



**Institut Provincial Supérieur
d'Enseignement
de Promotion Sociale de
Seraing**

Campus Verviers

Enseignement Supérieur de Type Court

Bachelier en Informatique de Gestion

Épreuve Intégrée

Sujet : *Gestion de projet*

Nom : *SOUSSI*

Prénom : *Sena*

Année académique 2022-2023



Dédicaces

Je commence par rendre grâce à DIEU qui m'avoir donné le courage et la
Patience pour arriver à ce stade.

Du profond de mon cœur, je dédie ce modeste travail à tous ceux qui me sont

Chers

A ma très chère mère

A mon très cher père

Qui n'ont jamais cessé, de formuler des prières à mon égard, de me soutenir et
de m'épauler pour que je puisse atteindre mes objectifs.

A mes chères sœurs

A mon cher frère

Pour ses soutiens moraux et leurs conseils précieux tout au long de mes études.

A mes chér(e)s ami(e)s

Pour leurs aides et supports dans les moments difficiles.

À toute l'équipe Delomid, qui m'ont encouragé et soutenu durant la

Période de ce stage en particulier Amir et Malak et Yannick .

Je dédie ce travail à tous ceux qui m'aiment et tous ceux que j'aime ..

Que Dieu vous bénisse ∞



Remerciements

Je souhaite également remercier mes professeurs et conseillers académiques qui ont su m'orienter et me conseiller tout au long de mon parcours, ainsi que lors de la réalisation de ce projet.

Un remerciement particulier va à ma famille et à mes amis pour leur soutien moral, leur encouragement et leur foi en mes capacités, sans lesquels ce projet n'aurait pas été possible.

Je remercie également tous ceux qui ont contribué de près ou de loin à la réalisation de ce projet, que ce soit par leur temps, leurs connaissances ou leurs encouragements

Table des matières

1.Contexte général.....	3
Introduction	10
1.1 Présentation de l'organisme d'accueil	10
1.1.1 Aperçu général de l'entreprise	10
1.1.2 Organigramme de l'entreprise	11
1.1.3 Objectifs de stage	12
1.1.4 Problèmes rencontrés	12
1.2 Objectifs du projet individuelle	14
1.3 Solutions proposées	14
1.4 Méthodologie et Langage de Modélisation Utilisés.....	14
1.4.1 Le processus unifié.....	14
1.4.2 Méthodologie Adoptée : Processus Unifié.....	15
1.4.3 Cycle de vie du Processus Unifié dans Mon Projet	15
1.4.4 Enchaînement d'activités	16
1.4.5 Diagramme de Gantt	17
<u>2.Analyse et spécification des besoins.....</u>	19
2.1 Spécifications des besoins	19
2.1.1 Identification des acteurs	19
2.1.2 Spécifications des besoins fonctionnels.....	20
2.2 Diagramme du cas d'utilisation général :	21
3.Conception	23
Introduction	23
3.1 Raffinement et spécification des cas d'utilisation	23
3.1.1 Raffinement du cas d'utilisation : S'inscrire et S'authentifie	23
3.1.1.1 Description textuelle de cas d'utilisation : S'inscrire.....	23
3.1.1.2 Description textuelle de cas d'utilisation : S'authentifier	24
3.1.2 Raffinement du cas d'utilisation : Gérer compte utilisateur	25
3.1.3 Raffinement du cas d'utilisation : Gérer projets	26
3.1.3.1 Description textuelle de cas d'utilisation : Ajouter projet.....	26
3.1.3.2 Description textuelle de cas d'utilisation : Consulter projet	27
3.1.3.3 Description textuelle de cas d'utilisation : Supprimer équipe.....	27
3.1.4 Raffinement du cas d'utilisation : Gérer tâches	28
3.1.4.1 Description textuelle de cas d'utilisation : Ajouter tâche	28
3.1.5 Raffinement du cas d'utilisation : Créer compte clients	29
3.1.5.1 Description textuelle de cas d'utilisation : Ajouter client	30
3.2 Diagramme de classes	30
3.3 Schéma relationnel de la base de données.....	31
3.4 Architecture à envisager : Le Modèle MVC/MVT	32
3.5 Diagramme de séquence.....	34
3.5.1 Définition	34
3.5.2 Diagramme de séquence de CU : S'authentifier	35
3.5.3 Diagramme de séquence de CU : Ajouter Projet	35
3.5.4 Diagramme de séquence de CU : Modifier Projet	36
3.5.5 Diagramme de séquence de CU : Ajouter tâche	37

3.5.6 Diagramme de séquence de CU : Ajouter client	38
<u>4.Réalisation</u>	40
Introduction	40
4.1 Environnement logiciel :	40
4.1.1 Environnement de développement	40
4.1.2 Environnement de conception	41
4.1.3 Environnement de création logo	41
4.1.3 Les Outils de Base des données :	41
4.2 Les frameworks et les langages de programmation	42
4.2. python	42
4.2.2 JS	42
4.2.3 Pusher	42
4.3 Les interfaces de l'application.....	42
4.3.1 Interface Inscription	42
4.3.2 Interface connexion	43
4.3.3 Interface Client	44
4.3.3.1 Mail	44
4.3.4 Interface développeur	45
4.3.4 .1 Page d'accueil	45
4.3.4.2 Consulter projet	45
4.3.4.3 Gérer message	46
4.3.4.4 Consulter notification de message.....	47
4.3.4.5 Consulter pipeline	47
4.3.4.6 Modifier pipeline	47
4.3.4.7 Ajouter tâche	48
4.3.4.8 Modifier tâche par pipeline	49
4.3.5 Interface Administrateur	50
4.3.5.1 Page d'accueil	50
4.3.5.2 Ajouter projet	50
4.3.5.3 Créer client	51
4.3.5.4 Consulter utilisateur	52
Conclusion.....	53
Conclusion générale et perspectives.....	54



Table des figures

<i>Figure 1.1.1. Logo de l'entreprise</i>	<i>11</i>
<i>Figure 1.1.1 Organigramme de l'entreprise</i>	<i>12</i>
<i>Figure 1.0.1 cycle de développement</i>	<i>16</i>
<i>Figure 1.1.4.4.1 Enchainements d'activités au cours du cycle de vie</i>	<i>16</i>
<i>Figure 1.0.2 UML (Unified Modeling Language).....</i>	<i>17</i>
<i>Figure 2.2.1Diagramme du cas d'utilisation général.....</i>	<i>21</i>
<i>Figure 3.1.1.1.1Raffinement de cas d'utilisation : S'inscrire</i>	<i>23</i>
<i>Figure 3.1.2.1Raffinement de cas d'utilisation : Gérer compte utilisateur</i>	<i>25</i>
<i>Figure 3.2.1 Diagramme de classes général.....</i>	<i>31</i>
<i>Figure 0.1 Schémas de l'architecture MVT.....</i>	<i>34</i>
<i>Figure 3.10.1 Diagramme de séquence de CU : Ajouter tâche</i>	<i>38</i>
<i>Figure 3.10.1Diagramme de séquence de CU : Ajouter client</i>	<i>39</i>
<i>Figure 4.0.1 Interface Inscription</i>	<i>43</i>
<i>Figure 4.0.10 Modifier pipeline</i>	<i>48</i>
<i>Figure 4.0.11 Ajouter tâche</i>	<i>49</i>
<i>Figure 4.0.14 Interface Administrateur</i>	<i>50</i>
<i>Figure 4.0.17 Ajouter projet.....</i>	<i>51</i>
<i>Figure 4.0.18 Créer client.....</i>	<i>52</i>
<i>Figure 4.19 Consulter utilisateur</i>	<i>53</i>



Liste des tableaux

Tableau 3-1 Raffinement de cas d'utilisation : S'inscrire	24
Tableau 3-2 Description de cas d'utilisation « S'authentifier »	25
Tableau 3-3 Description de cas d'utilisation « Refuser compte utilisateur »	25
Tableau 3-4 Description de cas d'utilisation « Ajouter projet ».	27
Tableau 3-5 Description de cas d'utilisation « Consulter projet »	27
Tableau 3-6 Description de cas d'utilisation « Supprimer équipe »	28
Tableau 3-7 Description de cas d'utilisation « Ajouter tâche ».	29
Tableau 3-8 Description textuelle de cas d'utilisation : Ajouter client	30



Introduction générale

L'Internet est le meilleur moyen de communiquer là-bas. Des nouvelles innovations le rendent plus rapide et plus fiable. Aujourd'hui, nous pouvons facilement communiquer avec n'importe qui dans le monde. La communication est devenue beaucoup plus facile qu'auparavant avec l'aide de diverses applications.

L'ère informatique a sensibilisé à la gestion d'entreprise en permettant l'automatisation de catégories des tâches qui étaient auparavant effectuées manuellement. En fait, les chefs de projet étaient satisfaits des programmes populaires tels que les feuilles de Calcul.

Le logiciel de gestion de projet est né dans les années 1980 pour regrouper toutes les fonctionnalités dont vous avez besoin dans un seul programme. À cette époque, le concept de réseaux d'information n'existait pas encore. En outre, ces programmes s'avéraient souvent complexes.

Avec l'avènement d'Internet et du Web, les choses ont complètement évolué. Aujourd'hui, presque tous les systèmes d'entreprise sont connectés au réseau. En conséquence, l'information peut être rapidement distribuée à la fois en interne et en externe, nécessitant un logiciel pour gérer les projets et leurs tâches.

Dans le cadre de ce développement, l'objectif principal de ce travail est de développer une plateforme

Web pour la gestion des projets qui contient 3 espaces, un espace administrateur permettant de gérer tous les comptes, les projets et tous ce qui appartient au projet, un

espace développeur qui permet de gérer projets et un espace client qui permet de consulter leurs projets et envoyer et recevoir des messages.

Ce rapport décrit clairement les différentes étapes pour effectuer les travaux demandés.

Notre démarche est organisée comme suit :

- Le premier chapitre intitulé « Contexte général » qui résume l'organisme d'accueil au sein duquel nous avons réalisé notre projet de fin d'études, le sujet du stage, l'étude de L'existant, la méthodologie
- Le chapitre suivant est intitulé « Analyse et spécification des besoins ». Dans ce chapitre, nous décrivons les besoins fonctionnels et non fonctionnels de notre projet
- Troisième chapitre, s'intitule « Conception » représente la démarche conceptuelle suivie
Pour la réalisation de notre application.
- Finalement dans le dernier chapitre, décrit l'environnement de travail, les outils utilisés et les interfaces principales qui mettent en évidence le fonctionnement de l'application développée.

Le rapport se termine par une conclusion qui résume notre projet et fournit quelques perspectives pour le futur.

Contexte général

Introduction

Dans ce chapitre, nous présentons l'organisme d'accueil et le cadre du projet. Ensuite, nous parlerons de l'objectif du stage et des problèmes rencontrés.

1.1 Présentation de l'organisme d'accueil

1.1.1 Aperçu général de l'entreprise

« **Delomid IT** », est une entreprise innovante et dynamique dans le domaine des technologies de l'information. Delomid IT se spécialise dans la fourniture de solutions informatiques complètes et sur mesure pour les entreprises de toutes tailles.

Parmi ses principaux services on trouve :

- Business intelligence
 - ✓ Intégration de données
 - ✓ Qualité de données
 - ✓ Référentiel client unique
 - ✓ Data visualisation
- Digital Online
 - ✓ Création de site web et d'application
 - ✓ Référencement web et SEO
 - ✓ Rédaction de contenu
 - ✓ Marketing web
 - ✓ Conception graphique

« Delomid IT » a eu le privilège de travailler avec de nombreux clients renommés dans divers secteurs, notamment dans les domaines de la finance, de la santé, de la

vente au détail et des services professionnels (Delhaize, Toyota, Renault, Orange, Electrabel etc.)

Les langages de programmation utilisée actuellement :

- Python / Talend /
- « **Dénomination sociale** :» Delomid IT
- « **Responsable** :» Amir Miandarbandi
- « **Secteur** :» Services et conseil informatiques.
- « **Siège social** :» Rue large 9, 1917 Luxembourg / Rue de la Loi 28, 1000 Belgium
- « **Contact** :» +32 2 544 04 45 (Belgique) / +352 621 47 17 46 (Luxembourg)



Figure 1.1.1. Logo de l'entreprise

1.1.2 Organigramme de l'entreprise

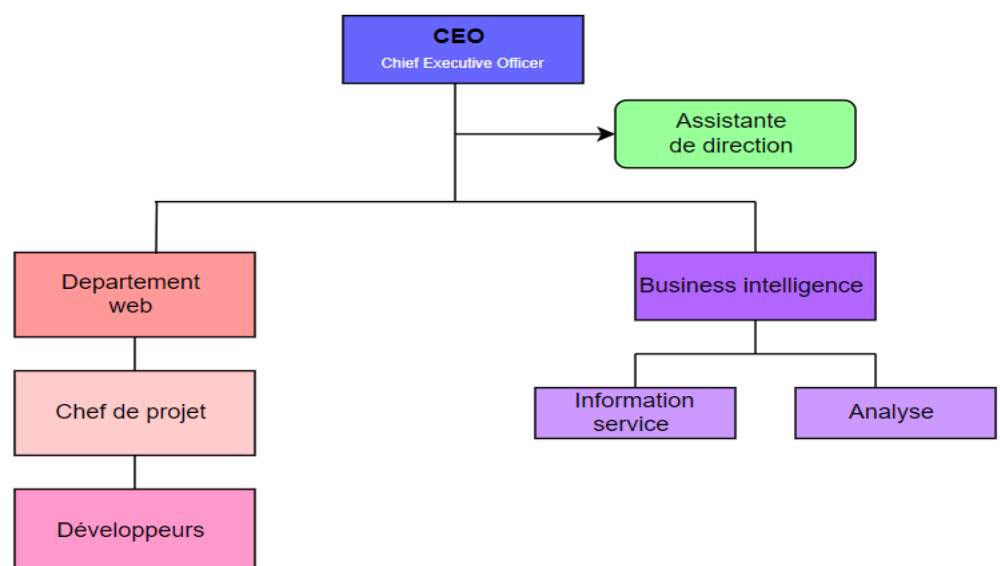


Figure 1.1.1 Organigramme de l'entreprise

1.1.3 Objectifs de stage

- ⇒ Mise en place d'un projet Odoo x Delomid axé sur l'interfaçage avec des logiciels comptables comme Bob ou Sage

Le but de ce stage est de se concentrer sur l'exportation de données de Odoo vers Bob (ou d'autres logiciels comptables comme Sage) et sur l'importation de pièces comptables dans ces logiciels comptables.

Les cas d'usage principaux sont :

- Importer des produits, des listes de clients et de fournisseurs vers Odoo en utilisant une API (fichiers csv).
- Charger des produits, des listes de clients et de fournisseurs vers Odoo en lisant directement un fichier Excel, éliminant ainsi la nécessité d'une saisie manuelle.
- Effectuer du reporting via Odoo en important les données de vente.
- Créer un modèle BI pour les contrôles de gestion.
- Importer des pièces comptables depuis les logiciels comptables.
- Les transactions entre Odoo et les logiciels comptables doivent être automatisées.

