



Rapport de stage chez Quadient

Maintenance et optimisation du site Web Portal de Parcel Pending : Correction des affichages, résolution de bugs et sécurisation du site

Réalisé par :

Antoni MARZEWSKI

Stagiaire

—

Sous la supervision de :

Nasreddine LATRECHE

Maître de stage

Marin BOUGERET

Tuteur universitaire de stage

BUT Informatique – Deuxième Année

Parcours - Réalisation d'applications : conception, développement, validation

Remerciements

Je tiens à remercier la société **Quadient** pour m'avoir offert l'opportunité d'effectuer mon stage au sein de leur équipe. Ce fut une expérience extrêmement enrichissante, tant sur le plan technique que professionnel.

Je souhaite exprimer ma profonde gratitude à **M. Aurélien ABRIEU** pour les démarches qu'il a entreprises afin de faciliter mon intégration dans l'entreprise. Un immense merci à **M. Nasreddine LATRECHE**, mon tuteur de stage, pour son accompagnement, l'organisation de mes tâches et son engagement à assurer le bon déroulement de mon apprentissage.

Je tiens aussi à remercier **M. Hassane AILALI** et **M. Paul-Antoine RIVA**, développeurs, pour leur précieuse aide dans la résolution de divers problèmes, qu'ils concernent le code ou les logiciels. Leur disponibilité et leurs explications m'ont permis d'acquérir de nombreuses connaissances et de mieux comprendre les bonnes pratiques du développement.

Enfin, un grand merci à **toute l'équipe de Quadient** pour son accueil, son soutien et cette expérience professionnelle.

Résumé

Dans un contexte de numérisation croissante et de sécurisation des services, Quadient propose la solution Parcel Pending*, un système de consignes automatiques pour la gestion des colis. Au cœur de cette solution, le Web Portal* permet aux gestionnaires de suivre livraisons et utilisateurs. Intégré à l'équipe Multi-Family*, mon stage a porté sur la maintenance et l'amélioration de ce portail : préparation de l'environnement, correction de bugs, optimisation fonctionnelle et renforcement de la sécurité (injections SQL, XSS, ACL). Ce rapport présente l'entreprise, les missions réalisées, les choix techniques et les apprentissages issus de cette expérience professionnelle.

Mots clés : sécurité des données, fiabilité, protection, Quadient, Parcel Pending, Web Portal, consignes automatiques, livraisons, utilisateurs, maintenance, correction de bugs, optimisation, injections SQL, failles XSS, failles ACL, environnement de développement, amélioration continue, Multi-Family, solution logicielle, outil numérique, gestion des colis, cyberattaques, accès sécurisé.

In a context of growing digitalization and service security, Quadient offers the Parcel Pending* solution, a system of automated lockers for parcel management. At the heart of this solution, the Web Portal* allows property managers to track deliveries and users. As part of the Multi-Family* team, my internship focused on maintaining and improving this portal: setting up the development environment, fixing bugs, enhancing functionality, and strengthening security (SQL injections, XSS, ACL vulnerabilities). This report presents the company, the tasks carried out, the technical choices made, and the key takeaways from this professional experience.

Keywords: data security, reliability, protection, Quadient, Parcel Pending, Web Portal, automated lockers, deliveries, users, maintenance, bug fixing, optimization, SQL injections, XSS vulnerabilities, ACL vulnerabilities, development environment, continuous improvement, Multi-Family, software solution, digital tool, parcel management, cyberattacks, secure access.

Sommaire

Remerciements	2
Résumé	3
Sommaire	4
Table de figures.....	5
Glossaire	6
1 - Introduction	7
2 - Contexte du stage	8
2.1 - Quadient – l'entreprise d'accueil	8
2.2 - Parcel Pending - la solution Quadient au cœur de mon stage.....	10
3 - Analyse du stage	11
3.1 - Positionnement au sein de l'entreprise.....	11
3.2 - Analyse de l'existant.....	12
3.2.1 - Architecture de Parcel Pending.....	12
3.2.2 - Compléments sur le Web Portal	14
3.2.3 - Objectifs du stage et mes missions	15
4 - Rapport Technique	16
4.1 - Préparation de l'environnement de travail.....	16
4.2 - Travail sur l'aspect visuel et fonctionnel du Web Portal	19
4.2.1 - Erreur d'affichage dans la barre de navigation du Web Portal	19
4.2.2 - Bug : Possible de supprimer une unit avec des utilisateurs actifs	21
4.3 - Protection du site contre les cyberattaques	25
4.3.1 - Prévention des attaques par injection SQL.....	25
4.3.2 - Prévention des attaques par XSS	29
4.3.3 - Prévention des failles liées aux ACL.....	32
5 - Méthodologie et organisation du projet	36
5.1 - Organisation du travail dans l'entreprise	36
5.2 - Flux de travail sur un Ticket.....	38
5.3 - Méthodes adoptées personnellement	39
5.4 - Apprentissages mobilisés	40
6 - Conclusion	41
6.1 - Un projet utile, intégré et maintenu par l'entreprise.....	41
6.2 - Une montée en compétences du BUT Informatique	41
6.3 - Une expérience formatrice, concrète et professionnalisante	42
Signatures	42
Bibliographie	43

Table de figures

Figure 1 : Diagramme circulaire Chiffre d'Affaires par Segments d'Activités Quadient 2024	9
Figure 2 : Diagramme circulaire Chiffre d'Affaires par Géographie Quadient 2024	9
Figure 3 : Graphique illustrant ma position chez Quadient	11
Figure 4 : Diagramme de cas d'utilisation de Parcel Pending	13
Figure 5 : Page exemple du Web Portal	14
Figure 6 : Utilisation de l'alias ppdc	18
Figure 7 : Description ticket de l'erreur d'affichage	19
Figure 8 : Barre de navigation avec bouton manquant, mais accès par manipulation de lien	19
Figure 9 : Ajout d'une condition concernant les Property Managers	20
Figure 10 : Barre de navigation corrigée	20
Figure 11 : Test suppression des units avant modification du code	21
Figure 12 : Vérification utilisateurs actifs dans une unit	22
Figure 13 : Méthode de vérification des utilisateurs actifs dans une unit	22
Figure 14 : Action si utilisateurs actifs trouvés	23
Figure 15 : Test suppression des units après modification	24
Figure 16 : Description de l'injection SQL	25
Figure 17 : Exportation d'une requête HTML d'une page	25
Figure 18 : Requête sans injection SQL	26
Figure 19 : Requête avec injection SQL	26
Figure 20 : Cast du paramètre propertyId	27
Figure 21 : Bind du paramètre propertyId	27
Figure 22 : Requête avec injection SQL après modification du code	28
Figure 23 : Description failles liées aux XSS	29
Figure 24 : Restriction du nom de la propriété cote client	29
Figure 25 : Requête avec balise 	30
Figure 26 : Affichage du prompt après attaque XSS	30
Figure 27 : Affichage de la propriété avec balise sur le site	31
Figure 28 : Traitement des paramètres name et street d'une propriété avant affichage	31
Figure 29 : Affichage de la propriété avec balise après modification du code	31
Figure 30 : Mauvais affichage de la propriété avec logique de création modifiée	32
Figure 31 : Description de la faille ACL	32
Figure 32 : Ajout d'une Property Management Company a un Property Manager	33
Figure 33 : Affichage du id et nom d'une Property Management Company dans HeidiSQL	33
Figure 34 : Vérification de présence des données dans la Property Management Company choisi	34
Figure 35 : Réponse à l'attaque ACL avant modification du code	34
Figure 36 : Vérification des Property Management Company accessibles a l'utilisateur connecté	35
Figure 37 : Affichage de l'erreur après modification du code concernant l'ACL	35
Figure 38 : Exemple de rapport quotidien sur teams	36
Figure 39 : Exemple de backlog* de mon équipe	37
Figure 40 : Exemple d'un pull request dans Jira	38
Figure 41 : Solutions proposées par moi lors du travail sur un ticket	39

Glossaire

Méthode Agile : Approche de gestion de projet favorisant la flexibilité, la collaboration et l'amélioration continue. Elle repose sur des cycles courts appelés sprints et met l'accent sur l'adaptation aux besoins des utilisateurs.

Ticket : Unité de travail dans un outil de gestion de projet (ex : Jira, Trello), représentant une tâche, une amélioration ou un bug à résoudre.

Sprint : Période de travail définie (généralement 1 à 4 semaines) dans un cadre Agile, durant laquelle l'équipe doit développer et livrer des fonctionnalités ou corrections spécifiques.

Parcel Pending : Service de Quadient proposant des consignes de colis automatiques pour faciliter la réception et la gestion des colis dans les immeubles résidentiels, entreprises et campus.

Locker : Consigne de colis automatique utilisée pour stocker temporairement des colis avant leur récupération par le destinataire.

Property : Une propriété utilisant la solution Parcel Pending. Il peut s'agir d'un immeuble résidentiel, d'un campus, d'un bureau d'entreprise, etc.

Unit : Une unité au sein d'une **propriété**, pouvant être assimilée à un appartement lorsque la propriété est un immeuble. Elle sert à regrouper les résidents vivant ensemble, leur permettant ainsi de partager l'accès aux colis et de les récupérer mutuellement.

Property Manager : Personne responsable de la gestion du Locker dans une propriété, notamment en surveillant son bon fonctionnement et en résolvant d'éventuels problèmes.

Property Management Company : Un regroupement de propriétés dans la base de données, utilisé pour donner accès à plusieurs **properties** en même temps à un **Property Manager**.

Web Portal : Site web de Parcel Pending, permettant aux administrateurs et Property Managers de gérer le service des consignes de colis automatiques (suivi des livraisons, gestion des utilisateurs, maintenance, etc.).

Multi-Family : Segment de la solution Parcel Pending destiné aux résidences multifamiliales, comme les immeubles d'appartements.

Daily Meeting : Une réunion d'équipe brève, tenue chaque jour, pour synchroniser les membres, suivre l'avancement des tâches et identifier les obstacles.

Backlog : Une liste priorisée des tâches, fonctionnalités ou éléments à réaliser dans un projet, généralement utilisée en gestion agile.

1 - Introduction

Ce rapport de stage présente le travail que j'ai réalisé au sein de l'entreprise **Quadient Shipping**, à **Cavaillon**, du **27 janvier au 4 avril 2025**, dans le cadre de ma deuxième année de formation en **BUT Informatique** (parcours RACDV) à l'IUT de Montpellier.

Quadient est un groupe technologique international, historiquement centré sur les solutions de traitement du courrier, qui s'est progressivement tourné vers les **services numériques et les consignes colis intelligentes**. L'une de ses solutions phares, **Parcel Pending**, permet aux entreprises et aux résidences d'installer des lockers automatisés afin de faciliter et sécuriser la réception de colis.

Durant ce stage, j'ai été intégré à l'équipe **Multi-Family**, composée de développeurs répartis entre la France et le Vietnam. Mon travail s'est articulé autour de l'application **Web Portal**, un site d'administration essentiel au bon fonctionnement de la solution Parcel Pending. Il permet notamment aux gestionnaires de propriétés de suivre les livraisons, de gérer les utilisateurs et de configurer les paramètres des consignes.

J'ai été chargé de **maintenir, améliorer et sécuriser** cet outil. Cela a impliqué la préparation de mon environnement de travail, la **correction de bugs fonctionnels**, l'**ajout de nouvelles fonctionnalités** et la **mise en place de protections contre les cyberattaques** (injections SQL, failles XSS, failles ACL).

Dans un premier temps, je présenterai brièvement l'entreprise et son organisation, ainsi que la solution Parcel Pending. Je détaillerai ensuite les différentes missions réalisées au cours de ces dix semaines de stage. Pour chaque tâche traitée, j'analyserai le contexte technique, les choix effectués, les résultats obtenus, ainsi que les axes d'amélioration possibles.

2 - Contexte du stage

2.1 - Quadient – l'entreprise d'accueil

2.1.1 - Présentation



Quadient est un groupe technologique français spécialisé dans les solutions de traitement du courrier, l'automatisation des processus métier, la gestion de l'expérience client et les consignes colis automatiques. Fondée en 1924 sous le nom de Neopost, elle a adopté le nom de Quadient en 2019 pour refléter sa diversification et son orientation vers les solutions numériques. L'entreprise est présente dans 29 pays et commercialise ses services dans 90 pays, avec un chiffre d'affaires annuel de 1,14 milliard d'euros en 2019. Elle emploie environ 5 700 personnes [1].

