectes à la recherche de projets stimulants.

### 1.3.3 Impact Attendu

Archimatch vise à révolutionner la manière dont les architectes sont choisis et engagés pour des projets architecturaux. En simplifiant et en accélérant le processus de mise en relation, la plateforme contribue à l'efficacité, à la transparence et à la réussite des projets architecturaux. Elle offre également aux architectes une opportunité de mettre en avant leur expertise et de trouver des projets correspondant à leurs compétences, favorisant ainsi la croissance de leur activité.

### 1.3.4 Étude de l'existant

#### 1.3.4.1 Solutions existantes

- **Archibien :**

Archibien Est une plateforme dédiée à aider les individus à concevoir et construire leur propre maison. Elle propose divers services incluant l'accès à des architectes, des ressources pour la conception de maison et un accompagnement tout au long du processus de construction. Les utilisateurs peuvent trouver de l'inspiration, se connecter avec des professionnels et obtenir un soutien pour concrétiser leurs projets architecturaux grâce à la plateforme Archibien.



FIGURE 1.2 – Archibien Logo

• **Archidvisor :**

Archidvisor Est une plateforme en ligne qui met en relation des particuliers ayant des projets de construction ou de rénovation avec des architectes. Les utilisateurs peuvent décrire leur projet sur le site, indiquer leurs besoins et leurs préférences, puis recevoir des propositions d'architectes adaptées à leurs besoins. Ils peuvent ensuite comparer les offres, consulter les profils des architectes et choisir celui qui correspond le mieux à leurs attentes. Archidvisor facilite ainsi le processus de recherche et de sélection d'un architecte pour un projet immobilier.



FIGURE 1.3 – Archidvisor Logo

### • Upwork :

Upwork offre une plateforme efficace et conviviale pour les architectes afin de trouver des opportunités de projet à travers le monde. Avec des fonctionnalités de création de profil détaillé, de recherche avancée de projets et de communication sécurisée avec les clients, les architectes peuvent facilement présenter leurs compétences, trouver des projets correspondant à leurs intérêts et collaborer en ligne de manière transparente. Grâce à des outils de gestion de projet intégrés et à des systèmes de paiement sécurisés, Upwork simplifie le processus de travail indépendant pour les architectes, leur permettant de se concentrer sur leur créativité et leur succès professionnel.



FIGURE 1.4 – Upwork Logo

### • Architizer :

Architizer est une plateforme pour les architectes, offrant une variété de fonctionnalités essentielles. Les architectes peuvent publier leurs projets et interagir avec une communauté mondiale. La recherche avancée de projets permet de trouver des opportunités correspondant aux compétences et intérêts des architectes, tandis que les ressources éditoriales fournissent des articles et des critiques d'architecture pour rester informé des dernières tendances. Architizer facilite également les collaborations entre architectes et clients, offrant un environnement complet pour promouvoir, découvrir et partager le meilleur de l'architecture contemporaine.



FIGURE 1.5 – Architizer Logo

### 1.3.5 Critique de l'existant

Nous avons relevé des critères à prendre en considération lors de la comparaison et l'évaluation des applications dans le Tableau comparatif 1.1. Nous explicitons par la suite les critères les plus pertinents pour notre solution.

- **Module gestion paramètre compte** : Le client peut ajuster ses paramètres de compte, il peut renforcer la sécurité de son compte en modifiant son mot de passe.
- **Module gestion et suivi projet** : La plateforme offre au client un suivi détaillé et une gestion fiable des architectes, présenté dans une boxe sous sa publication. Le client peut supprimer un projet et voir l'historique des architectes. Ce module indique le statut de l'architecte, passant par trois phases distinctes
- **Module gestion box de proposition projets** : l'architecte a la main d'accéder a un box ou il peut visualiser des projets publiés par les clients et qui sont recommandés selon plusieurs critères bien étudiés
- **Module gestion profil architecte** : Le module de gestion du profil architecte permet à l'architecte de visualiser et de gérer ses informations personnelles, ses projets et des services proposés
- **Portefeuille de réalisations architecte** : Le module de Portefeuille de réalisations

architecte permet aux architectes de la plate-forme de présenter leurs réalisations a fin d'améliorer leurs visibilités et avoir plus de projet dans leurs box de propositions

- **Module de gestion et suivi projet sélectionné** : L'architecte peut sélectionner un ou plusieurs projets sur la plateforme et ce module lui permet de suivre l'état, le statut et l'avancement de son projet et rester en communication directe avec ses clients

- **Module de notification** : es utilisateurs seront informés de tout ce qui se passe sur la plateforme

- **Module de signalement** : Ce module permet aux client de signaler un architecte suite a un comportement non respectueux ainsi donne la main aux architectes de signaler les annonces publiées par les clients

- **Module note architecte** : Ce module permet aux client de donner leurs avis a propos les architectes après avoir compléter leur travaux en donnant des notes

- **Module d'abonnement et paiement** : Ce module permet aux architectes de sélectionner un plan d'abonnement selon leurs préférences et réaliser l'opération de paiement en utilisant plusieurs méthodes

- **Module gestion des réunions démo** : Ce module permet aux administrateurs de la plate-forme de bien organiser les réunions de démonstration avec les architectes afin de finaliser l'opération de créations de leurs comptes

- **Module gestion des parrainages** : Ce module permet aux architecte d'envoyer des invitations aux autres architectes pour rejoindre la plateforme

## CHAPITRE 1. CONTEXTE GÉNÉRAL ET ÉTUDE PRÉALABLE

---

	<b>Archibien</b>	<b>Archidvisor</b>	<b>Upwork</b>	<b>Architizer</b>
Module gestion paramètre compte	✓	✓	×	×
Module gestion et suivi projet	×	×	×	×
Module gestion box de proposition projets	×	×	×	×
Module gestion profil architecte	✓	✓	✓	×
Recherche des projets selon le profil	×	×	×	×
Portefeuille de réalisations architecte	✓	✓	✓	✓
Notifications	✓	×	✓	✓
Module de gestion et suivi projet sélectionné	×	×	×	×
Module de signalement	×	×	×	×
Module note architecte	×	×	×	×
Module d'abonnement et paiement	×	✓	✓	×
Module gestion des réunions démo	×	×	×	×
Module gestion des parrainages	×	×	×	×

TABLE 1.1 – Tableau comparatif

## **1.4 Choix méthodologique et conceptuel**

### **1.4.1 Méthodologie de conception**

Pour la spécification et la conception de ce travail, nous avons utilisé le langage de modélisation UML (Unified Modeling Language)[9], qui est un langage de modélisation qui permet de décrire les exigences, de documenter les systèmes et d'esquisser l'architecture logicielle.

UML est organisé autour de 3 axes différents, chacun étant dédié à la présentation d'un concept spécifique du système logiciel.

Cependant, nous ne montrerons que certains diagrammes que nous avons trouvés utiles et suffisants pour comprendre le projet, à savoir les diagrammes de cas d'utilisation, les diagrammes de classes, et les diagrammes de séquence.

## **1.5 Méthodologie de travail**

Avant de réaliser chaque projet, il est nécessaire de choisir une méthodologie de travail afin d'aboutir à un logiciel fiable, adaptable et efficace. Et pour pouvoir choisir la méthodologie la plus adéquate nous avons procédé à une comparaison entre la méthode 2Tup et la méthode agile Scrum

### **1.5.1 Méthode 2TUP : Two Tracks Unified**

La méthode 2 Tracks Unifed est un processus de développement logiciel qui met en œuvre la méthode du processus unifié (c'est-à-dire construit sur UML, itératif, centré sur l'architecture et conduit par les cas d'utilisation). Le 2TUP propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels. Il commence par une étude préliminaire qui consiste essentiellement à identifier les acteurs qui vont interagir avec le système à construire, les messages qu'échangent les acteurs et le système, à produire le cahier de charges et à modéliser le contexte. Le processus



s'articule ensuite autour de trois phases essentielles comme indique la figure 1.5 :

- **Une branche fonctionnelle** : elle capitalise la connaissance du métier de l'entreprise. Cette branche capture des besoins fonctionnels, ce qui produit un modèle focalisé sur le métier des utilisateurs finaux..
- **Une branche technique** : capitalise un savoir-faire technique et/ou des contraintes techniques. Les techniques développées pour le système sont indépendamment des fonctions à réaliser.
- **Une phase de réalisation** : elle consiste à réunir les deux branches, permettant de mener une conception applicatif et enfin la livraison d'une solution adaptée aux besoins

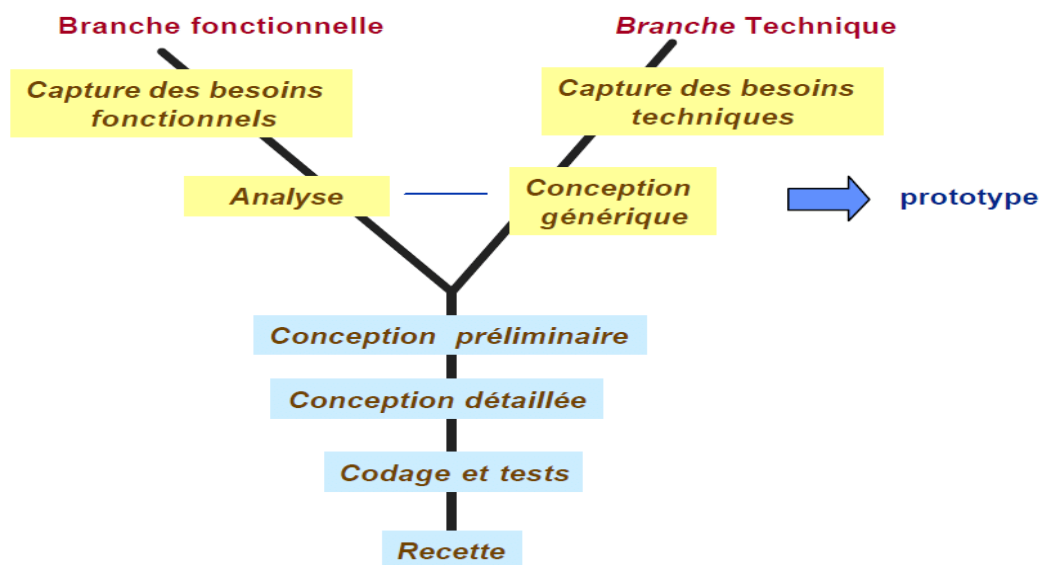


FIGURE 1.6 – Méthodologie 2TUP

### 1.5.2 Méthodologie Agile

Les méthodes agiles sont des outils de gestion de projet très populaires parmi les développeurs de logiciels et d'applications. Depuis peu, de nombreuses agences de com-

munication et services marketing les utilisent pour repenser leurs produits et services. L'une de leurs caractéristiques est que les clients sont au cœur du processus de développement de produits répondant aux besoins du marché. De plus, les chefs d'entreprise l'utilisent pour créer des outils pour améliorer le bien-être des employés.

