

IUT de Montreuil

<https://www.iut.univ-paris8.fr/>



Agitel

<https://www.agitel.fr/>



## **Rapport de stage du BUT Informatique**

---

# Administrateur systèmes & réseaux

---

**Maître de stage**

Mathis JUIGNE

[m.juigne@agitel.fr](mailto:m.juigne@agitel.fr)

**Tuteur enseignant**

Nedra NAUWYNCK

[n.nauwynck@iut.univ-paris8.fr](mailto:n.nauwynck@iut.univ-paris8.fr)

**Étudiant**

Priyank SOLANKI

[priyank.slki@gmail.com](mailto:priyank.slki@gmail.com)

**Promotion 2023 - 2024**

date de début : 15 mai 2024

date de fin : 19 juillet 2024

## Table des matières

<b>Introduction.....</b>	<b>2</b>
<b>Remerciements.....</b>	<b>3</b>
<b>Résumé.....</b>	<b>4</b>
<b>I - Entreprise Agitel.....</b>	<b>5</b>
1. Présentation de l'entreprise.....	6
2. Environnement professionnel.....	7
3. Description sommaire des missions.....	7
<b>II - Projet Dashboard Atera.....</b>	<b>10</b>
1. Contexte, enjeux et objectif.....	10
2. Première version.....	12
2.1. Cahier des charges.....	12
2.2. Visuel.....	13
2.3. Code.....	17
3. Technologies utilisés.....	20
4. Problèmes et axes d'amélioration.....	24
<b>III - Projet Interface compteurs.....</b>	<b>25</b>
1. Contexte, enjeux et objectif.....	25
2. Première version.....	26
2.1. Cahier des charges.....	26
2.2 Visuel.....	27
2.3 Code.....	30
3. Technologies utilisés.....	33
<b>IV - Conclusion.....</b>	<b>34</b>
1. Bilan professionnel.....	34
2. Bilan personnel.....	35
<b>Table des figures.....</b>	<b>36</b>
<b>Lexiques.....</b>	<b>38</b>

## Introduction

La 3ème année de BUT Informatique s'achève avec un stage de 14 semaines. Ce stage est nécessaire afin de valider la 3ème année et a pour objectif d'apporter de l'expérience à l'étudiant dans un milieu professionnel, d'approfondir ses connaissances acquises tout au long de l'année et d'en acquérir de nouvelles. J'ai réussi à trouver ce stage grâce à mon oncle et Perrine Lozach qui s'est chargé de mon recrutement chez Agitel, qui a eu confiance en moi pour un stage de 14 semaines.

Mon stage s'est déroulé au siège d'Agitel une PME<sup>[1]</sup> comptant environ 15 employés située à Paris. Agitel est une entreprise française spécialisée dans la création et l'administration de réseaux de téléphones et de systèmes de paiement, elle accompagne les entreprises dans la création et la gestion de leurs réseaux. Elle possède des concurrents tels que Atos ou bien Capgemini, proposant des services similaires. Cependant Agitel se démarque de par sa fiabilité, sa disponibilité, sa compréhension, son anticipation de l'évolution des besoins clients et de sa réactivité à chaque demande.

Mon stage a été effectué dans le service administration systèmes et réseaux de l'entreprise, au poste de stagiaire administrateur systèmes et réseaux. En tant qu'administrateur systèmes et réseaux, j'étais en relation avec l'équipe du service en distanciel située à Toulon et à Rennes, et j'étais également encadré par différents membres du service.

Durant mon stage, mes missions ont été le développement d'interface web ainsi qu'un petites missions annexes qui est l'installation de Windows Server sur des KVM. Grâce à ces missions j'ai eu l'occasion de découvrir et d'utiliser des logiciels, tels que Atera et Odoo qui sont des outils de gestion ou encore Burp Suite. J'ai également eu l'occasion de réutiliser des outils tels que Postman, PostgreSQL, GitHub, VS Code, etc que j'avais déjà utilisés lors de mes projets universitaires.

En premier lieu, le rapport s'ouvrira sur une présentation de l'entreprise. La seconde partie du rapport portera sur mon premier projet de mon stage, puis une troisième partie sera consacrée à mon deuxième projet de mon stage. Enfin, le rapport se conclura sur l'apport professionnel et personnel du stage.

## **Remerciements**

Tout d'abord, avant de commencer, j'aimerais remercier Perrine Lozach, de m'avoir donné une chance dans l'entreprise Agitel pour ma deuxième expérience professionnelle, de m'avoir fait confiance pour un stage de 3 mois. Je tiens également à remercier toute l'équipe d'administration système et réseaux de l'entreprise pour leur accueil et leur sympathie.

Également, je tiens à remercier mon tuteur enseignant, Nedra Nauwynck, de m'avoir accompagné durant cette période de stage. Je tiens aussi à remercier Isabelle Bonneton et Jeremy Marcinkowski pour nous avoir apporté leur aide dans la rédaction du rapport de stage et la préparation de la soutenance. Enfin, je souhaite remercier l'IUT de Montreuil de m'avoir permis de vivre ma deuxième expérience professionnelle en tant que stagiaire en informatique dans le cadre de mon BUT Informatique.

## **Résumé**

This report summarizes my three-month internship experience at Agitel. This internship was a crucial part of my third year in a Bachelor's degree in Computer Science, as it provided me with my second hands-on experience with a company and allowed me to expand my knowledge.

I did this internship as a system and network administrator, this internship was a great opportunity for me to discover this job. It allowed me to understand the importance of this role in an enterprise.

Before starting this internship, I already had an idea of what an internship was like after my first experience in my second year of a Bachelor's degree at UCAR, so I already knew how it was organized, but I had a very vague idea of what a system and network administrator's job was.

During my internship, my mission was the development of web interfaces, I also contributed to a smaller mission which is the installation of windows server on KVMs.

This internship provided me with a valuable opportunity to explore the working world. It helped me understand the various challenges faced by a company and the importance of teamwork. Apart from gaining technical knowledge, it gave me a clearer understanding of how a company operates.

# I - Entreprise Agitel

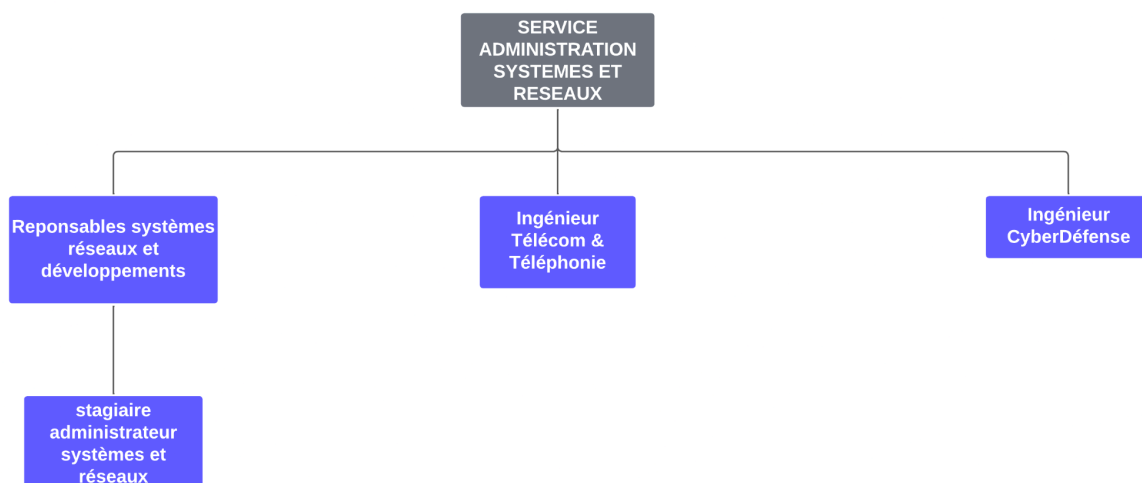
## 1. Présentation de l'entreprise

Agitel est une PME<sup>[1]</sup> qui compte environ 15 employés, elle opère dans l'import-export d'équipements informatiques et accompagne les entreprises dans la création et la gestion de leurs réseaux de téléphonie et de systèmes de paiement. Les locaux du siège sont situés à Paris, dans un appartement haussmannien, dans le 8ème arrondissement, ils sont constitués d'un rez-de-chaussée et d'un étage.

L'entreprise a été fondée en 2007 par Benjamin Vanhaeverbeek dans le but de proposer aux entreprises un accompagnement complet dans la création et gestion de leurs réseaux. Elle possède le statut de SASU<sup>[2]</sup> (Société par Actions Simplifiée Unipersonnelle).

L'objectif d'Agitel est de proposer à ses entreprises clientes la création et l'administration de leurs réseaux informatiques, que ce soit de téléphonie ou de paiement, avec un accompagnement 7 jours sur 7, un conseil et une assistance personnalisée et une anticipation de leurs évolutions, tout en leur faisant part des technologies les plus performantes.

Le service d'administration systèmes et réseaux de l'entreprise se situe en distanciel. Les membres opèrent à Toulon et à Rennes. Ce service compte 4 membres : Mathis Juigne, responsable systèmes, réseaux et développements, Alexandre Allard-Jacquín, ingénieur en télécom et téléphonie, Valentin Lelievre, ingénieur en cyberdéfense, et moi, stagiaire administrateur systèmes et réseaux.



*Organigramme du service administration systèmes et réseaux d'Agitel*

Pour finir Agitel c'est plus de 454 références, 25 collaborateurs et plus de 10 clients dont :

- Ba&sh
- G20
- APRIL
- Sinéquanone
- etc...

## **2. Environnement professionnel**

Au sein de l'entreprise, des logiciels de communication sont utilisés, Microsoft Teams afin de pouvoir communiquer facilement que ce soit par message ou par appel, et Outlook pour envoyer des mails, des documents, etc... Ces logiciels sont communément utilisés dans les entreprises pour leur simplicité et leur facilité d'utilisation.