2.1.2 - Organisation

L'entreprise est structurée autour de quatre domaines d'activité principaux [2] :

1. **Solutions liées au courrier (Mail)** : offre des équipements et services pour le traitement et l'affranchissement du courrier.
2. **Expérience client & Automatisation (Digital)** : fournit des solutions logicielles pour améliorer les interactions entre les entreprises et leurs clients, ainsi que des outils d'automatisation des processus financiers et documentaires.
3. **Consignes colis automatiques (Lockers*)** : développe des solutions de consignes intelligentes pour la gestion sécurisée des colis.

2.1.3 - Dimension commerciale

Quadient génère son chiffre d'affaires principalement à travers la vente de services et de solutions logicielles en mode SaaS, la vente d'équipements et de licences, ainsi que des revenus de location. En 2023/2024, 69 % du chiffre d'affaires provenaient des services Mail, 23 % du Digital et 8 % des Lockers [3].

Chiffre d'Affaires 2024 par Segments d'Activités

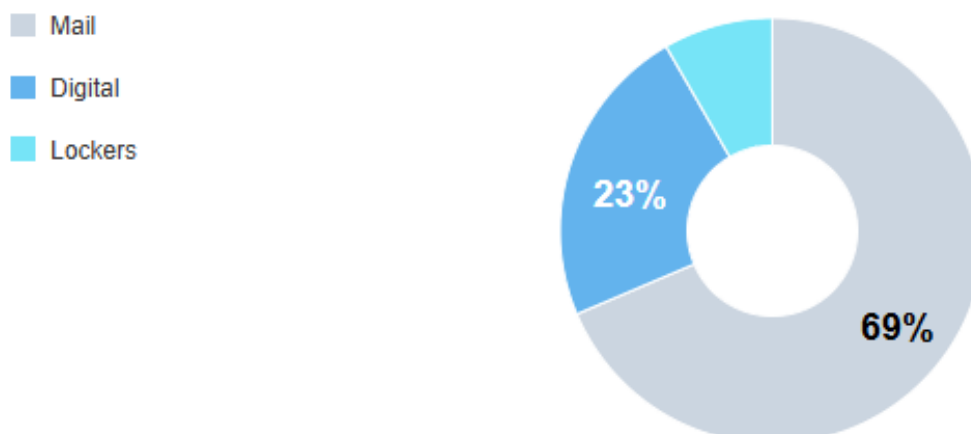


Figure 1 : Diagramme circulaire Chiffre d'Affaires par Segments d'Activités Quadient 2024

Quadient opère principalement aux États-Unis, qui représentent 57 % de son chiffre d'affaires. La région France-Benelux contribue à hauteur de 17 %, tandis que l'Allemagne, l'Autriche, la Suisse, l'Italie, le Royaume-Uni et l'Irlande génèrent ensemble 16 % des revenus (cf. Figure 1) [3].

Chiffre d'Affaires 2024 par Géographie

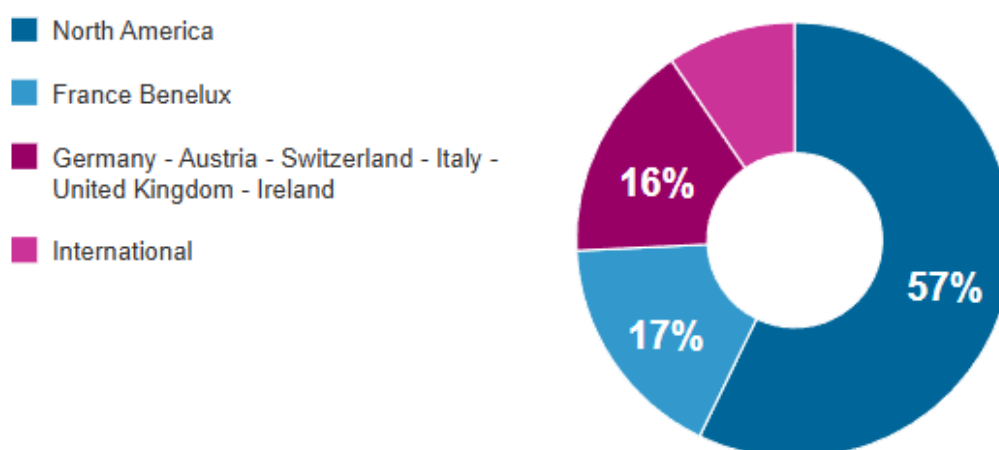


Figure 2 : Diagramme circulaire Chiffre d'Affaires par Géographie Quadient 2024

Parmi ces revenus, 62,9 % étaient récurrents (prestations de maintenance et services professionnels), 26,2 % provenaient de la vente d'équipements et de licences, et 10,9 % de revenus de location (cf. Figure 2) [3].

2.2 - Parcel Pending - la solution Quadient au cœur de mon stage

2.2.1 - Présentation

Parcel Pending* by Quadient est une solution de consignes colis intelligentes conçue pour simplifier et sécuriser la livraison et le retrait des colis. Elles permettent aux utilisateurs de récupérer leurs colis à leur convenance, 24h/24 et 7j/7, réduisant ainsi les risques de perte ou de vol. En 2022, Quadient comptait plus de 15 100 consignes colis installées dans le monde [4].



2.2.2 - Organisation

L'entreprise propose une gamme de solutions de consignes intelligentes adaptées aux besoins spécifiques de chaque marché. L'organisation de Parcel Pending se divise en **2 services** pour servir divers secteurs [5] :

Carrier - Retail solution :

- L'enseignement supérieur (University of Florida, Miami University)
- Magasins de détail (Decathlon, 7-Eleven)
- Transporteurs (DHL, UPS)
- Bureaux d'entreprise (Microsoft, Mazda, Netflix)

Multi - Family solution :

- Les résidences multifamiliales
- Appartements

2.2.3 - Clientèle cible

Les utilisateurs finaux de Parcel Pending sont principalement des résidents d'immeubles, des employés de bureaux, des clients de commerces de détail et des étudiants, qui bénéficient d'une solution pratique pour la réception de leurs colis.

3 - Analyse du stage

3.1 - Positionnement au sein de l'entreprise

Quadient propose une large gamme de services. Avant d'analyser l'existant, il me semble essentiel de préciser mon rôle au sein de l'entreprise durant ce stage. J'ai travaillé sur l'une des solutions proposées, **Parcel Pending**, au sein de l'équipe **Multi-Family**, composée de six personnes. Mon rôle était celui de **développeur web**, où j'ai contribué au développement et à l'amélioration et la maintenance du **Web Portal***. Tout cela sera décrit et expliqué par la suite.

Voici un simple graphique illustrant ma position (cf. Figure 3) :

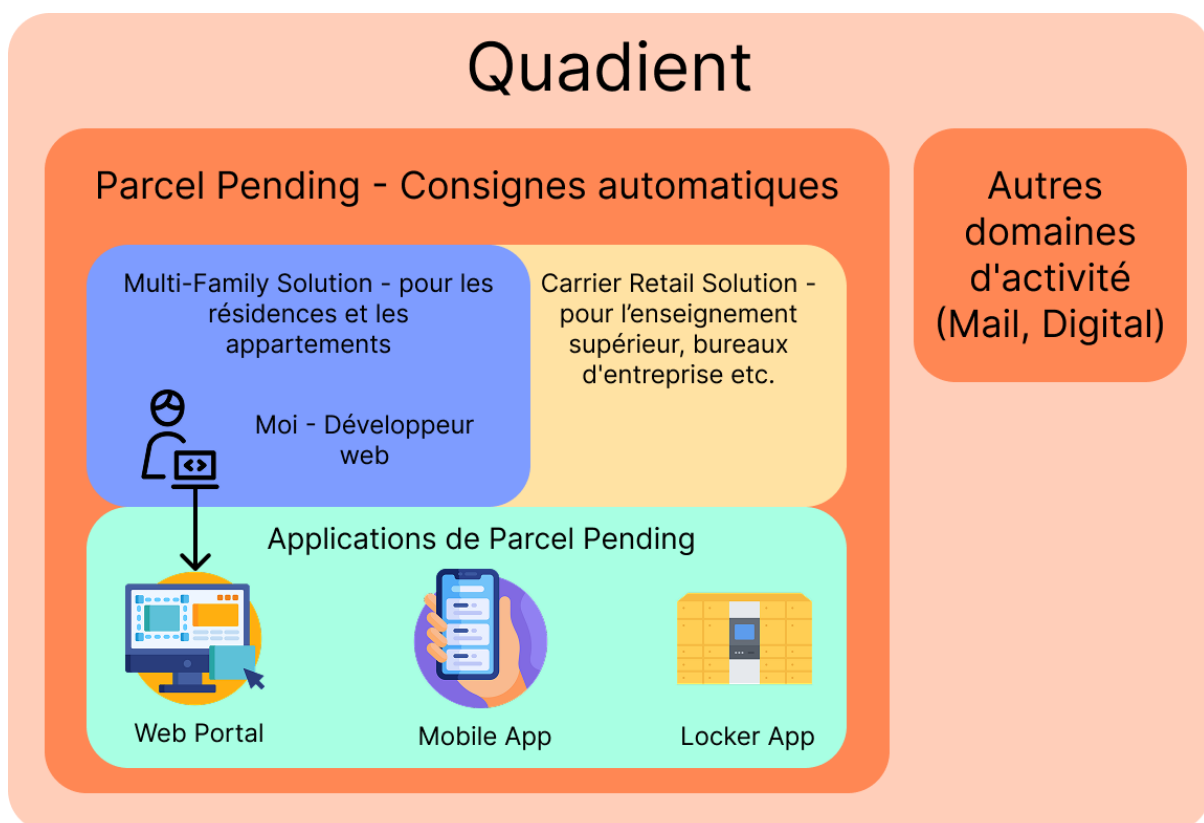


Figure 3 : Graphique illustrant ma position chez Quadient

3.2 - Analyse de l'existant

3.2.1 - Architecture de Parcel Pending

Parcel Pending repose sur une architecture logicielle composée de plusieurs applications, chacune ayant un rôle spécifique dans la gestion et l'utilisation des consignes automatiques :

Web Portal : Il s'agit du site web permettant aux administrateurs et aux gestionnaires de suivre l'activité des consignes, gérer les utilisateurs, configurer les paramètres des livraisons et analyser les données d'utilisation. Il offre une interface complète pour superviser le fonctionnement des consignes en temps réel. Il donne également accès aux résidents pour gérer leurs comptes et leurs activités.



Locker App : Ce logiciel est installé directement sur les consignes automatiques. Il est utilisé par les livreurs pour déposer les colis et par les utilisateurs finaux pour les récupérer via un code ou un QR code ou via Bluetooth en utilisant l'appli mobile. Il assure la communication entre les consignes et les autres systèmes, garantissant ainsi la traçabilité et la sécurité des colis.



Mobile App : Destinée aux clients finaux, cette application permet aux utilisateurs de recevoir des notifications sur la disponibilité de leurs colis, de générer des QR codes pour le retrait et d'accéder à l'historique de leurs livraisons. Elle améliore l'expérience utilisateur en simplifiant l'accès aux consignes.



Base de données et Cloud S3 : Hébergeant toutes les parties API et de stockage, le Cloud S3 permet de lier efficacement les trois applications (Web Portal, Locker App et Mobile App). Il assure une gestion centralisée des données, garantissant la sécurité, la traçabilité et la disponibilité des informations en temps réel pour toutes les parties prenantes.



Ce diagramme de cas d'utilisation (cf. Figure 4) précise de manière compréhensible et concise les principales fonctions des trois applications proposées par **Parcel Pending**.