### 1.5.2.1 Méthode SCRUM

SCRUM[8] est l'une des principales méthodes de gestion de projets agiles. Sa principale caractéristique est de se concentrer sur les personnes et leurs interactions, les fonctions logicielles et la coopération avec les clients. C'est une bonne solution qui prend en compte le processus, la documentation complète, les négociations contractuelles et les plans préétablis. Soutenue par les piliers de la transparence, de la vérification et de l'adaptation, l'approche SCRUM vise à réduire les difficultés telles que le manque de planification, les exigences changeantes et la portée inexacte. De plus, SCRUM repose sur une livraison rapide, fréquente et continue de logiciels fonctionnels, une coopération continue entre l'équipe et l'entreprise, l'excellence technique et la simplicité.

### 1.5.2.2 Déroulement de la méthode SCRUM

Scrum est basé sur des itérations appelées sprints (développement à court terme avec des objectifs prédéfinis). L'objectif est une partie du référentiel d'exigences appelé backlog de produit, qui est fourni et maintenu par le propriétaire du produit. Le référentiel est composé de fonctions qui sont continuellement priorisées. Avant chaque sprint, la fonction la plus prioritaire entre dans le backlog de sprint, et devient donc l'objectif à atteindre lors du sprint (itération). Le sprint commence toujours par le plan, en commençant par la discussion entre le propriétaire du produit et le backlog du produit. Après la réunion, la tâche est confirmée et le sprint peut commencer.

L'équipe de développement est dirigée par un Scrum Master dont le but est de résoudre les obstacles, de participer au développement si nécessaire et de tout mettre en place. Mettre en œuvre pour s'assurer que l'objectif est atteint pendant le sprint.

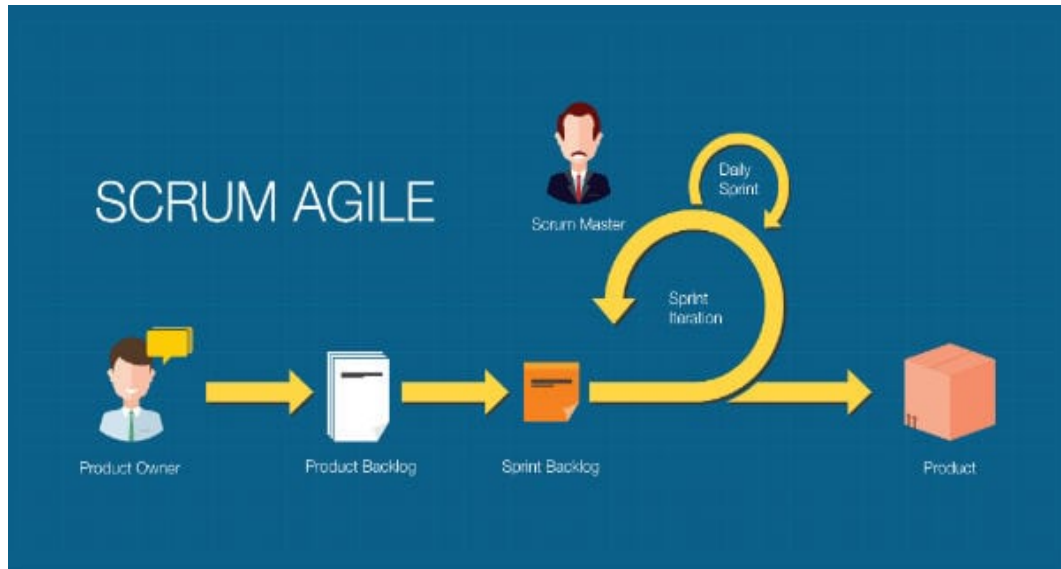


FIGURE 1.7 – Processus SCRUM

Chaque sprint augmente la valeur ajoutée du produit en ajoutant de nouvelles fonctionnalités qui peuvent être livrées aux clients.

### Les rôles SCRUM :

- Équipe SCRUM : une équipe de 10 personnes maximum. L'équipe est auto-organisée et chacun reprend les tâches d'un autre membre.
- Product owner : le correspondant métier, qui représente l'entreprise en gérant la priorité et le développement du backlog produit. Il doit se permettre de répondre aux questions de l'équipe Scrum.
- Scrum Master : il cumule plusieurs rôles : coach, chef de projet, développeur, coordinateur... Il doit exécuter le rituel Scrum dans son équipe et s'assurer qu'ils ne soient pas entravés par des faits extérieurs tout en assurant des rapports.

### 1.5.3 Méthodologie adoptée

Nous avons décidé d'adopter le processus Agile Scrum pour la suite de notre travail car il offre une approche itérative et collaborative qui correspond parfaitement à la nature

de notre projet. Scrum divise le travail en itérations appelées "sprints", généralement de deux à quatre semaines, au cours desquelles des fonctionnalités spécifiques sont développées, testées et livrées. Ce processus favorise une communication transparente et régulière entre les membres de l'équipe grâce à des réunions quotidiennes appelées "stand-up meetings". De plus, Scrum permet une adaptation continue aux changements grâce à des revues régulières et des rétrospectives à la fin de chaque sprint, ce qui correspond à notre besoin de flexibilité dans un environnement en évolution constante. Enfin, la responsabilité collective de l'équipe dans la réalisation des objectifs du sprint assure une motivation élevée et une meilleure efficacité dans l'accomplissement des tâches.

### **1.6 Conclusion**

En guise de conclusion, dans ce chapitre nous avons mis notre projet dans son cadre général en mettant l'accent sur son objectif. Ensuite, nous avons clôturé par la méthodologie de développement que nous avons adoptée. Dans le chapitre suivant nous allons spécifier les différents besoins fonctionnels et non fonctionnels auxquels doit répondre notre plateforme.

## **Chapitre 2**

### **Analyse et spécification des besoins**

### 2.1 Introduction

Dans ce chapitre nous allons aborder tout ce qui est en rapport avec l'analyse du projet, ainsi que la spécification des besoins fonctionnels et non fonctionnels de notre système.

### 2.2 Analyse et identification des besoins

Nous allons commencer par identifier les acteurs. Plus tard, nous listerons et détaillerons toutes les exigences fonctionnelles et non fonctionnelles sur lesquelles nous allons travailler pour les adapter à notre plateforme dans le but de satisfaire les besoins de nos utilisateurs.

#### 2.2.1 Identification des acteurs

Nous présentons tous les acteurs qui interagissent avec notre système dans le tableau 2.1 ci dessous :

Acteur	Roles
Internaute	L'internaute représente le visiteur initial de la plateforme "Archimatch". Il a la possibilité de créer un compte, que ce soit en tant qu'architecte ou en tant que client.
Client	Il peut gérer ses annonces publiées pour la recherche d'architecte, il peut aussi consulter les réalisations des architectes intéressés par son projet, évaluer les devis proposés, et prendre des décisions éclairées sur le choix de son architecte.

Architecte	Ils peuvent rechercher des projets qui correspondent à leur domaine d'expertise. Ils ont la possibilité de contacter les clients en soumettant des devis pour les projets qui les intéressent. De plus, les architectes ont la responsabilité de créer et de gérer leur profil professionnel pour attirer les clients.
Administrateur de ArchiMatch	C'est l'administrateur de la plateforme, il a toutes les permissions et tous les droits d'accès. Il peut gérer les comptes et les demandes et Il est chargé de superviser et de suivre les activités sur la plateforme.

TABLE 2.1 – Tableau des acteurs du système

### 2.2.2 Identification des besoins fonctionnel

Les besoins fonctionnels sont les fonctions fournies par le système afin de répondre aux exigences des utilisateurs. Nous allons présenter les besoins fonctionnels par acteur comme suit :

#### 2.2.2.1 Besoins fonctionnels de l'internaute

- **Demander un compte d'Architect** : L'Architect qui peut envoyer une demande de création de son propre compte afin de jouir des fonctionnalités fournies par la plateforme.
- **Créer un compte client** : l'internaute peut créer son propre compte client sur la plateforme pour profiter les fonctionnalités fournies.
- **Se connecter a son espace** : l'internaute peut utiliser son compte crée pour pouvoir accéder a son espace

- **Visualiser les réalisations des architectes** : l'internaute peut accéder à une liste des catégories publiées dans l'espace vitrine pour visualiser les réalisations des architectes de la plate-forme ainsi que leurs profils
- **Contacteur l'équipe de la plateforme** : le visiteur peut contacter l'équipe pour poser ses questions concernant la solution, proposer des offres de partenariat, etc.

**Remarque** : chacun des acteurs suivants doit s'authentifier afin d'accéder à son espace et d'utiliser les opérations fournies.

### 2.2.2.2 Besoins fonctionnels du client

- **Gérer le profil** : il a le droit de consulter, modifier ses informations de profil : coordonnées personnelles (nom, prénom, etc.) .
- **Gérer les annonces** : le client peut publier, mettre à jour et supprimer une annonce, il peut ainsi définir le statut de ces annonces.
- **Donner un avis par rapport un architecte** : le client peut évaluer le travail de l'architecte et donner son avis.
- **Consulter un architecte** : il peut afficher plus de détails de l'architecte (présentation, description, travail réalisés, etc.).
- **Signaler un architecte** : Le client a le droit de signaler un architecte en donnant la raison et des informations supplémentaires en laissant un message

### 2.2.2.3 Besoins fonctionnels de l'architecte

- **Avoir des propositions des projets** : la plateforme propose des projets à l'architecte selon les détails de leurs profils, leurs expériences et leurs réalisations de son côté l'architecte a la main à visualiser ces projets dans le but de sélectionner un ou plusieurs parmi son box de propositions
- **Consulter une annonce** : il peut afficher plus de détails sur une annonce (besoin, détails d'exécution, adresse, etc.).
- **Gérer la liste des annonces sélectionnées** : l'architecte peut accéder à ces projets sélectionnés et en cours de réalisations et les organiser.



- **Gérer le profil** : il a le droit de consulter, modifier ses informations de profil : coordonnées personnelles (nom, prénom, etc.).
- **Choisir un plan d'abonnement** : l'architecte à la possibilité de choisir un plan d'abonnement parmi les plans réalisé par l'administrateur.
- **Gérer des devis** : L'architecte peut envoyer des devis au clients pour commencer a travailler sur le projet et il peut également supprimer ou modifier le devis
- **Signaler un client** : L'architecte a le droit de signaler un client en donnant la raison et des informations supplémentaires en laissant un message .