Comme outils de planification, l'entreprise utilise le logiciel Odoo, qui propose plusieurs services dont le CRM<sup>[3]</sup> (Customer Relationship Management) qui est le service principalement utilisé par l'entreprise. L'entreprise utilise aussi l'outil Atera qui est un RMM<sup>[4]</sup> (Remote Monitoring and Management), qui permet de gérer différentes demandes des clients via un système de ticket. De plus Agitel possède une hotline<sup>[5]</sup> qui s'occupe de la prise en charge des clients pour toute sollicitation via appel téléphonique, permettant ainsi de garder un contact permanent avec la clientèle.

Au sein du service administration système et réseaux, c'est principalement Mathis Juigne qui se charge de la répartition des tâches. Lors de ma mission principale j'ai été principalement encadré par Valentin Lelievre. Durant la phase de développement du projet, un compte rendu journalier était prévu en fin de journée pour constater l'avancée du projet, les différents points de blocage, les changements/améliorations, etc...

### 3. Description sommaire des missions

Durant ce stage, ma mission principale était le développement d'une interface web en PHP. Le but de ce projet était de créer une interface affichant un tableau de bord donnant des informations globales sur les tickets du logiciel Atera. Agitel utilise Atera comme outil de gestion de tickets. L'entreprise a beaucoup de clients, ce qui génère de nombreux tickets et peut rendre leur gestion complexe. C'est là qu'intervient l'idée de ce projet : permettre aux techniciens et aux agences d'avoir un aperçu clair de l'ensemble des tickets pour mieux s'organiser dans leur gestion et leur traitement.

Pour initier ce projet, un cahier des charges a été présenté, permettant de conceptualiser le projet à l'aide de maquettes et de consignes simples et précises.

La réalisation de ce projet a nécessité plusieurs outils :

- PHP : langages de programmation utilisés.
- VSCode : IDE<sup>[6]</sup> utilisé pour coder.
- Wampserver : serveur web permettant le lancement du projet.
- Atera : utilisation de l'API afin de récupérer les données principales.
- Odoo : utilisation de l'API afin de récupérer une partie des données.
- Postman : permettant de tester les API.
- GitHub : utilisé dans la gestion, le suivi et la sauvegarde du projet.
- PostgreSQL : utilisé dans la création d'une base de données.
- DataGrip : utilisé dans la gestion de la base de données.
- Burp Suite Community Edition : utilisé pour analyser le trafic HTTP(S)<sup>[7]</sup>.

Lors du développement, plusieurs étapes se sont distinguées. Tout d'abord il a fallu s'informer sur le logiciel Atera et sa documentation pour comprendre le logiciel et son API. Ensuite, il y a eu la phase de développement. Après cela, il a fallu documenter le projet et le code avec un document technique et un document administratif. Enfin, il y a eu la phase de déploiement pour son utilisation.

J'ai aussi été assigné à des tâches plus petites durant mon stage, comme l'installation de Windows Server<sup>[8]</sup> sur des KVM.



*Une KVM (Kernel-based Virtual Machine) est une technologie de virtualisation intégrée au noyau Linux, permettant de transformer un serveur Linux en un hyperviseur<sup>[9]</sup> capable de créer et de gérer des machines virtuelles.*

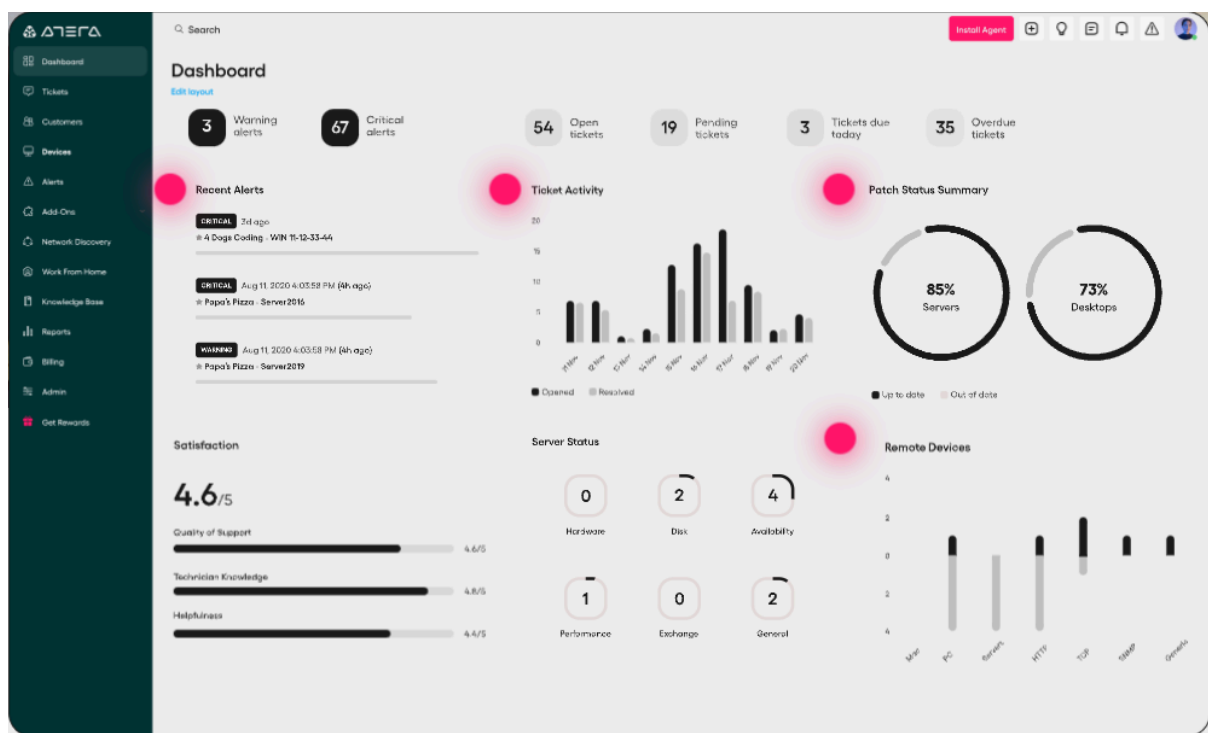
Le but de cette tâche était de suivre la procédure d'installation de la version Windows Server Datacenter 2022 (expérience de bureau). Cette installation visait à fournir une base solide pour la gestion de l'infrastructure IT<sup>[10]</sup> de l'entreprise, en profitant des fonctionnalités avancées offertes par cette version de Windows Server.

## II - Projet Dashboard Atera

### 1. Contexte, enjeux et objectif

Agitel est une entreprise spécialisée dans la gestion de réseaux pour divers clients, répondant à une multitude de demandes, allant de la correction de bugs à la résolution de problèmes techniques. Ces demandes peuvent parfois être chronophages et s'accumuler rapidement, rendant leur gestion difficile. Pour remédier à ce problème, l'entreprise utilise Atera, un outil de gestion permettant de centraliser et d'optimiser le traitement des demandes de support.

Atera est un logiciel de gestion des services informatiques basé sur le cloud<sup>[11]</sup>. Il offre une gamme de fonctionnalités pour aider les entreprises à gérer leurs opérations informatiques, y compris la surveillance à distance, la gestion des tickets, la gestion des actifs, la sauvegarde et bien d'autres encore. Atera est conçu pour les prestataires de services informatiques et les équipes informatiques internes afin de simplifier et d'automatiser les processus de gestion informatique.



Interface du logiciel Atera

Atera est utilisé par l'entreprise comme outil de gestion de tickets. Les clients peuvent créer des tickets afin de faire des demandes qui sont ensuite traitées par des techniciens. Cela facilite la gestion des demandes et permet une meilleure organisation dans leur traitement.

Cependant l'entreprise possède plusieurs clients, et donc accumule rapidement beaucoup de tickets, pour un technicien il est facile de se perdre dans cet amas de tickets. Afin d'aider les techniciens, le service d'administration système et réseaux à eux l'idée de mettre en place une interface web permettant de visualiser différentes informations sur les tickets sous forme de tableau de bord, cela permettra aux techniciens de mieux visualiser cet amas de tickets et ainsi prendre de meilleures décisions dans la gestion et le traitement de ceux-ci.

L'objectif de ce nouveau projet est de concevoir une interface accessible aux différentes agences et techniciens permettant une visualisation rapide, claire et précise de différentes informations sur les tickets. Cela leur permettra d'avoir une meilleure connaissance des tickets, lesquels prioriser, avoir une meilleure organisation, etc...