1.1.4 Problèmes rencontrés

- Changement inattendu de direction du projet :

Au début de mon stage, j'avais pour mission d'intégrer les produits et les stocks d'Odoo vers différents sites d'annonces. C'était un projet complexe qui nécessitait une compréhension approfondie des API de divers sites d'annonces ainsi que des fonctionnalités d'Odoo.

Après avoir consacré du temps à la recherche, à la planification et au début de la mise en œuvre, j'ai été informé d'un pivot majeur dans la direction du projet. La nouvelle mission était axée sur l'interfaçage entre Odoo et des logiciels comptables comme Bob ou Sage.

- **Impacts du changement :**

➤ **Réorientation des efforts** : J'ai dû rapidement m'adapter pour comprendre les nouvelles exigences et les spécificités de l'interfaçage avec les logiciels comptables.

➤ **Temps investi** : Le travail initial et les efforts investis dans la première direction du projet ne pouvaient pas être intégralement exploités pour le nouveau projet. Cela a également posé des défis en termes de gestion du temps et de respect des échéances.

➤ **Adaptabilité technique** : Changer de projet a exigé une courbe d'apprentissage pour comprendre et s'adapter aux nouvelles technologies et API associées aux logiciels comptables.

1.2 Objectifs du projet individuelle

Je propose de développer une application dédiée à la gestion de projet. Cette application aura pour but de planifier les pipelines et de faciliter les interactions entre les développeurs au sein de la société. De plus, elle permettra de mettre le client en contact avec chaque étape du projet.

1.3 Solutions proposées

Face aux défis mentionnés précédemment, j'ai été inspiré tout au long de ce projet par différentes plateformes existantes pour répondre de manière optimale aux besoins de ce service. Ma solution repose principalement sur la conception d'une application web. Cette application permet à l'utilisateur de créer son propre compte et de gérer les messages par projet.

L'outil que j'ai développé donne au chef de projet la capacité de gérer les projets, les pipelines, les tâches associées, et l'équipe du projet. Il offre également aux développeurs la possibilité de recevoir des notifications en temps réel lorsqu'ils sont ajoutés à une tâche, de suivre le projet, et de gérer leurs tâches assignées. De plus, cette application permet aux clients de suivre l'avancement du projet en temps réel.

1.4 Méthodologie et Langage de Modélisation Utilisés

Durant la réalisation de mon projet, j'ai choisi une méthodologie qui m'assurerait un résultat fiable. En particulier, j'ai opté pour la méthodologie du "processus unifié". Cette approche m'a permis de maintenir un contrôle solide sur le projet, même lorsque sa complexité augmentait.

1.4.1 Le processus unifié

Le processus unifié est une méthode de développement pour les logiciels orientés objets. C'est une méthode générique, itérative et incrémentale, contrairement à la méthode séquentielle, Merise. Il s'agit d'un guide méthodologique permettant d'éclaircir les différentes étapes du processus afin de réduire la complexité du projet et d'améliorer la qualité du livrable tout en ayant le contrôle pour prendre en charge les modifications qui interviennent au cours du développement sans qu'elles ne posent de problèmes.

1.4.2 Méthodologie Adoptée : Processus Unifié

Dans le cadre de mon projet, j'ai opté pour le processus unifié en raison de ses caractéristiques distinctives qui correspondent bien aux exigences et à la nature de mon projet. Voici les principales caractéristiques du processus unifié et leur pertinence pour mon travail :

- **Guidé par les cas d'utilisation** : J'ai basé mon travail sur les exigences des utilisateurs pour définir des cas d'utilisation précis. Cette approche m'a aidé à comprendre et à prioriser les besoins de chaque utilisateur.
- **Centré sur l'architecture** : La vision claire de l'architecture dès le début m'a permis d'anticiper la mise en œuvre de tous les cas d'utilisation. Cela a facilité la réalisation et l'adaptation du projet à mesure qu'il évoluait.
- **Itérative et incrémentale** : J'ai abordé mon projet par étapes, en me concentrant sur des parties spécifiques lors de chaque itération. À la fin de chaque phase, j'évaluais si les objectifs définis avaient été atteints.

1.4.3 Cycle de vie du Processus Unifié dans Mon Projet

J'ai organisé le développement de mon projet en suivant le cycle de vie du processus unifié, qui se divise en quatre phases majeures. Voici comment j'ai adapté et appliqué ces phases à mon projet :

- **Phase de création** : Permet de définir le produit et les objectifs du projet ;
- **Phase d'élaboration** : se à clarifier les besoins, définir l'architecture du produit et à en valider la faisabilité
- **Phase de construction** : Vise à réaliser et mettre en œuvre le produit et les livrables associés
- **Phase de transition** : Vise à livrer le produit de sorte qu'il soit prêt à être utilisé.

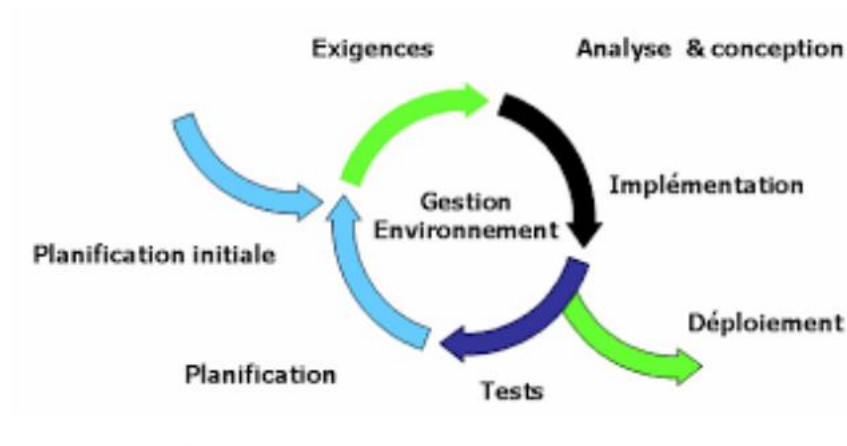


Figure 1.0.1 cycle de développement

1.4.4 Enchaînement d'activités

Le processus unifié propose les enchaînements d'activité suivants :

- **Exigences** : Recherche des acteurs et des cas d'utilisation, priorisation, description des cas d'utilisation, prototypages des interfaces utilisateur et structuration du modèle des cas d'utilisation ;
- **Analyse** : Analyse de l'architecture, des cas d'utilisation et des classes ;
- **Conception** : Conception de l'architecture, des cas d'utilisation, des classes et d'un sous-système ;
- **Implémentation** : Implémentation de l'architecture et d'un sous-système ;
- **Tests** : Planification, conception et exécution des tests

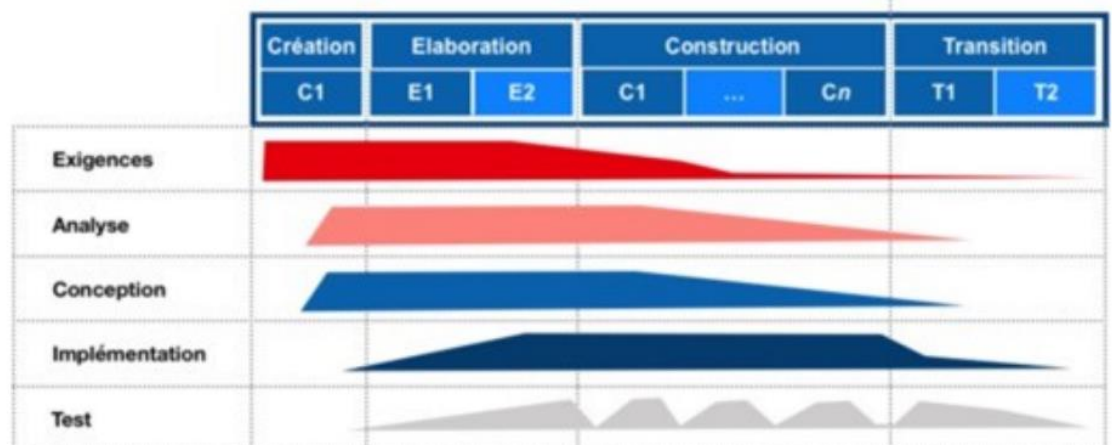


Figure 1.1.4.4.1 Enchaînements d'activités au cours du cycle de vie

1.4.5 Diagramme de Gantt

La planification est une des étapes importantes avant de commencer le développement du projet. Elle m'a aidé à déterminer, à ordonner les tâches nécessaires et à estimer le temps requis pour chacune d'elles. Pour assurer une planification efficace et un suivi rigoureux de l'avancement de mon projet, j'ai choisi d'utiliser le diagramme de Gantt. C'est un outil qui facilite non seulement la planification, mais offre aussi une visualisation claire de la séquence et de la durée des différentes tâches au cours de la réalisation du projet.

1.5 UML

UML, est un langage de modélisation normalisé basé sur les concepts orientés objet et peut être utilisé par toutes les méthodes de développement orienté objet. UML offre un standard de modélisation afin de visualiser graphiquement, décrire et spécifier le système logiciel à développer



Figure 1.0.1 UML (Unified Modeling Language)

Dans le cadre de la modélisation UML, bien qu'il existe 14 types de diagrammes, j'ai choisi d'utiliser les quatre diagrammes suivants pour mon projet, car ils étaient les plus pertinents pour représenter les aspects essentiels de l'application :

- **Diagramme de cas d'utilisation** : Pour modéliser les interactions entre les utilisateurs et le système.
- **Diagramme de classes** : Pour définir la structure de l'application.

- **Diagramme de séquence** : Pour représenter les interactions entre objets ou composants.

L'un des principaux avantages de cette méthodologie est qu'elle offre une vision abstraite de haut niveau. Cette abstraction facilite grandement la communication entre les différents acteurs du projet, qu'il s'agisse des utilisateurs, des spécialistes métier ou des développeurs.

Conclusion

Ce chapitre a défini le cadre général du projet et a offert un aperçu de l'état actuel des solutions similaires. De plus, il a établi une vue d'ensemble de notre sujet d'étude. Dans le chapitre suivant, je vais me pencher sur la spécification et l'analyse des besoins de l'application.

Analyse et spécification des besoins

Introduction

Après avoir examiné l'existant, nous entrons dans la phase cruciale de la mise en place des fondations du système. Pour clarifier clairement les besoins des utilisateurs, nous identifierons les acteurs et les cas d'utilisation, et nous définirons les besoins fonctionnels et non fonctionnels.

2.1 Spécifications des besoins

2.1.1 Identification des acteurs

Un acteur est une entité externe interagissant avec le système. Les acteurs peuvent avoir des rôles variés et peuvent intervenir dans divers scénarios. Ci-dessous se trouve une description des acteurs en relation avec le système :

- **L'invité** : Tout utilisateur non inscrit ou tout type d'utilisateur de notre application.
- **Le Client** : Il peut consulter la liste de ses projets, leurs pipelines, leurs tâches, et communiquer avec l'équipe de son projet. Il reçoit également des notifications en temps réel lorsqu'un nouveau message est envoyé.
- **Le développeur** : Utilisateur inscrit, il peut suivre la liste des projets, gérer ses tâches, son profil, ses messages et recevoir des notifications lorsqu'il est ajouté à une tâche ou qu'un nouveau message lui est adressé.

- **Le chef de projet** : Utilisateur inscrit, il a accès à la gestion de son profil, des projets, des pipelines, des tâches, et des messages. Il reçoit aussi des notifications pour de nouveaux messages en temps réel.
- **L'administrateur** : Il est responsable du contrôle de l'application. Il crée les comptes des utilisateurs, gère les projets, les pipelines, les tâches, les messages, et reçoit des notifications en temps réel pour de nouveaux message

2.1.2 Spécifications des besoins fonctionnels

Pour cette application, les cas d'utilisations les plus pertinents sont les suivants :

- **L'invité** :

- **S'inscrire** : Il s'agit de remplir le formulaire d'inscription.

- **Le client** :

- **S'authentifier** : Il s'agit de saisir les informations de connexion.
- **Gérer profile** : Le client peut consulter son profil et le modifier (information divers).
- **Gérer des messages** : Le client peut envoyer et recevoir des messages entre l'équipe du projet pour connaître les nouvelles sur son projet en temps réel et aussi modifier et supprimer son message.
- **Consulter projets** : Le client peut consulter ses projets avec ses informations.

- **Le développeur** :

- Le développeur hérite les mêmes fonctionnalités que le client.
- **Suivi projets** : Le développeur peut suivre les projets.
- **Gérer tâches** : Le développeur peut gérer des tâches.

- **Le chef de projet** :

- Le chef de projet hérite les mêmes fonctionnalités que le développeur.
- **Gérer projets** : Le développeur peut gérer les projets.
- **Gérer pipelines** : Le chef de projet peut gérer pipelines.

- **L'administrateur** :

- L'administrateur hérite les mêmes fonctionnalités que Le chef de projet.
- **Gérer comptes utilisateur** : L'administrateur peut consulter, accepter et refuser les Comptes utilisateurs.
- **Gérer comptes client** : L'administrateur peut créer un compte pour le client et L'envoyer un email qui contient l'email et le mot de passe de son compte.

2.2 Diagramme du cas d'utilisation général :

Les cas d'utilisation décrivent le comportement d'un système du point de vue de l'utilisateur. Ils offrent un aperçu fonctionnel du système, mettant en avant les objectifs et les besoins des utilisateurs par rapport à ce dernier. Ainsi, le diagramme de cas d'utilisation offre une modélisation visuelle de ces interactions et fonctionnalités.

La figure ci-après illustre le diagramme de cas d'utilisation global de notre plateforme, mettant en évidence les principales fonctionnalités associées à chaque acteur.