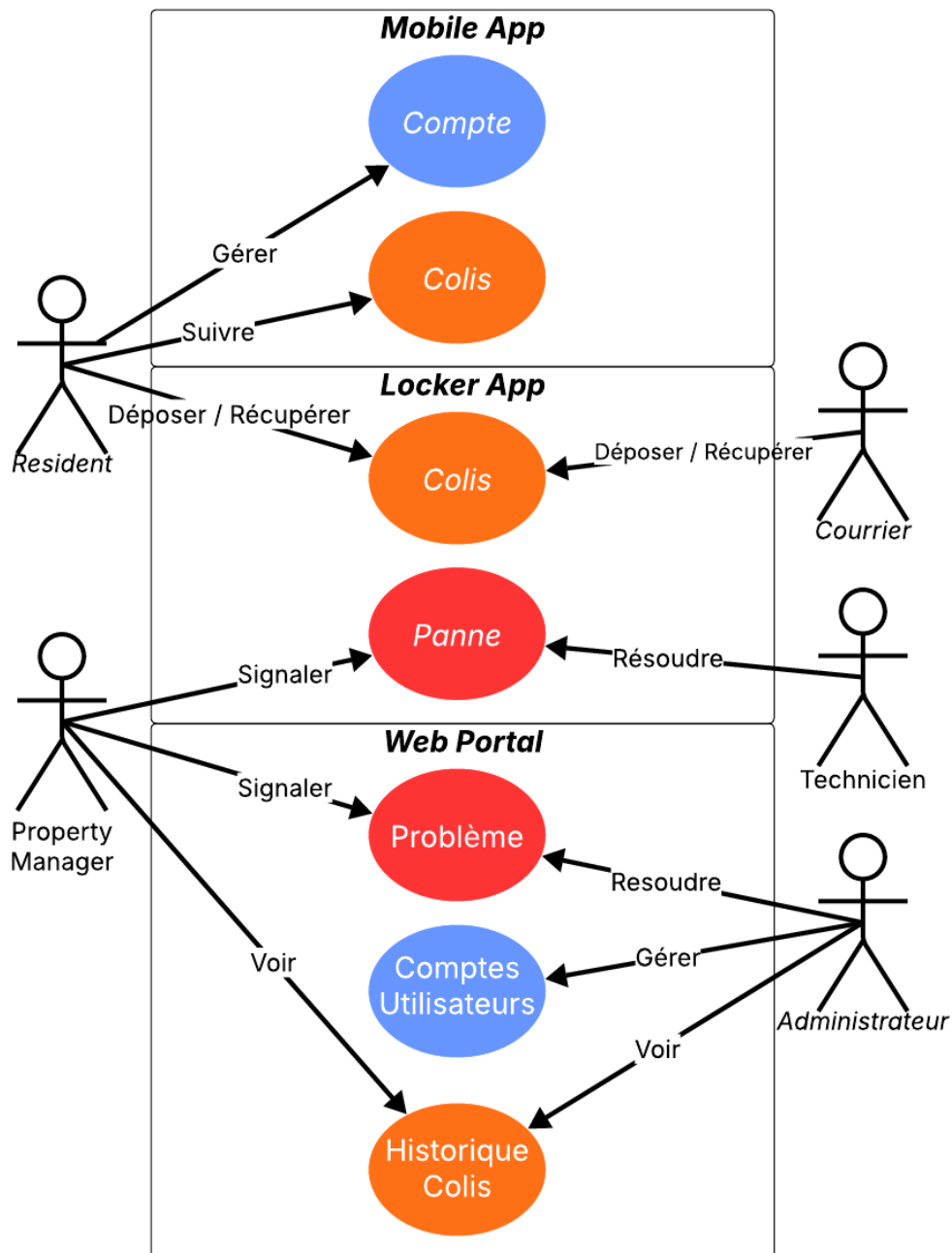


Figure 4 : Diagramme de cas d'utilisation de Parcel Pending

3.2.2 - Compléments sur le Web Portal

Le site est principalement développé en PHP et repose sur une base de données MySQL. J'ai également remarqué l'application des principes SOLID dans le code. Bien que le site soit beaucoup plus vaste et complexe que ceux sur lesquels j'ai pu travailler à l'IUT, il était relativement facile de s'y repérer et d'y trouver des similitudes.

Le Web Portal est principalement utilisé par les administrateurs et les gestionnaires de propriétés. Il offre un contrôle quasi total sur la gestion des consignes des colis automatiques ainsi que sur les comptes utilisateurs.

Voici une capture d'écran de l'une des fonctionnalités du Web Portal (cf. Figure 5) :

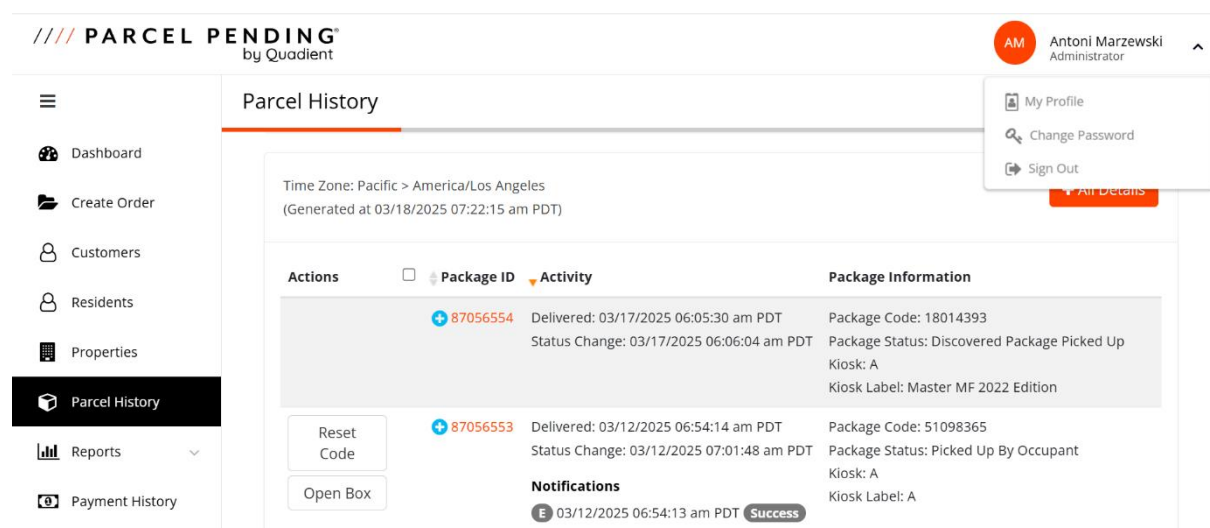


Figure 5 : Page exemple du Web Portal

À gauche, on voit le menu proposant de nombreuses autres fonctionnalités, telles que la gestion des résidents, des propriétés, l'historique des paiements, etc. En haut à gauche, l'utilisateur actuellement connecté est affiché. La partie centrale de l'écran est dédiée à l'affichage du contenu de la fonctionnalité en cours d'utilisation, ici l'historique des colis.

Le Web Portal étant vaste et riche en fonctionnalités, je ne les décrirai pas toutes. Celles sur lesquelles j'ai travaillé seront détaillées dans le rapport technique.

Étant donné que Parcel Pending n'est pas open source, je ne peux pas partager davantage d'informations sur sa structure et son architecture, car ces éléments sont strictement confidentiels.

3.2.3 - Objectifs du stage et mes missions

L'objectif du stage était d'assurer le bon fonctionnement du Web Portal. Il n'y avait pas de cahier des charges explicite défini au début du stage, car Quadient adopte la méthode **Agile*** comme méthode de gestion de projet. Les **tickets*** étaient créés et attribués à moi de manière systématique. Toutefois, mon tuteur a défini trois étapes distinctes pour regrouper mes tâches de manière chronologique.

1. Préparer mon environnement de travail :

- Installer tous les outils nécessaires
- S'assurer que je possède les accès à tous les services nécessaires

2. Travailler sur l'aspect visuel et fonctionnel du site, incluant :

- Modification de l'affichage et gestion des droits d'accès pour certains utilisateurs
- Résolution des bugs
- Ajout de nouvelles fonctionnalités

3. Sécuriser le site contre les cyberattaques, notamment :

- Protection contre les injections SQL
- Prévention des attaques XSS (Cross-site Scripting)
- Gestion des ACL (Contrôle d'Accès basé sur les Rôles)

Cette répartition a permis de s'assurer que je travaille sur plusieurs aspects du site, allant de l'installation et la configuration de mon environnement à l'optimisation des performances et à la sécurisation du Web Portal.

4 - Rapport Technique

Dans cette partie, j'expliquerai le travail que j'ai effectué durant mon stage. Je décrirai uniquement les tâches les plus intéressantes, en présentant au moins deux par section définie dans la partie "**Objectifs du stage et mes missions**".

4.1 - Préparation de l'environnement de travail

Ma première mission a été de préparer mon environnement de développement sur un PC portable fourni par Quadient. Cette étape a impliqué la configuration de plusieurs outils essentiels à mon travail.

J'ai installé et configuré :

Git

Git est un système de gestion de versions permettant de suivre les modifications du code, de travailler en collaboration et de gérer différentes branches d'un projet



Le compte GitLab et l'accès au projet du Web Portal m'ont été fournis par l'entreprise dès mon arrivée. J'étais déjà familier avec **Git** grâce aux projets réalisés à l'IUT.

WSL (Windows Subsystem for Linux)

WSL permet d'exécuter un environnement Linux (Ubuntu dans mon cas) directement sur Windows sans avoir besoin d'une machine virtuelle ou d'un dual boot.



Après avoir installé WSL, j'ai pu ouvrir un terminal Ubuntu et générer une clé SSH avec **ssh-keygen** afin de lier ma machine et mon IDE au dépôt GitLab du Web Portal.

PHPStorm

PHPStorm est un environnement de développement intégré (IDE) spécialisé pour PHP. Il offre de nombreuses fonctionnalités avancées comme l'autocomplétion du code, le débogage intégré et l'intégration avec des outils de gestion de bases de données.



J'ai pu utiliser un autre IDE mais j'ai choisi PHPStorm car c'est ce que j'utilise à l'IUT.

Zscaler

Zscaler est une solution de cybersécurité qui sécurise l'accès aux ressources en ligne et protège contre les menaces potentielles. Son utilisation garantit une connexion sécurisée aux services et outils internes de l'entreprise.



Zscaler peut être considéré comme une alternative aux VPN traditionnels, mais avec une approche plus moderne et sécurisée basée sur le **Zero Trust Network Access (ZTNA)**.

HeidiSQL

HeidiSQL est un client de gestion de bases de données qui permet d'interagir facilement avec MySQL. Il facilite l'exploration, la modification et l'exécution de requêtes SQL.



La version locale de la base de données était hébergée sur ma machine, ce qui nécessitait de télécharger une copie de celle-ci. Pour accéder aux autres bases de données, j'utilisais les identifiants fournis par l'entreprise.

Postman

Postman est un outil de développement utilisé pour tester, développer et documenter des API. Il permet d'envoyer des requêtes HTTP, d'analyser les réponses et de simuler des interactions avec des services web. Postman offre une interface conviviale pour automatiser les tests d'API, vérifier leur bon fonctionnement etc.



J'ai utilisé Postman principalement pour envoyer des requêtes HTTP personnalisées au Web Portal afin de simuler des cyberattaques.

Jira

Jira est un outil de gestion de projet et de suivi des tickets développé par Atlassian. Il est principalement utilisé pour le développement logiciel, permettant aux équipes d'organiser leurs tâches, suivre les bugs et gérer les sprints en méthodologie Agile.



J'ai utilisé Jira pour gérer mes tickets, plus de détails sur cela dans la partie Méthodologie de travail.

Docker

Docker est une plateforme de virtualisation légère permettant de créer, déployer et gérer des conteneurs. Il assure un environnement de développement homogène, évitant les problèmes liés aux différences de configuration entre les machines des développeurs et les serveurs de production.



J'utilisais Docker pour gérer les conteneurs qui hébergeaient en local les différents services dont j'avais besoin. Pour faciliter cette tâche, un alias "**ppdc**" a été créé.

```
ppdc='cd ~/www/quadient && docker compose -p parcel_pending -f  
phpconfig/hassane/docker/docker-compose.yml
```

Cet alias permettait de me rendre rapidement dans le répertoire `~/www/quadient` et de lancer la configuration Docker spécifiée dans le fichier `docker-compose.yml` avec le projet `parcel_pending`.