### 2.2.2.4 Besoins fonctionnels de l'administrateur de ArchiMatch

- **Gérer les demandes** : L'admin peut accepter ou refuser les demandes de création de comptes des architectes.
- **Gérer les comptes** : Le super admin a la possibilité d'ajouter, supprimer ou modifier des comptes des autres admin.
- **Gérer les abonnements** : L'admin a la possibilité d'ajouter, supprimer ou modifier les plans d'abonnement pour les architectes.
- **Suivre les projets sélectionnés** : l'admin peut consulter les projets sélectionnés par les architectes.
- **consulter les clients qui ont publié les annonces** : l'admin peut consulter les détails des clients qui ont publié les annonces.
- **gérer les autorisations des comptes admin** : le super admin a le droit de créer des comptes administrateur en les donnant des autorisations et des droit d'accès restreints .
- **gérer les propositions des projets pour les architectes** : l'admin peut consulter la liste de projets proposés au architecte et il peut modifier les paramètre de box de propositions
- **gérer les signalements** : l'admin peut consulter et traiter les signalement reçus soit par les architectes ou par les clients
- **Vérifier les profils architectes** : l'admin peut consulter les profils des architectes et les vérifier pour les publier dans l'espace vitrine

- **Vérifier les blogs** : l'admin peut ajouter modifier ou supprimer des blogs informatives qui vont être publier dans l'espace vitrine

### 2.2.3 Identification des besoins non fonctionnels

Un besoin non fonctionnel est une exigence qui caractérise une propriété souhaitée (qualité) du système. Notre solution doit répondre aux critères suivants :

- **Sécurité** : cette propriété est l'un des plus importants critères que nous allons assurer par l'attribution de la liste des permissions et des droits d'accès aux différents espaces selon le type de compte que l'utilisateur possède.
- **Validité** : le produit réalisé doit satisfait ses fonctionnalités ; qui répond exactement aux besoins des utilisateurs.
- **Facilité de l'utilisation et Attractivité** : consiste à offrir à l'utilisateur une bonne expérience. Des interfaces claires, attractive, faciles à comprendre et y naviguer.
- **Maintenabilité et extensibilité** : la décomposition que nous allons appliquer sur notre application peut nous aider à adapter rapidement notre application aux nouvelles exigences. Ainsi nous permettrons de relever et de résoudre aisément les problèmes des modules.
- **Évolution** : l'équipe de la plateforme doit être à l'écoute des nouvelles demandes des besoins de ses utilisateurs dans le but de les satisfaire.

## 2.3 Conception

La conception est une étape préalable essentielle avant le développement, car elle détermine la structure, l'architecture et la manière dont les besoins identifiés dans le chapitre précédent seront mis en œuvre.

### 2.3.1 Spécification des besoins fonctionnels

La conception est une étape fondamentale dans la réalisation d'un projet. Dans cette section, nous allons présenter le diagramme de cas d'utilisation global de l'application

### **2.3.1.1 Diagramme de cas d'utilisation général**

Le diagramme de cas d'utilisation permet de détailler les fonctionnalités offertes par notre système ainsi que leurs associations avec les acteurs décrits auparavant

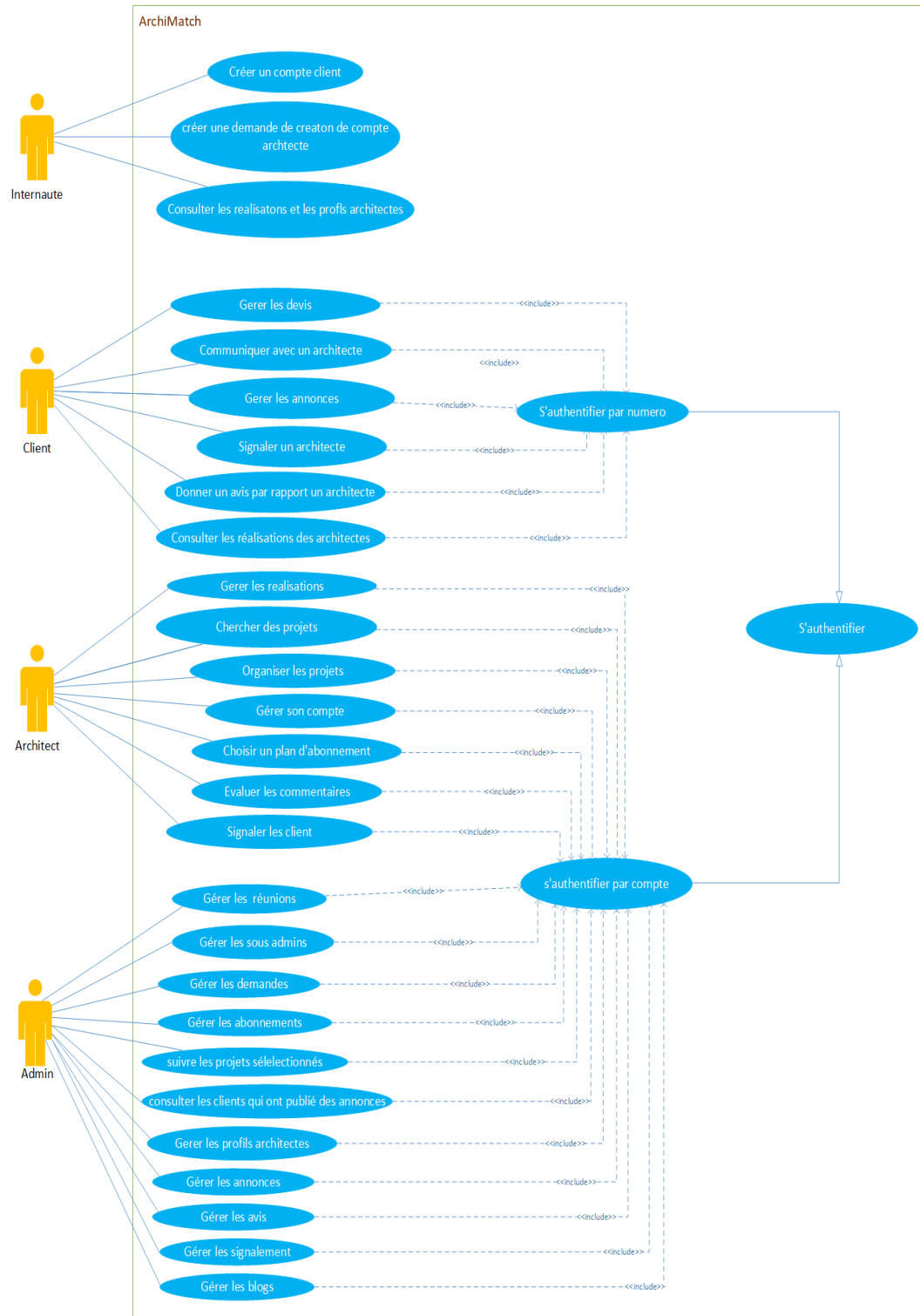


FIGURE 2.1 – Diagramme de cas d'utilisation général

## 2.4 Backlog de produit

Le tableau ci-dessous montre le Backlog produit de notre projet.

Product Backlog		
Equipe	Gafsi Sarra Islem et Ayeb Mohamed	
Date	01/02/2024	
Release	authentification et création de compte	
Sprint	Features	User story
1	<p>Demande de création de compte architecte</p> <p>Première connexion architecte (choisir le plan d'abonnement, choisir le mode de paiement)</p> <p>Login Admin</p>	<ul style="list-style-type: none"><li>• En tant qu'architecte, je peux soumettre la demande de création de compte au travers un formulaire stepper</li><li>• En tant qu'architecte, je peux me connecter à mon compte pour la première fois et définir son mot de passe.</li><li>• En tant qu'administrateur, je peux me connecter à mon compte via un lien en utilisant mon adresse e-mail et mon mot de passe.</li></ul>

2	<p>Gestion des Admin et permission</p> <p>Gestion des réunion</p>	<ul style="list-style-type: none"><li>• En tant que Super admin je peux créer un nouveau compte pour les sous admins.</li><li>• En tant que Super admin, je peux attribuer des permissions spécifiques à chaque sous admin.</li><li>• En tant que Super admin, je peux supprimer un sous admin.</li><li>• En tant que Super admin, je peux visualiser les détails des sous admins.</li><li>• En tant que Super admin, je peux modifier les informations des sous admins.</li><li>• En tant qu'administrateur, je peux gérer mes indisponibilités en utilisant un calendrier.</li><li>• En tant qu'administrateur, je peux visualiser et filtrer les réunions planifiées par mois, semaine et jour.</li><li>• En tant qu'administrateur, je peux accéder à une liste détaillée des réunions prévues pour la journée.</li><li>• En tant qu'administrateur, je peux planifier des entretiens d'acceptations .</li></ul>
3	Gestions des demandes de création de compte	<ul style="list-style-type: none"><li>• En tant qu'administrateur, je peux Accepter les nouveaux demande de creation de comptes d'architectes.</li><li>• En tant qu'administrateur, je peux Refuser les demande de creation de comptes vérifiés.</li><li>• En tant qu'administrateur, je peux Consulter les demande de creation de comptes vérifiés.</li></ul>

## CHAPITRE 2. ANALYSE ET SPÉCIFICATION DES BESOINS

---

	Gestions des abonnements coté Admin	<ul style="list-style-type: none"><li>• En tant qu'administrateur, je peux ajouter/modifier/supprimer un plan d'abonnement sur la plateforme.</li><li>• En tant qu'administrateur, je peux voir la liste des demandes d'abonnement soumises par les architectes.</li><li>• En tant qu'administrateur, je peux valider/annuler l'activation l'abonnement après vérification du paiement par virement.</li></ul>
Release	Gestion des Profils	
Sprint	Features	User story
1	Profil Architecte	<ul style="list-style-type: none"><li>• En tant qu'architecte, je peux afficher/modifier mes réalisations, projets précédents et compétences dans mon profil en cas de vérification de mon compte.</li></ul>
	Gestion des réalisations coté architecte	<ul style="list-style-type: none"><li>• En tant qu'architecte, je peux lister mes réalisations.</li><li>• En tant qu'architecte, je peux Ajouter des réalisations.</li><li>• En tant qu'architecte, je peux Supprimer mes réalisations.</li></ul>
2	Paramètre architecte	<ul style="list-style-type: none"><li>• En tant qu'architecte, je peux visualiser et modifier les paramètres de mon compte .</li></ul>

	Paramètre client	<ul style="list-style-type: none"><li>• En tant que Client je peux gérer mes coordonnées en modifiant mon nom,prénom,Email ,numéro</li><li>• En tant que client , je peux me gérer et configurer les notifications reçu</li><li>• En tant que client , je peux mettre a jour mon mot de passe</li></ul>
3	Gestions des architectes coté Admin	<ul style="list-style-type: none"><li>•En tant qu'administrateur, je peux visualiser la progression d'achèvement des profils des architectes.</li><li>•En tant qu'administrateur, je peux accéder à une liste d'architecte dont ses progressions au cours d'achèvement.</li><li>•En tant qu'administrateur, je peux valider/refuser le profil d'un architecte lorsqu'il a complète son profil.</li><li>•En tant qu'administrateur, je peux fournir des instructions pour compléter la validation de compte au cas d'échec.</li></ul>
4	Visualisation des profils des architecte en vitrine	<ul style="list-style-type: none"><li>•En tant que client, je peux visiter le profil de l'architecte associé a un projet.</li></ul>