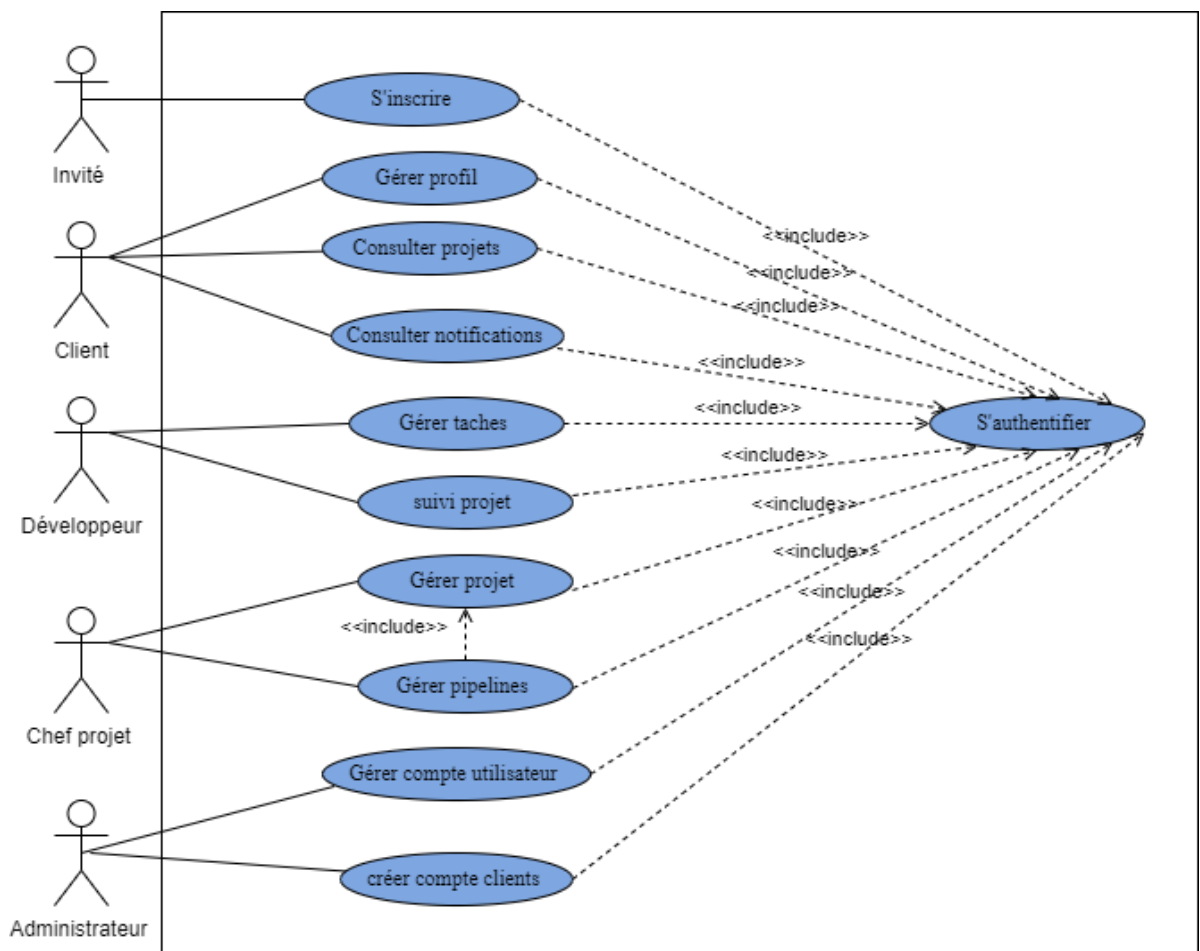


Figure 2.2.1 Diagramme du cas d'utilisation général.

Conclusion

Au terme de ce deuxième chapitre, nous avons minutieusement identifié les acteurs et défini les besoins fonctionnels de mon application. Par ailleurs, nous avons établi et décrit le diagramme de cas d'utilisation, fournissant une vue globale des interactions entre les acteurs

et le système. Dans le chapitre suivant, nous nous attacherons à élaborer l'architecture de notre système et à en détailler la conception.

Cette reformulation rend le texte légèrement plus fluide tout en conservant l'essentiel de votre message.

Conception

Introduction

Dans ce chapitre, je vais parler de comment j'ai conçu mon projet. D'abord, j'ai réfléchi à comment l'application doit fonctionner et j'ai utilisé des outils pour montrer ces idées. J'ai dessiné des schémas qui montrent comment les différentes parties de l'application vont interagir entre elles. Je vais vous montrer ces schémas et vous expliquer ce qu'ils représentent.

3.1 Raffinement et spécification des cas d'utilisation

Dans cette partie nous allons détailler les cas d'utilisation les plus prioritaires en décrivant, dans chaque cas, les préconditions et les postconditions ainsi que le scénario principal, et Éventuellement les exceptions liées à ce cas.

3.1.1 Raffinement du cas d'utilisation : S'inscrire et S'authentifie

3.1.1.1 Description textuelle de cas d'utilisation : S'inscrire

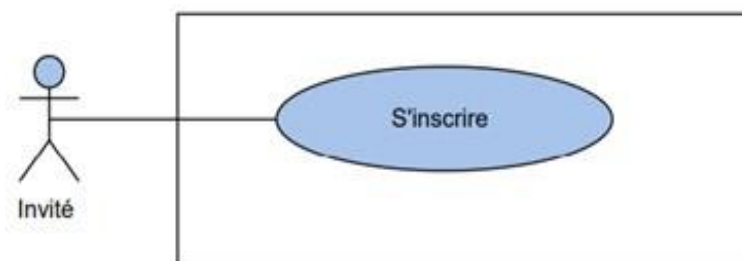


Figure 3.1.1.1 Raffinement de cas d'utilisation : S'inscrire

Cas d'utilisateur	S'inscrire
Résumé	L'utilisateur introduit ses données personnelles pour avoir un compte
Acteur	Invité
Préconditions	Opération inscription choisi
Post-condition	Utilisateur inscrit
Scénario nominale	<ol style="list-style-type: none"> 1. L'utilisateur demande un formulaire D'inscription 2. L'utilisateur introduit ses données personnelles 3. L'utilisateur clique sur le bouton Inscription 4. Le système vérifie la saisie 5. Le système enregistre l'utilisateur sur la plateforme 6. Le système affiche espace login
Exception	<p>4.1 Données non valides :</p> <p>(a) Message d'erreur est affiché par le système</p> <p>(b) Retour l'étape 2</p>

Tableau 3-0-1 Raffinement de cas d'utilisation : S'inscrire

3.1.1.2 Description textuelle de cas d'utilisation : S'authentifier

Cas d'utilisation	S'authentifier.
Résumé	L'acteur introduit son username et son mot de passe pour accéder au système.
Acteur	L'utilisateur (Administrateur, développeur et client).
Précondition	L'acteur doit avoir un username et un mot de passe.

Post-condition	Acteur authentifié.
Scénario nominal	<ol style="list-style-type: none"> 1. L'utilisateur clique sur le bouton se connecter. 2. Le système affiche l'interface d'authentification 3. L'acteur saisi son email et son mot de passe. 4. Le système vérifie les données entrées par l'utilisateur 5. Le système affiche l'interface accueil.
Exception	<p>4.1 username ou mot de passe erronés.</p> <p>(a) Message d'erreur affiché par le système.</p> <p>(b) Retour à l'étape 3.</p>

Tableau 3-0-2 Description de cas d'utilisation « S'authentifier »

3.1.2 Raffinement du cas d'utilisation : Gérer compte utilisateur

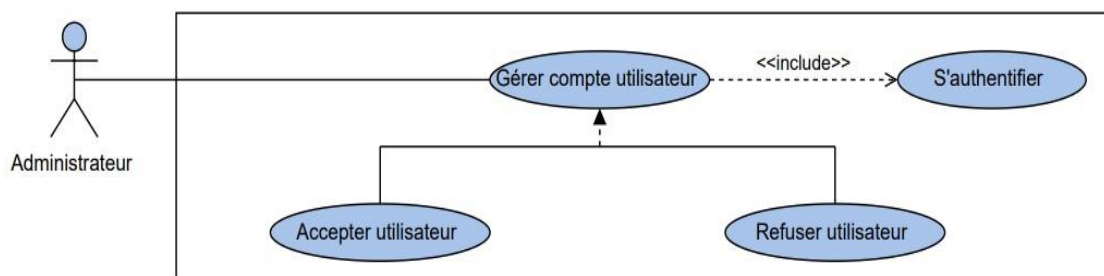


Figure 3.1.2.1 Raffinement de cas d'utilisation : Gérer compte utilisateur

Cas utilisation	Refuser demande
Résumé	L'administrateur a le droit supprimer l'utilisateur
Acteur	Administrateur
Précondition	Acteur authentifié. Opération de refus choisie
Post-Condition	Utilisateur refusé
Scénario nominal	<ol style="list-style-type: none"> 1. <<include>> s'authentifier 2. <<include>> gérer les utilisateurs 3. L'acteur clique sur l'icône a supprimé 4. Le système efface la demande de l'utilisateur 5. L'utilisateur sera effacé par le système

Tableau 3-0-3 Description de cas d'utilisation « Refuser compte utilisateur »

3.1.3 Raffinement du cas d'utilisation : Gérer projets

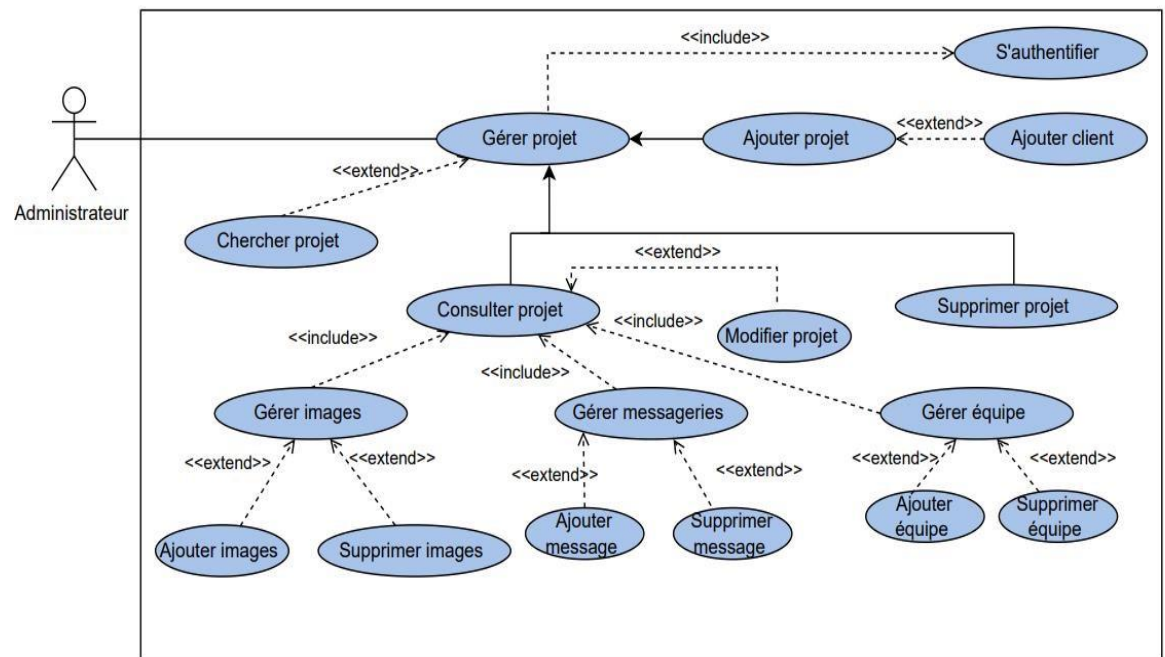


Figure 3.1.3 Raffinement de cas d'utilisation : Gérer projet

3.1.3.1 Description textuelle de cas d'utilisation : Ajouter projet

Cas utilisation	Ajouter projet
Résumé	L'acteur peut ajouter un projet
Acteur	Administrateur et chef de projet
Précondition	Acteur authentifié. Opération d'ajout choisie
Post-Condition	Projet ajouté
Scénario nominal	<ol style="list-style-type: none"> 1. <<include>> s'authentifier 2. <<include>> Consulter interface projet 3. L'acteur clique sur le bouton ajouter projet 4. Le système affiche un formulaire comportant les champs nécessaires pour l'ajout de projet 5. L'acteur saisie les données 6. L'acteur clique sur le bouton ajouter 7. Le système vérifie la validité de saisie 8. Le système ajoute un nouveau projet dans la base de données

Exception	<p>5.1 SI client n'existe pas :</p> <ul style="list-style-type: none"> ⇒ Cliquez sur le lien pour ajouter client ⇒ Retour à l'étape 2 ⇒ <p>7.1 Champ(s) manquant(s) :</p> <p>→ retour à l'étape 5.</p>
------------------	---

Tableau 3-0-4 Description de cas d'utilisation « Ajouter projet ».

3.1.3.2 Description textuelle de cas d'utilisation : Consulter projet

Cas utilisation	Consulter Projet
Résumé	L'acteur peut consulter liste des projets
Acteur	Administrateur, chef de projet, développeur et client
Précondition	Acteur authentifié. Opération consulter choisie
Post-Condition	Projet consulté
Scénario nominal	<ol style="list-style-type: none"> 1. <<include>> s'authentifier 2. L'acteur demande l'interface de consulter liste projet 3. Le système affiche l'interface de consulter liste projets
	<ol style="list-style-type: none"> 4. L'acteur clique sur le projet 5. Le système affiche l'interface de projet demandé avec ses informations relatives

Tableau 3-0-5 Description de cas d'utilisation « Consulter projet »

3.1.3.3 Description textuelle de cas d'utilisation : Supprimer équipe

Cas utilisation	Supprimer équipe
Résumé	L'acteur peut supprimer un membre de l'équipe
Acteur	Administrateur et chef de projet
Précondition	Acteur authentifié. Opération supprimé choisie
Post-Condition	Équipe supprimé

Scénario nominal	<ol style="list-style-type: none"> 1. <<include>> s'authentifier 2. L'acteur cherche le projet a demandé 3. L'acteur clique sur le projet demandé 4. Le système affiche l'interface pour consulter les informations dédiées au projet. 5. L'acteur clique sur icone pour supprimer un membre d'équipe 6. Le système affiche un message d'avertissement indiquant l'accord de suppression 7. L'acteur valide la suppression.
Exception	<p>6 l'acteur annule la suppression :</p> <p>⇒ Retour à l'étape 4 de scénario principal</p>

Tableau 3-0-6 Description de cas d'utilisation « Supprimer équipe »

3.1.4 Raffinement du cas d'utilisation : Gérer tâches

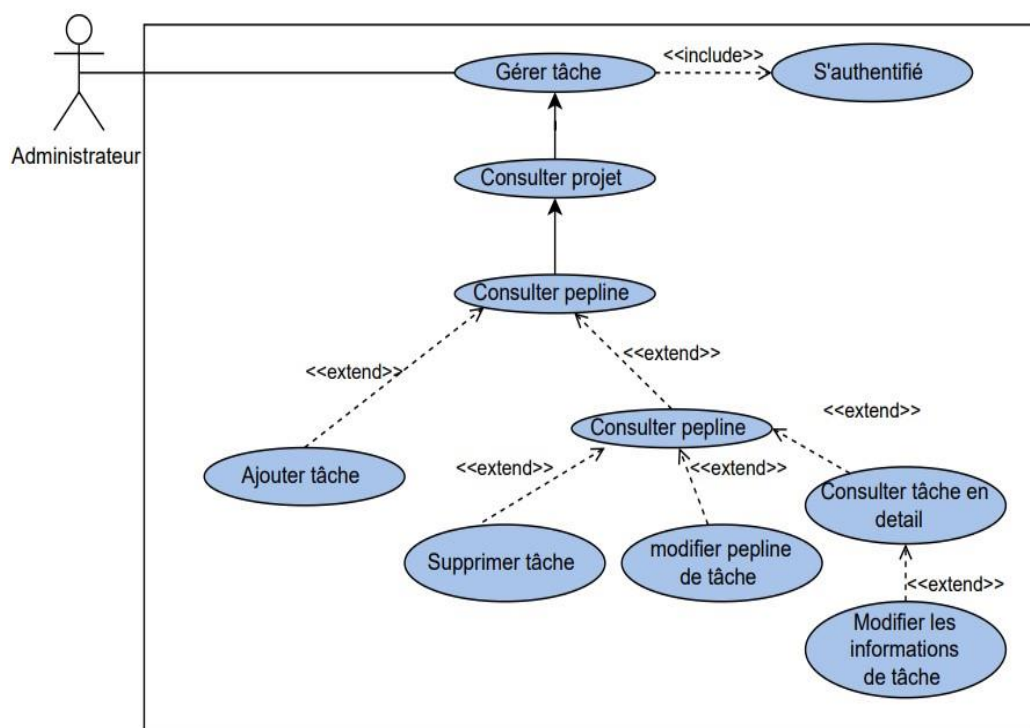


Figure 3.1.4 Raffinement de cas d'utilisation : Gérer tâche

3.1.4.1 Description textuelle de cas d'utilisation : Ajouter tâche

Cas utilisation	Ajouter tâche.
Résumé	L'acteur peut ajouter une tâche
Acteur	Administrateur, chef de projet et développeur.
Précondition	Acteur authentifié. Opération d'ajout choisie.
Post-Condition	Tâche ajouté
Scénario nominal	<ol style="list-style-type: none"> 1. « Include » s'authentifier. 2. « Include » Consulter interface projet. 3. « Include » Consulter interface pipeline. 4. Dans la liste de pipeline choisi l'acteur cliquer sur le bouton Ajouter tâche. 5. Le système affiche un formulaire comportant les champs Nécessaires pour l'ajout de tâche. 6. L'acteur saisie les données. 7. L'acteur clique sur le bouton ajouter. 8. Le système vérifie la validité de saisie. 9. Le système ajoute une nouvelle tâche dans la base de Données.
Exception	<p>4 Si pipeline n'existe pas :</p> <p>(a) Clique sur le bouton en haut pour ajouter pipeline.</p> <p>(b) Retour à l'étape</p> <p>4. 8 Champ(s) manquant(s) :</p> <p>(a) Retour à l'étape 6</p>

Tableau 3-0-7 Description de cas d'utilisation « Ajouter tâche ».