Voici un exemple de commande utilisant cet alias, cela permet de relancer tous les conteneurs (cf. Figure 6) :

```
antoni@FR04-51765X3:~/www/quadient$ ppdc restart  
[+] Restarting 5/5  
✓Container wp Started  
✓Container api Started  
✓Container parcel_pending_mailcatcher_1 Started  
✓Container reverse-proxy Started  
✓Container db Started  
antoni@FR04-51765X3:~/www/quadient$
```

Figure 6 : Utilisation de l'alias `ppdc`

4.2 - Travail sur l'aspect visuel et fonctionnel du Web Portal

4.2.1 - Erreur d'affichage dans la barre de navigation du Web Portal

Conception

Actual result:

the ""customer-parcel-history"" is not displayed, we can see it as admin but not as PM

Expected result:

The PM can see the report ""customer-parcel-history"" in the report list

Figure 7 : Description ticket de l'erreur d'affichage

D'après cette description du ticket (cf. Figure 7), le problème était qu'un utilisateur **Property Manager*** ne voyait pas la section **"Customer Parcel History"** dans la barre de navigation, cependant j'ai observé il pouvait y accéder en manipulant le lien (cf. Figure 8) :

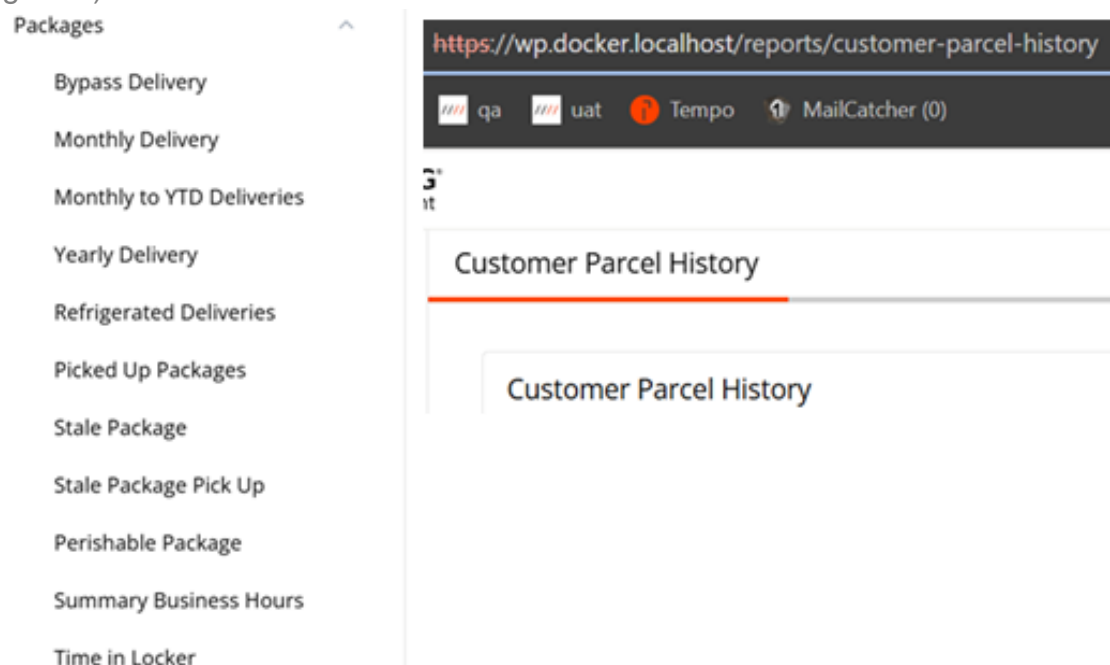


Figure 8 : Barre de navigation avec bouton manquant, mais accès par manipulation de lien

D'après ces observations, j'ai constaté que les comptes **Property Manager** avaient bien accès à cette page et qu'il s'agissait uniquement d'un problème d'affichage.

Réalisation

J'ai retrouvé le fichier navigation.phtml dans le projet, il définit les éléments affichés dans la barre de navigation en fonction de l'utilisateur connecté. J'ai identifié la section responsable de l'affichage de **Customer Parcel History** et constaté qu'il manquait une condition vérifiant si l'utilisateur connecté était un [Property Manager](#). J'ai donc ajouté cette condition, comme le montre la capture d'écran suivante (cf. Figure 9) :

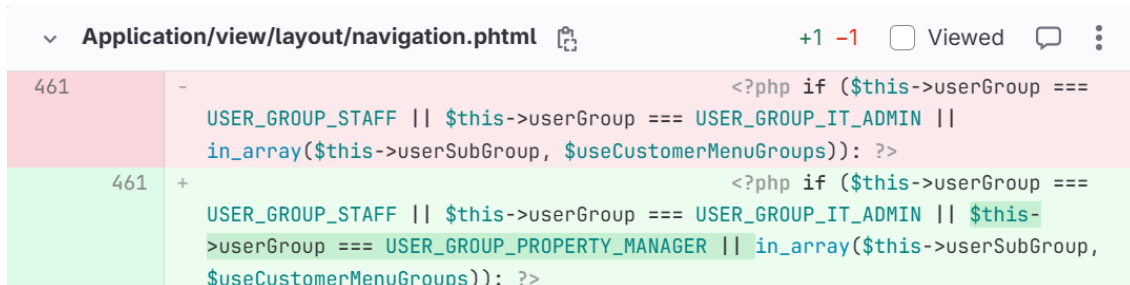


Figure 9 : Ajout d'une condition concernant les Property Managers

Validation

Après cette modification, **Customer Parcel History** s'affichait correctement dans la barre de navigation (cf. Figure 10) :

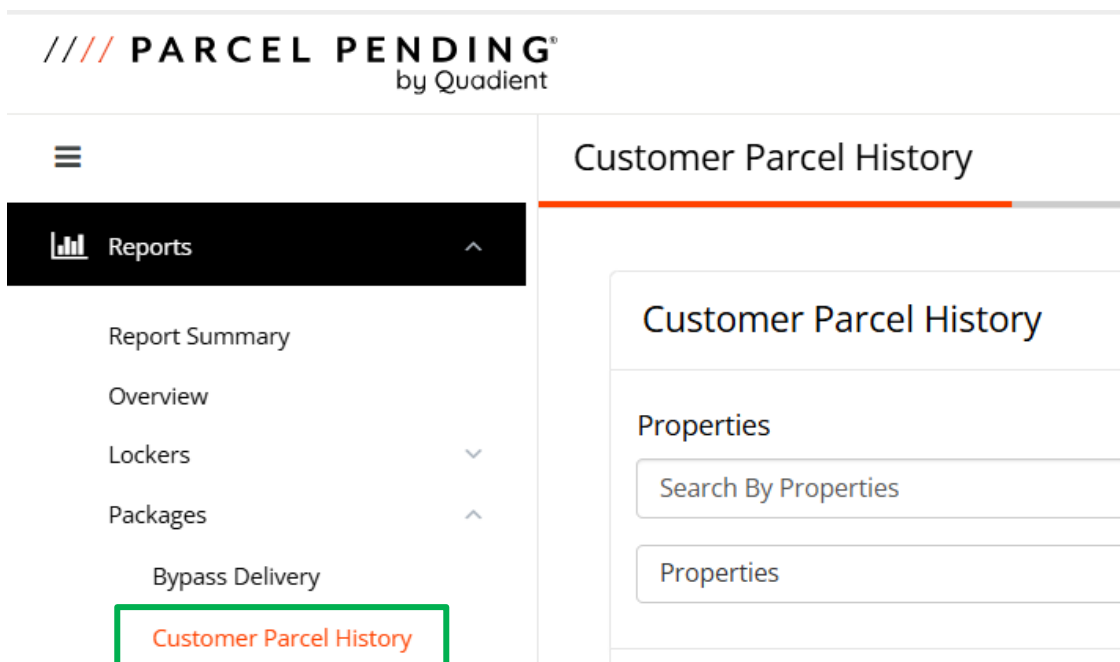


Figure 10 : Barre de navigation corrigée

Apprentissages critiques

- AC21.01 - Élaborer et implémenter les spécifications fonctionnelles et non fonctionnelles à partir des exigences
- AC21.04 - Vérifier et valider la qualité de l'application par les tests

4.2.2 - Bug : Possible de supprimer une unit avec des utilisateurs actifs

Conception

Après une conversation avec le rapporteur du ticket, j'ai compris que la possibilité de supprimer une **unit*** contenant des utilisateurs actifs était un bug. Je devais donc empêcher cette action.

Tout d'abord, j'ai identifié **quatre units** qui m'ont permis de reproduire le bug et de tester ma solution :

- Une contenant des utilisateurs **actifs et inactifs**
- Une avec uniquement des utilisateurs **actifs**
- Une avec uniquement des utilisateurs **inactifs**
- Une sans aucun utilisateur

Ainsi, j'ai pu couvrir tous les scénarios possibles. Je les ai renommées afin de pouvoir les identifier facilement (cf. Figure 11).

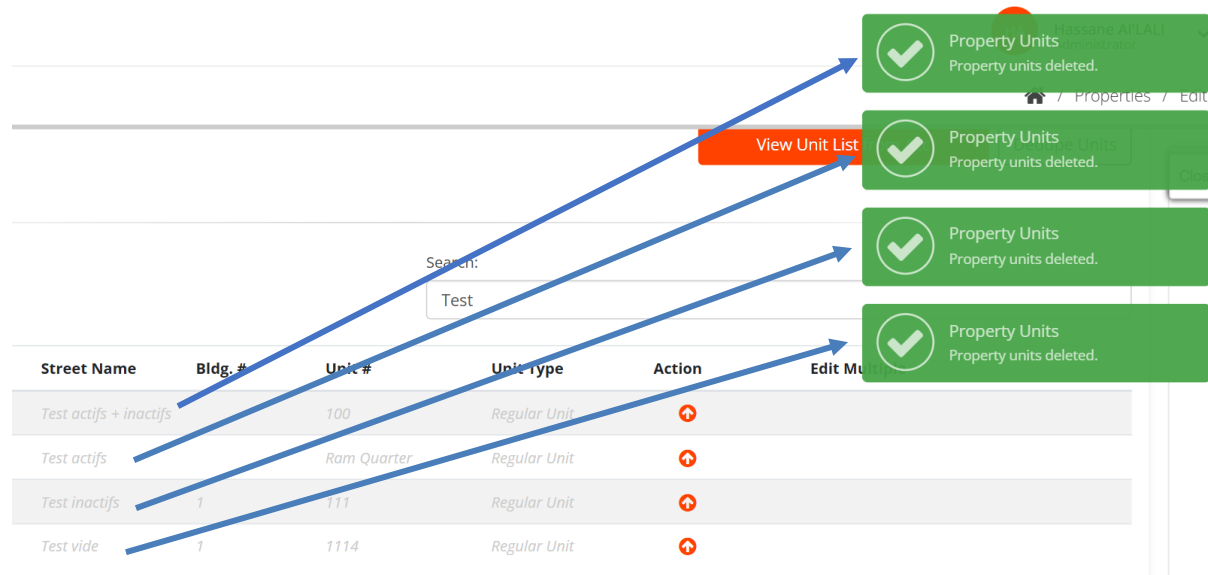


Figure 11 : Test suppression des units avant modification du code

En effet, il était possible de les supprimer toutes, confirmant ainsi la présence du bug.

Réalisation

J'ai trouvé la méthode responsable de la suppression des **units**, **deleteUnitAction()**, dans le fichier **PropertiesController.php**. J'y ai ajouté du code permettant de vérifier s'il existe des utilisateurs actifs dans l'**unit** choisie pour la suppression (cf. Figure 12).

```
if($unit) {
    // Check for active residents using QueryBuilder
    $now = new \DateTime();
    $now = $now->format('Y-m-d');

    $qb = $this->entityManager->getRepository(ResidentAddress::class)->createQueryBuilder('ra');
    $qb->where('ra.propertyUnits = :unitId')
        ->andWhere('ra.moveInDate <= :now')
        ->andWhere('ra.moveOutDate >= :now OR ra.moveOutDate is NULL')
        ->setParameter('unitId', $id)
        ->setParameter('now', $now);

    $activeResidents = $qb->getQuery()->getResult();

    if (count($activeResidents) > 0) {
        echo json_encode([
            SUCCESS => false,
            'message' => $this->translator()->translate('Unable to delete a unit with active users.')
        ]);
        exit();
    }
}
```

Figure 12 : Vérification utilisateurs actifs dans une unit

Cependant, après avoir poussé ce code, la personne chargée de valider de mon ticket m'a demandé de déplacer cette logique dans une fonction du fichier **ResidentAddressRepository.php**, car selon le modèle MVC, les requêtes SQL ne devraient pas se trouver dans les contrôleurs. C'était une erreur de ma part. J'ai donc effectué ce changement (cf. Figure 13) :

```
Application/src/Repository/ResidentAddressRepository.php +25 -0 Viewed

74 + public function hasActiveResidents($unitId)
75 + {
76 +     $now = new \DateTime();
77 +     $now = $now->format('Y-m-d');
78 +
79 +     $qb = $this->createQueryBuilder('ra');
80 +     $qb->select('COUNT(ra.id)')
81 +         ->join('ra.users', 'u')
82 +         ->where('ra.propertyUnits = :unitId')
83 +         ->andWhere('ra.moveInDate <= :now')
84 +         ->andWhere('ra.moveOutDate >= :now OR ra.moveOutDate is NULL')
85 +         ->andWhere('u.status = :activeStatus')
86 +         ->setParameter('unitId', $unitId)
87 +         ->setParameter('now', $now)
88 +         ->setParameter('activeStatus', 1); // Only consider users with status = 1 (active)
89 +
90 +     return (int)$qb->getQuery()->getSingleScalarResult() > 0;
```

Figure 13 : Méthode de vérification des utilisateurs actifs dans une unit

Fonctionnement de la méthode (cf. Figure 14)

- Elle prend un paramètre \$unitId qui représente l'identifiant de l'unité à vérifier
- Elle crée une date actuelle au format 'Y-m-d'
- Elle construit une requête SQL qui :
 - Compte le nombre de résidents
 - Joint la table des utilisateurs
 - Filtre par l'unité spécifiée
 - Vérifie que la date d'emménagement est antérieure ou égale à aujourd'hui
 - Vérifie que la date de déménagement est postérieure à aujourd'hui OU qu'il n'y a pas de date de déménagement
 - Vérifie que le statut de l'utilisateur est actif (1)
- Elle retourne true si au moins un résident actif est trouvé, sinon false

```
// Use the hasActiveResidents method to check for active residents
$hasActiveResidents = $this->entityManager-
>getRepository(ResidentAddress::class)->hasActiveResidents($id);

if ($hasActiveResidents) {
    return new JsonModel([
        'status' => false,
        'message' => $this->translator()->translate('Unable to delete a unit
with active users.')
```

Figure 14 : Action si utilisateurs actifs trouvés

Si des résidents actifs sont trouvés, l'application renvoie un message d'erreur indiquant qu'il est impossible de supprimer une unité avec des utilisateurs actifs.