## CHAPITRE 2. ANALYSE ET SPÉCIFICATION DES BESOINS

---

	Gestions des avis client	<ul style="list-style-type: none"><li>• En tant qu'architecte, je peux fournir un devis et l'envoie au client.</li><li>• En tant qu'architecte, je peux négocier avec le client (au cas de refus du devis)</li><li>• En tant qu'architecte, je peux être notifié d'un rappel pour envoyer un devis au client.</li></ul>
Release	Gestion des Annonces	
Sprint	Features	User story
1	Gestion des annonces clients	<ul style="list-style-type: none"><li>• En tant qu'administrateur, je peux superviser les annonces publiées.</li><li>• En tant qu'administrateur, je peux consulter les listes de projets disponibles.</li><li>• En tant qu'administrateur, je peux filtrer les annonces avec un filtre des statuts.</li><li>• En tant qu'administrateur, je peux envoyer au client un e-mail lors de la sélection de son annonce par un architecte.</li></ul>

	Gestion des annonces coté client	<ul style="list-style-type: none"><li>•En tant que client, je peux Ajouter/Annuler une annonce.</li><li>•En tant que client, je peux lister tous les architectes qu'ont sélectionné mon annonce et voir leurs coordonnées et contact. . .</li><li>•En tant que client, je peux lister mes annonces</li><li>•En tant que client, je peux mettre à jour le statut de recherche de chaque annonce.</li><li>•En tant que client, je peux mettre à jour le statut globale de chaque architecte.</li><li>•En tant que client, je peux modifier les annonces pour mes projets.</li></ul>
2	<p>Gestions des sélections coté client</p> <p>Box de proposition coté architectes</p>	<ul style="list-style-type: none"><li>•En tant que client, je peux accepter un devis.</li><li>•En tant que client, je peux refuser un devis</li><li>•En tant que client, je peux recevoir des devis de la part des architectes.</li><li>•En tant que client, je peux consulter les devis envoyés par les architectes.</li><li>•En tant qu'architecte, je peux sélectionner un projet proposer par la plateforme.</li><li>•En tant qu'architecte, je peux sélectionner un projet proposer directement par un client au cas où j'ai un abonnement Pro Premium.</li></ul>



4	Gestions des signalement	<ul style="list-style-type: none"><li>•En tant qu'architecte, je peux signaler un projet/avis.</li><li>•En tant que client, je peux signaler le travail d'un architecte en remplissant les données nécessaires.</li><li>•En tant que client, je peux fournir les details sur l'incident rencontré.</li><li>•En tant que client, je peux signaler le profil architecte en donnant les raisons necessaires.</li><li>•En tant qu'administrateur, je peux examiner les détails de signalement.</li><li>•En tant qu'administrateur, je peux envoyer des avertissements officiels aux utilisateur concerné.</li><li>• En tant qu'administrateur, je peux recevoir un signalent des problèmes, des bogues ou des comportements inattendus dans la plateforme.</li></ul>
---	--------------------------	--

### 2.5 Architecture physique

L'architecture physique de notre application, représente l'ensemble des composantes matérielles constituant l'application web. Dans ce contexte, notre application est constituée principalement d'un utilisateur, une partie applicative qu'est composée de la partie front-end, back-end et une partie de persistance composée d'un serveur de base de donnée et les noeuds du Elasticsearch, la figure 2.2 ci-dessous.

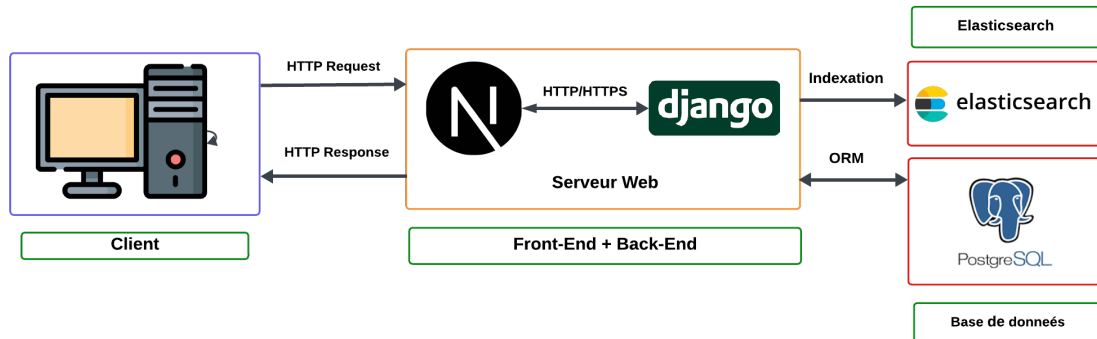


FIGURE 2.2 – Architecture physique

## 2.6 Patron d'architecture

Un patron d'architecture est une solution générale et réutilisable à un problème d'architecture récurrent. Dans ce cadre, nous allons présenter le patron d'architecture pour les deux parties frontend et backend.

### 2.6.1 Partie Backend

Pour la partie Backend, nous avons utilisé l'architecture MVT. En effet, l'architecture MVT s'inspire de l'architecture MVC très utilisée dans les frameworks Web. Son objectif est de séparer les responsabilités de chaque pôle afin que chacun focalise sur ses tâches. Elle représente une architecture orientée autour de trois pôles.

- **Modèle** : interagit avec une base de données via l'ORM.
- **Vue** : reçoit une requête HTTP et renvoie une réponse HTTP adéquate (par exemple si la requête est une interaction avec une base de données, la vue appelle un modèle pour récupérer les items demandés).
- **Template** : est un fichier HTML récupéré par la vue et envoyé au visiteur avec les données des modèles.

### 2.6.1.1 Les avantages de l'architecture MVT :

- Une meilleure organisation du code.
- Une conception claire et efficace grâce à la séparation des tâches effectuées par les différents éléments.
- Une plus grande souplesse pour organiser le développement du site entre les différents développeurs.

La figure 2.3 ci-dessous montre comment les différentes composantes de l'architecture MVT de Django interagissent pour répondre à la requête d'un utilisateur.

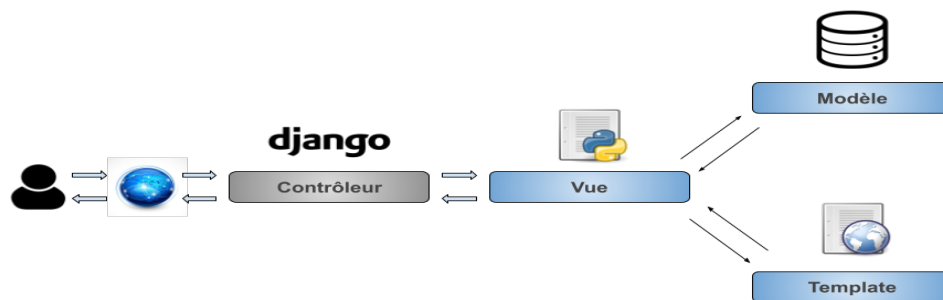


Schéma de l'architecture MVT

FIGURE 2.3 – Architecture MVT

### 2.6.2 Partie Frontend

Pour la partie Frontend, nous avons utilisé l'architecture Redux qui est une librairie Javascript qui nous permet de réduire la complexité du code en appliquant la restriction sur la manière et le temps de la mise à jour de l'état. Elle peut se produire comme le montre la figure 2.4 ci-dessous.

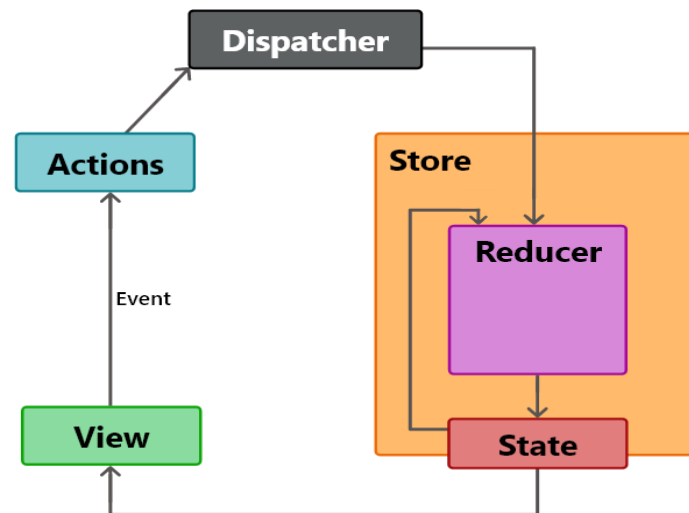


FIGURE 2.4 – Architecture Redux

- **Views** : ce sont des classes ou des fonctions Javascript appelés composants dans lesquelles nous définissons la logique de l'interface utilisateur, elles acceptent d'autres propriétés (props) à venir d'autres composants et renvoient la façon dont notre interface utilisateur prendra forme. Il y a un composant racine qui enveloppe tous les autres composants comme des enfants qu'on nomme (app.js).
- **Redux store** : il s'agit d'un conteneur de gestion d'état global pour notre application React. Tout composant connecté au store Redux peut envoyer un appel d'action, après que ces derniers transmettent des propriétés aux réducteurs (reducers), chaque réducteur (reducer) a une action définie et un état donné. Le rôle des réducteurs (reducers) est de fournir le nouvel état aux composants.

### 2.6.3 Environnement logiciel

Dans ce qui suit, nous présentons les outils logiciels utilisés lors du développement de notre application.

Logo	Technologie
	Dans la partie backend, nous avons opté pour l'utilisation du framework web Django, qui est basé sur le langage de programmation Python. Ce framework est open source et simple à utiliser. Il a aussi une communauté qui est extrêmement vivante.[3]
 PostgreSQL	Pour la base de données, nous avons choisi PostgreSQL, car elle offre des solutions efficaces pour la gestion des problématiques complexes, garantit une grande capacité d'interrogation, assure l'indépendance des données et permet d'éliminer la redondance.[7]
 elasticsearch	Elasticsearch est un logiciel utilisant Lucene pour l'indexation et la recherche de données. Il fournit un moteur de recherche distribué et multi-entité à travers une interface REST. [4]
	Pour la partie frontend, nous avons opté pour Next.js, car cette bibliothèque JavaScript offre une approche efficace pour le développement d'applications web côté client. Next.js facilite la création d'interfaces utilisateur interactives en fournissant des fonctionnalités avancées telles que le rendu côté serveur, le routage dynamique, et la génération de pages statiques.[6]





	<p>Pour la gestion de projet agile, nous avons choisi ClickUp, une plateforme polyvalente et intuitive qui permet d'organiser les projets de manière flexible grâce à ses tableaux, listes et cartes personnalisables. Avec des fonctionnalités avancées telles que les vues en liste, les tableaux de bord personnalisables et le suivi du temps et des ressources.[2]</p>
	<p>Pour le contrôle de suivi et la gestion de versionnement du projet, nous avons préféré GitHub. Cette plateforme de développement collaborative est basée sur les fonctionnalités du logiciel Git. Elle simplifie l'exploitation, le développement et la collaboration entre les équipes tout en offrant des fonctionnalités avancées pour la gestion de la sécurité et l'approche CI/CD.[5]</p>

TABLE 2.2 – Environnement logiciel

## 2.7 Conclusion

Dans cette section, nous avons examiné la spécification des besoins fonctionnels et non fonctionnels, ainsi que la conception de notre application. Cette étude s'est concrétisée à travers la création de diagrammes de cas d'utilisation, visant à rendre l'étape suivante claire et compréhensible.