3.1.5 Raffinement du cas d'utilisation : Créer compte clients

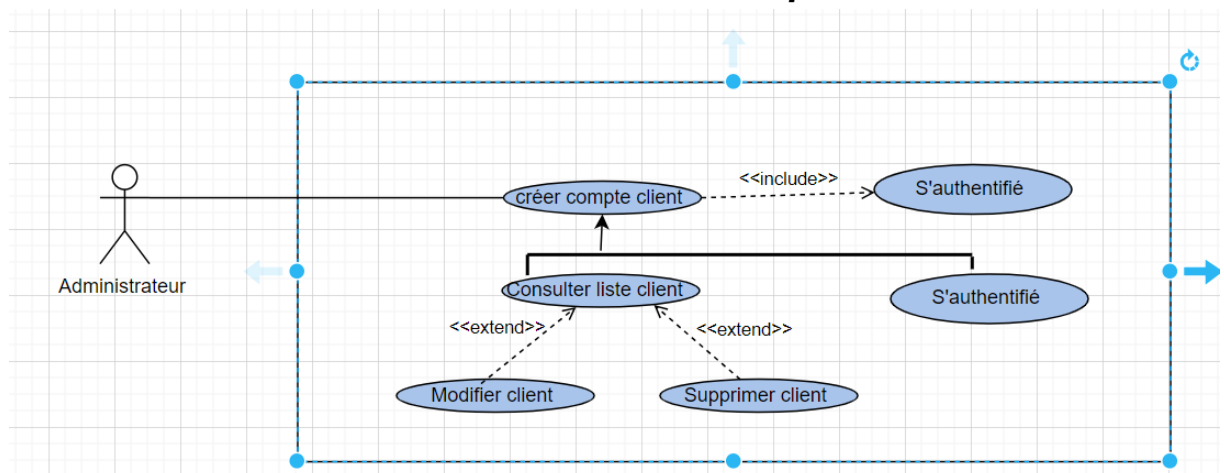


Figure 3.5 Raffinement du cas d'utilisation : Créer compte clients

3.1.5.1 Description textuelle de cas d'utilisation : Ajouter client

Cas utilisation	Ajouter client.
Résumé	L'acteur peut ajouter un client.
Acteur	Administrateur et chef de projet
Précondition	Acteur authentifié. Opération d'ajout choisie.
Post-Condition	Client ajouté.
Scénario nominal	<ol style="list-style-type: none"> 1. « Include » s'authentifier. 2. « Include » Consulter interface client. 3. Dans la l'interface client l'acteur clique sur le bouton 4. Ajouter client. 5. Le système affiche un formulaire comportant les champs 6. Nécessaires pour l'ajout d'un client. 7. L'acteur saisie les données. 8. L'acteur clique sur le bouton ajouter. 9. Le système vérifie la validité de saisie. 10. Le système ajoute un nouveau client dans la base de données
Exception	7 Champ(s) manquent(s) : (a) Retour à l'étape 5.

Tableau 3-8 Description textuelle de cas d'utilisation : Ajouter client

3.2 Diagramme de classes

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il est le seul obligatoire lors d'une telle modélisation. Il permet de fournir une représentation abstraite des objets du système qui vont interagir pour réaliser les cas d'utilisation. Le diagramme de classes est un diagramme statique qui représente le point de vue logique du logiciel à travers le concept de classe et relations entre classes.

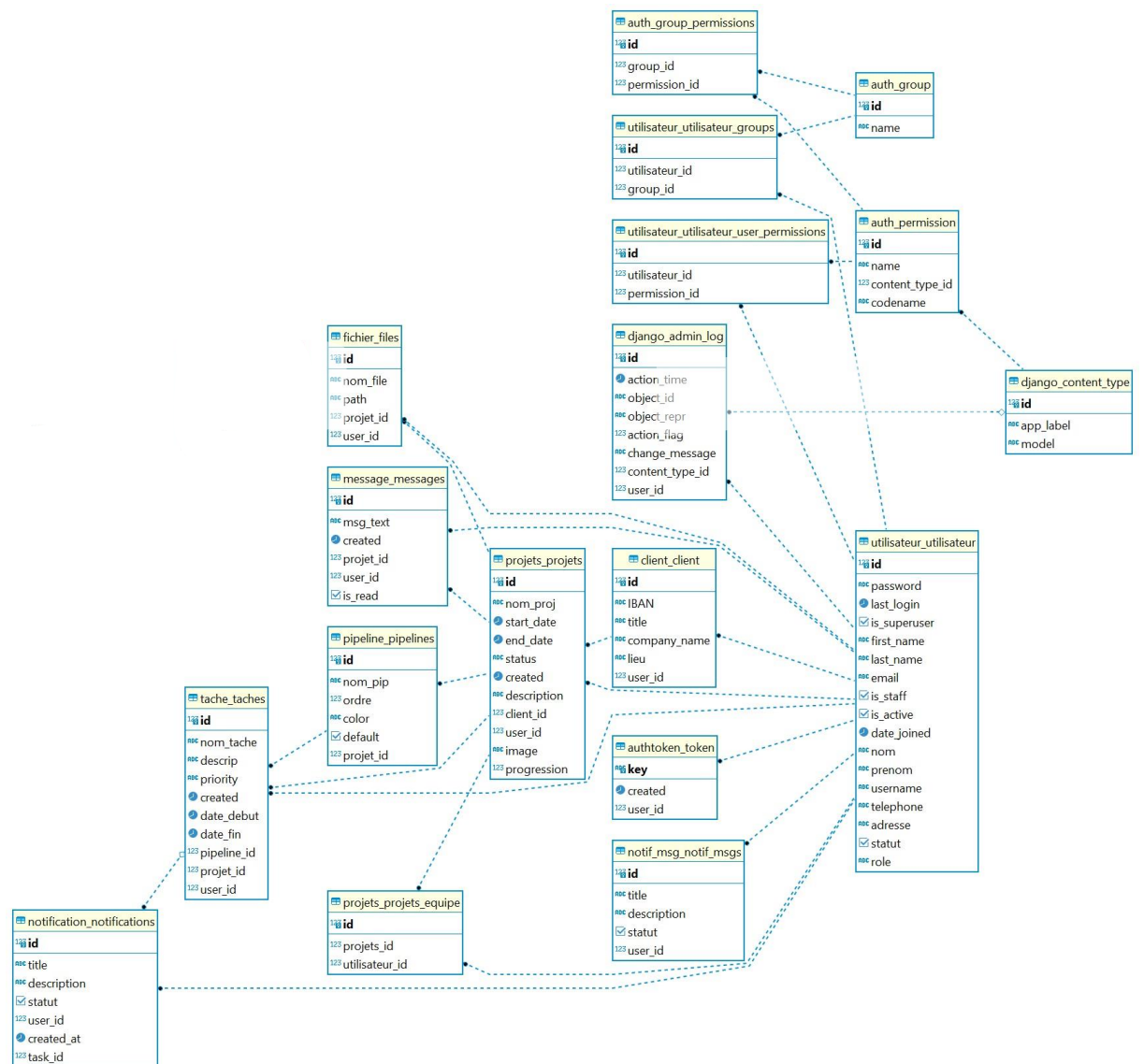


Figure 3.2.1 Diagramme de classes général.

3.3 Schéma relationnel de la base de données

Le schéma relationnel de notre application est le suivant :

- Utilisateur (id, Nom, Prenom, Telephone, Adresse, Email, Password, Statut, Type)
- Client (id, Eban, Title, Company_name, Lieu, #user_id)
- Projets (id, Nom_proj, Deadline, Status, Created, Description, #user_id, #client_id)
- Equipes (#user_id, #projet_id, date_crea_proj)
- Pipelines (id, Nom_pip, Ordre, Color, Default, #projet_id)
- Tâches (id, Nom_tache, Descrip, Priority, Created, Deadline, #user_id, #projet_id, #pipeline_id)
- Messages (id, Msg_text, Created, #projet_id, #user_id)
- Notifications (id, Title, Description, Statut, #user_id)

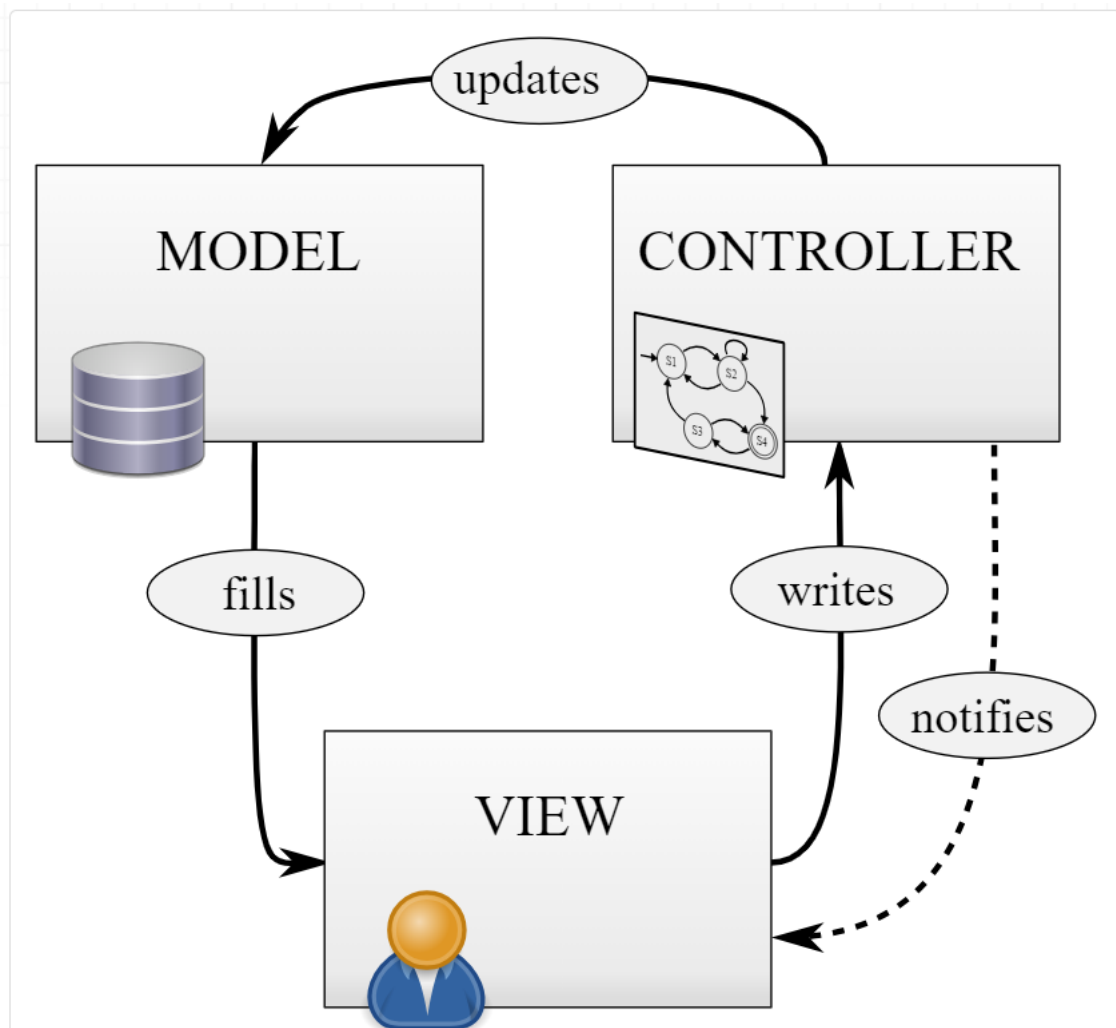
- Notif_Msgs (id, Title, Description, Statut, #user_id)

3.4 Architecture à envisager: Le Modèle MVC/MVT

Le design pattern Modèle-Vue-Contrôleur ou MVC est un type d'architecture logicielle destiné aux interfaces graphiques lancé en 1978 et très populaire pour les applications web. Le motif est composé de trois types de modules assurant différents rôles :

- Le modèle (Model) représente, souvent sous forme d'objet, les données à utiliser par l'application et utilise un ORM pour l'interaction avec la base de données.
- La vue (View) contient la présentation des données via une interface graphique.
- Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

L'idée importante dans ce design pattern est de séparer distinctement les tâches effectuées par les différents éléments. Ainsi la conception et la maintenance sont simplifiées.



Django utilise l'architecture MVT (modèle-vue-Template) qui s'inspire de MVC :

- **Le modèle** interagit avec une base de données via un ORM. Tous les modèles sont réunis dans un fichier python `models.py`.
- **La vue** reçoit une requête HTTP et renvoie une réponse HTTP convenable (par exemple si la requête est une interaction avec une base de données, la vue appelle un modèle pour récupérer les items demandés). Les vues se trouvent dans le fichier `views.py`
- **Le Template** est un fichier HTML récupéré par la vue et envoyé au visiteur avec les données des modèles.

La figure ci-dessous montre comment les différents composants de l'architecture MVT de Django interagissent pour répondre à la requête d'un utilisateur. Ici le contrôleur ne correspond pas au contrôleur du MVC, mais à Django en lui-même qui gère en interne tout ce qui est liée au choix de la vue à laquelle envoyer la requête HTTP, ...