Validation

J'ai refait la même démarche que lors de la reproduction du bug, c'est-à-dire que j'ai essayé de supprimer toutes les **units** de test :

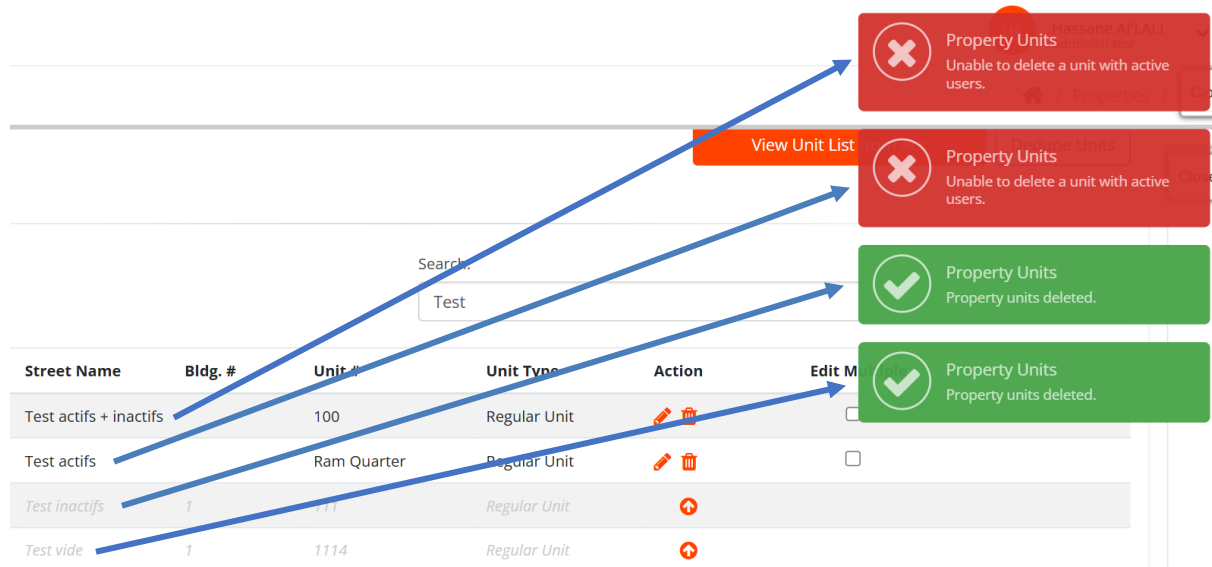


Figure 15 : Test suppression des units après modification

On constate qu'il est impossible de supprimer les **units** qui possèdent des utilisateurs actifs, mais qu'il est possible de supprimer celles qui n'en ont pas. La fonctionnalité fonctionne donc comme prévu.

Apprentissages critiques

- **AC21.01** - Élaborer et implémenter les spécifications fonctionnelles et non fonctionnelles à partir des exigences
- **AC21.03** - Adopter de bonnes pratiques de conception et de programmation (application du design pattern MVC dans ce cas)
- **AC21.04** - Vérifier et valider la qualité de l'application par les tests
- **AC23.01** - Concevoir et développer des applications communicantes (communication avec la base de données)

4.3 - Protection du site contre les cyberattaques

4.3.1 - Prévention des attaques par injection SQL

L'injection SQL est une attaque où un pirate insère du code SQL malveillant dans une requête pour manipuler une base de données. Elle peut permettre de voler, modifier ou supprimer des données sensibles.

Conception

D'après un rapport provenant d'une entreprise spécialisée dans l'identification des failles de sécurité des sites, la page `reports/monthly-package-data` est vulnérable à une injection SQL sur le paramètre `propertyId` (cf. Figure 16).

SQL injection at `"/reports/monthly-package-data"`:

The endpoint `"/reports/monthly-package-data"` is also vulnerable with the parameter `"propertyId"`.

Figure 16 : Description de l'injection SQL

J'ai utilisé Postman pour envoyer une requête avec du code SQL dans le paramètre `propertyId` afin de simuler une cyberattaque et confirmer l'existence d'une vulnérabilité.

Pour importer une requête dans Postman il suffit de (cf. Figure 17) :

- **Se rendre** sur la page concernée.
- **Ouvrir** les outils de développement de votre navigateur en appuyant sur **F12**.
- **Accéder** à l'onglet **Réseau** (Network) des outils de développement.
- **Cliquer** sur le bouton qui envoie la requête.
- Une fois la requête envoyée, elle **apparaître** dans l'onglet **Réseau**.
- **Faire** un clic droit sur la requête correspondante et **sélectionner** l'option pour la copier en format **cURL (bash)**.
- Cliquer sur le bouton **Import** dans Postman, coller la requête et confirmer

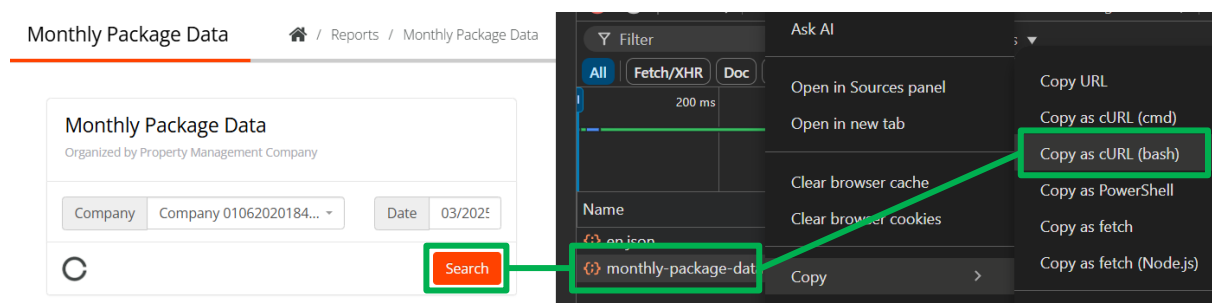


Figure 17 : Exportation d'une requête HTML d'une page

On envoie la requête de base et on vérifie qu'elle fonctionne (cf. Figure 18) :

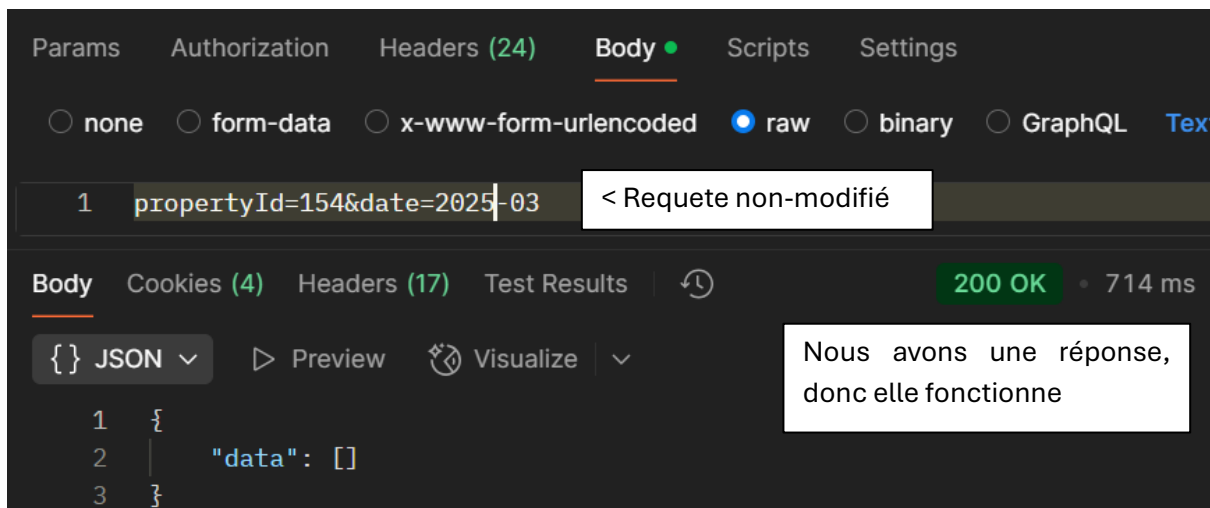


Figure 18 : Requête sans injection SQL

Dans ce cas, il n'y a pas de données dans la réponse, mais cela est simplement dû au fait qu'aucune donnée n'est disponible pour cette propriété et cette période.

Après avoir envoyé la requête de base et confirmé qu'elle fonctionne, on peut ajouter du code SQL à l'un des paramètres concernés par la vulnérabilité (cf. 19) :

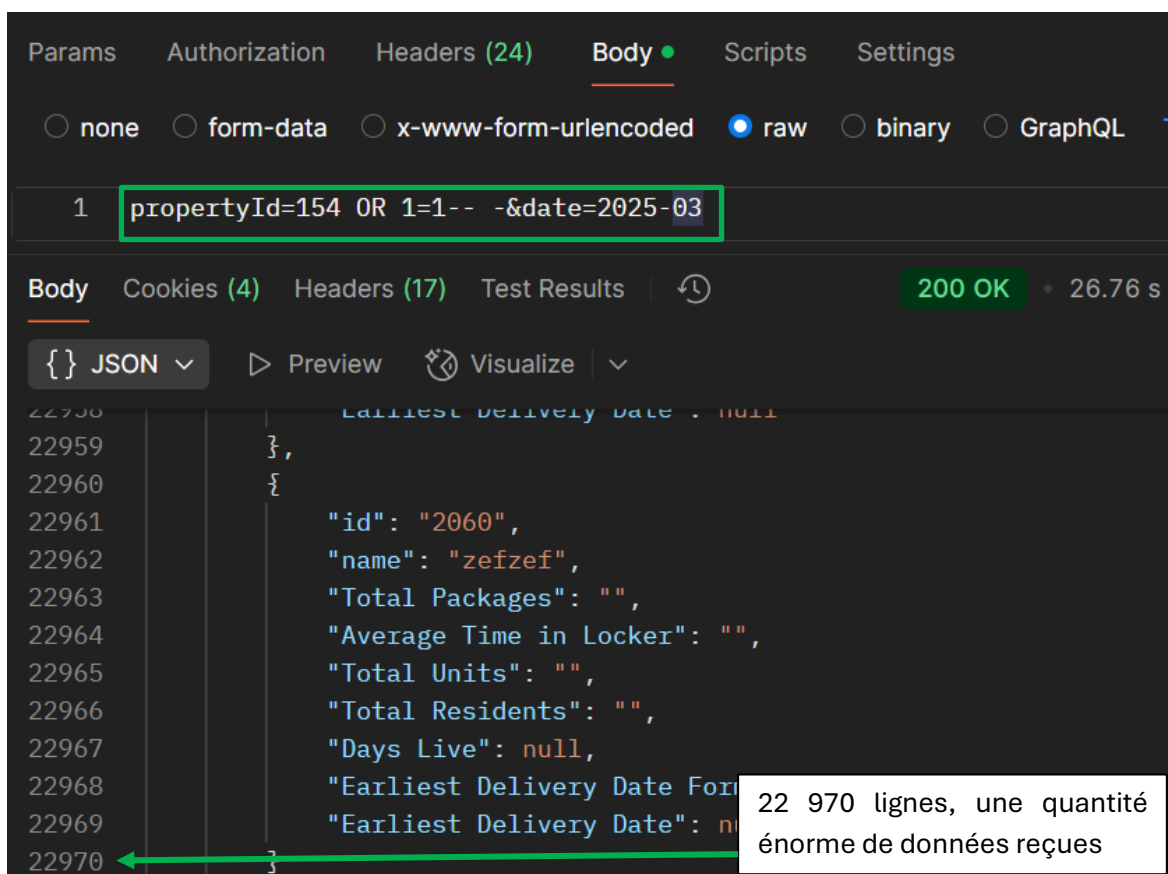


Figure 19 : Requête avec injection SQL

L'injection SQL `propertyId=154 OR 1=1-- --&date=2025-03` exploite une faille en insérant `OR 1=1`, une condition toujours vraie, pour contourner les restrictions et afficher toutes les données. Le `-- --` commente le reste de la requête, désactivant toute condition supplémentaire. Par conséquent, l'attaquant peut accéder à **toutes les données de la table**, ce qui explique la réponse contenant 22 970 lignes.