## **Chapitre 3**

### **Réalisation du Release 1**

### 3.1 Introduction

Dans ce chapitre, nous allons organiser notre première release en sprints, dans lequel nous allons traiter les users Stories de nos sprints afin de produire un ensemble d'incrémentes potentiellement livrable. Nous allons présenter la phase d'analyse et la solution conceptuelle en exposant les différents diagrammes qui décrivent l'interaction entre le système et l'utilisateur.

### 3.2 Sprint 1

#### 3.2.1 Backlog du sprint 1

Durant cette partie, nous allons détailler les différentes tâches à effectuer afin d'atteindre les objectifs de sprint 1.

Nom	Description	Estimation
Demande de création de compte architecte	<ul style="list-style-type: none"><li>En tant qu'architecte, je peux soumettre la demande de création de compte au travers un formulaire step-per.</li></ul>	jours
Première connexion architecte (choisir le plan d'abonnement, choisir le mode de paiement)	<ul style="list-style-type: none"><li>En tant qu'architecte je peux recevoir un mail de confirmation de création de compte pour pouvoir accéder et remplir les étapes de premières connexion.</li></ul>	jours

Login Admin	<ul style="list-style-type: none"> <li>• En tant qu'administrateur, je peux me connecter à mon compte via un lien en utilisant mon adresse e-mail et mon mot de passe.</li> </ul>	jours
-------------	---	-------

TABLE 3.1 – Backlog Sprint 1

## 3.2.2 Analyse des besoins

### 3.2.2.1 Diagramme de cas d'utilisation détaillé

Pour mieux comprendre les fonctionnalités à réaliser durant ce sprint, nous allons présenter le diagramme de cas d'utilisation du sprint 1 dans la figure 3.1 ci-dessous.

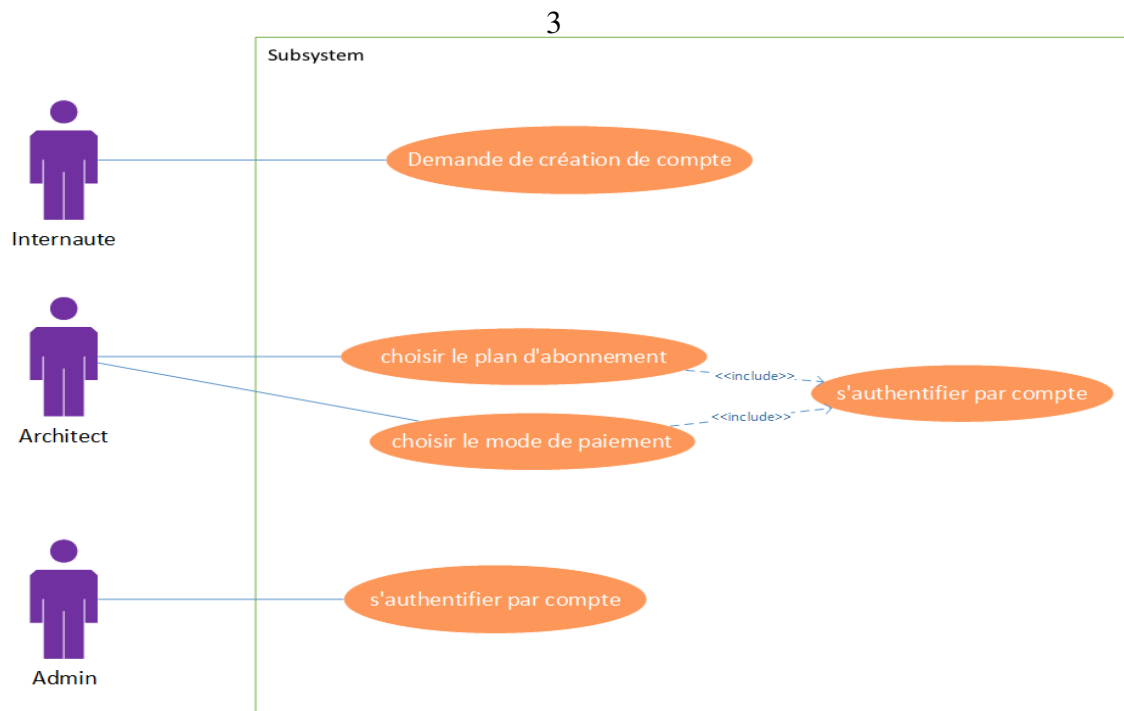


FIGURE 3.1 – Diagramme de cas d'utilisation du Sprint1

### 3.2.2.2 Diagramme de classe détaillé

Nous allons présenter le diagramme de classe du sprint 1 dans la figure 3.2 ci-dessous.

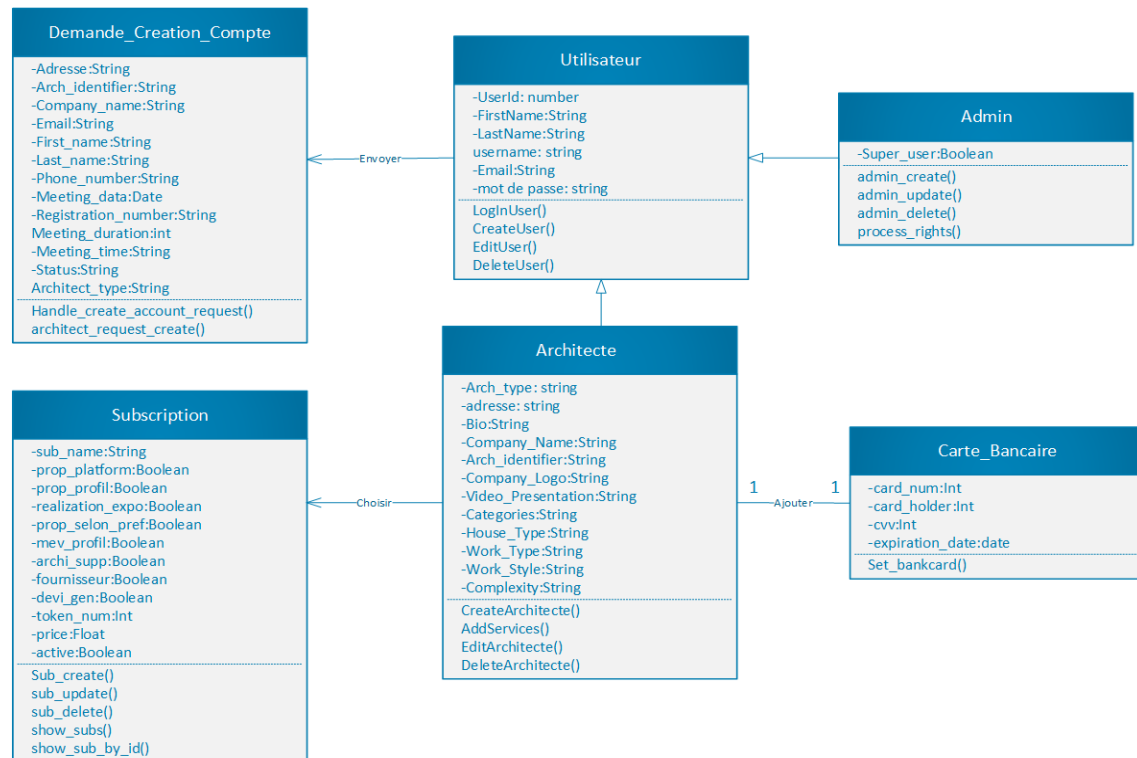


FIGURE 3.2 – Diagramme de classe Release 1 sprint 1

### 3.2.2.3 Diagramme de séquence du scénario « Demande de création de compte architecte »

La figure 3.3 ci-dessous illustre le diagramme de séquence du scénario « Demande de création de compte architecte » qui montre comment soumettre une demande de création de compte architecte.

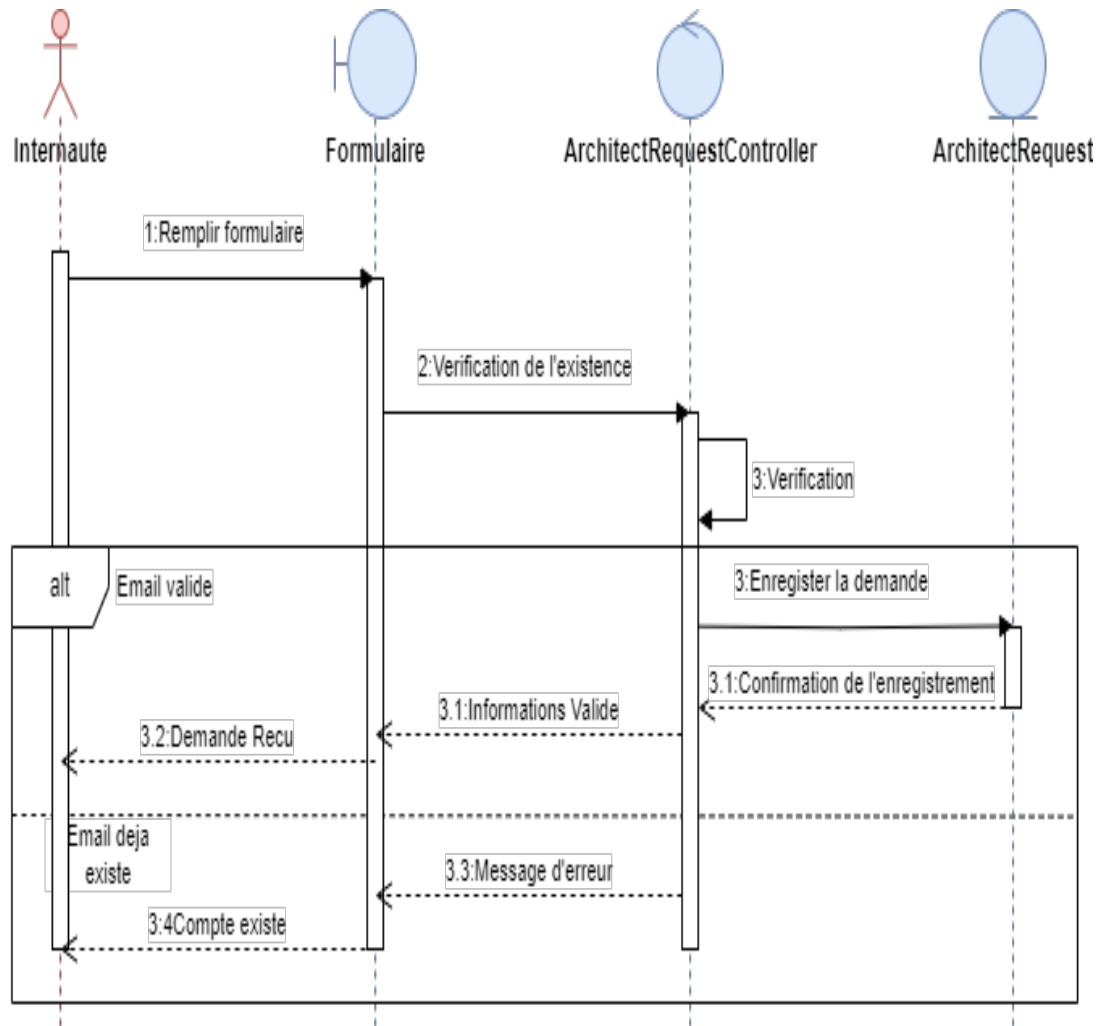


FIGURE 3.3 – Diagramme de séquence Demande de création de compte architecte

#### 3.2.2.4 Diagramme de séquence du scénario « Première connexion architecte »

La figure 3.4 ci-dessous illustre le diagramme de séquence du scénario « Première connexion architecte » qui montre les étapes du premier connexion architecte.