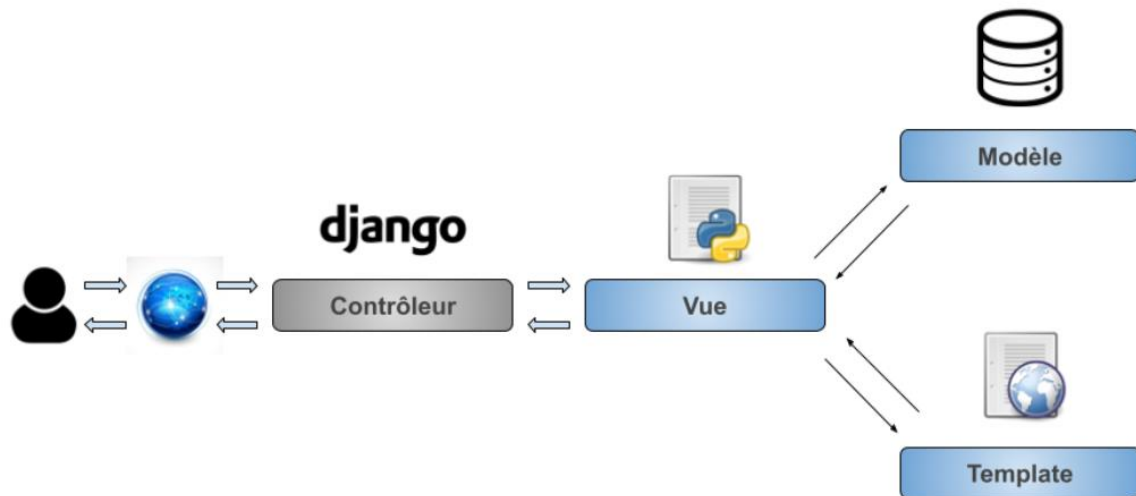


Figure 0.1 Schémas de l'architecture MVT

3.5 Diagramme de séquence

3.5.1 Définition

Le diagramme de séquence modélise le comportement d'un système à travers des interactions entre ces objets. Les objets vont interagir entre eux en envoyant et recevant des messages par réaliser un cas d'utilisation.

L'organisation des messages dans le diagramme de séquences se fait selon leurs ordres Chronologiques durant la réalisation d'un cas d'utilisation [7].

Je vais maintenant présenter quelques diagrammes de séquence qui sont les plus utiles pour la suite de mon travail.

3.5.2 Diagramme de séquence de CU : S'authentifier

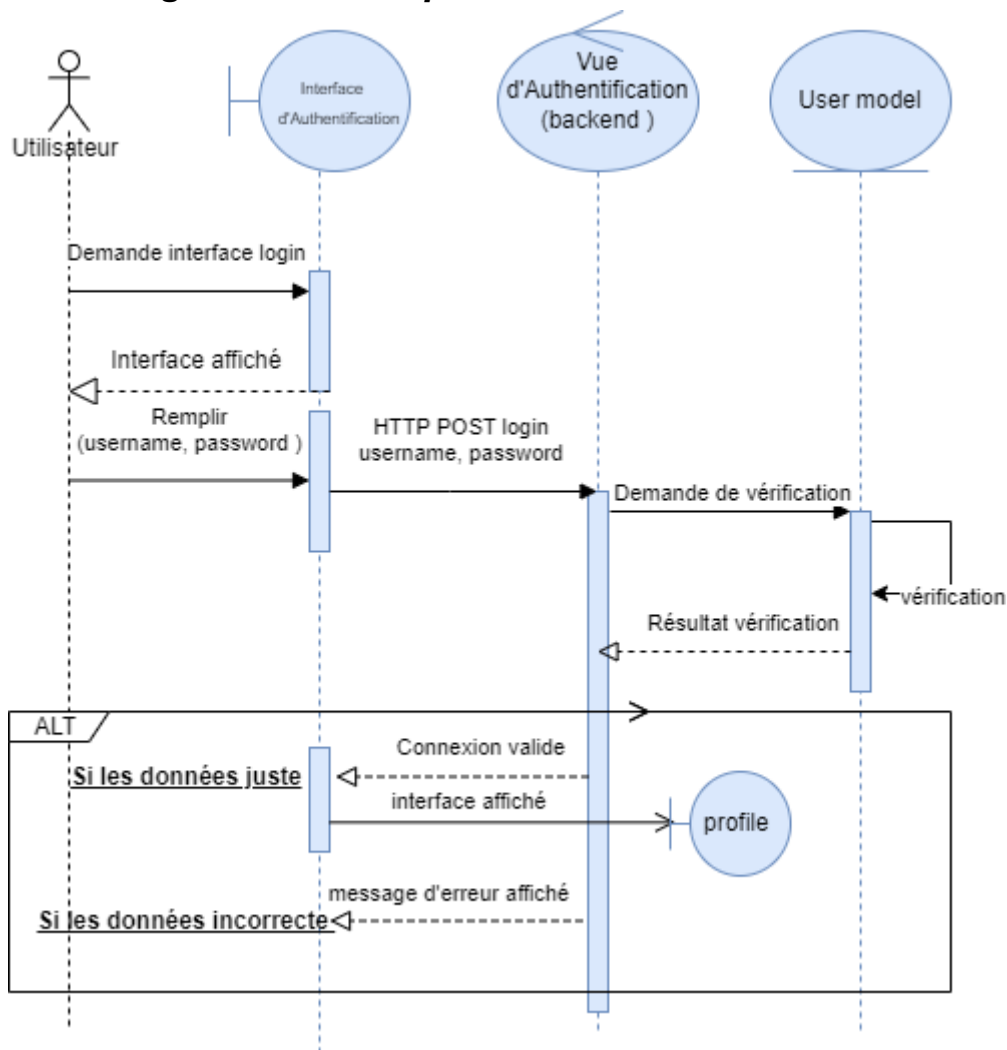


Figure 3.8 Diagramme de séquence de CU : S'authentifier

3.5.3 Diagramme de séquence de CU : Ajouter Projet

Le diagramme de séquence présenté à la Figure 3.9 décrit le processus d'ajout d'un projet. L'administrateur a la possibilité de saisir les détails nécessaires du projet. À la suite de cela, une requête HTTP de type « POST » est envoyée à la vue « ProjetView ». Cette vue vérifie alors si les données fournies sont correctes. En cas d'incohérence, un message d'erreur est renvoyé, invitant à la correction des données. Si les informations sont correctes, le projet est alors créé.

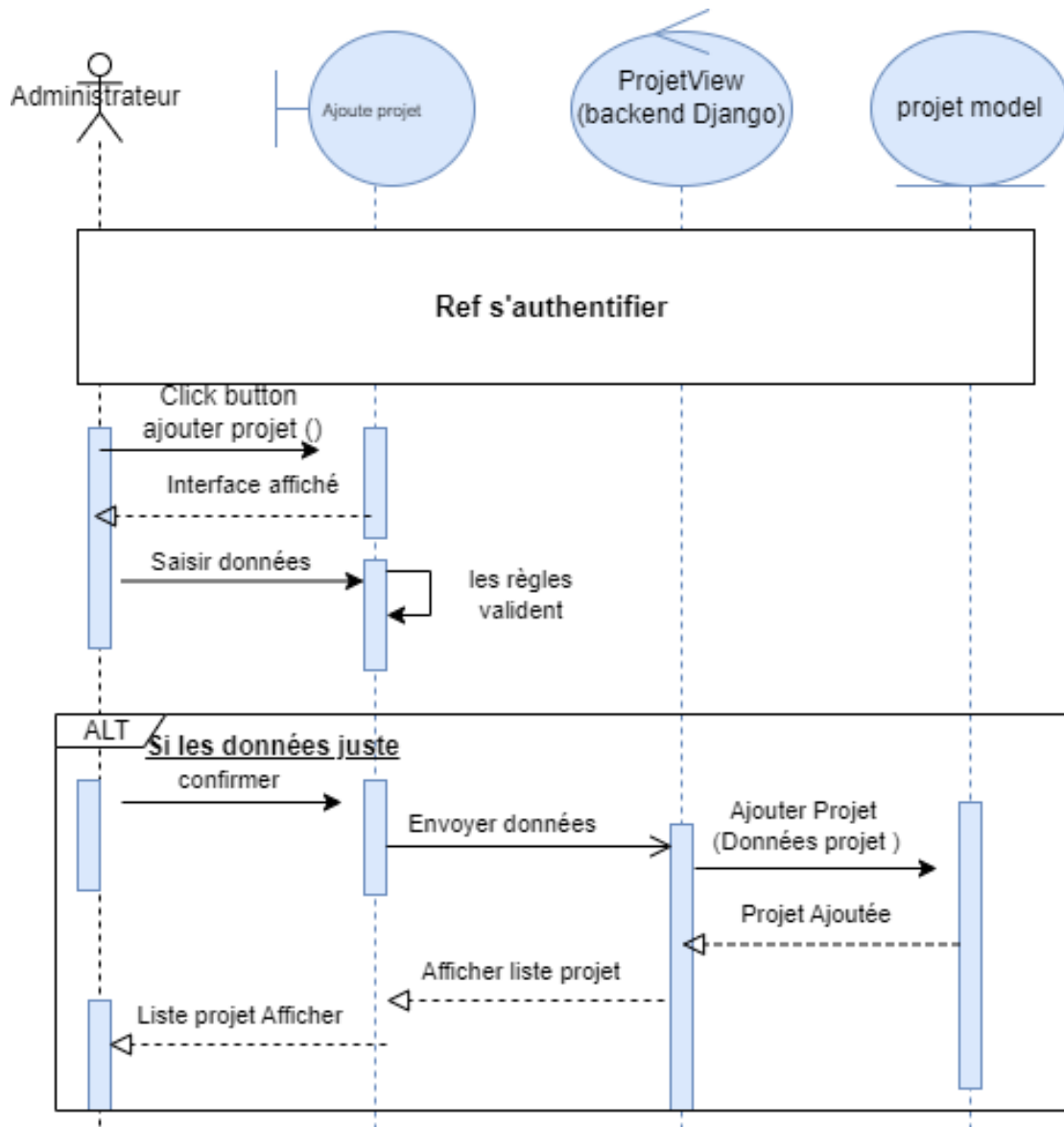


Figure 3.9 Diagramme de séquence de CU : Ajouter projet

3.5.4 Diagramme de séquence de CU : Modifier Projet

Le diagramme de séquence illustré dans Figure 3.10, explique le processus de modifier Projet, l'administrateur peut consulter le projet qui veut le modifier et entrer les détails À modifier pour projet. Une requête HTTP de type « PUT » est envoyé à la vue (contrôleur).

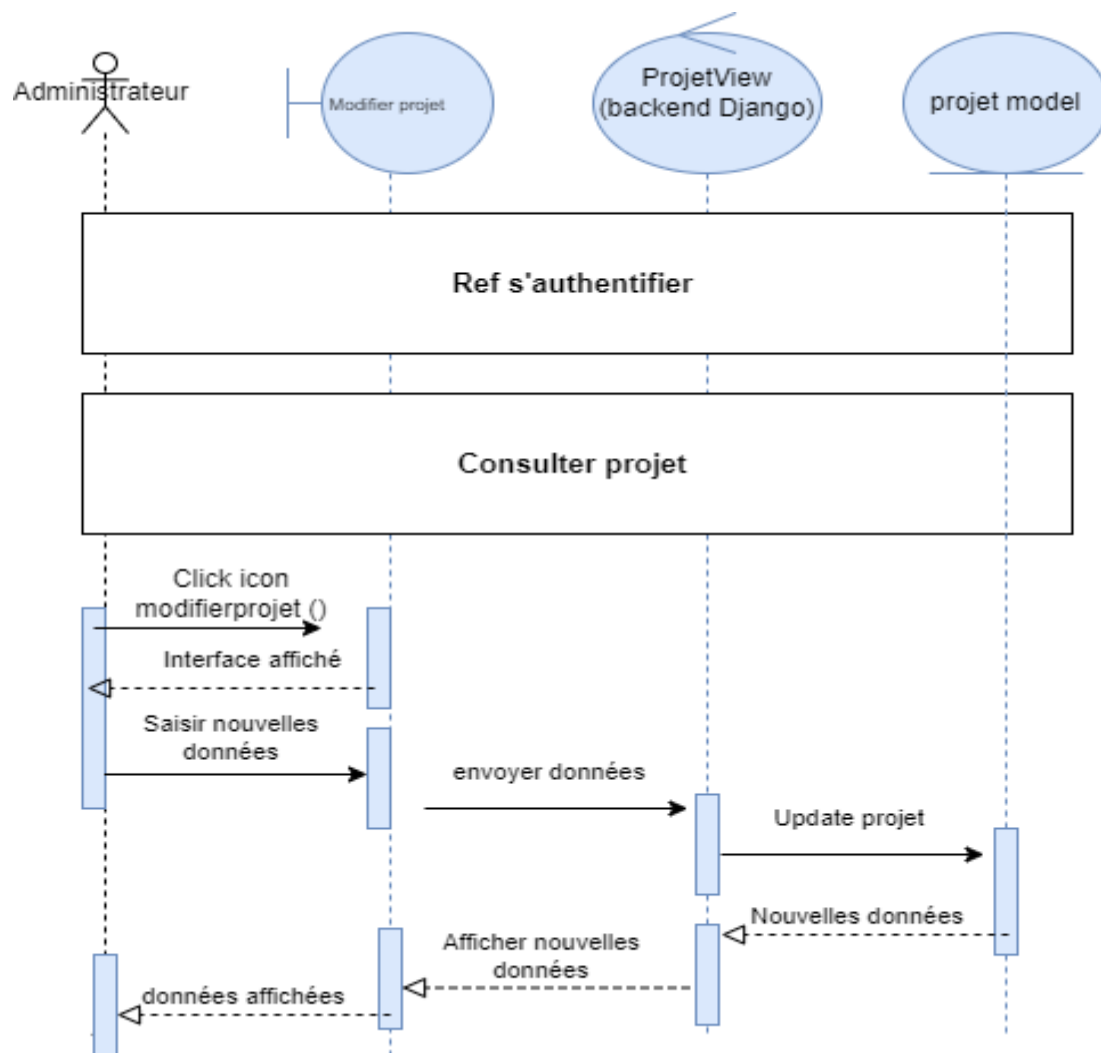


Figure 3.10 Diagramme de séquence de CU : Modifier projet

3.5.5 Diagramme de séquence de CU : Ajouter tâche

Le diagramme de séquence illustré dans Figure 3.11, explique le processus d'ajouter tâche, L'administrateur ou le chef de projet ou développeur peut également entrer les détails requis de tâche. Une requête HTTP de type « POST » envoyée à la vue « TacheView ». Cette vue vérifie alors si les données fournies sont correctes. En cas d'incohérence, un message d'erreur est renvoyé, invitant à la correction des données. Si les informations sont correctes, le tâche est alors créé.