Réalisation

Pour empêcher cette injection SQL, il suffit de caster le paramètre `propertyId` en entier (cf. Figure 20). Cela garantit que seule une valeur numérique valide est prise en compte, ignorant toute autre partie de la chaîne de caractères ou retournant `0` en l'absence de nombre.

```
if($request->isPost()){
    $id = $this->params()->fromPost('propertyId');
    $id = (int) $this->params()->fromPost('propertyId');
    $date = $this->params()->fromPost('date');
```

Figure 20 : Cast du paramètre `propertyId`

Pour renforcer la sécurité, j'ai également **bindé** ce paramètre, c'est-à-dire que j'ai utilisé une **requête préparée** avec un paramètre lié (binding) (cf. Figure 21). Cela empêche toute tentative d'injection SQL, car la valeur du paramètre est traitée comme une donnée et non comme une partie du code SQL. Ainsi, même si un attaquant tente d'injecter du code malveillant, celui-ci sera simplement interprété comme une valeur sans affecter la requête.

```
WHERE property.id_property_mgmt = ".$id." AND (pr
property_units.unit_number is null)
WHERE property.id_property_mgmt = :propertyId AND
property_units.unit_number is null)
$stmt = $conn->prepare($sql);
$stmt->bindValue('propertyId', $id, \PDO::PARAM_INT);
$stmt->bindValue('timeformat', TIME_FORMAT_MYSQL);
```

Figure 21 : Bind du paramètre `propertyId`

Validation

On constate qu'après cette modification, l'injection SQL ne fonctionne plus (cf. Figure 22).

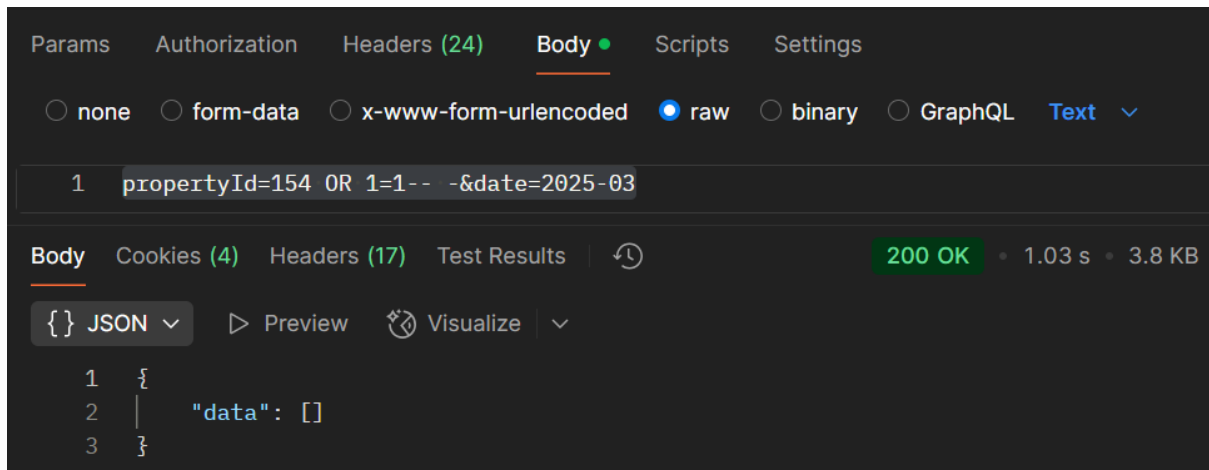


Figure 22 : Requête avec injection SQL après modification du code

Nous avons donc réussi à protéger cette page contre les attaques par injection SQL sur le paramètre `propertyId`.

Apprentissages critiques

- AC23.03 | Sécuriser les services et données d'un système (contre une injection SQL dans ce cas)
- AC21.04 - Vérifier et valider la qualité de l'application par les tests

4.3.2 - Prévention des attaques par XSS

Le **Cross-Site Scripting (XSS)** consiste à injecter du code JavaScript/HTML/CSS malveillant dans une page web pour voler des données ou manipuler l'affichage.

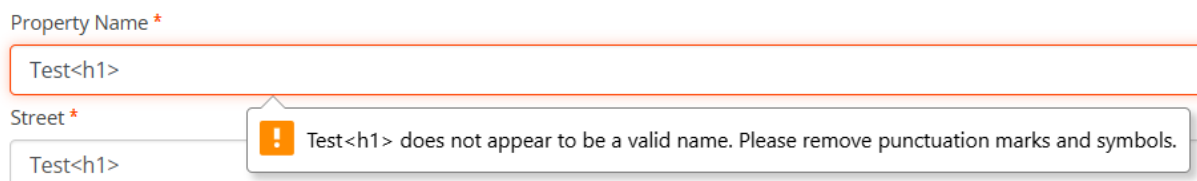
Conception

D'après le même rapport sur les failles de sécurité (cf. Figure 23), si les valeurs des propriétés **name** et **street** contiennent du code HTML, celui-ci sera interprété dans la page **/manager/add** :

Endpoint	Parameter(s)
/properties	name, street
/manager/add	Reflection

Figure 23 : Description failles liées aux XSS

Pour reproduire cette attaque, j'ai créé une propriété contenant une balise HTML dans les paramètres **name** et **street**. Cependant, je n'ai pas pu le faire directement via le navigateur, car les champs du formulaire de création de propriété n'acceptent pas les caractères spéciaux (cf. Figure 24) :



Property Name *

Test<h1>

Street *

Test<h1>

! Test<h1> does not appear to be a valid name. Please remove punctuation marks and symbols.

Figure 24 : Restriction du nom de la propriété côté client

Mais cette restriction n'est qu'une vérification côté client, ce qui signifie qu'elle peut être contournée à l'aide d'un outil comme Postman, comme l'aurait fait un attaquant. J'ai donc copié la requête de création de propriété, l'ai importée dans Postman et ai réussi à créer une propriété contenant du code HTML dans les champs **name** et **street** (cf. Figure 25).

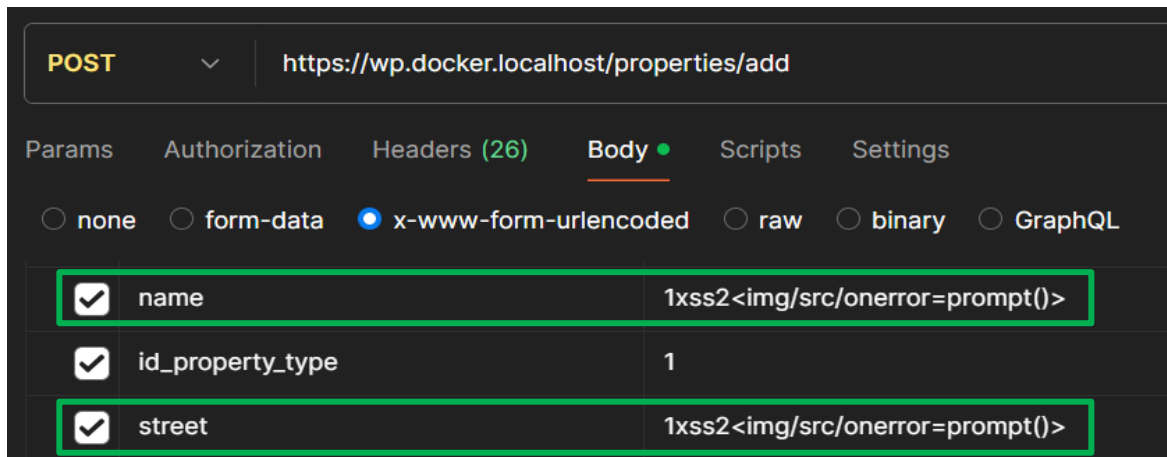


Figure 25 : Requête avec balise ``

La balise `<img/src/onerror=prompt()>` est utilisée ici pour essayer d'injecter du code JavaScript.

- L'attribut `src` est mal formé (pas d'URL valide), ce qui provoque une erreur de chargement de l'image.
- L'attribut `onerror` est déclenché car le chargement de l'image échoue.
- L'événement `onerror` exécute alors `prompt()`, ce qui affichera une boîte de dialogue si le script est exécuté.

Cela permet de tester facilement si le site est vulnérable à une attaque XSS. Si une boîte de dialogue `prompt()` s'affiche, cela signifie que le code a été interprété par le navigateur, indiquant une faille de sécurité.

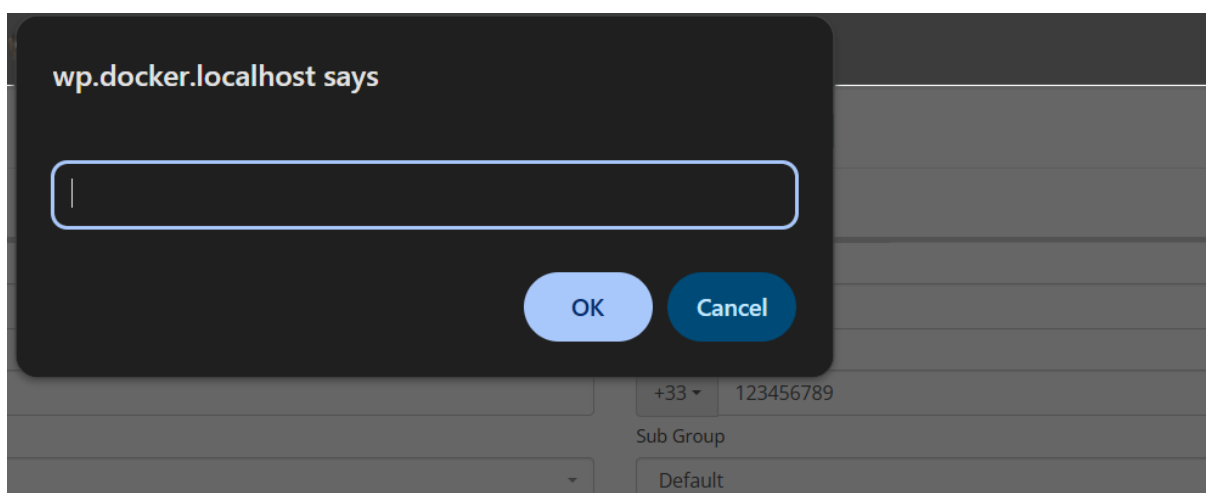


Figure 26 : Affichage du prompt après attaque XSS

Sur cette image (cf. Figure 26), on voit que la boîte de dialogue `prompt()` s'est bien affichée. Cependant, la balise `` n'apparaît pas dans l'affichage des champs `name` et `street` de la propriété en bas à gauche de l'image, mais plutôt dans la liste des options de sélection, ici (cf. Figure 27) :

Access to Properties

Only To Selected Properties

Select Property (city based)

Property

1xss2<img/src/onerror=prompt()-1xss2<img/src/onerror=prompt()->

Add Access

Property Permissions

No Properties have been saved as

Figure 27 : Affichage de la propriété avec balise sur le site

Ces informations m'ont aidé à mieux comprendre l'origine du problème. Plus précisément, je sais maintenant que les champs à échapper sont intégrés à l'intérieur d'une balise HTML `<select>`.