### CHAPITRE 3. RÉALISATION DU RELEASE 1

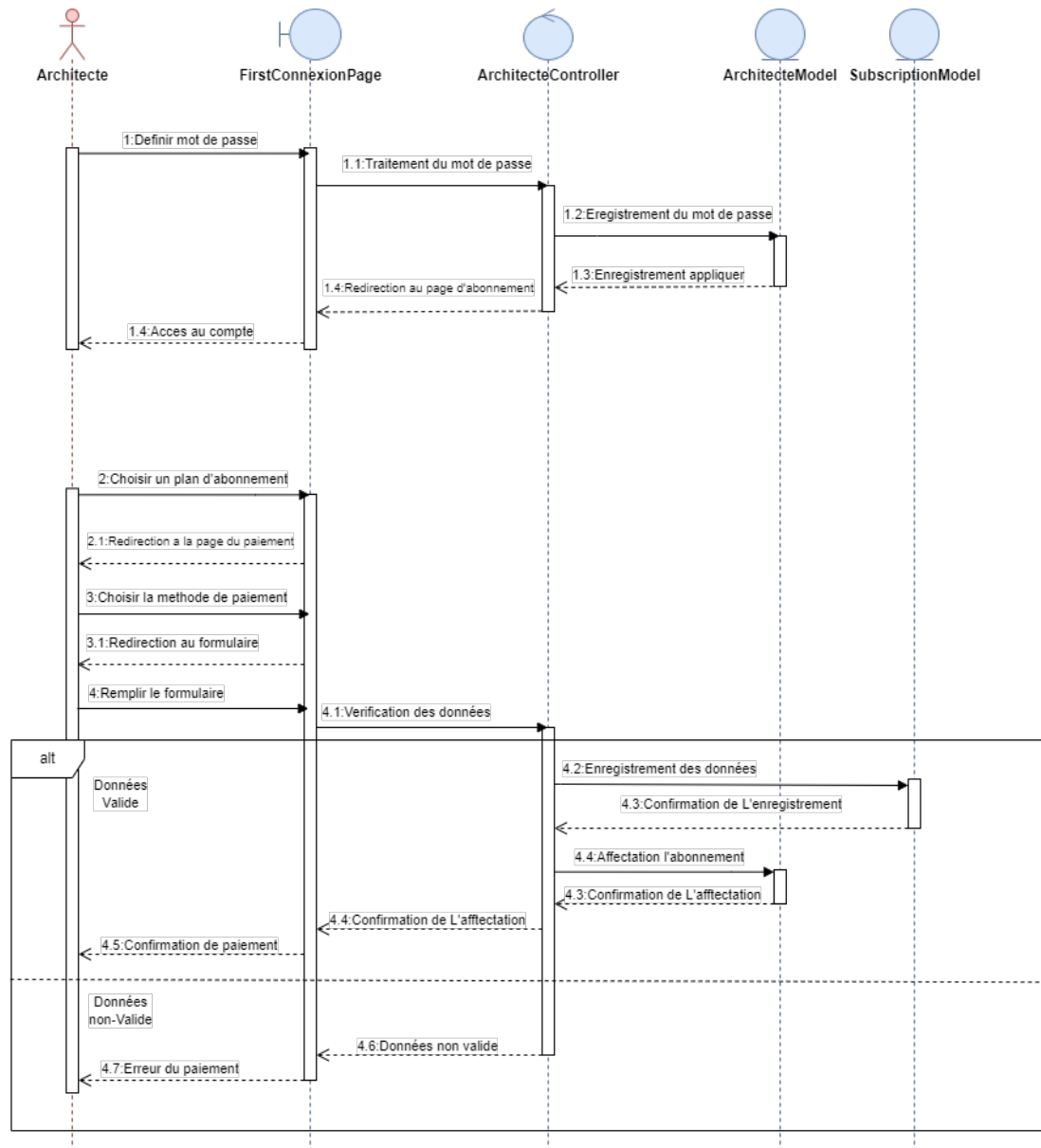


FIGURE 3.4 – Diagramme de séquence Première connexion architecte

## 3.3 Sprint 2

### 3.3.1 Backlog du sprint 2

Durant cette partie nous allons détailler les différentes tâches à effectuer afin d'atteindre les objectifs de sprint 2.

Nom	Description	Estimation
Gestion des Admin et permission	<ul style="list-style-type: none"> <li>• En tant que Super admin je peux créer un nouveau compte pour les sous admins.</li> <li>• En tant que Super admin, je peux attribuer des permissions spécifiques à chaque sous admin.</li> <li>• En tant que Super admin, je peux supprimer un sous admin.</li> <li>• En tant que Super admin, je peux visualiser les détails des sous admins.</li> <li>• En tant que Super admin, je peux modifier les informations des sous admins.</li> </ul>	jours
Gestion des réunion	<ul style="list-style-type: none"> <li>• En tant qu'administrateur, je peux gérer mes indisponibilités en utilisant un calendrier.</li> <li>• En tant qu'administrateur, je peux visualiser et filtrer les réunions planifiées par mois, semaine et jour.</li> <li>• En tant qu'administrateur, je peux accéder à une liste détaillée des réunions prévues pour la journée.</li> <li>• En tant qu'administrateur, je peux planifier des entretiens d'acceptations.</li> </ul>	jours

TABLE 3.2 – Backlog Sprint 2



### 3.3.2 Analyse des besoins

#### 3.3.2.1 Diagramme de cas d'utilisation détaillé

Dans la figure 3.5 ci-dessous, nous illustrons le diagramme de cas d'utilisation du sprint 2.

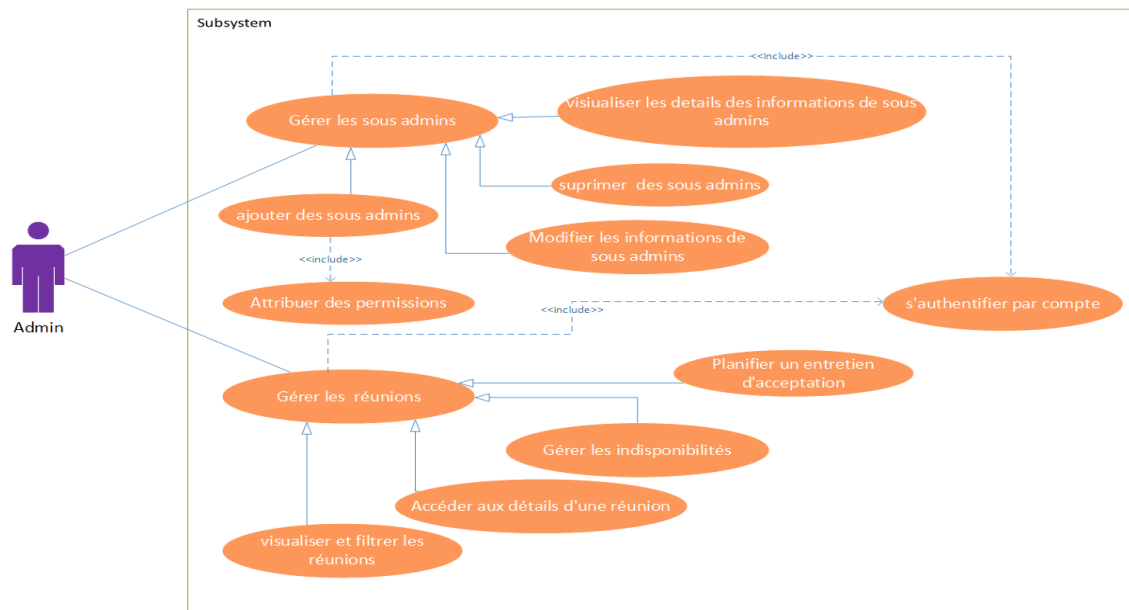


FIGURE 3.5 – Diagramme de cas d'utilisation du Sprint2

#### 3.3.2.2 Diagramme de classe détaillé

Nous allons présenter le diagramme de classe du sprint 2 dans la figure 3.6 ci-dessous.

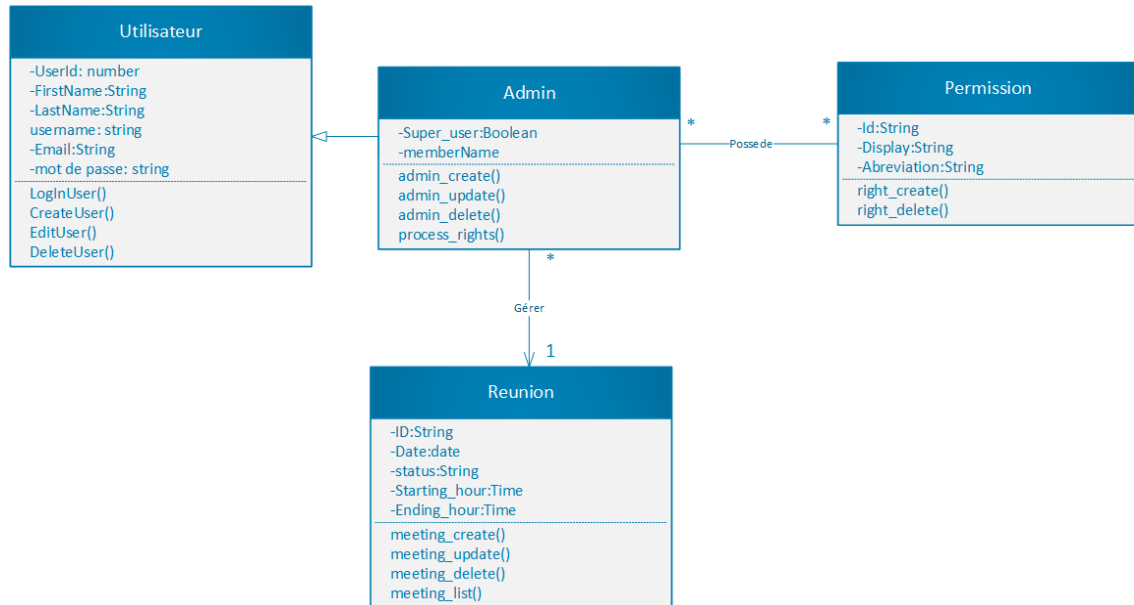


FIGURE 3.6 – Diagramme de classe Release 1 sprint 2

### 3.3.2.3 Diagramme de séquence du scénario «Gestion des Admin et permission »

La figure 3.7 ci-dessous illustre le diagramme de séquence du scénario « Gestion des Admin et permission » qui montre les étapes de l'ajout d'un administrateur.