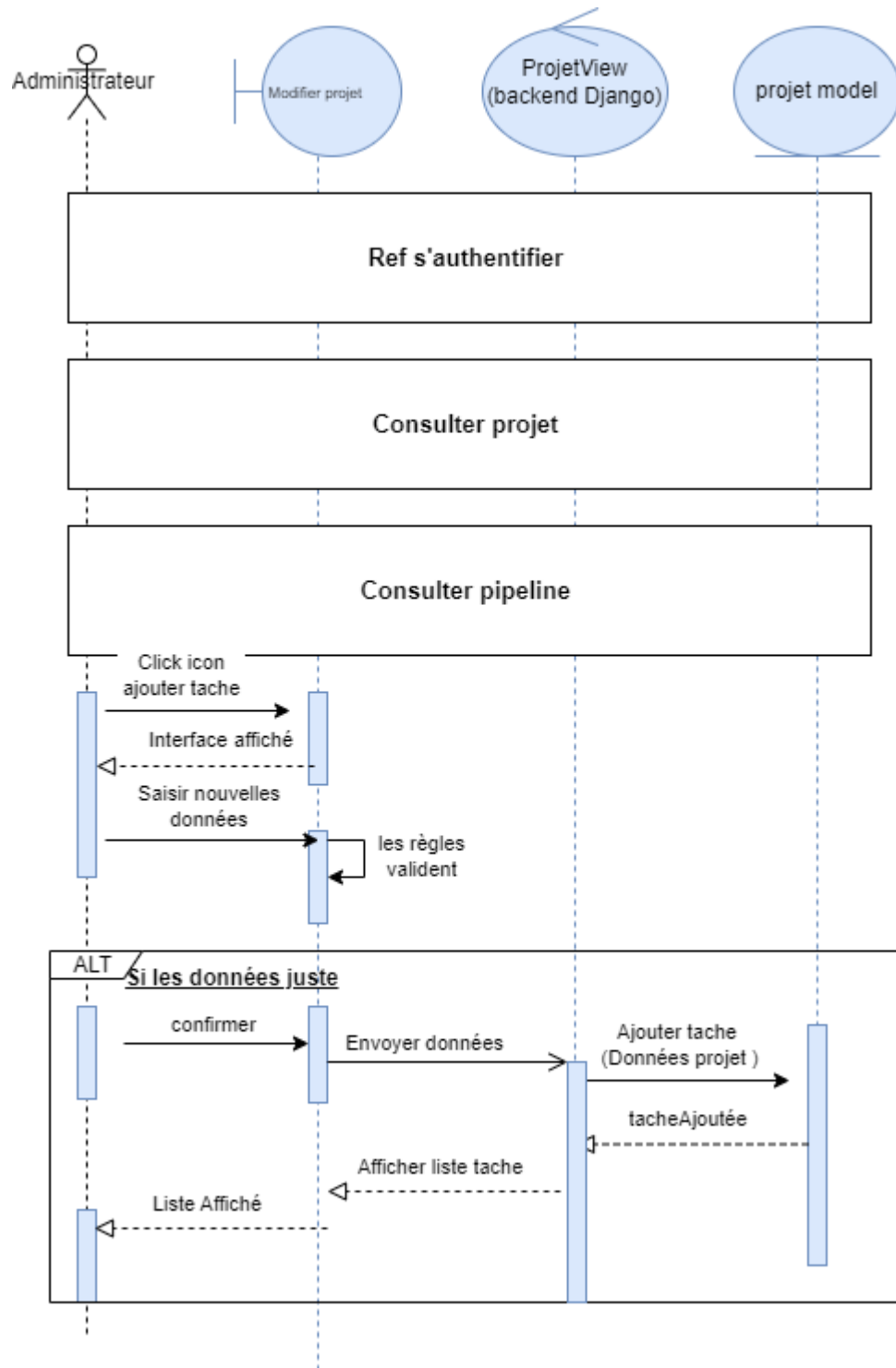


Figure 3.10.1 Diagramme de séquence de CU : Ajouter tâche

3.5.6 Diagramme de séquence de CU : Ajouter client

Le diagramme de séquence illustré dans Figure 3.12, explique le processus de création d'un Client. L'administrateur saisie les données nécessaires et le système va les vérifier. Une requête http de type « POST » est transmise à la vue ClientView ». Contenant les données

Saisis et au fur et à mesure un email sera envoyé lors de la création qui contient l'adresse email et le mot de passe de compte.

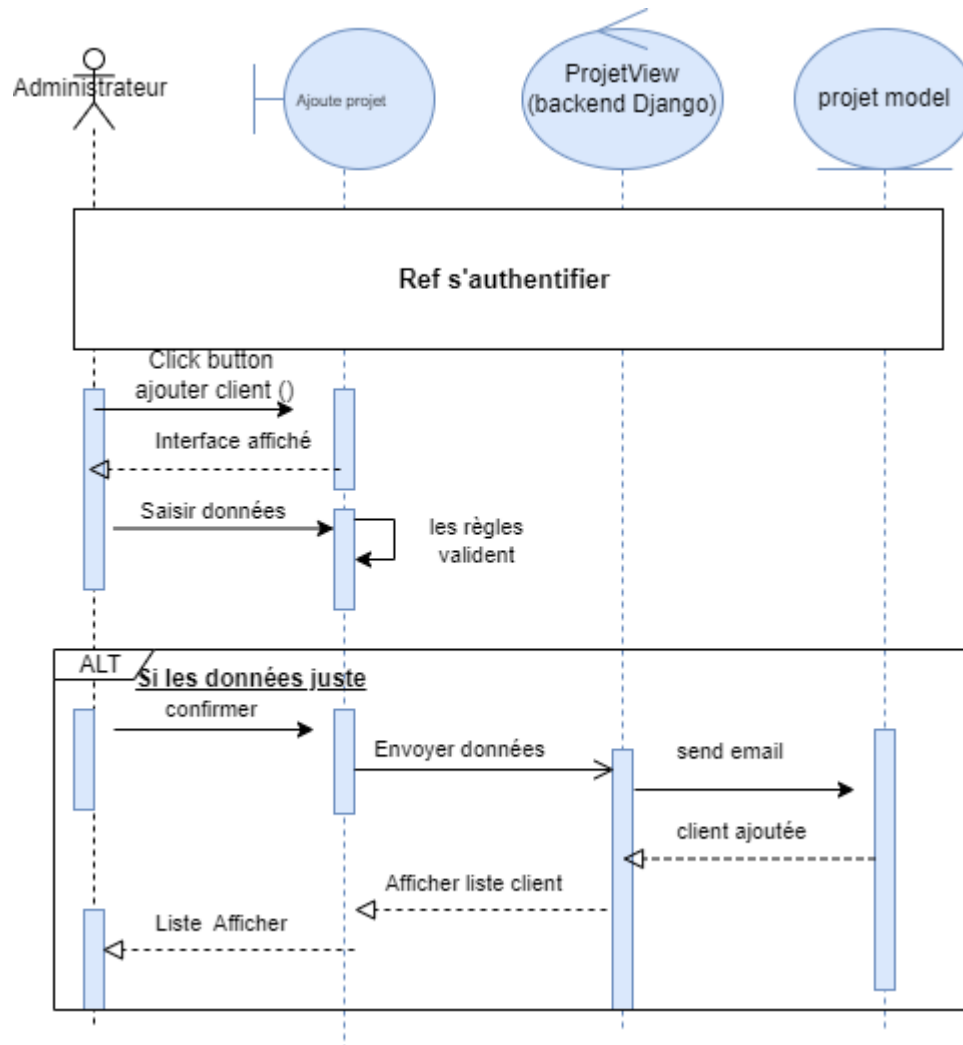


Figure 3.10.1 Diagramme de séquence de CU : Ajouter client

Conclusion

Au cours de ce chapitre, j'ai abordé l'architecture de mon application et la modélisation UML du système. J'ai détaillé différents diagrammes : le diagramme raffiné avec sa description, le diagramme de classes, et les diagrammes de séquence. Ces éléments ont permis de spécifier de manière approfondie les aspects fonctionnels de mon système. Dans le chapitre suivant, je présenterai la mise en œuvre effective de mon application.

Introduction

Ce chapitre présente d'abord l'environnement de développement logiciel. Ce

Qui suit décrit les tâches effectuées pour répondre à nos besoins, sur la base de la conception

Déjà décrite. La dernière partie est consacrée à la validation et à l'utilisation de la solution

Implémentée.

4.1 Environnement logiciel :

4.1.1 Environnement de développement



PyCharm est un environnement de développement intégré utilisé pour programmer en Python. Il permet l'analyse de code et contient un débogueur graphique. Il permet également la gestion des tests unitaires, l'intégration de logiciel de gestion de versions, et supporte le développement web avec Django.

4.1.2 Environnement de conception



Est une application gratuite en ligne qui permet de dessiner

Des diagrammes ou des organigrammes. Cet outil vous propose de concevoir toutes sortes de diagrammes, de dessins vectoriels, de les enregistrer au format XML puis de les exporter. Draw.io est un véritable couteau suisse de la frise chronologique, de la carte mentale et des diagrammes de tout genre.

4.1.3 Environnement de création logo



Canva est un site Internet qui permet de créer et de

Personnaliser les designs pour tout type de projet, de façon simple et intuitive. Il s'avère très utile surtout pour celles et ceux qui n'ont pas de compétence graphique particulières.

4.1.3 Les Outils de Base des données :



Dbeaver est un logiciel permettant l'administration et le requêtage de base de données. Pour les bases de données relationnelles, il utilise un driver JDBC. Pour les autres bases de données, il utilise des pilotes de base de données propriétaire.

4.2 Les frameworks et les langages de programmation

4.2. Python



Python est un langage de programmation interprété, multiparadigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet.

4.2.2 JS



JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Avec les langages HTML et CSS, JavaScript est au cœur des langages utilisés par les développeurs web³. Une grande majorité des sites web l'utilisent⁴, et la majorité des navigateurs web disposent d'un moteur JavaScript⁵ pour l'interpréter.

4.2.3 Pusher



Pusher est une couche en temps réel entre vos serveurs et vos clients. Pusher maintient des connexions persistantes avec les clients de sorte que dès que vos serveurs ont de nouvelles données qu'ils souhaitent transmettre aux clients, ils peuvent le faire, instantanément via Pusher.

4.3 Les interfaces de l'application

Dans cette partie, nous allons présenter les différentes fenêtres qui constituent notre application.

4.3.1 Interface Inscription

Nous allons présenter l'interface « S'inscrire » dans Figure 4.1 L'utilisateur remplit le formulaire d'inscription.

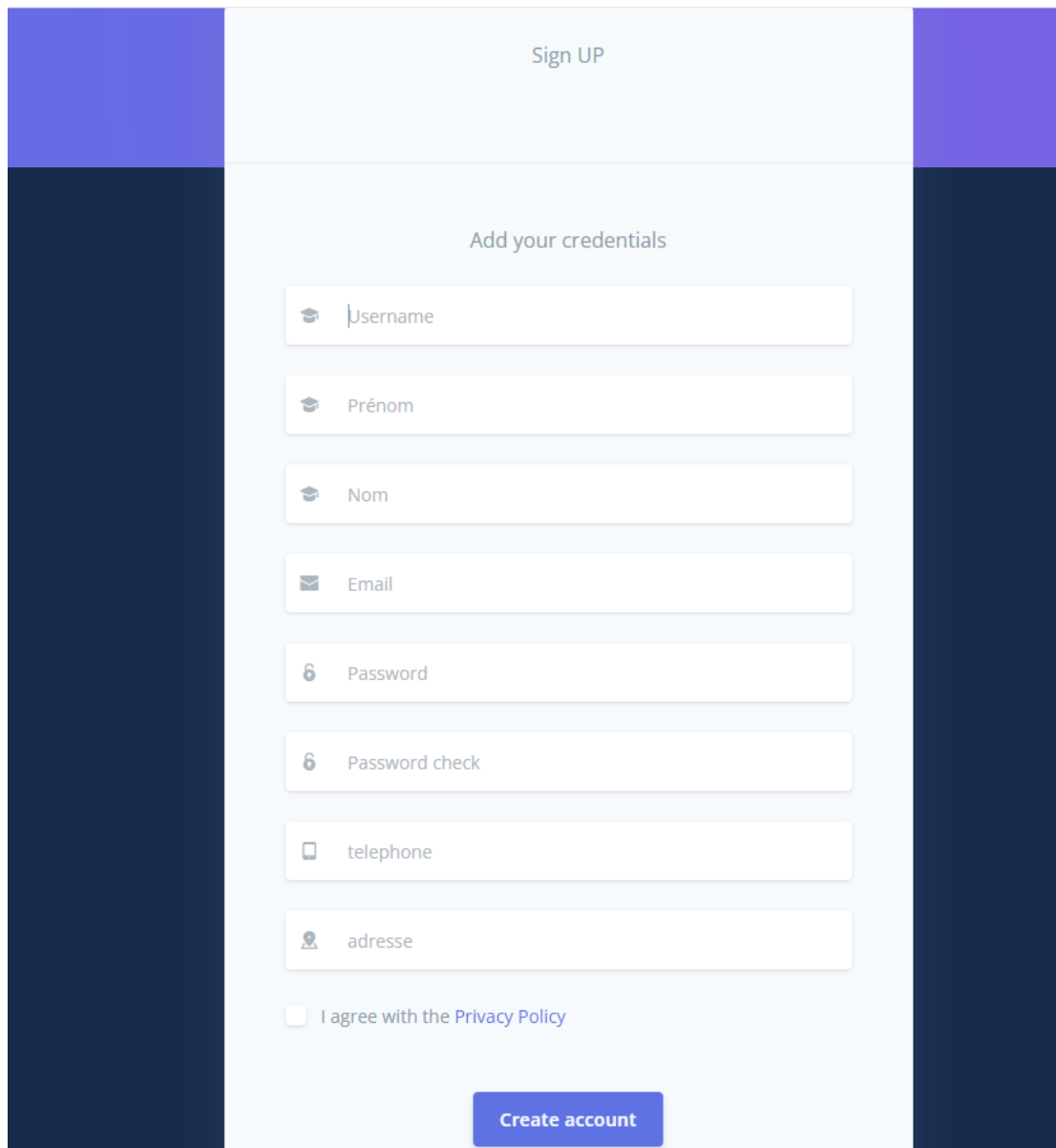
The image shows a 'Sign UP' form interface. At the top, the text 'Sign UP' is centered in a light blue header. Below this, the instruction 'Add your credentials' is centered. The form consists of several input fields, each with a small icon on the left: 'Username' (user icon), 'Prénom' (person icon), 'Nom' (person icon), 'Email' (envelope icon), 'Password' (key icon), 'Password check' (key icon), 'telephone' (phone icon), and 'adresse' (location pin icon). Below the input fields, there is a checkbox labeled 'I agree with the Privacy Policy'. At the bottom, there is a blue button with the text 'Create account'. The form is set against a light blue background with dark blue vertical bars on the left and right sides.