Réalisation

J'ai donc retrouvé le code concerné dans une balise `<select>` dans la vue `add.phtml`, située dans le dossier des vues `manager`. Pour traiter correctement les entrées, j'ai d'abord décodé les entités HTML à l'aide de `htmlspecialchars_decode()`, puis utilisé `strip_tags()` afin de supprimer toutes les balises HTML de la chaîne de texte (cf. Figure 28) :

```
<select id="form-select-property-main" class="form-control form-select-searchable" name="id_property">
  <?php foreach ($properties as $pp) : ?>
    <option value="<?=$pp['id'] ?>" <?=$isset($formValues['id_property']) && $formValues['id_
    <?=$strip_tags(htmlspecialchars_decode(string: $pp['name'] . ' - ' . $pp['street'])) ?>
    </option>
  <?php endforeach; ?>
</select>
```

Figure 28 : Traitement des paramètres `name` et `street` d'une propriété avant affichage

Validation

Access to Properties

Only To Selected Properties

Select Property (city based)

Property

1xss2 - 1xss2

Add Access

Property Permissions

No Properties have been save

Figure 29 : Affichage de la propriété avec balise après modification du code

On remarque qu'après cette modification, la balise HTML ne s'affiche plus (cf. Figure 29), tout comme le prompt. Nous avons donc réussi à protéger cette page contre les attaques XSS.

Remarque importante : Ma première solution consistait à appliquer la même approche lors de la création de la propriété. Cependant, un autre développeur m'a expliqué que modifier une fonction aussi triviale pouvait engendrer de nombreux problèmes. Par curiosité, j'ai tout de même testé cette solution, et je me suis rapidement rendu compte qu'il avait raison : la propriété que j'avais créée après l'ajout de `htmlspecialchars_decode()` et `strip_tags()` s'affichait mal sur de nombreuses pages (cf. Figure 30) :

Actions	Name	Street	City	State	Zip/Postal Code	Cr PI
Profile" data-toggle="tooltip">🔍 " data-toggle="tooltip">🔍 " data-toggle="tooltip">🔍	1+ATest1">	1+ATest2">	Irvine	CA	12345	+33
🔍 🔍 🔍	1+ATest3"> <img/src/onerror=prompt()>	1+ATest3"> <img/src/onerror=prompt()>	Irvine	CA	12345	+33

Figure 30 : Mauvais affichage de la propriété avec logique de création modifiée

4.3.3 - Prévention des failles liées aux ACL

Les **Access Control Lists (ACL)** définissent les permissions des utilisateurs sur un système. Une mauvaise configuration peut exposer des données sensibles ou permettre un accès non autorisé.

Conception

D'après le même rapport, il est possible d'exploiter une faille de contrôle d'accès sur la page `/reports/registration-pending` (cf. Figure 31). En effet, il est possible de modifier l'identifiant de la **Property Management Company*** dans la requête et d'accéder à des données auxquelles l'utilisateur ne devrait normalement pas avoir accès. Cela signifie que, même si l'interface du site ne propose que les **Property Management Companies** accessibles à l'utilisateur, il ne vérifie pas correctement, côté serveur, si celui-ci a réellement les permissions nécessaires.

ACL issue at "/reports/registration-pending":

Indeed, the `"/reports/registration-pending"` endpoint lacks proper access control allowing any property manager to retrieve information belonging to another tenant by abusing the parameter `"propertyMgmtId"`.

Figure 31 : Description de la faille ACL

Pour reproduire cela, j'ai créé un compte **Property Manager** ayant accès uniquement à la **Property Management Company** identifiant `propertyMgmtId = 4` (cf. Figure 32 et 33) :

Property Access

Access to Properties

Only To Selected Property Management Company

Select Property Management Company

Property Management Company

Engineering Property Management Company

Add Access

Figure 32 : Ajout d'une Property Management Company à un Property Manager

```
1 SELECT * FROM property_mgmt WHERE NAME LIKE "Engi%"
```

#	id	name	street	id_city
1	4	Engineering Property Management Company	44736 Corte Valencia	11403

Figure 33 :Affichage du id et nom d'une Property Management Company dans HeidiSQL

J'ai vérifié que les données de cette **Property Management Company** s'affichent correctement (cf. Figure 34) :

Search By Property Management Companies: Engineering Property Management Company (LO...
Have Not Registered For Over: 1 Day
Search Clear

Show: 25 entries

Search:

Property ID	User ID	Resident Name	Pending Days	Email	Phone	Reminders Sent	Last Received At	Property
10	4962	Test3, VStripe3	1655			0		UK QA property with fee:
10	4959	Test, Vstripe	1655			0		UK QA property

Figure 34 : Vérification de présence des données dans la Property Management Company choisie

Ensuite, j'ai copié et importé la même requête dans Postman afin de pouvoir la modifier :

propertyMgmtId: 5
200 OK • 484 ms • 68.7 KB • Save Response

HTML Preview Visualize

Property ID	User ID	Resident Name	Pending Days	Email	Phone	Reminders Sent	Last Received At	Property
2	47	Stolp6, Kevin6	1795			0		Uk QA Parcel Pending
2	48	Stolp7, Kevin7	1795			0		Uk QA Parcel Pending

Figure 35 : Réponse à l'attaque ACL avant modification du code

J'ai envoyé la requête avec `propertyMgmtId = 5` au lieu de 4. Ce **Property Manager** ne devrait pas avoir accès à `propertyMgmtId = 5`, pourtant le serveur a tout de même renvoyé des données (cf. Figure 35). Cela confirme donc une faille de contrôle d'accès.

Réalisation

J'ai analysé la méthode `registrationPendingAction()` et réalisé que je pouvais utiliser une variable déjà présente dans le code, `$propertyMgmtOptions`, qui sert à afficher les propriétés accessibles à l'utilisateur dans la liste. J'ai donc ajouté une condition vérifiant que l'ID envoyé dans la requête est bien présent dans cette variable (cf. Figure 36). Sinon, une erreur est générée et l'utilisateur est redirigé vers la page `reports`. Bien que cette redirection ait peu d'importance (puisque l'affichage de cette erreur indique une tentative d'attaque), il s'agit néanmoins d'une bonne pratique en matière de sécurité.

```
public function registrationPendingAction()

    if (!in_array($formData['propertyMgmtId'], array_keys($propertyMgmtOptions))) {
        $this->flashMessenger()
            ->setNamespace('error')
            ->addMessage('You do not have access to the specified property management.');
```

Figure 36 : Vérification des Property Management Company accessibles à l'utilisateur connecté

Validation

J'ai renvoyé la requête qui permettait d'accéder à des données non autorisées pour l'utilisateur. Cette fois, aucune donnée n'est affichée, l'utilisateur est correctement redirigé et un message d'erreur s'affiche (cf. Figure 37).

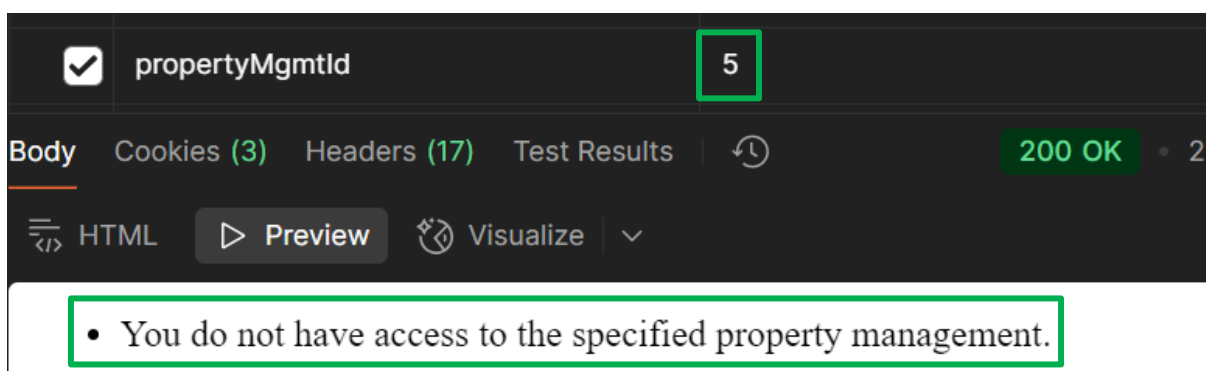


Figure 37 : Affichage de l'erreur après modification du code concernant l'ACL

5 - Méthodologie et organisation du projet

5.1 - Organisation du travail dans l'entreprise

Chez Quadient, l'organisation du travail repose sur une méthode Agile adaptée au contexte international de l'équipe, répartie entre la France et le Vietnam. Les développements sont découpés en **sprints* de deux semaines**, au cours desquels les tâches sont planifiées, réalisées et révisées.

Chaque sprint débute par une planification où les membres de l'équipe sélectionnent leurs tickets Jira en concertation avec leur manager. Le suivi quotidien s'effectue par **daily meetings*** en format texte sur **Microsoft Teams**. Les **daily meetings** sont effectués par écrit, car ce format est à la fois rapide, simple et efficace. Il permet également de conserver une trace chronologique claire dans le chat de Microsoft Teams, facilitant ainsi le suivi de l'avancement. Ces échanges sont rédigés en **anglais**, l'équipe étant **multinationale**.

Voici un exemple d'un de mes rapports quotidiens (cf. Figure 38) :

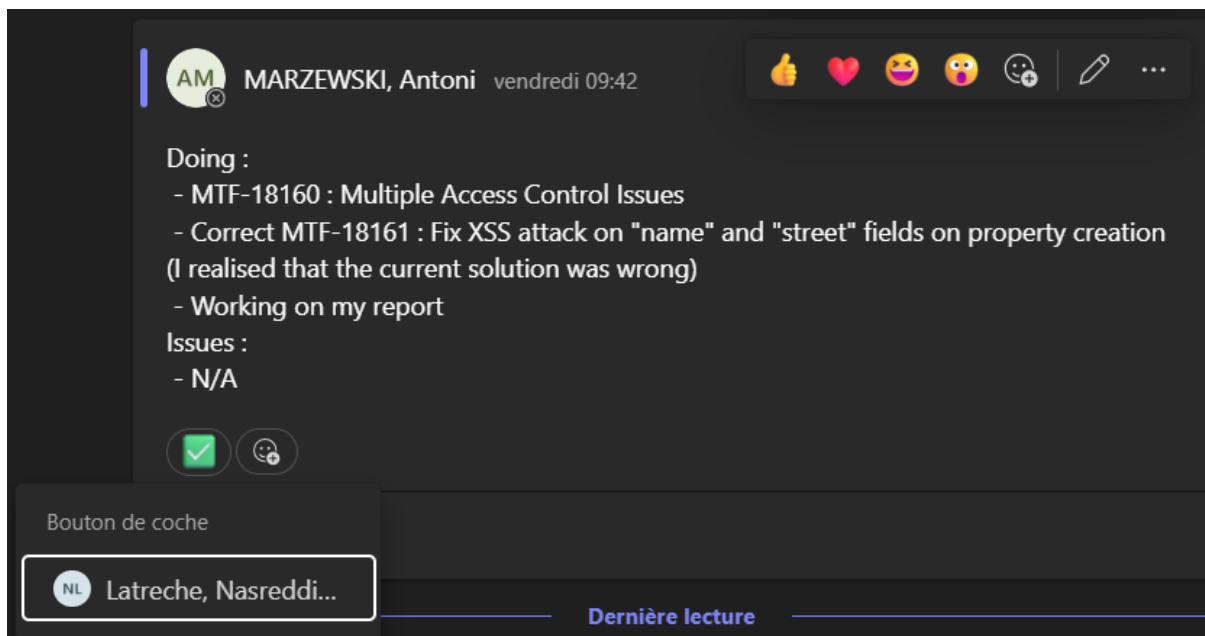


Figure 38 : Exemple de rapport quotidien sur teams

L'environnement de travail comprend plusieurs environnements :

- **Dev** : environnement personnel, isolé, pour le développement et les tests individuels.
- **QA** : partagé avec l'équipe de validation, pour les tests fonctionnels et de régression.
- **UAT** : utilisé par les clients internes pour les tests finaux avant la mise en production.