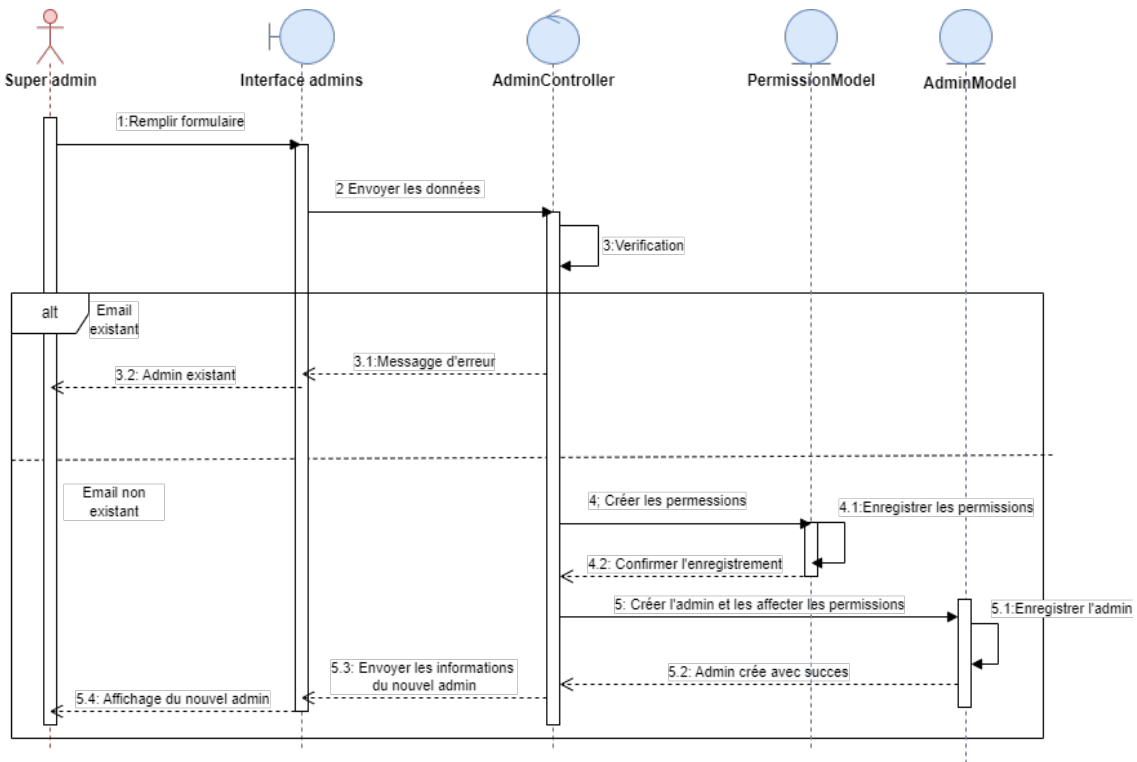


FIGURE 3.7 – Diagramme de séquence Gestion des Admin et permission

### 3.3.2.4 Diagramme de séquence du scénario « Gestion des réunions »

La figure 3.8 ci-dessous illustre le diagramme de séquence du scénario « Gestion des réunions » qui montre les étapes de l'ajout d'une réunion.

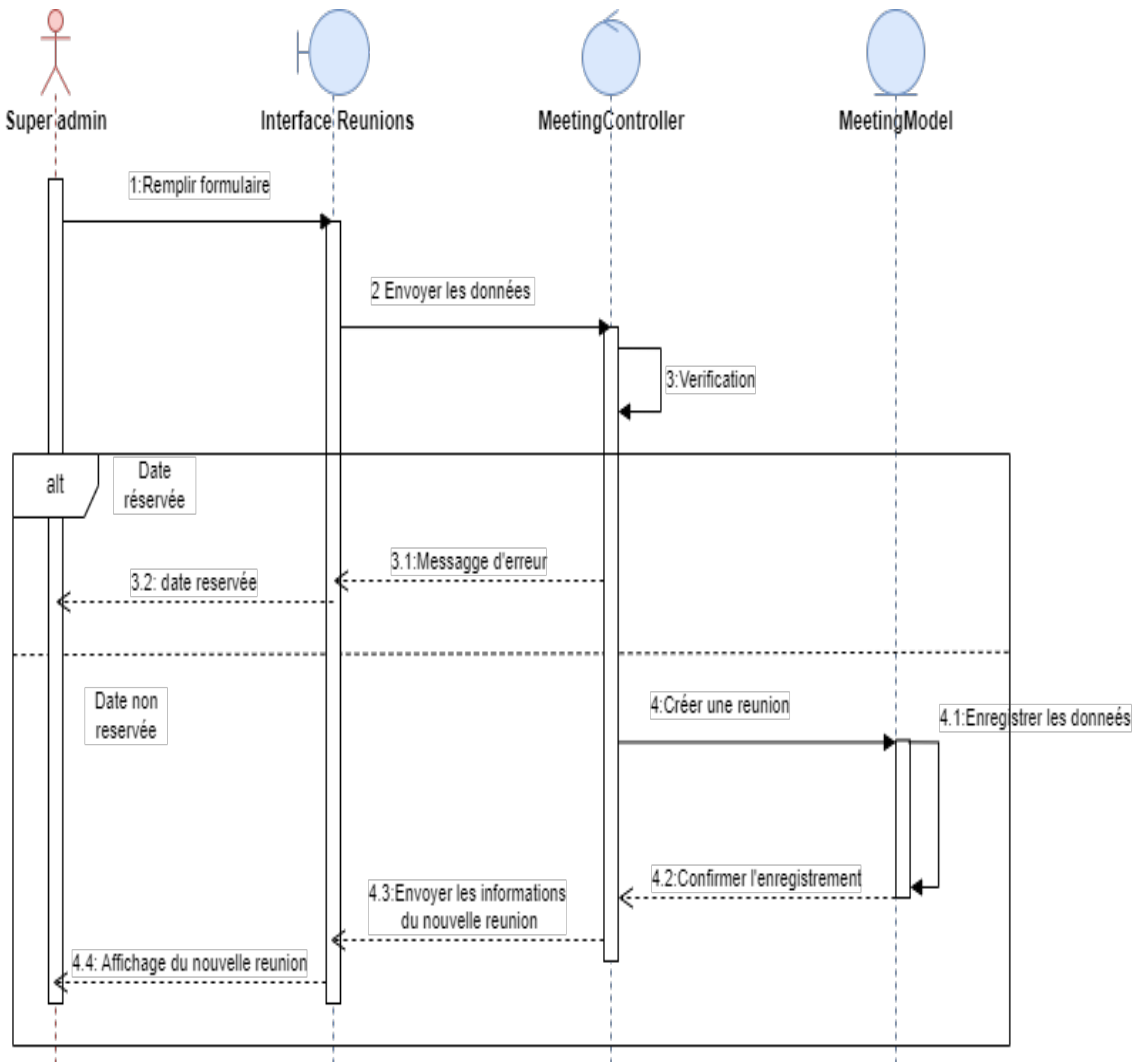


FIGURE 3.8 – Diagramme de séquence Gestion des réunion

## 3.4 Sprint 3

### 3.4.1 Backlog du sprint 3

Durant cette partie nous allons détailler les différentes tâches à effectuer afin d'atteindre les objectifs de sprint 3.

Nom	Description	Estimation
Gestions des demandes de création de compte	<ul style="list-style-type: none"><li>• En tant qu'administrateur, je peux Accepter les nouveaux demande de creation de comptes d'architectes.</li><li>• En tant qu'administrateur, je peux Refuser les demande de creation de comptes vérifiés.</li><li>• En tant qu'administrateur, je peux Consulter les demande de creation de comptes vérifiés.</li></ul>	jours
Gestions des abonnement coté Admin	<ul style="list-style-type: none"><li>• En tant qu'administrateur, je peux ajouter/modifier/supprimer un plan d'abonnement sur la plateforme.</li><li>• En tant qu'administrateur, je peux voir la liste des demandes d'abonnement soumises par les architectes.</li><li>• En tant qu'administrateur, je peux valider/annuler l'activation l'abonnement après vérification du paiement par virement.</li></ul>	jours

TABLE 3.3 – Backlog Sprint 3

## 3.4.2 Analyse des besoins

### 3.4.2.1 Diagramme de cas d'utilisation détaillé

Dans la figure 3.9 ci-dessous, nous illustrons le diagramme de cas d'utilisation du sprint 3.

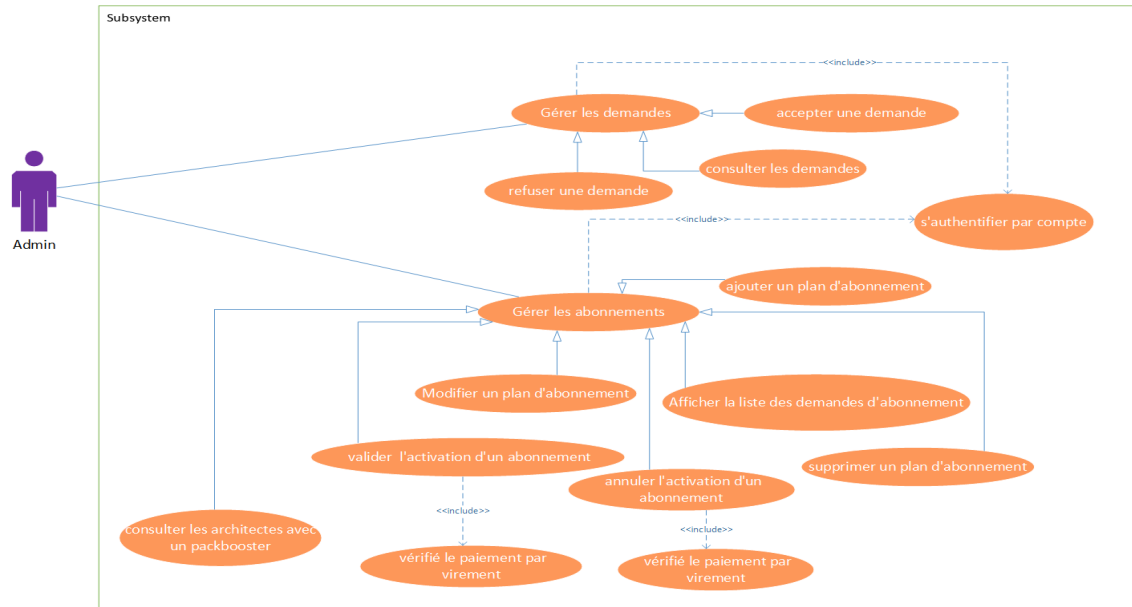


FIGURE 3.9 – Diagramme de cas d'utilisation du Sprint3

### 3.4.2.2 Diagramme de classe détaillé

Nous allons présenter le diagramme de classe du sprint 3 dans la figure144444444444445 ci-dessous.