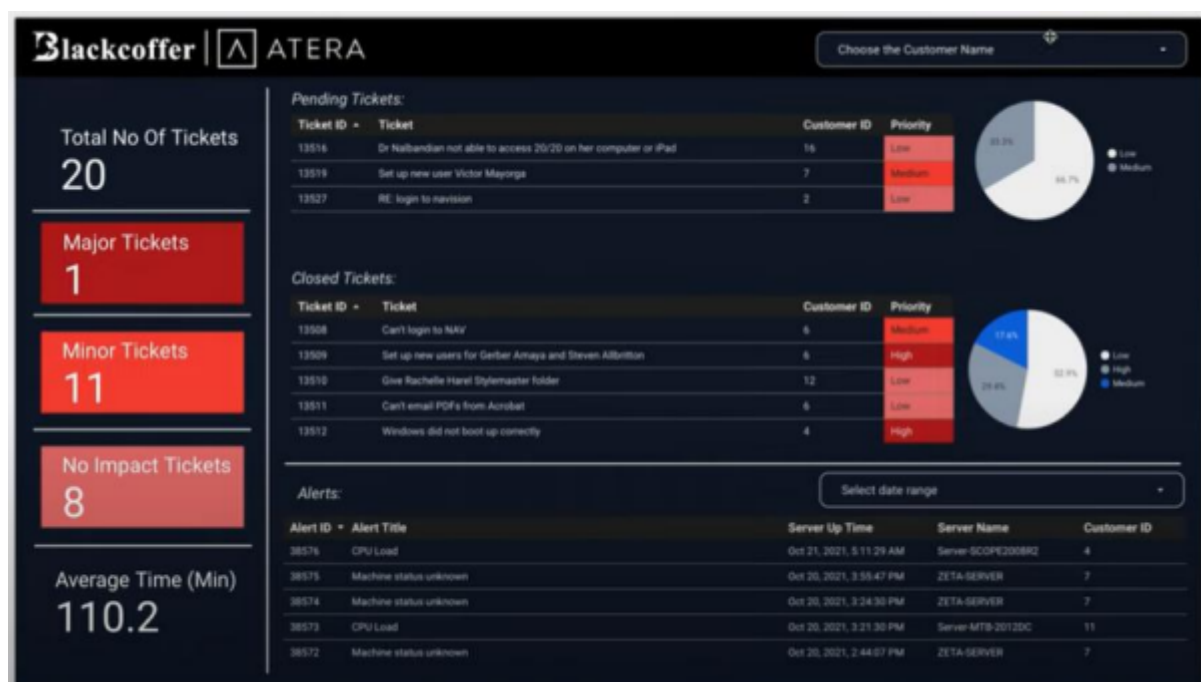
Pour le développeur, les enjeux sont différents : il faut permettre à l'utilisateur une lecture facile et rapide des différentes informations présentes sur le tableau de bord, il faut assurer la sécurité du code afin d'éviter toute fuite de données, assurer de la sécurité des accès afin d'éviter toute attaque, assurer l'exactitude des informations présentes, bien organiser et documenter le code source pour qu'il puisse être repris et amélioré ou modifié pour d'éventuelle version supérieur et surtout s'assurer du bon fonctionnement de l'interface pour une utilisation simple et efficace.

## 2. Première version

### 2.1. Cahier des charges

Le projet à été réalisé seul, par moi même. J'ai été assisté, encadré et suivi par Valentin Lelievre tout au long du projet et j'ai également reçu de l'aide de la part de Léo Pasquereau pour la prise en main du logiciel Odoo. Dans la réalisation de ce projet, un cahier des charges devait être respecté. Il présentait des maquettes et des directives, notamment :

- Utilisation du langage PHP pour l'interface.
- Mise en place d'une page d'authentification avant d'accéder aux données.
- Aucune information ne doit être disponible en clair dans le code.
- L'ensemble du code doit être documenté dans un autre document permettant le suivi de la conceptualisation et la compréhension par d'autres collaborateurs.
- Sécuriser le code avec du chiffrement et contre les vulnérabilités possibles.



Maquette 1 : page du tableau de bord



Maquette 2 : page tickets par clients

## 2.2. Visuel

Passons à la présentation du projet terminé. L'interface présente 3 pages, tout d'abord une page de connexion

**Agitel Groupe**

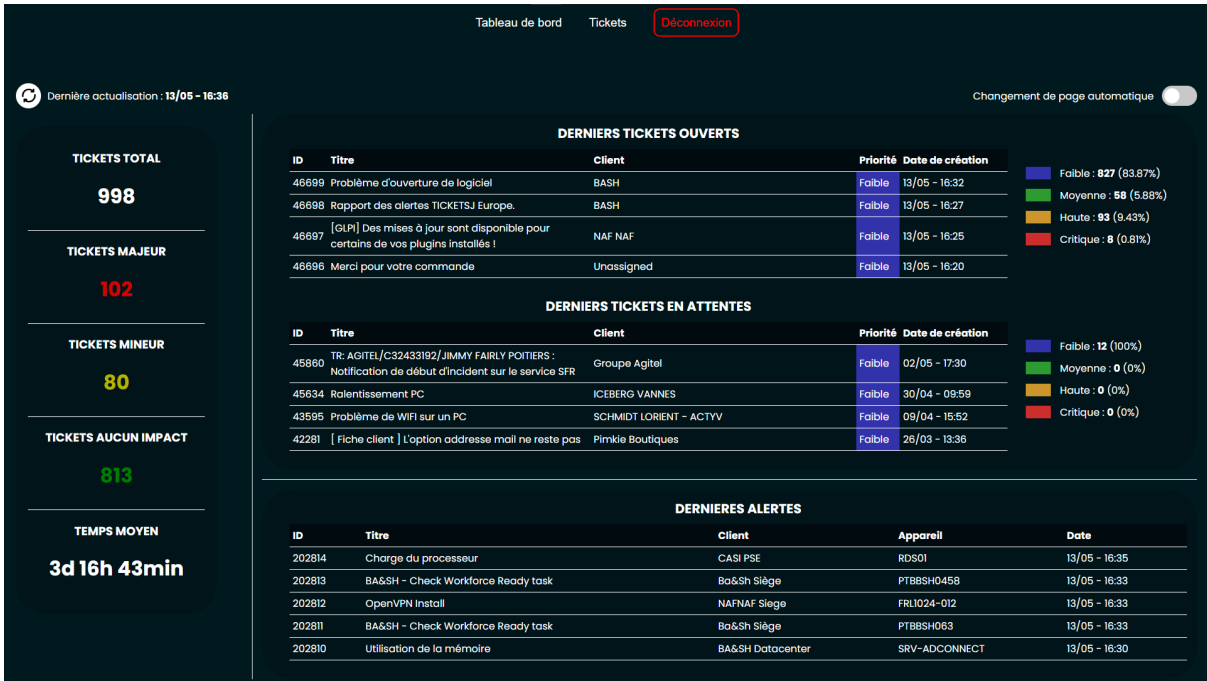
Identifiant

Mot de passe

Se connecter

Page de connexion de l'interface

Un formulaire simple et basique permettant à l'utilisateur de s'authentifier avant d'accéder aux données. Elle garantit la sécurité en empêchant l'accès non autorisé aux informations sensibles.



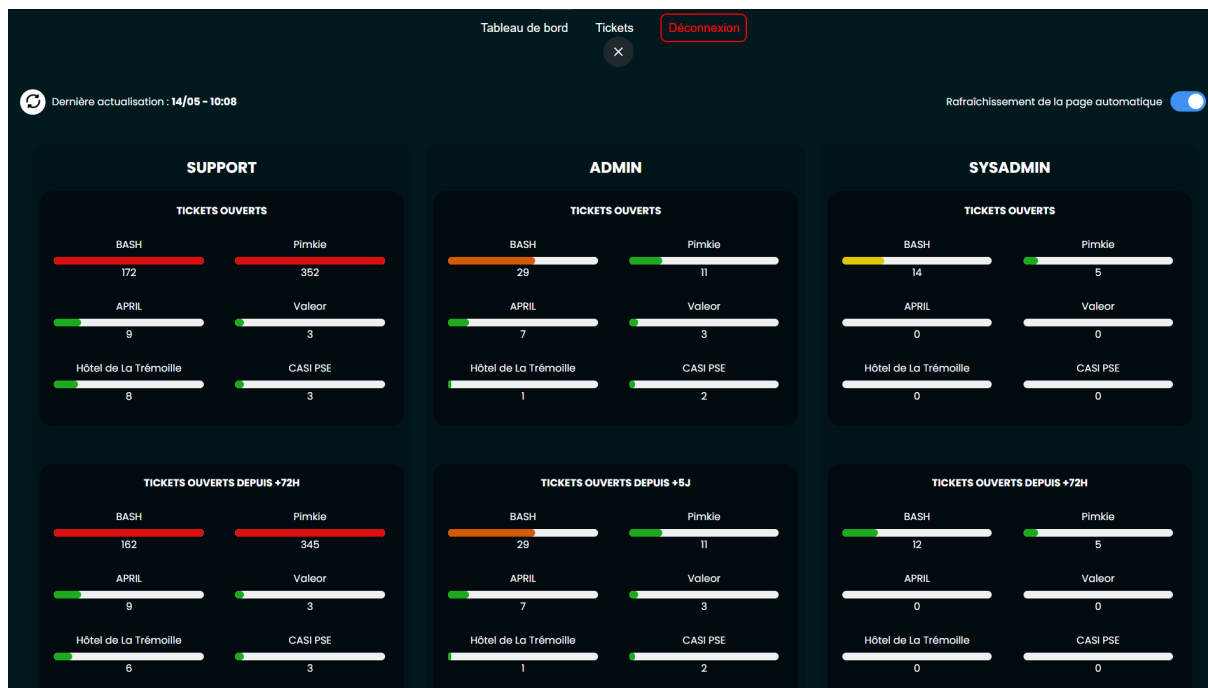
Page tableau de bord de l'interface

Une fois connecté, l'utilisateur accède à la première page, le tableau de bord. Cette page correspond à la première maquette du cahier des charges et reprend la même disposition des informations, mais avec un design différent. Sur cette page on peut y retrouver :

- **Tickets total** : Nombre total de tickets ouverts et en attente.
- **Tickets Majeur/Mineur/Aucun impact** : Nombre de tickets ouverts et en attente par impact.
- **Temps moyen** : Indication du temps moyen de réponse initiale aux tickets.
- **Derniers tickets ouverts** : Tableau comportant : l'id, le titre, le client, la priorité et la date de création des 4 derniers tickets ouverts.
- **Statistiques des tickets ouverts** : Nombre et pourcentage par priorité (faible, moyenne, haute et critique).
- **Derniers tickets en attente** : Tableau comportant : l'id, le titre, le client, la priorité et la date de création des 4 derniers tickets en attente.