Figure 4.0.1 Interface Inscription

4.3.2 Interface connexion

L'authentification est un mécanisme de sécurité du système. Grâce à elle, chaque utilisateur authentifié peut accéder à notre application. Pour s'authentifier, l'utilisateur doit saisir son username et son mot de passe, comme illustré à la

figure.

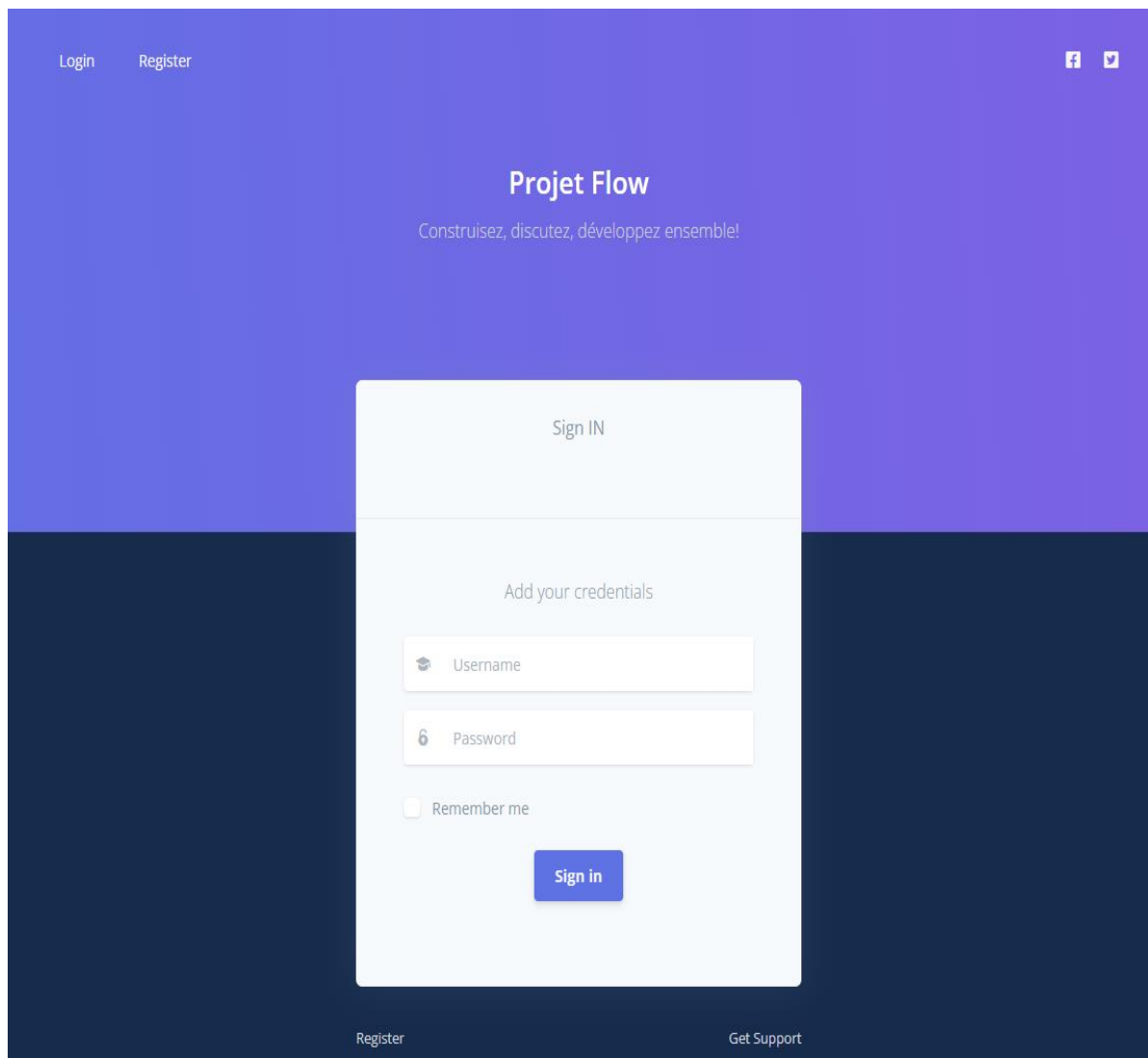


Figure 4.2 interface Connexion

4.3.3 Interface Client

4.3.3.1 Mail

Lors de la création du compte client, ce dernier reçoit sur son email principal un email et un Mot de passe pour connecter à son interface.

Bienvenue à notre service

Voici vos identifiants de connexion:

Votre Pseudo: Sanasoussi

Email: sanasoussi55@gmail.com

Mot de passe: pbkdf2_sha256\$600000\$gBb5abk2aE6v6VjdLNwfRV\$kYsz2teGMgaIFLzuSalXWr8Ku+w6yEyP/wQbq8vSDBc=

Figure 4.4 Mail

4.3.4 Interface développeur

4.3.4.1 Page d'accueil

Je vais commencer par présenter la page d'accueil du développeur, illustrée à la figure 4.5. La partie supérieure de cette figure indique le nombre de projets et de tâches assignés au développeur connecté. La partie inférieure liste toutes les tâches affectées à ce développeur

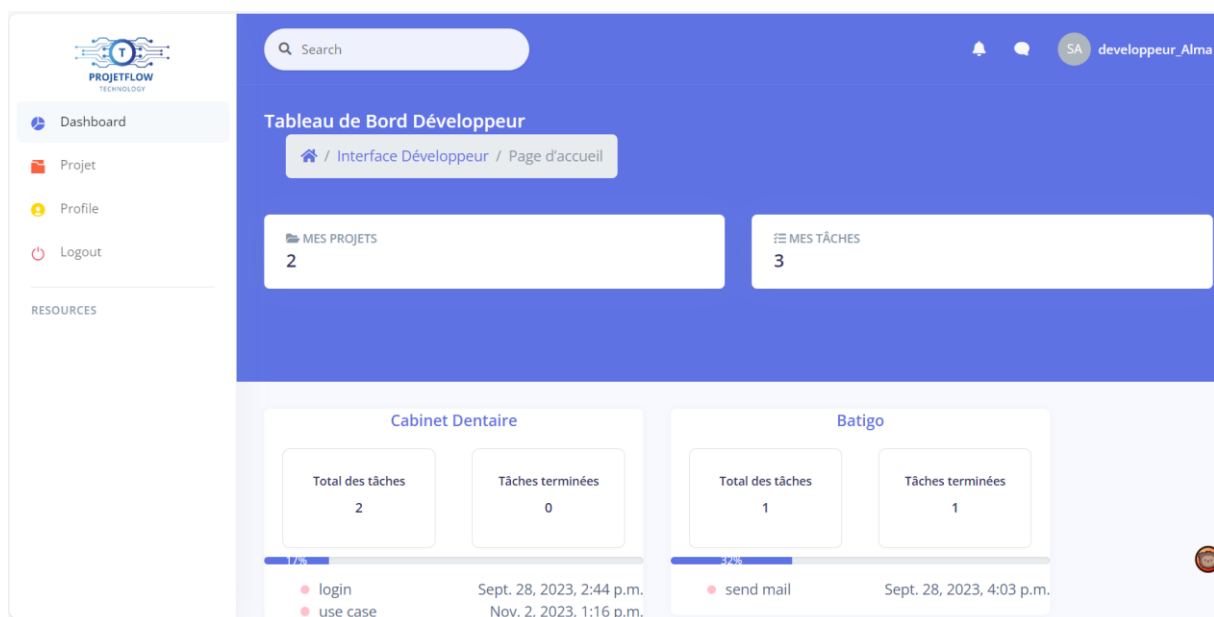


Figure 4.5 Page d'accueil

4.3.4.2 Consulter projet

Je présente l'interface « Consulter Projet », illustrée à la figure 4.6, qui détaille les informations du projet. Elle montre le descriptif du projet avec ses détails tels que le nom du client, le chef de projet, la date de création et la date limite, le statut et la progression du projet. On y trouve également la section de messagerie de l'équipe du projet, une liste d'images, une liste de développeurs qui peuvent y être ajoutés.

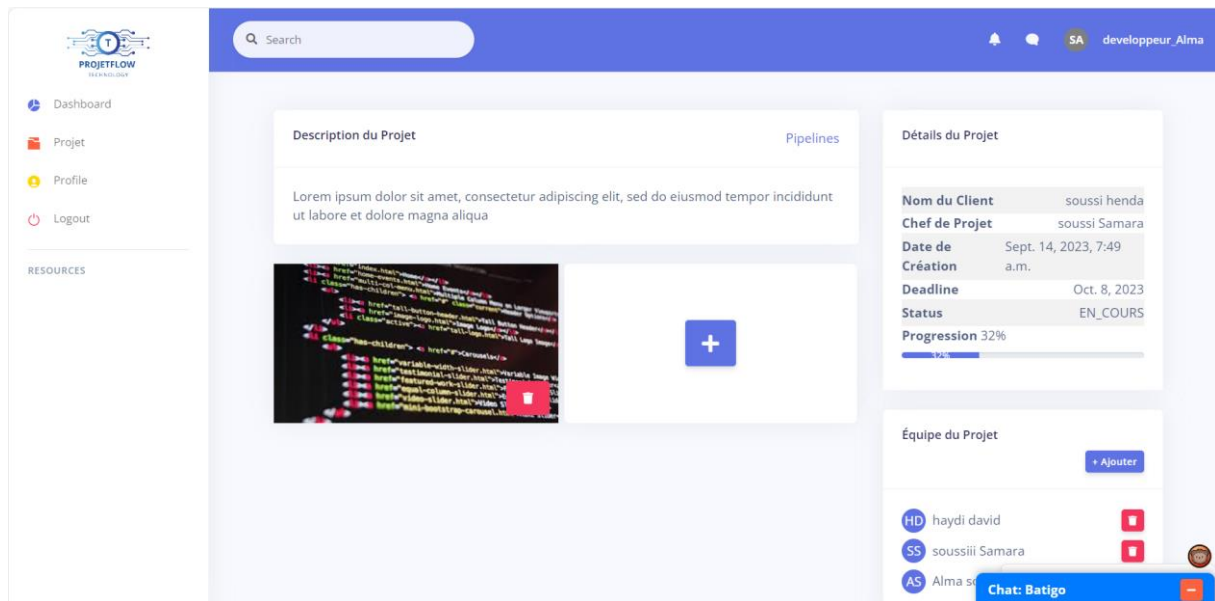


Figure 4.6 Consulter projet

4.3.4.3 Gérer message

Je présente la section « Gérer Message », illustrée à la figure 4.7. Elle offre à l'équipe du projet la possibilité d'envoyer et de recevoir des messages entre ses membres. Dans cette partie, j'utilise "Pusher", qui permet d'envoyer une notification en temps réel lors de la réception d'un message.

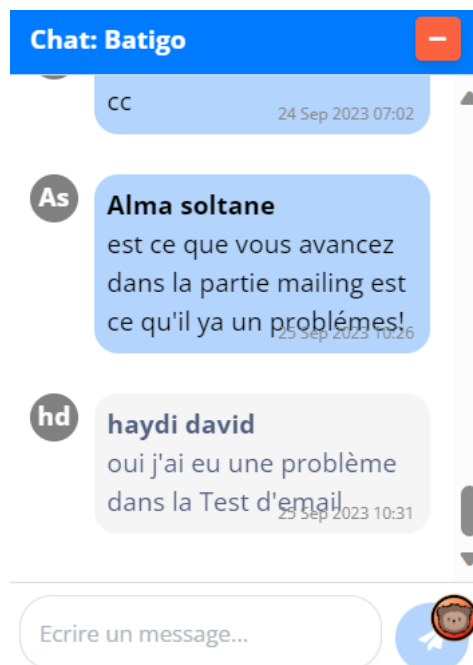


Figure 4.7 Gérer message

4.3.4.4 Consulter notification de message

L'interface « consulter notification de message » de la figure 4.8 indique le nombre des Messages qui n'ont pas été consultés par le profil ouvert dans le chat de l'équipe du projet. Lorsqu'on clique sur l'icône message pour le consulter, le numéro du notification-message Devient zéro et une liste qui s'ouvre avec l'information de la notification. Afin d'effacer tout Les messages de la notification, on peut cliquer sur bouton "clear all".



Figure 4.8 Consulter notification de message

4.3.4.5 Consulter pipeline

Pour donner suite à la consultation du projet on clique sur l'icône en haut qui affiche interface pipeline présenter dans la figure 4.9. Cette interface définit la liste des pipelines avec des couleurs différentes. Chaque pipeline contient une liste des tâches. En fait, cette partie donne la main à chaque utilisateur (administrateur et/ou chef de projet) d'ajouter, modifier et supprimer un pipeline.

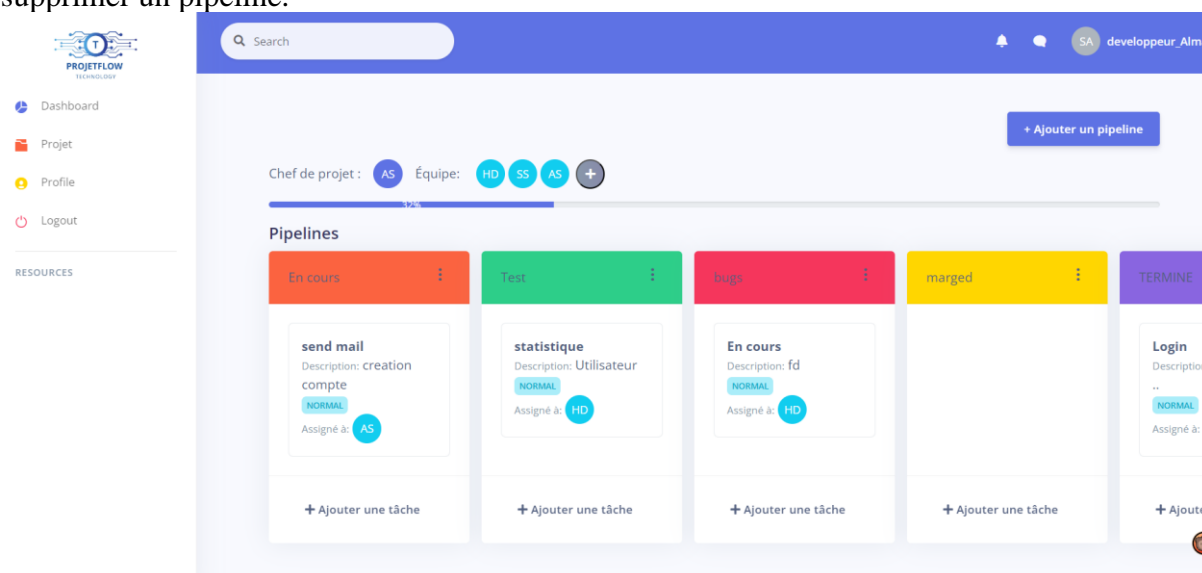


Figure 4.9 Consulter pipeline

4.3.4.6 Modifier pipeline

Cette figure 4.10 montre comment chaque utilisateur (administrateur et/ou chef de projet) Peut modifier pipeline par son nom et couleur de ticket.