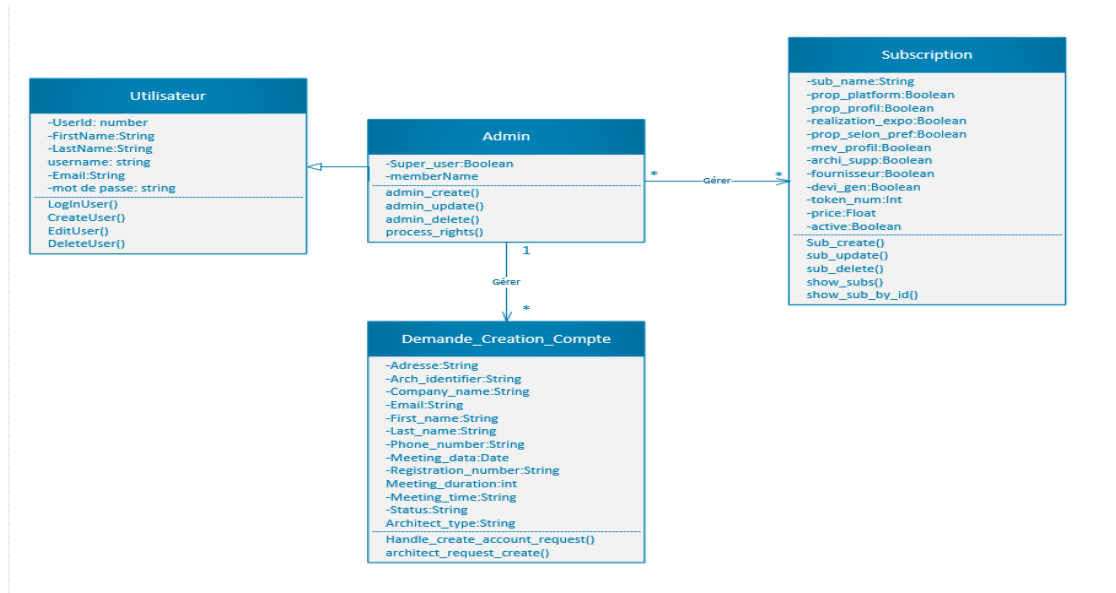


FIGURE 3.10 – Diagramme de classe Release 1 sprint 3

### 3.4.2.3 Diagramme de séquence du scénario « Gestions des demandes de création de compte »

La figure 3.11 ci-dessous illustre le diagramme de séquence du scénario « Gestions des demandes de création de compte » qui montre les étapes de la verification ou le refus d'un demande de creation du compte architecte.

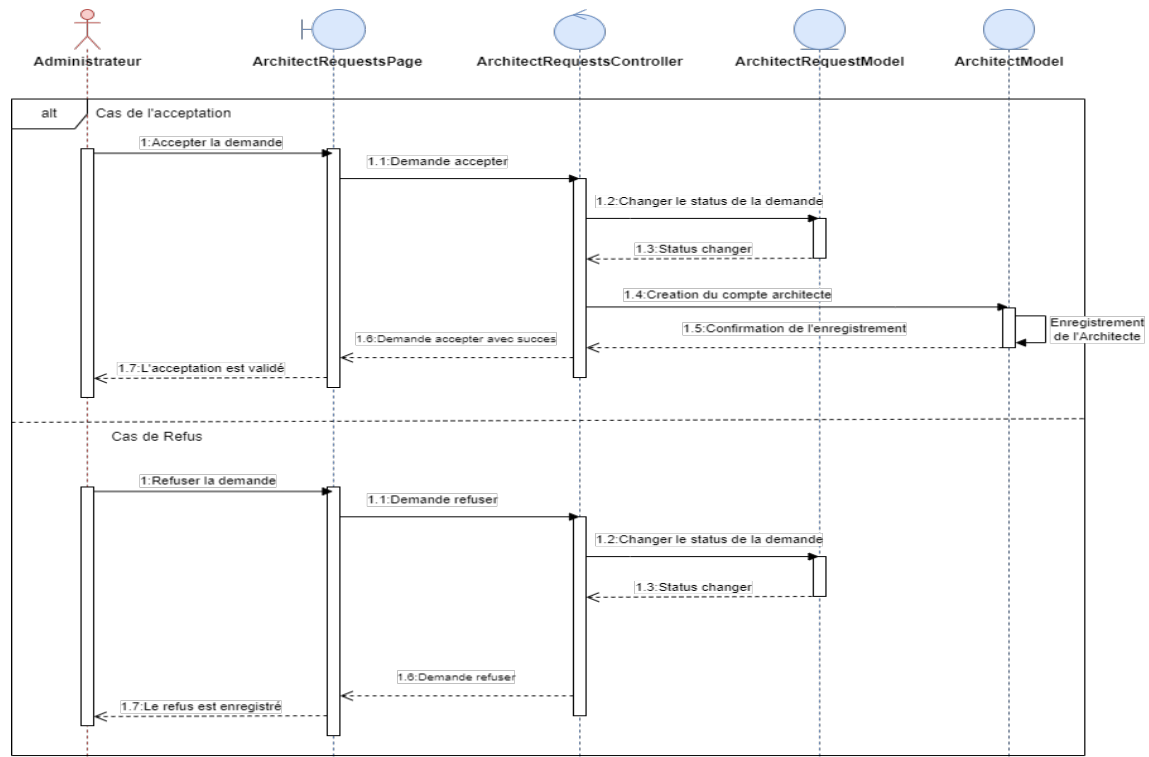


FIGURE 3.11 – Diagramme de séquence Gestions des demandes de création de compte

### 3.4.2.4 Diagramme de séquence du scénario « Gestions des abonnement coté Admin »

La figure 3.12 ci-dessous illustre le diagramme de séquence du scénario « Gestions des abonnement coté Admin » les étapes de creation d'un plan d'abonnement.



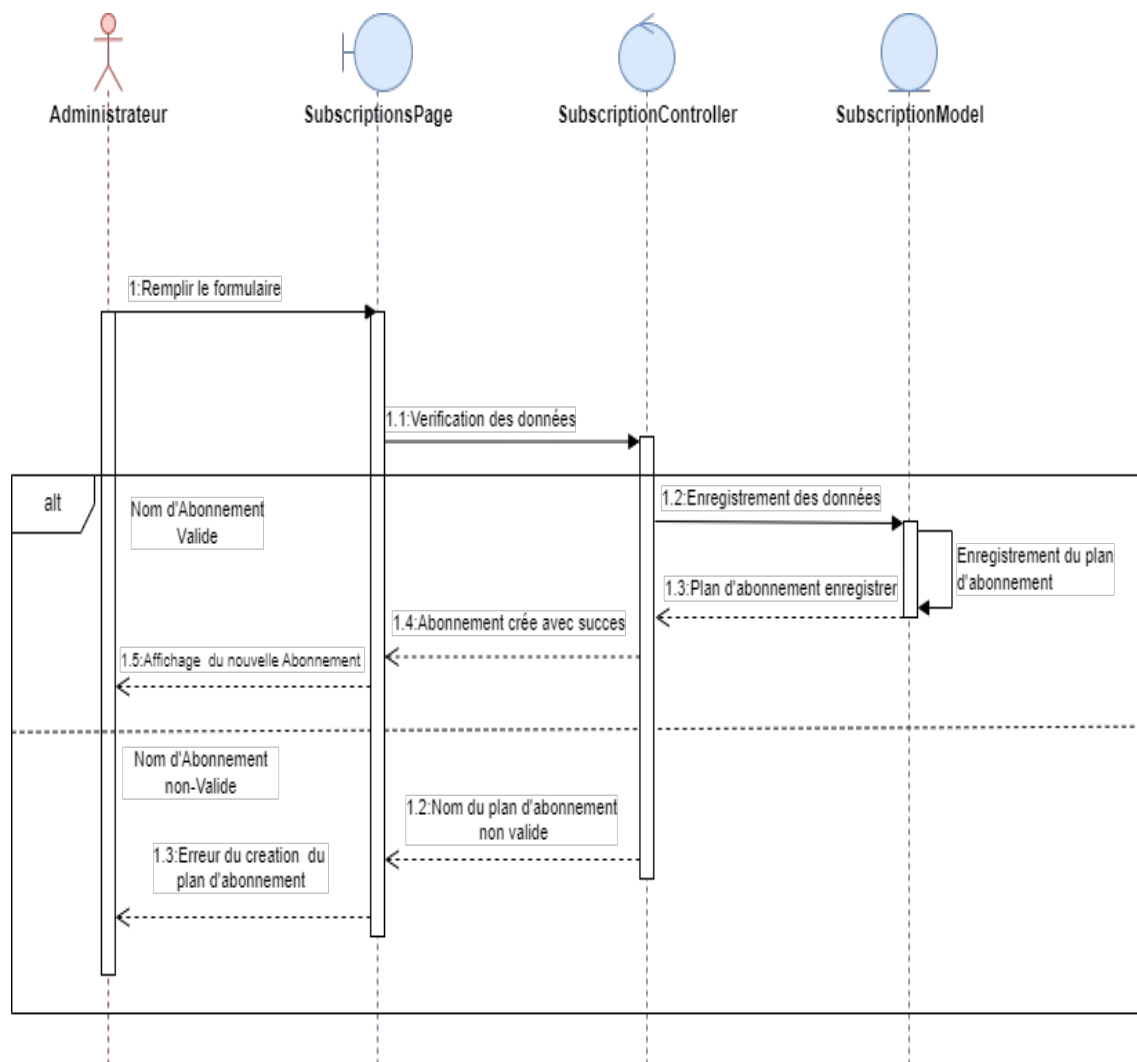


FIGURE 3.12 – Diagramme de séquence Gestions des abonnement coté Admin

# Webographie

- [1] *Cleverttech*. URL : <http://www.cleverttech-france.fr>.
- [2] *ClickUp*. URL : <https://en.wikipedia.org/wiki/ClickUp>.
- [3] *Django*. URL : <https://en.wikipedia.org/wiki/Django>.
- [4] *Elasticsearch*. URL : <https://en.wikipedia.org/wiki/Elasticsearch>.
- [5] *GitHub*. URL : <https://en.wikipedia.org/wiki/GitHub>.
- [6] *NEXTJS*. URL : <https://fr.wikipedia.org/wiki/Next.js>.
- [7] *PostgreSQL*. URL : <https://en.wikipedia.org/wiki/PostgreSQL>.
- [8] *SCRUM*. URL : <https://www.scrum.org/resources/what-is-scrum/>.
- [9] *UML*. URL : [https://fr.wikipedia.org/wiki/UML\\_\(informatique\)](https://fr.wikipedia.org/wiki/UML_(informatique)).