- **Statistiques des tickets en attente** : Nombre et pourcentage par priorité (faible, moyenne, haute et critique).
- **Dernières alertes** : Tableau comportant : l'id, le titre, le client, l'appareil et la date des 5 dernières alertes.

Cette page permet aux techniciens de visualiser rapidement l'état actuel des tickets, les priorités et les alertes, facilitant ainsi une gestion efficace et une prise de décision éclairée.



*Page tickets clients*

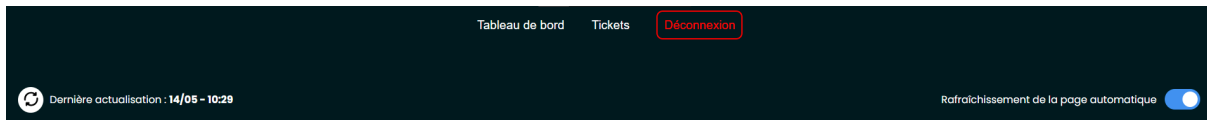
La deuxième page principale reprend les informations de la deuxième maquette avec une disposition différente. Elle est divisée en trois parties distinctes : SUPPORT, ADMIN et SYSADMIN, représentant respectivement les tickets destinés au support, à l'administratif et à l'administration système.

Dans chaque onglet, les utilisateurs peuvent voir le nombre de tickets ouverts pour chaque gros client, notamment :

- BASH
- Pimkie
- APRIL
- Valeor
- Hôtel de la Trémoille
- CASI PSE

De plus, une barre de progression est incluse pour évaluer le volume des tickets et déterminer s'il est élevé ou non.

Dans la partie SUPPORT et SYSADMIN, une case dédiée affiche le nombre de tickets dépassant un délai de 72 heures, tandis que dans la partie ADMIN, cette limite est de 5 jours. Cette fonctionnalité permet aux techniciens de suivre les tickets en retard et de prendre les mesures nécessaires pour assurer un traitement rapide et efficace.



*Header de l'interface*

Le header représente la partie supérieure de chaque page de l'interface. Il offre à l'utilisateur plusieurs fonctionnalités essentielles, notamment :

- Barre de navigation : Permet à l'utilisateur de changer de page et de se déconnecter facilement.
- Bouton de rafraîchissement : Permet à l'utilisateur de mettre à jour la page et voir la date et l'heure du dernier rafraîchissement.
- Switch de mode automatique : Offre deux options de mode automatique :
  - Mode "changement de page automatique" : Change automatiquement de page en alternant les deux pages principales avec un intervalle de 2 minutes.
  - Mode "rafraîchissement de page automatique" : Rafraîchit les données de la page toutes les 2 minutes.

Pour assurer une meilleure compréhension du projet et faciliter sa maintenance et son amélioration ultérieure, un document administratif et technique a été rédigé. Ces documents fournissent des informations détaillées sur le projet, son fonctionnement, son architecture et son code source. Ils sont disponibles en annexes et comprennent :

- Document administratif : Présente une vue d'ensemble du projet, ses objectifs, ses fonctionnalités principales, et ses utilisateurs cibles.
- Document technique : Décrit en détail l'architecture logicielle du projet, y compris les technologies utilisées, la structure du code source, les bases de données et les API et les instructions pour l'installation.

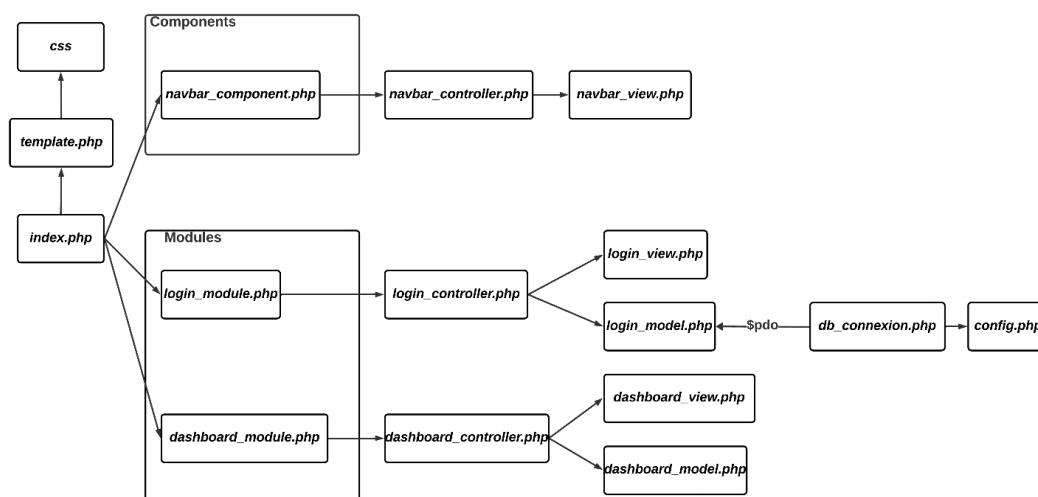


## 2.3. Code

Le projet adopte une architecture MVC (Modèle Vue Contrôleur), qui est destiné aux interfaces graphiques. Cette architecture permet de séparer le modèle de la vue pour une meilleure lisibilité et compréhension du code. Elle est composé de trois parties :

- **Modèle** : Cette partie représente les données de l'application. Dans notre cas, le modèle est responsable d'interagir avec la base de données et les API. Il est chargé de manipuler les données et de les récupérer.
- **Vue** : La vue représente l'interface utilisateur et est responsable de l'affichage des données à l'utilisateur. Elle est chargée de présenter les informations de manière claire et compréhensible, en utilisant des éléments visuels tels que des formulaires, des tableaux, des graphiques, etc. La vue communique avec le contrôleur pour récupérer les données à afficher, mais n'est pas responsable de leur manipulation directe.
- **Contrôleur** : Le contrôleur agit comme un intermédiaire entre le modèle et la vue. Il reçoit les demandes de l'utilisateur depuis l'interface utilisateur (vue), puis fait appel au modèle approprié pour récupérer ou manipuler les données nécessaires. Une fois les opérations sur les données effectuées, le contrôleur renvoie les résultats à la vue pour l'afficher à l'utilisateur.

Voici une représentation complète de l'architecture du projet.



Représentation de l'architecture du projet sous forme de diagramme

Afin de donner accès à notre interface, la meilleure solution est d'utiliser une base de données, pour garantir la sécurité, la gestion et l'administration des comptes.

```

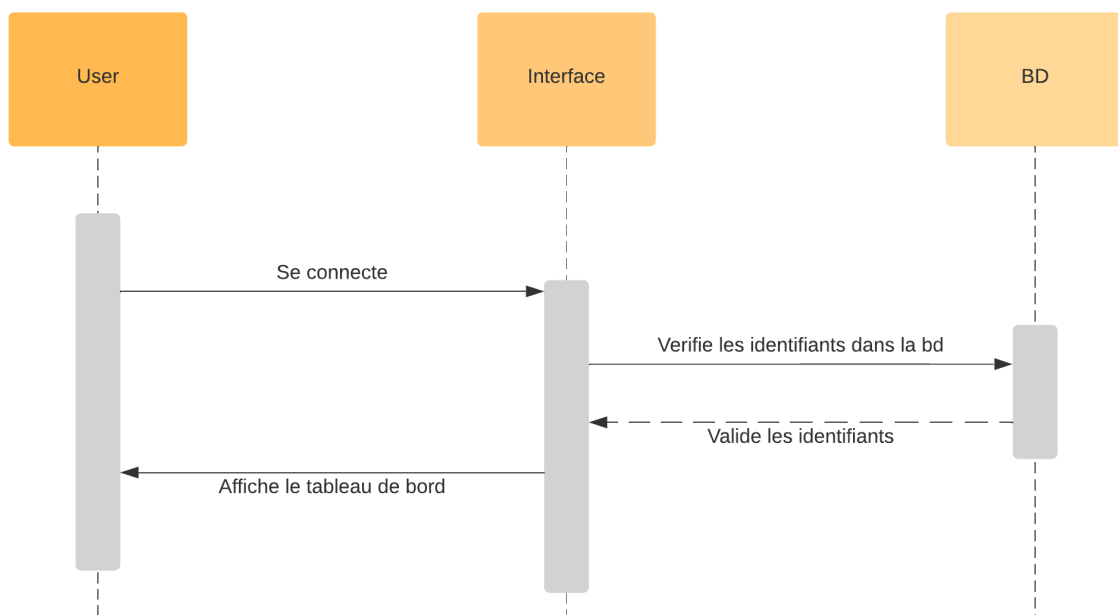
/* USERS TABLE */
CREATE TABLE Users(
    id SERIAL PRIMARY KEY,
    login varchar(32),
    password varchar(255),
    atera_api_key varchar(255),
    odoo_db_name varchar(255),
    odoo_username varchar(255),
    odoo_api_key varchar(255)
);

```

*Script SQL de la création de la base de données*

Notre base de données possède qu'une seule table, la table Users, qui contient les champs suivants :

- **id**, clé primaire et unique de la table, permet de distinguer chaque instance.
- **login**, le nom d'utilisateur.
- **password**, le mot de passe.
- **atera\_api\_key**, la clé API d'Atera.
- **odoo\_db\_name**, le nom de la base de données Odoo.
- **odoo\_username**, le nom d'utilisateur d'Odoo.
- **odoo\_api\_key**, la clé API d'Odoo.