Modifier le Pipeline

Nom du Pipeline:

En cours

Couleur du Pipeline:

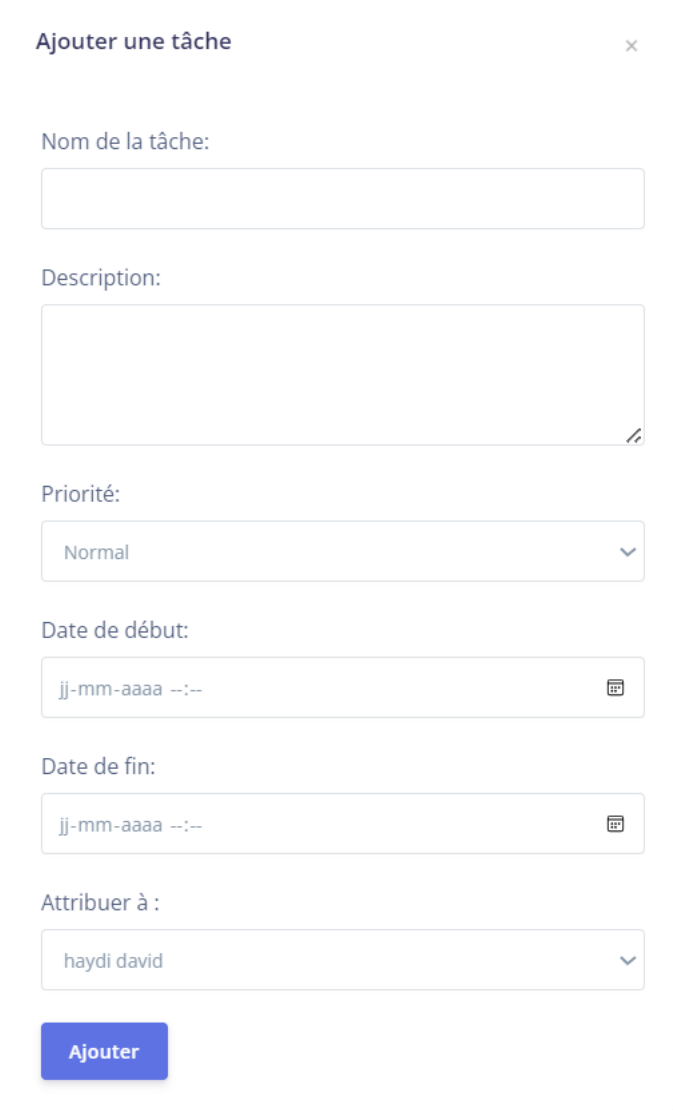
Orange Green Light Blue Purple Yellow Red

Modifier

Figure 4.0.10 Modifier pipeline

4.3.4.7 Ajouter tâche

Je présente l'interface « Ajouter Tâche », visible à la figure 4.11. Elle offre à chaque utilisateur la possibilité d'ajouter une tâche. À la suite de cela, une notification est envoyée au développeur concerné pour l'informer qu'une tâche lui a été attribuée



Ajouter une tâche

Nom de la tâche:

Description:

Priorité:
Normal

Date de début:
jj-mm-aaaa --:--

Date de fin:
jj-mm-aaaa --:--

Attribuer à :
haydi david

Ajouter

Figure 4.0.11 Ajouter tache

4.3.4.8 Modifier tâche par pipeline

Je présente l'interface « Modifier Tâche par Pipeline », illustrée à la figure 4.13. Dans cette section, lorsqu'un développeur termine une tâche, il doit la déplacer vers une autre liste du pipeline, par exemple, de la liste "En cours" vers la liste "Test". Pour ce faire, le développeur clique sur le cadre de la tâche puis peut la glisser vers un autre pipeline associé à ce projet. Le statut de la tâche est alors mis à jour.

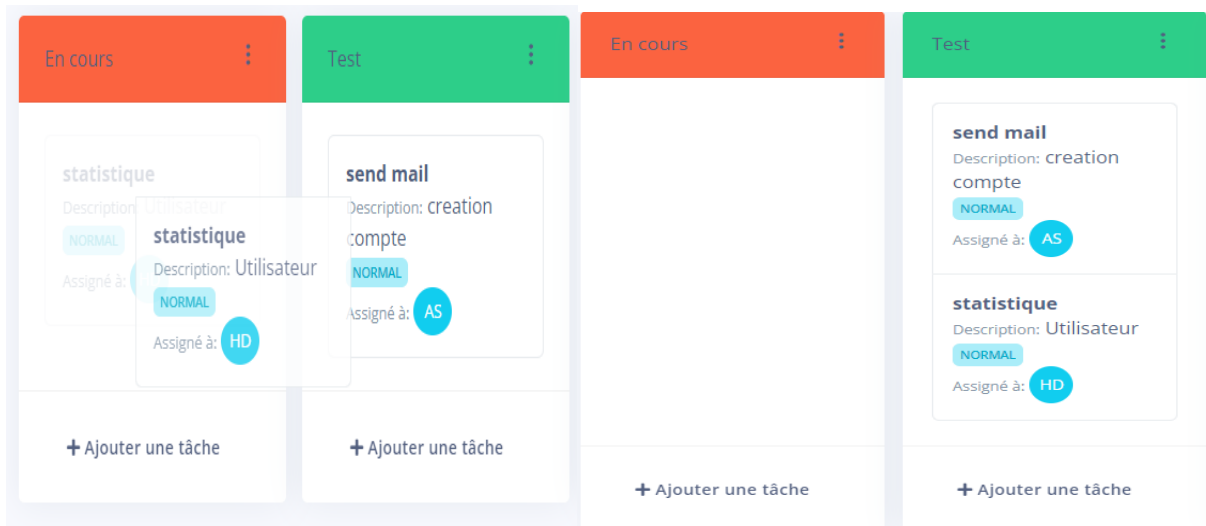


Figure 4.13 Modifier tâche par pipeline

4.3.5 Interface Administrateur

4.3.5.1 Page d'accueil

Je commence par présenter la page d'accueil illustrée à la figure 4.14, qui se compose de trois sections principales :

La partie supérieure : elle représente les totaux des comptes utilisateurs, des projets, des tâches et des clients.

La section centrale : elle contient tous les projets avec leurs informations respectives.

La partie inférieure : elle présente la liste des clients.

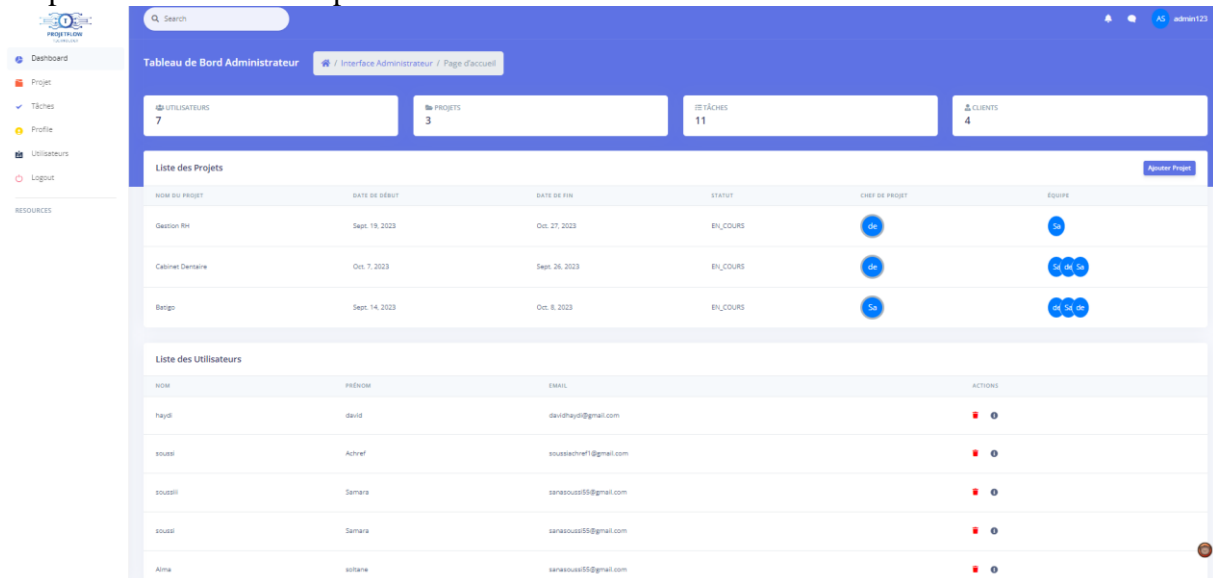


Figure 4.0.14 Interface Administrateur

4.3.5.2 Ajouter projet

Je présente l'interface « Ajouter Projet », illustrée à la figure 4.17, qui permet à l'administrateur d'ajouter un projet. Le formulaire comprend plusieurs champs obligatoires,

dont : le nom du projet, le nom du client, la date de création du projet, le deadline, le chef de projet, la description, et le statut. Si le client n'est pas déjà enregistré, on peut cliquer sur l'icône 'plus' pour l'ajouter.

Ajouter un nouveau projet

Nom de projet:

Nom client:

Si ce client n'est pas enregistré, cliquez sur l'icône plus pour l'ajouter.

Description:

description

Start date:

End date:

Chef de projet:

Ajouter des membres à l'équipe:

- ☐ developper_David
- ☐ Sanasoussi111
- ☐ developpeur_Alma
- ☐ Sanasoussi11111

Statut:

Image: Aucun fichier n'a été sélectionné

Figure 4.0.17 Ajouter projet

4.3.5.3 Créer client

Pour créer un nouveau compte d'un client, on doit remplir le formulaire comme il est présenté dans la figure 4.18. Tous les champs du formulaire sont obligatoires.

The form is titled 'Créer client' and is organized into several sections. At the top, there are two input fields for 'Nom' and 'Prénom'. Below these is the 'Adresse:' section with an 'adresse' input field. The 'Telephone:' section has a 'telephone' input field. A section header 'Informations Client' is followed by an 'Iban:' section with an 'IBAN' input field. The 'Title:' section has a 'title' input field. The 'Company name:' section has a 'company_name' input field. The 'Lieu:' section has a 'lieu' input field. A section header 'Connexion' is followed by an 'Email:' section with an 'Email' input field. The 'Password1:' section has a 'Password' input field. The 'Password2:' section has a 'Password check' input field. At the bottom center is a blue button labeled 'Ajouter'.

Figure 4.0.18 Créer client

4.3.5.4 Consulter utilisateur

L'interface « consulter utilisateur » est présentée dans la figure 4.19. Cette interface permet à l'administrateur de consulter, refuser et supprimer un compte utilisateur.

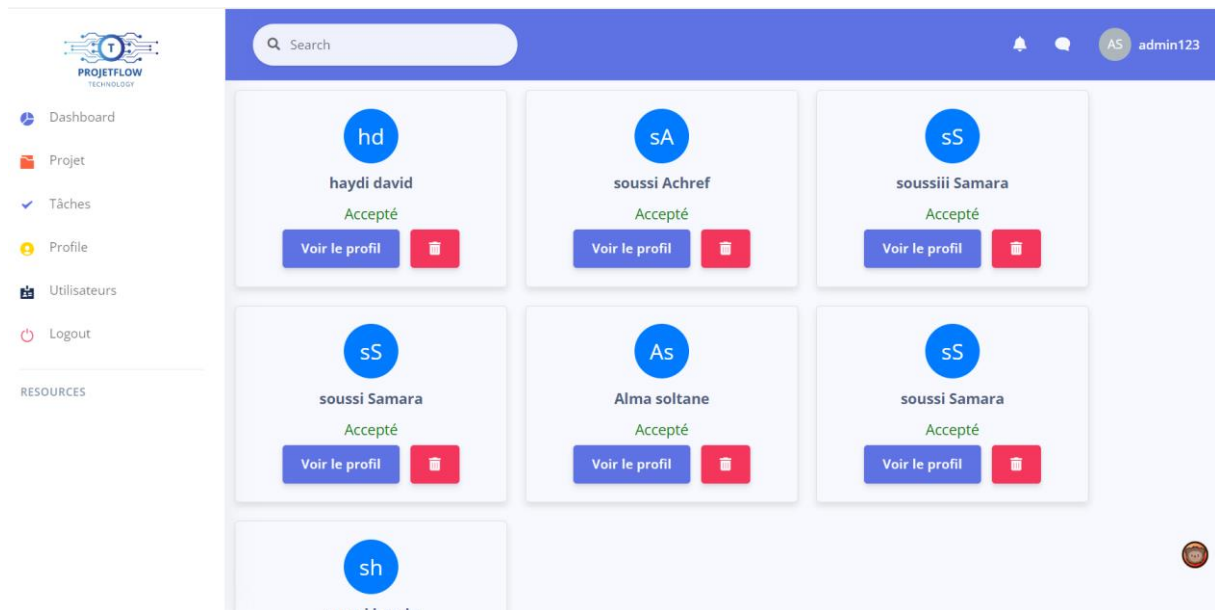


Figure 4.19 Consulter utilisateur

Conclusion

Dans ce chapitre, nous avons tout d'abord annoncé les environnements logiciels Utilisés tout au long de ce projet. Ensuite, nous avons détaillé les étapes de l'implémentation de la solution. Finalement, à travers un ensemble de tests unitaires et d'intégration, nous avons validé l'ensemble des modules développés.



Conclusion générale et perspectives

Mon stage au sein de Delomid m'a fourni une expérience pratique précieuse et m'a permis de développer mes compétences techniques et professionnelles. Durant cette période, j'ai eu l'opportunité de terminer mes tâches avec efficacité, ce qui m'a encouragé à envisager un projet individuel pour obtenir mon diplôme.

Le projet "ProjetFlow" est le fruit de cette ambition. Il représente mon engagement à mettre en pratique mes connaissances, à aborder les problématiques réelles de la gestion de projet, et à concevoir une solution adaptée. Le développement de cette application m'a offert l'occasion d'approfondir mes compétences en matière de conception, de planification et de mise en œuvre.

En termes de perspectives, j'envisage d'améliorer davantage "ProjetFlow", en intégrant des fonctionnalités supplémentaires et en optimisant les processus existants. De plus, les retours d'expérience et les besoins futurs des utilisateurs guideront les améliorations à apporter. J'espère également pouvoir mettre en œuvre les connaissances et les compétences acquises durant mon stage et mon projet de fin d'études dans de futurs rôles professionnels, continuant ainsi mon parcours de croissance et d'apprentissage dans le domaine de l'informatique.



Néographie

UML c'est quoi? URL : <https://www.futura-sciences.com/tech/definitions/informatiqueuml-3979/>

Diagramme de classe : URL : <https://laurent-audibert.developpez.com/Cours-UML/?page=diagramme-classes>

<https://explorweb.github.io/cours2018A/cours/django/>

Le diagramme de séquence : URL : <https://www.cybermedian.com/fr/what-is-modelview-controller-mvc-framework-model-mvc-with-uml-robustness-analysis/>

Draw.io : URL : <https://drawio-app.com/>

Canva : URL : <https://www.canva.com/>

Pusher : URL : <https://fr.quish.tv/create-realtime-application-with-pusher-js-with-nodejs>