Des **diagrammes GANTT ou PERT** n'étaient pas utilisés, la méthode Agile privilégiant la flexibilité et la communication continue sur une planification rigide. Néanmoins, Jira permettait un suivi précis de l'avancement des tâches, avec des statuts comme **To Do, Blocked, In Progress, Ready for QA et Done** :

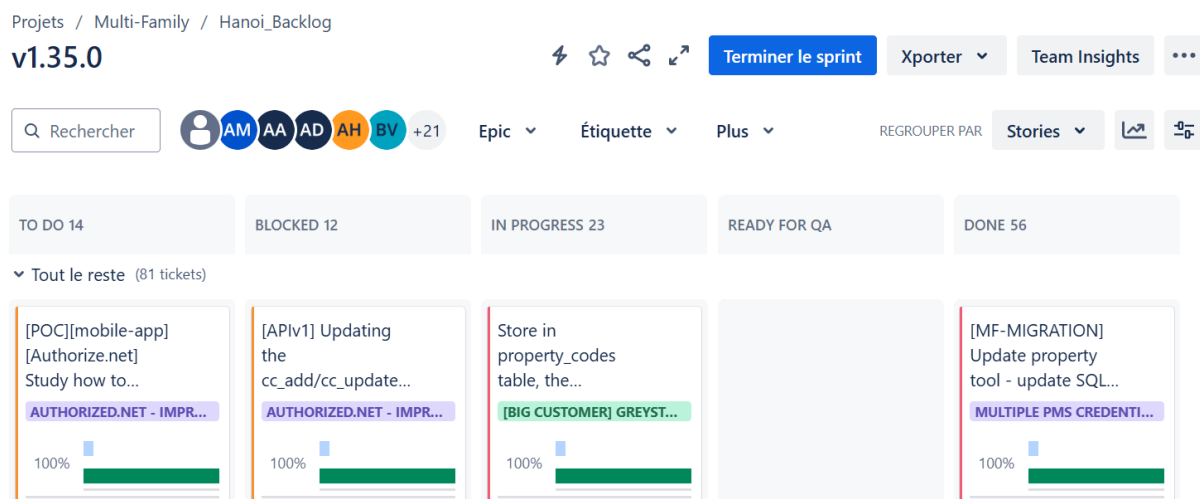


Figure 39 : Exemple de backlog* de mon équipe

5.2 - Flux de travail sur un Ticket

Une fois une tâche attribuée, voici les étapes suivies :

1. **Clonage de la branche** : Récupération de la dernière version du code en clonant la branche du sprint en cours.
2. **Création d'une nouvelle branche à partir de la courante** : Création d'une branche à partir de la branche du sprint en cours, nommée avec le code du ticket suivi d'une courte description, séparée par des tirets (par exemple, **MTF-1234-fix-sql-injection-in-monthly-package-data**).
3. **Analyse du problème** : Reproduction du bug ou compréhension de la nouvelle fonctionnalité à implémenter.
4. **Développement et tests locaux** : Implémentation des modifications nécessaires (correction de bugs, nouvelles fonctionnalités, optimisations), suivie de tests manuels.
5. **Création d'une Pull Request (PR)** : Une fois le développement terminé et testé, soumission d'une PR sur GitLab pour validation.
6. **Validation et fusion** : Révision par le Reviewer et fusion du code dans la branche principale après approbation.
7. **Déploiement en QA** : Mise à disposition du code pour des tests supplémentaires avant passage en UAT et production.

Voici un exemple d'un **pull request** dans Jira (cf. Figure 40) :



Figure 40 : Exemple d'un pull request dans Jira

En haut de cette image, on peut voir le nom de la **pull request** ainsi que l'identifiant du ticket correspondant, "**MTF-18161**". On y trouve également la branche cible dans laquelle je souhaite fusionner mon travail, qui correspondait le plus souvent à la prochaine version du site. Un peu plus bas, on peut voir mon message de commit, qui décrit les modifications que j'ai effectuées.

La section "**Assignee**" indique la personne qui propose ce changement, en l'occurrence moi, tandis que "**Reviewer**" désigne la personne que choisi pour examiner et valider mes modifications. Cette personne peut consulter les détails des modifications en cliquant sur la section "Changes" et y ajouter des commentaires pour proposer des améliorations ou autres suggestions.

5.3 - Méthodes adoptées personnellement

Dès le début du stage, j'ai fait preuve d'une **forte autonomie et d'un esprit d'initiative**. Je me suis **autoformé à des outils internes** (Zscaler, Docker, HeidiSQL, etc.) en m'appuyant sur la documentation de l'entreprise et en posant des questions ciblées à mes collègues.

Lors des développements, j'ai systématiquement :

- Analysé les tickets de manière proactive,
- Cherché des informations dans la base de code et les bases de données,
- Proposé des solutions lors des revues de code, par exemple (cf. Figure 41) :

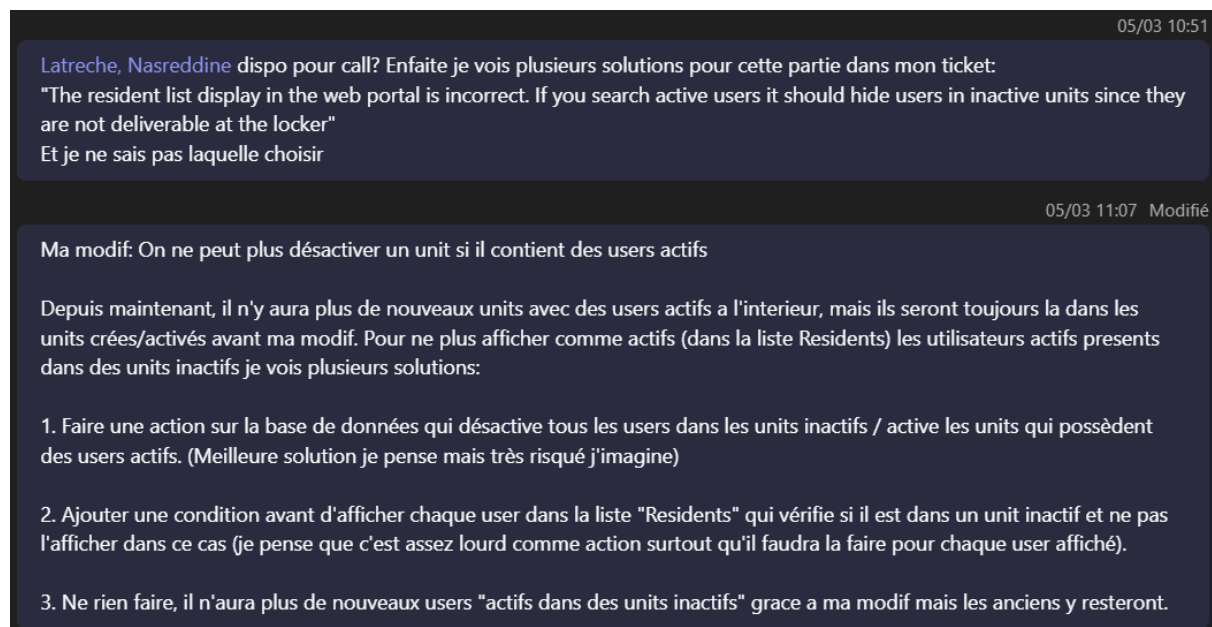


Figure 41 : Solutions proposées par moi lors du travail sur un ticket

Ma **proactivité a été remarquée** notamment lors de la correction de failles de sécurité (XSS, injections SQL, ACL), où j'ai simulé des attaques via Postman, analysé les failles, proposé des correctifs, et amélioré la robustesse du code. J'ai également su **appliquer des principes d'architecture MVC et de clean code** dans toutes mes contributions.

5.4 - Apprentissages mobilisés

Concernant la méthodologie de travail, les apprentissages critiques suivants ont été particulièrement développés pendant le stage :

- **AC25.04 | Définir et mettre en œuvre une démarche de suivi de projet**
→ Participation active aux sprints, suivi de tickets Jira, daily reports, échanges concernant mes propositions de code.
- **AC26.03 | Mobiliser les compétences interpersonnelles pour travailler dans une équipe informatique**
→ Travail en équipe répartie, communication efficace, interactions en anglais, intégration fluide dans l'équipe Agile.

6 - Conclusion

6.1 - Un projet utile, intégré et maintenu par l'entreprise

Durant ces dix semaines de stage chez **Quadient**, j'ai contribué à la maintenance, à l'optimisation et à la sécurisation du **Web Portal de Parcel Pending**, une application essentielle pour les gestionnaires de consignes de colis. L'ensemble des correctifs et des améliorations que j'ai réalisés ont été **validés par l'équipe de développement, testés en environnement QA**, puis intégrés au code principal. Mon travail est donc **repris par l'entreprise et utilisé en production**, ce qui atteste à la fois de sa qualité et de sa pertinence.

Plusieurs de mes interventions ont permis **d'augmenter la fiabilité et la sécurité du système d'information** :

- Correction de bugs fonctionnels affectant l'interface utilisateur (affichage, gestion d'unités),
- Résolution de failles de sécurité critiques (injection SQL, XSS, contournement des droits d'accès),
- Amélioration de la logique de gestion des utilisateurs selon les rôles définis (ACL).

Ce projet a donc eu un impact direct sur le fonctionnement du portail, tout en respectant les standards techniques et organisationnels de l'équipe Agile.

6.2 - Une montée en compétences du BUT Informatique

Ce stage m'a offert l'occasion d'appliquer concrètement les **6 compétences du BUT Informatique**, dans des situations variées et réalistes :

- **Réaliser un développement d'application** : j'ai développé de nouvelles fonctionnalités et corrigé du code métier existant dans un projet de grande ampleur, en respectant les standards internes et les principes du **design pattern MVC**.
- **Optimiser des applications** : par exemple, en sécurisant une page vulnérable aux injections SQL, ou en corrigeant une faille XSS via une gestion rigoureuse de l'affichage côté client.
- **Administrer des systèmes informatiques communicants complexes** : j'ai configuré un environnement de développement professionnel incluant **Docker**, **WSL**, **GitLab**, **Postman** et des outils de sécurité comme **Zscaler**, tout en manipulant des API REST et des conteneurs applicatifs.
- **Gérer des données de l'information** : à travers l'analyse et l'écriture de requêtes SQL complexes, la consultation de la base via **HeidiSQL**, et la sécurisation des interactions avec la base de données.

- **Conduire un projet** : j'ai suivi la méthode **Agile** en sprint de deux semaines, organisé mes tickets via **Jira**, rédigé des rapports quotidiens, produit des **pull requests documentées**, et suivi l'ensemble du cycle de développement jusqu'à la mise en production.
- **Travailler dans une équipe informatique** : au sein d'une équipe internationale répartie entre la France et le Vietnam, j'ai appris à **collaborer à distance en anglais**, à **respecter les workflows collectifs**, et à intégrer les retours constructifs du reviewer de mes tickets.

6.3 - Une expérience formatrice, concrète et professionnalisante

Ce stage a été une véritable **mise en situation professionnelle**, bien au-delà d'un simple exercice académique. J'ai appris à **gagner en autonomie**, à **prendre des initiatives**, **travailler en équipe** et à **proposer des solutions concrètes**.

Il m'a permis de comprendre les exigences réelles d'un développement en entreprise, tant au niveau **technique** qu'au niveau **organisationnel** et **humain**. Grâce à cette expérience, je me sens aujourd'hui bien mieux préparé à intégrer un projet de développement informatique complexe, que ce soit en alternance, en poursuite d'étude ou dans le monde professionnel.

Signatures

N.LATRECHE
Vu le 26 mars 2025

A stylized, handwritten signature in black ink, consisting of a large, rounded 'O' shape with a small loop at the top right.

Antoni Marzewski

A handwritten signature in black ink, written in a cursive style, reading 'Marzewski'.

Bibliographie

[1] "Quadient," *Wikipédia*, [En ligne]. Disponible sur : <http://fr.wikipedia.org/wiki/Quadient>. [Consulté le 25 mars 2025].

[2] "Tout savoir sur Quadient : salaires, stage, recrutement," *Planète Grandes Écoles*, [En ligne]. Disponible sur : <https://www.planetegrandesecoles.com/tout-savoir-sur-quadient-salaires-stage-recrutement>. [Consulté le 25 mars 2025].

[3] "Quadient S.A. : Activité de la société," *Zonebourse*, [En ligne]. Disponible sur : <https://ch.zonebourse.com/cours/action/QUADIENT-S-A-4674/societe/>. [Consulté le 25 mars 2025].

[4] "Le transporteur canadien Purolator s'équipe de consignes intelligentes Parcel Pending de Quadient," *Option Finance*, 22 février 2022. [En ligne]. Disponible sur : <https://www.optionfinance.fr/info-financiere-en-continu/d/2022-02-22-le-transporteur-canadien-purolator-sequipe-de-consignes-intelligentes-parcel-pending-de-quadient.html>. [Consulté le 25 mars 2025].

[5] "À propos de Parcel Pending," *Parcel Pending by Quadient*, [En ligne]. Disponible sur : <https://www.parcelpending.com/ca-fr/about/>. [Consulté le 25 mars 2025].