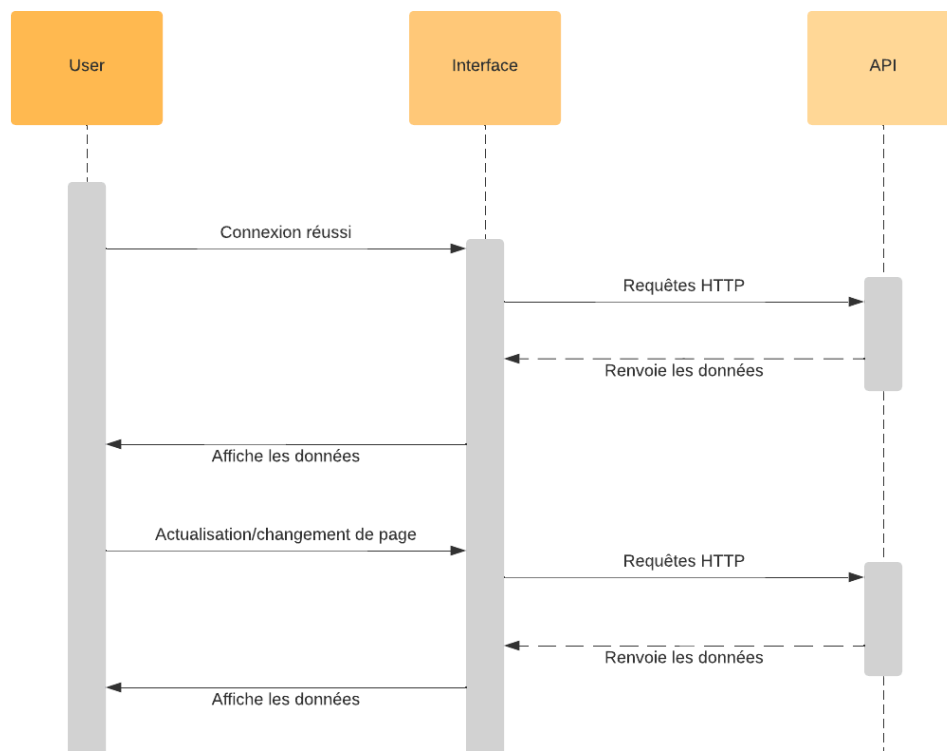


*Diagramme de séquence de l'interaction entre l'interface et la base de données*

Actuellement, voici comment fonctionne l'interaction entre l'interface et la base de données.

Lorsqu'un utilisateur tente de se connecter, l'interface interroge la base de données pour vérifier si les identifiants fournis sont corrects. Si les identifiants sont valides, l'interface récupère les données associées à cet utilisateur depuis la base de données. Ces données comprennent la clé API d'Atera, le nom de la base de données Odoo, le nom d'utilisateur d'Odoo et la clé API d'Odoo.

Ensuite, ces informations sont stockées dans des variables de session pour maintenir la connexion de l'utilisateur tout au long de sa session. Une fois que les données sont récupérées et stockées avec succès, l'interface redirige l'utilisateur vers la page du tableau de bord.



*Diagramme de séquence de l'interaction entre l'interface et les API*

Dès la connexion l'interface effectue des requêtes à l'API d'Atera et à l'API d'Odoo afin de récupérer les données actualisées nécessaires à l'affichage. À chaque actualisation de la page ou changement de page, les requêtes sont refaites à l'API pour garantir les informations les plus récentes disponibles.

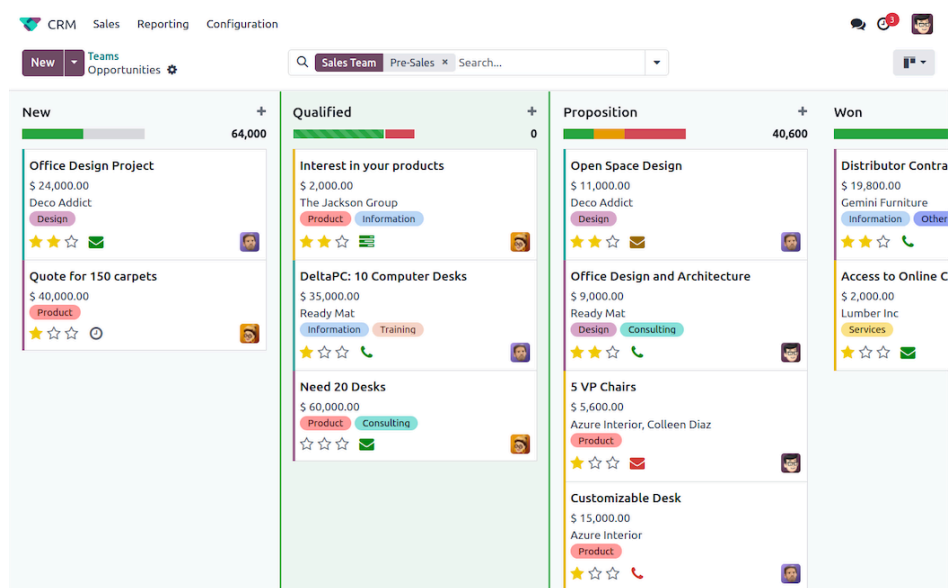
### 3. Technologies utilisés

Comme mentionné précédemment, les données de l'interface sont principalement basées sur le logiciel Atera. Cependant, une petite partie des données (la partie ADMIN) proviennent du logiciel Odoo.



Partie ADMIN de la page ticket clients de l'interface

Pour séparer les tickets administratifs des autres, l'entreprise utilise Odoo afin d'avoir une meilleure gestion pour ces tickets là.



Service CRM du logiciel Odoo

Afin de récupérer les données des tickets pour notre interface utilisateur, nous utilisons les API proposées par Atera et Odoo.

Une API (Interfaces de Programmation Applicatives) est une interface logicielle qui permet à différents systèmes informatiques de communiquer entre eux. Elles définissent les règles et les protocoles à suivre pour l'échange de données. Pour récupérer les données via ces API, nous utilisons des requêtes HTTP<sup>[7]</sup> spécifiques aux ressources que nous souhaitons obtenir, dans notre cas, les tickets. Les requêtes renvoient les données sous format de données textuelles, principalement du JSON<sup>[12]</sup>. De plus, pour les API d'Atera et Odoo, il est nécessaire d'avoir une clé API générée à partir du compte sur lequel nous voulons récupérer les données. Cette clé agit comme une authentification et permet de valider notre identité.

`"https://app.atera.com/api/v3/tickets?page=2&itemsInPage=50&ticketStatus=open"`

Exemple de requête HTTP

```

1 {
2     "string": "text",
3     "number": 1.0,
4     "boolean": true,
5     "array": [],
6     "object": {
7         "key": "value"
8     }
9 }
```

Exemple du format JSON

Les API d'Atera et d'Odoo sont complètement différentes, elles possèdent des points forts et points faibles.

	Atera	Odoo
<b>Points fort</b>	Documentation Utilisation Intégration	Modularité Performance
<b>Points faible</b>	Performance Limitations fonctionnelles	Complexité Intégration

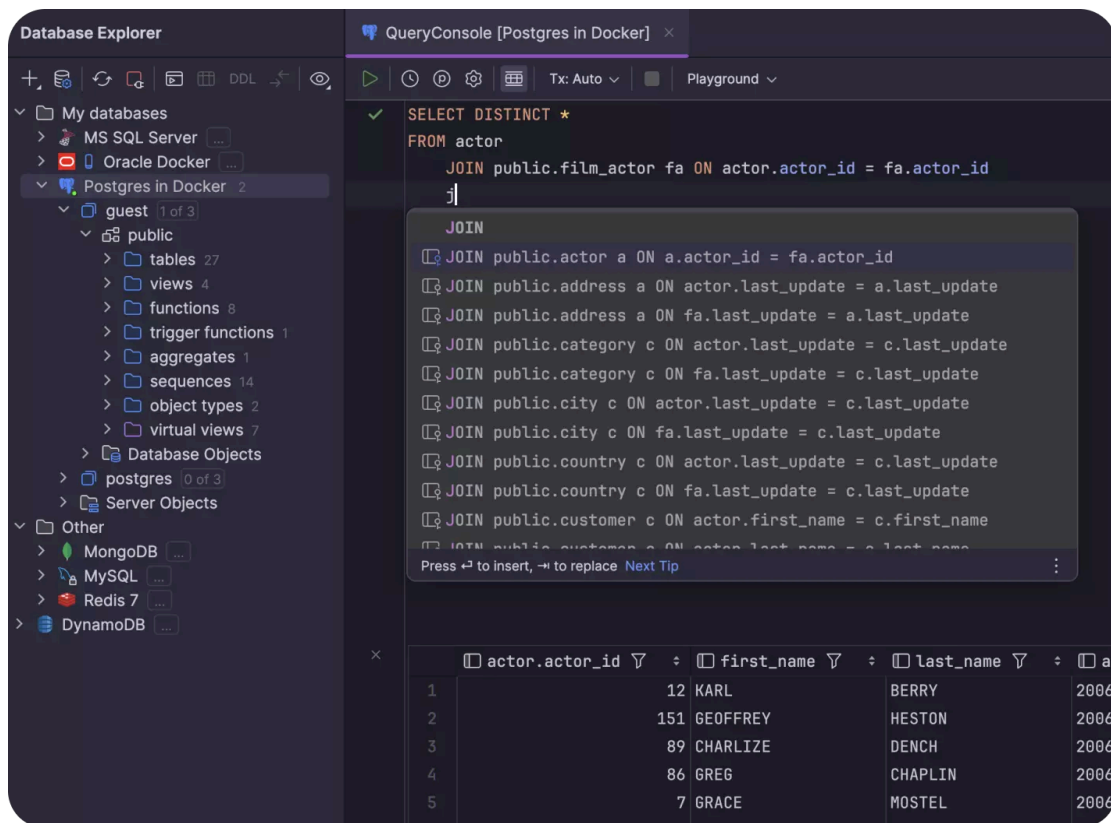
L'API d'Atera propose une documentation simple avec une liste de toutes les requêtes HTTP<sup>[7]</sup> utilisables via le système Swagger<sup>[13]</sup>. Elle est très facile à intégrer car nous avons seulement besoin d'une clé API pour l'utiliser. Cependant, elle présente une limitation fonctionnelle avec des requêtes trop générales, ce qui nécessite une manipulation des données pour obtenir des informations plus précises. En revanche, l'API d'Odoo possède une grande modularité qui offre la possibilité de récupérer des informations précises. Cependant, sa documentation est très vague et nécessite plus de paramètres pour son utilisation, en plus de la clé API, il faut, le nom de la base de données et le nom d'utilisateur, ce qui rend l'intégration plus difficile.

Pour le développement et le test des API, ainsi que pour l'exploration des données, j'ai utilisé le logiciel Postman. Postman est un outil polyvalent permettant de tester des API facilement. Il offre une interface conviviale et facile d'utilisation et permet une visualisation des réponses afin de faciliter leur exploitation.

Pour la réalisation du projet, nous avons choisi d'utiliser le langage PHP, un langage de développement web, en raison de sa polyvalence et de sa large adoption dans le domaine du développement web. De plus, il offre une grande flexibilité pour la manipulation des données et la création d'interfaces utilisateur dynamiques. Ces caractéristiques en font un choix attrayant pour notre projet.

Afin de permettre des accès à l'interface et de permettre d'associer les comptes Atera et Odoo à un compte de notre interface, la meilleure solution est l'utilisation d'une base de données PostgreSQL. PostgreSQL est un système de gestion de base de données relationnelle (SGBDR) open source, puissant, facile à intégrer et conforme aux normes SQL<sup>[14]</sup>. Les accès à l'interface sont directement gérés sur la base de données et les données de la base sont chiffrées pour une meilleure sécurité.

Pour administrer notre base de données, nous avons utilisé Datagrip, un outil offrant une interface conviviale pour la conception, le développement et la gestion des bases de données PostgreSQL. Il nous permet d'exécuter des scripts SQL<sup>[14]</sup>, de visualiser les schémas de base de données et d'effectuer des opérations de maintenance avec facilité.



Interface du logiciel DataGrip

Afin de tester notre code et d'avoir une visualisation du rendu de celui-ci, j'ai utilisé Wampserver comme environnement de développement local. Wampserver fournit un serveur web utilisable en local et permet l'exécution d'applications web développées en PHP.

Pour intercepter le trafic HTTP/HTTPS<sup>[7]</sup>, j'ai utilisé Burp Suite qui est un outil permettant de voir et d'analyser le flux des requêtes HTTP/HTTPS<sup>[7]</sup>, dans notre cas il à été utile pour voir si aucune requête aux API était visible par l'utilisateur, car les requêtes comporte des données sensible.

L'entièreté du projet à été codée sur Visual Studio Code, un éditeur de code gratuit et facile d'utilisation. De plus, le projet à été enregistré sur mon GitHub pour une meilleure gestion du code et une meilleure visualisation de l'avancée et du suivi de celui-ci.



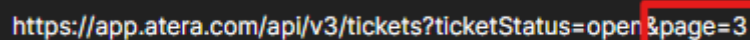
Logo : Atera - Odoo - Postman - PHP - PostgreSQL - DataGrip - Wampserver - Burp Suite - VS Code - GitHub

#### 4. Problèmes et axes d'amélioration

Pendant la réalisation de ce projet, un problème majeur est survenu. Comme mentionné précédemment, l'API d'Atera présente une limitation fonctionnelle, car elle ne permet que de récupérer des données générales et non assez précises, telles que le nombre de tickets ayant un impact majeur, moyen ou mineur.

Pour accéder à ces informations, il est nécessaire de parcourir tous les tickets et de compter le nombre de ceux qui ont un impact majeur, moyen ou mineur. Cela est très chronophage en raison du grand nombre de tickets à traiter.

De plus, l'API fonctionne avec un système de pagination qui limite la récupération à 50 tickets par page, soit 50 tickets par requête. Par conséquent, plusieurs requêtes doivent être envoyées à l'API pour parcourir toutes les pages.



`https://app.atera.com/api/v3/tickets?ticketStatus=open&page=3`

*Système de pagination via les requêtes HTTP*

Cependant, nous souhaitons que notre interface fournisse les données en temps réel provenant d'Atera et d'Odoo. Pour cela, il est essentiel d'actualiser fréquemment les données afin que les utilisateurs disposent des informations les plus récentes.

Pour remédier à ce problème, la meilleure solution consiste à parcourir l'ensemble des tickets une seule fois à chaque actualisation, puis à calculer toutes les informations précises en une seule opération. Malgré cette optimisation, le temps nécessaire pour rafraîchir les données reste relativement long, soit environ 25 secondes. Cependant, pour une interface en arrière-plan, ce délai reste acceptable.



### **III - Projet Interface compteurs**

#### **1. Contexte, enjeux et objectif**

Non seulement spécialisée dans la gestion de réseaux, Agitel est une entreprise qui propose également comme service un système de compteurs de passage à ses clients. Ce système a pour but de comptabiliser le nombre de personnes entrant dans une boutique cliente sur une ou plusieurs journées. Ces données sont stockées et insérées dans une base de données.

Afin de donner un retour permanent sur les statistiques générées par ce système de compteurs de passage à leurs clients et d'avoir un système de gestion de ceux-ci. Le service administration système et réseaux a eu l'idée de mettre en place une interface permettant une visualisation globale des compteurs et de leurs statistiques aux agences associées ainsi qu'à différents membres de chez Agitel dont le responsable d'agence.

L'objectif de ce projet est de développer une interface accessible aux différentes agences clientes ainsi qu'aux responsables d'agence de chez Agitel. Cette interface a pour but de pouvoir présenter une page aux clients avec leurs compteurs et leurs statistiques, tandis que les comptes Agitel auront une vue sur tous les compteurs présents dans la base de données et posséderont en plus un système de gestion des ressources, dont les comptes pour permettre les accès, les compteurs pour leur gestion et les alertes.

L'enjeu ici pour le développeur est de pouvoir proposer un système de connexion sécurisé afin d'éviter toute attaque, sécuriser le code afin d'éviter les fuites de données, proposer une interface claire et facile à comprendre afin d'avoir la meilleure expérience utilisateur possible, s'assurer de l'exactitude des données présentes dans l'interface, fournir une bonne documentation afin de donner la possibilité de pouvoir reprendre le code pour l'améliorer pour d'éventuelles versions supérieures et enfin s'assurer du bon fonctionnement de l'interface pour une utilisation facile et agréable.

## **2. Première version**

### **2.1. Cahier des charges**

Le projet est réalisé seul, par moi-même et j'ai été encadrée et suivie par Antoine Lebon, responsable des agences, tout au long de la réalisation du projet. Cette fois-ci il n'y avait pas de cahier des charges mais seulement des directives et des fonctionnalités à développer, dont voici :

- Une interface client qui présentes :
  - Le nom de l'entreprise
  - Un graphique camembert du nombre de compteurs opérationnels sur nombre total de compteurs
  - Un graphique en barres des 10 meilleurs boutiques de l'entreprise
  - La liste des compteurs
- Une interface pour les membres d'Agitel qui présente les fonctionnalités suivantes :
  - Gestion des comptes de l'interface
  - Gestions des compteurs
  - Visualisation des alertes

De plus, une maquette a été fournie pour apporter un design à respecter pour l'interface.



Maquette 1 : Interface client

## 2.2 Visuel

Passons à la présentation du projet terminé. L'interface possède 5 pages.  
La page de connexion.

**Interface compteurs**

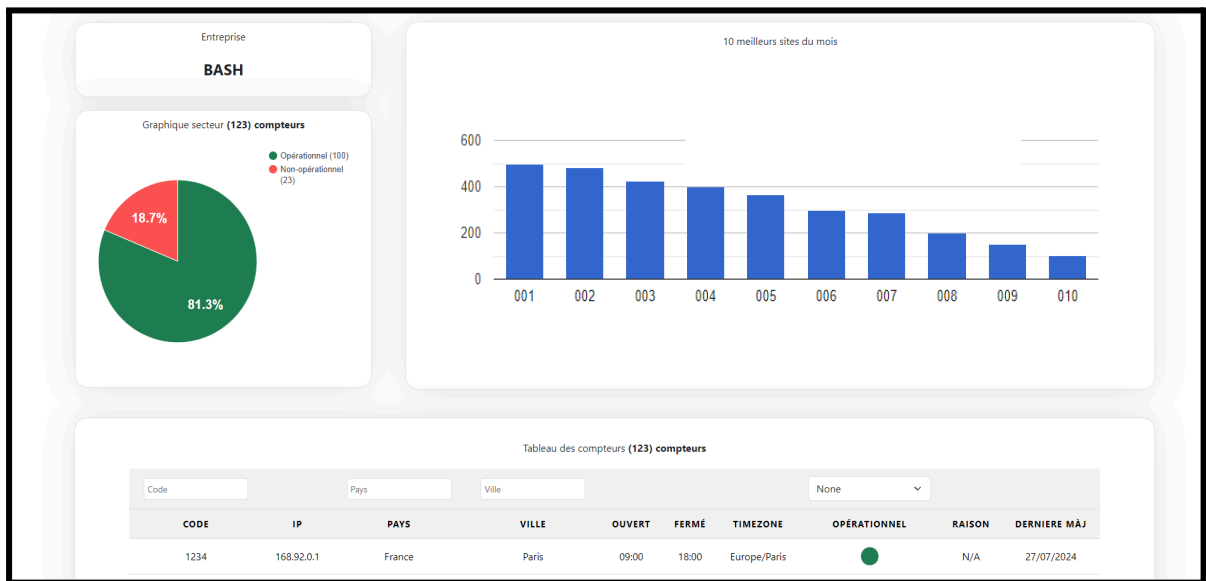
Identifiant

Mot de passe

**Se connecter**

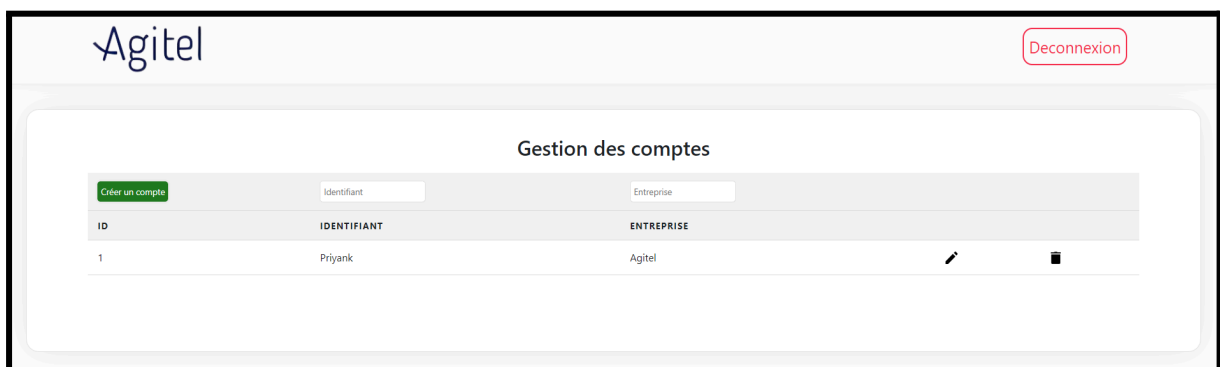
Page de connexion de l'interface

Un formulaire simple et basique permettant à l'utilisateur de s'authentifier avant d'accéder aux données. Elle garantit la sécurité en empêchant l'accès non autorisé aux informations sensibles.



Page tableau de bord de l'interface

Une page de tableau de bord qui contient : le nom de l'entreprise du compte, un graphique en bars représentant les 10 boutiques qui ont fait le plus d'entrées du mois actuel, un graphique camembert qui représente le pourcentage des compteurs opérationnels et non opérationnels, ainsi qu'un tableau listant tous les compteurs de passage de l'entreprise avec des fonctions de recherche associées.



Page de gestions des comptes de l'interface

Une page destinée à la gestion des comptes afin de gérer les accès des différents utilisateurs. Cette page est uniquement accessible pour les comptes Agitel qui représentent les administrateurs. Sur cette page nous pouvons voir tous les comptes de l'interface avec leurs identifiants ainsi que l'entreprise à laquelle ils sont associés. Nous pouvons effectuer une

recherche par identifiant ou/et par entreprise et enfin nous pouvons créer un nouveau compte, ainsi que modifier ou supprimer un compte déjà existant.

Gestion des compteurs									
BASH	Code	Pays	Ville	None	<a href="#">Ajouter un compteur</a>				
ENTREPRISE	CODE	IP	PAYS	VILLE	OUVERT	FERMÉ	OPÉRATIONNEL	RAISON	
BASH	001	168.92.0.1	France	Paris	08:00	21:00	<div></div>	Installation incomplète	<div></div>
BASH	002	168.92.0.2	France	Paris	08:00	21:00	<div></div>	N/A	<div></div>
BASH	003	168.92.0.3	France	Paris	08:00	21:00	<div></div>	N/A	<div></div>

Page de la gestion des compteurs de l'interface

Une page destinée aux comptes Agitel pour la gestion des compteurs, avec un tableau répertoriant tous les compteurs de la base de données avec un système de recherche adapté. Un bouton permettant d'ajouter un nouveau compteur, et la possibilité de modifier un compteur.

Agitel		Accueil	Comptes	Compteurs	Alertes	Deconnexion
Alertes						
BASH						
ENTREPRISE	CODE COMPTEUR	ALERTE	DATE			
BASH	1203	Le compteur de l'entreprise BASH à fait 0 entrée le 22/07/2024	23/07/2024			

Page alerte de l'interface

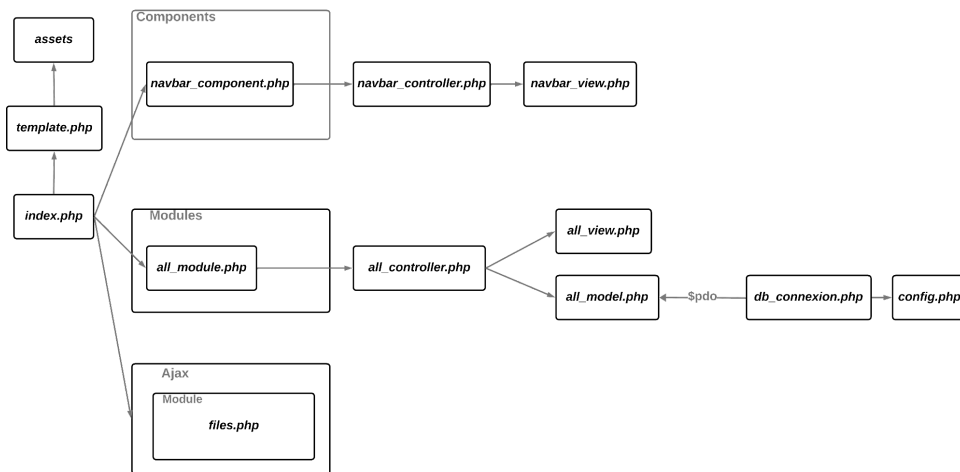
Une page alerte, qui affiche une alerte pour chaque boutique qui a fait 0 entrée le jour précédent.

## 2.3 Code

Comme le projet dashboard, celui-ci adopte une architecture MVC (Modèle Vue Contrôleur) destiné aux interfaces web, pour les mêmes raisons. Il permet de séparer le modèle de la vue pour une meilleure lisibilité et compréhension du code.

Cependant notre projet utilise de l'AJAX (Asynchronous JavaScript And XML), l'AJAX est une méthode qui utilise plusieurs technologies de développement web permettant de rendre une page dynamique interactive, en effectuant des requêtes aux serveurs web sans passer par de nouvelles pages web. Elle permet donc d'actualiser des données sans rafraichir la page.

Voici une représentation de l'architecture.



Représentation de l'architecture du projet sous forme de diagramme

Notre projet communique avec plusieurs bases de données, une base de données PostgreSQL qui permet la gestion des utilisateurs.

```

DROP SCHEMA IF EXISTS counters CASCADE;
CREATE SCHEMA counters;
SET search_path TO counters;

/* USERS TABLE */
CREATE TABLE Users(
    id SERIAL PRIMARY KEY,
    login varchar(32),
    password varchar(255),
    company varchar(32)
);

```

*Script SQL de la base de données pour les utilisateurs*

Cette base de données possède une table Users, qui représente les comptes d'accès à l'interface, elle comporte :

- **Id**, identifiant unique du compte
- **Login**, identifiant à entrer pour se connecter
- **Password**, le mot de passe à entrer pour se connecter
- **Company**, l'entreprise à laquelle appartient le compte

Concernant l'autre base de données, elle m'a été fournie par l'entreprise, qui stocke les données dans une base de données MySQL.

```

CREATE TABLE sites(
    enterprise varchar(50) PRIMARY KEY,
    code varchar(50) PRIMARY KEY,
    ip_till varchar(50),
    country varchar(50),
    city varchar(50),
    open datetime,
    close datetime,
    timezone varchar(50),
    ignore integer,
    ignore_reason varchar(255),
    last_update datetime
);

CREATE TABLE counting(
    enterprise varchar(50) PRIMARY KEY,
    code varchar(50) PRIMARY KEY,
    day_date datetime PRIMARY KEY,
    count Integer,
);

```

*Script SQL de la base de données des compteurs*

La base de données des compteurs comporte plusieurs tables, dont les principales utilisées sont **sites** et **counting**. La table **sites** représente tous les compteurs de passage, et la table **counting** répertorie toutes les données sur les entrées comptabilisées par les compteurs pour chaque boutique de chaque entreprise.

Pour la table *sites*, les principaux champs utilisés sont :

- **enterprise**, le nom de l'entreprise
- **code**, le code boutique
- **ip\_till**, l'adresse IP du compteur
- **country**, le pays de la boutique
- **city**, la ville de la boutique
- **open**, l'heure d'ouverture de la boutique
- **close**, l'heure de fermeture de la boutique
- **timezone**, la zone horaire de la boutique
- **ignore**, l'opérationnalité du compteur
- **ignore\_reason**, la raison de la désactivation du compteur
- **last\_update**, la date de la dernière modification du compteur

Pour la table *counting*, les champs sont :

- **enterprise**, le nom de l'entreprise
- **code**, le code boutique
- **day\_date**, la date des données enregistrées
- **count**, le nombre d'entrées que le compteurs à comptabilisées

Pour l'interaction avec les bases de données, on utilise des requêtes préparées. Cette méthode a pour but de garantir une meilleure sécurité en évitant les injections SQL<sup>[14]</sup>.

Cette méthode possède deux phases : la préparation et l'exécution.

Lors de la première phase, la requête SQL<sup>[14]</sup> est envoyée au serveur avec les paramètres marqués comme des espaces réservés pour que le serveur puisse analyser, compiler et optimiser le plan d'exécution de la requête sans l'exécuter. Ensuite, dans la deuxième phase, les valeurs des paramètres sont envoyées séparément et le serveur utilise la requête préparée pour l'exécuter.



```
try {
    $countStmt = $pdo->prepare("SELECT COUNT(*) FROM counters.Users WHERE login LIKE ? AND company LIKE ?");
    $countStmt->execute(array($loginLike, $companyLike));
    $total = $countStmt->fetchColumn();

    echo json_encode([
        'status' => 'success',
        'total_pages' => ceil($total/$per_page)
    ]);
} catch (PDOException $e) {
    echo json_encode(['status' => 'error', 'message' => "Erreur de requête : " . $e->getMessage()]);
}
```

*Exemple de requête prepare*

### 3. Technologies utilisés

Les technologies utilisées sont relativement les mêmes que pour le projet Dashboard, pour les mêmes raisons.

Pour la réalisation du projet, nous avons choisi d'utiliser le langage PHP. Nous avons utilisé une base de données PostgreSQL. Les accès à l'interface sont directement gérés sur la base de données. Pour administrer notre base de données, nous avons utilisé DataGrip. Afin de tester notre code et d'avoir une visualisation du rendu de celui-ci, j'ai utilisé WampServer comme environnement de développement local. L'ensemble du projet a été codé sur Visual Studio Code. De plus, le projet a été enregistré sur mon GitHub.



*Logo : PHP - PostgreSQL - DataGrip - Wampserver - VSCode - GitHub*

## **IV - Conclusion**

### **1. Bilan professionnel**

Ce stage de trois mois au sein d'Agitel a été une expérience très enrichissante sur le plan professionnel. Il m'a permis d'appliquer les connaissances acquises tout au long de mes années de BUT Informatique, notamment en développement web, dans un environnement professionnel réel.

Tout d'abord, j'ai pu découvrir le fonctionnement d'une entreprise comme Agitel et comprendre ses objectifs et les enjeux auxquels elle est confrontée. J'ai travaillé au sein du service d'administration système et réseaux, ce qui m'a donné une vision globale des différentes activités liées à la gestion des infrastructures. J'ai pu observer comment les équipes s'organisent, communiquent et collaborent pour assurer le bon fonctionnement des systèmes informatiques.

En tant qu'administrateur système et réseaux, j'ai eu l'opportunité d'être encadré par une équipe passionnée et motivée, ce qui m'a permis d'apprécier pleinement mon stage. J'ai appris à élaborer un projet, travailler en autonomie, comprendre les besoins clients et y apporter des solutions.

Une partie importante de mon stage a été consacrée au développement d'une interface web en PHP, en suivant un cahier des charges. J'ai pu acquérir une expérience pratique dans l'élaboration d'un projet web. Cela m'a permis de renforcer mes compétences techniques et de comprendre les différentes étapes de celui-ci.

En conclusion, ce stage chez Agitel a été une expérience formatrice, très épanouissante, qui m'a permis de consolider mes connaissances théoriques et d'acquérir de nouvelles compétences pratiques dans le domaine de l'informatique. J'ai pu appréhender les enjeux d'un administrateur système et réseaux au sein d'une entreprise et développer des compétences précieuses en gestion de projet.

## **2. Bilan personnel**

Ce stage de deux mois au sein d'Agitel a été une expérience très enrichissante sur le plan personnel. Il m'a permis de me confronter à un environnement professionnel réel et de développer mes compétences techniques et professionnelles.

Dans l'organisation du travail, les tâches attribuées m'ont été accessibles. J'ai également bénéficié du soutien de mes collègues en cas de blocage ou autres.

Ayant un cursus relativement centré sur le développement, ce stage m'a permis de faire face aux différentes étapes de l'élaboration d'un réel projet de développement au sein d'une entreprise, en plus de me familiariser avec le fonctionnement du service administration systèmes et réseaux. Ce stage a été très enrichissant, me permettant de découvrir de nombreux aspects dans la création d'un projet et de mettre en pratique mes connaissances en développement web dans un environnement réel.

J'ai également pu comprendre le fonctionnement d'une entreprise comme Agitel. J'ai pu appréhender l'organisation d'une équipe au sein d'une entreprise et la nécessité de l'utilisation d'outils et de logiciels informatiques. J'ai également consolidé de nombreuses connaissances acquises dans mes études, tout en en développant de nouvelles.

Les poursuites d'études dans ce domaine sont très larges. J'ai pu être en relation avec divers postes liés à l'administration systèmes et réseau. Cela m'a permis de découvrir un large éventail de postes dans ce domaine, chacun ayant son rôle et ses objectifs spécifiques.

## Table des figures

**Figure 1 :** Organigramme du service administration systèmes et réseaux de l'entreprise Agitel. **Page 6**

**Figure 2 :** Page de présentation du site Atera. **Page 10**

**Figure 3 :** Première maquette du cahier des charges du projet Atera. **Page 12**

**Figure 4 :** Deuxième maquette du cahier des charges du projet Atera. **Page 13**

**Figure 5 :** Page de connexion de l'interface du projet Atera. **Page 13**

**Figure 6 :** Page du tableau de bord de l'interface du projet Atera. **Page 14**

**Figure 7 :** Page de tickets par gros clients de l'interface du projet Atera. **Page 15**

**Figure 8 :** Header de l'interface du projet Atera. **Page 16**

**Figure 9 :** Diagramme de blocs représentant toute l'architecture du projet du projet Atera. **Page 17**

**Figure 10 :** Script SQL de la création de la base de données PostgreSQL du projet Atera. **Page 18**

**Figure 11 :** Diagramme de séquence représentant l'interaction entre l'interface et la base de données du projet Atera. **Page 18**

**Figure 12 :** Diagramme de séquence représentant l'interaction entre l'interface et les API d'Atera et Odoo du projet Atera. **Page 19**

**Figure 13 :** Partie ADMIN de la page tickets par gros clients du projet Atera. **Page 20**

**Figure 14 :** Présentation du service CRM du logiciel Odoo du projet Atera. **Page 20**

**Figure 15 :** Exemple de lien HTTP utilisé pour faire une requête à l'API d'Atera. **Page 21**

**Figure 16 :** Exemple du format JSON. **Page 21**

**Figure 17 :** Présentation du logiciel DaraGrip. **Page 23**

**Figure 18 :** Logos des différents logiciels et outils utilisés pour le projet Atera. **Page 23**

- Figure 19** : Exemple de requêtes HTTP avec le système de pagination. **Page 24**
- Figure 20** : Maquette du projet Counters. **Page 27**
- Figure 21** : Page de connexion du projet Counters. **Page 27**
- Figure 22** : Page du tableau de bord du projet Counters. **Page 28**
- Figure 23** : Page de gestion des comptes du projet Counters. **Page 28**
- Figure 24** : Page de gestion des compteurs du projet Counters. **Page 29**
- Figure 25** : Page des alertes du projet Counters. **Page 29**
- Figure 26** : Diagramme de blocs représentant toute l'architecture du projet du projet Counters. **Page 30**
- Figure 27** : Script SQL de la base de données PostgreSQL du projet Counters. **Page 31**
- Figure 28** : Script SQL de la base de données MySQL du projet Counters. **Page 31**
- Figure 29** : Exemple de requête SQL préparée. **Page 33**
- Figure 30** : Logos des différents logiciels et outils utilisés pour le projet Counters. **Page 33**

## Lexiques

[1] - **PME** (Petites ou Moyennes Entreprises) : Entreprises avec moins de 250 personnes et un chiffre d'affaires annuel n'excédant pas 50 millions d'euros ou un total du bilan n'excédant pas 43 millions d'euros. **Page 3**

[2] - **SASU** (Société par Actions Simplifiée Unipersonnelle) : Société à vocation commerciale constituant une alternative intéressante pour l'entrepreneur ne souhaitant pas s'installer en nom propre. **Page 6**

[3] - **CRM** (Customer Relationship Management/Gestion de la Relation Client) : Stratégie de gestion des relations et interactions d'une entreprise avec ses clients ou clients potentiels. **Page 7**

[4] - **RMM** (Remote Monitoring and Management/Surveillance et Gestion à Distance) : Type de logiciel utilisé par les professionnels de l'informatique pour surveiller et gérer à distance l'infrastructure informatique des utilisateurs. **Page 7**

[5] - **Hotline** : Ligne d'assistance ou centre d'appels, mis en place dans le but de se charger du traitement des sollicitations des clients de l'entreprise et des prospects. **Page 7**

[6] - **IDE** (Integrated Development Environment/Environnement de Développement Intégré) : Ensemble d'outils qui permet d'augmenter la productivité des programmeurs qui développent des logiciels. **Page 8**

[7] - **HTTP/HTTPS** (Hypertext Transfer Protocol (Secure)/Protocole de Transfert Hypertexte (Sécurisé)) : Protocole servant à transmettre des documents hypermédias, comme HTML. On rajoute le S pour permettre au visiteur de vérifier l'identité du site web auquel il accède, grâce à un certificat d'authentification émis par une autorité tierce, réputée fiable. **Page 8**

[8] - **Windows Server** : Logiciel permettant de développer une infrastructure d'applications, de réseaux et de services web connectés. **Page 8**

[9] - **Hyperviseur** : Plate-forme de virtualisation permettant à plusieurs systèmes d'exploitation de fonctionner en parallèle dans une seule machine physique. **Page 9**

[10] - **Infrastructure IT** : Ensemble des composants matériels, logiciels et réseau nécessaires au fonctionnement et à la gestion des services et des environnements informatiques d'une entreprise. **Page 9**

[11] - **Cloud** : Forme de service consistant à exploiter la puissance de calcul ou de stockage de serveurs informatiques distants par l'intermédiaire d'un réseau. **Page 10**

[12] - **JSON** (JavaScript Object Notation/Notation des Objets du langage JavaScript) : Format de données textuel dérivé de la notation des objets du langage JavaScript, qui est un langage de programmation de script. **Page 21**

[13] - **Swagger** : Langage de description d'interface permettant de décrire des API exprimées à l'aide de JSON. **Page 22**

[14] - **SQL** (Structured Query Language/Langage de Requêtes Structurées) : Langage informatique normalisé servant à exploiter des bases de données relationnelles. **Page 22**