

Rapport d'alternance

2022 - 2023

Stephane Hilaricus

MAISON ET CITES – IUT DE LENS | DOUAI - LENS

Table des matières

Rapport de synthèse	2
Introduction.....	2
Présentation de l'entreprise.....	3
Développement.....	4
Vous et l'entreprise	5
Mission réalisée.....	6
Conclusion	7
Annexes	8
Liste des abréviations et sigles utilisés.....	8
Autres annexes pertinentes	9
Rapport technique	12
Introduction.....	12
Développement.....	13
Détail des fonctionnalités attendues de votre programme.....	13
Outils collaboratifs utilisés	17
Descriptif de la démarche adoptée	18
Cas d'utilisation	19
Descriptif des méthodes importantes.....	20
Descriptions des données et structures des données.....	21
Descriptif des algorithmes utilisés	22
Les missions.....	27
Les bugs que j'ai résolus.....	27
Protocoles utilisés	29
Aspects liés à la sécurité (si applicable).....	30
Conclusion	32
Annexes.....	33
Liste des abréviations et sigles utilisés.....	33
Rappel du formalisme	34
Autres annexes pertinentes	36

Rapport de synthèse

Introduction

Ce rapport d'alternance a pour objectif principal de présenter les compétences professionnelles que j'ai acquises au cours de mon alternance à "Maisons et Cités", dans le cadre de ma Licence Professionnelle DIOC. L'obtention de ce diplôme requiert la démonstration de compétences spécifiques, que j'ai eu l'opportunité de développer et d'affiner tout au long de ma période d'alternance.

"Maisons et Cités" est le plus grand bailleur social des Hauts-de-France, ce qui nécessite un service informatique robuste et efficace. J'ai eu le privilège d'occuper le poste d'assistant développeur au sein de cette entreprise, sous la supervision de mon tuteur, Sylvain Harle. Le service informatique, initialement situé à Billy-Montigny, sera délocalisé à Douai, au nouveau siège de la société, à partir du 9 juin.

Mon alternance, qui s'est déroulée du 5 septembre 2022 au 18 août 2023, m'a permis de m'immerger dans un environnement professionnel stimulant et enrichissant. Le service informatique est composé d'une vingtaine de collaborateurs, répartis en deux domaines : le domaine des réseaux, de la sécurité et du paramétrage système, et le pôle de gestion applicative, au sein duquel j'ai travaillé.

Au cours de ces mois, j'ai non seulement acquis des compétences techniques en travaillant avec divers progiciels et en comprenant les méthodes utilisées dans le monde professionnel, mais j'ai également développé des compétences socio-professionnelles. Ce rapport détaille ces compétences et l'expérience que j'ai acquise au cours de mon alternance à "Maisons et Cités".

Présentation de l'entreprise

"Maisons et Cités" est le plus grand bailleur social des Hauts-de-France, une région située dans le nord de la France. En tant que bailleur social, l'entreprise joue un rôle crucial dans la fourniture de logements abordables pour les personnes et les familles à revenus modestes.

"Maisons et Cités" gère un large éventail de propriétés, allant des maisons individuelles aux immeubles d'appartements, et s'efforce de fournir des logements de qualité à ses locataires. L'entreprise est engagée dans la rénovation et la modernisation de ses propriétés pour améliorer la qualité de vie de ses locataires et pour assurer que les logements répondent aux normes actuelles en matière d'efficacité énergétique et de confort.

L'entreprise dispose d'un service informatique robuste, qui est essentiel pour gérer efficacement son vaste portefeuille de propriétés. Le service informatique est divisé en deux domaines : le domaine des réseaux, de la sécurité et du paramétrage système, et le pôle de gestion applicative. Le service informatique est actuellement situé à Billy-Montigny, mais sera délocalisé à Douai, au nouveau siège de la société, à partir du 9 juin.

"Maisons et Cités" est une entreprise qui valorise le développement des compétences et l'apprentissage continu de ses employés, y compris ses stagiaires. En tant que bailleur social, l'entreprise est également profondément engagée dans la promotion de l'équité sociale et la création de communautés durables.

Développement

Mon rôle au sein de la société "Maisons et Cités" consiste à assister le développeur, Sylvain Harle. Ma mission principale est de maintenir une application web existante. Pour ce faire, j'ai dû apprendre le langage de programmation utilisé pour son développement, à savoir Java avec le Framework Spring et ReactJs. Pour me former, j'ai suivi un cours sur ReactJs sur les plateformes OpenClassroom et YouTube ([Annexe 1](#)). J'ai ensuite renforcé mes compétences en Java en codant des mini-programmes pour m'exercer.

L'application en question permet de recueillir des fichiers qui fonctionnent par paires, c'est-à-dire un fichier PDF et un fichier XML. Ces fichiers sont des Diagnostics de Performance Energétique (DPE, [Annexe 2](#)). Une fois recueillis, les fichiers sont envoyés à différents endroits : un Gestionnaire Electronique de Documents (GED) et Ikos, l'ERP (Enterprise Resource Planning) de l'entreprise. L'application collecte les données des différents DPE afin de fournir toutes les informations nécessaires pour que chaque employé puisse atteindre ses objectifs. L'application pourrait évoluer pour intégrer d'autres types de diagnostics. Pour éditer le code, j'utilise un Environnement de Développement Intégré (IDE). L'application est stockée sur des serveurs et fonctionne avec un "middleware", un logiciel qui fournit des services et fonctionnalités unifiés aux applications.

Comme mentionné précédemment, j'ai acquis certaines compétences et mentalités du milieu professionnel au cours de mon alternance. Notamment, j'ai appris l'importance du travail en équipe. Par exemple, en cas de problème, je devais en informer le pilote de projet concerné, et si c'était un problème de serveur, je prévenais un administrateur réseau. J'ai également acquis certaines compétences techniques, comme le déploiement d'une application, la maintenance de celle-ci, ainsi que la rédaction d'études d'amélioration.

Vous et l'entreprise

Au sein de "Maisons et Cités", j'occupe le poste d'assistant développeur, un rôle qui m'a permis de m'immerger dans le monde professionnel de l'informatique et de développer une gamme de compétences techniques et socio-professionnelles. Mon tuteur, Sylvain Harle, m'a guidé tout au long de ce processus, me fournissant un soutien et des conseils précieux.

Mon intégration dans l'équipe a été une expérience enrichissante. J'ai eu l'opportunité de travailler avec une équipe de professionnels dévoués, qui m'ont aidé à comprendre les nuances du travail en équipe dans un environnement professionnel. J'ai également eu l'occasion de participer à des réunions d'équipe et de collaborer sur des projets, ce qui m'a permis de comprendre comment les différents rôles au sein de l'équipe se complètent pour atteindre les objectifs communs.

En termes de compétences techniques, j'ai eu l'opportunité de travailler sur une application web existante, ce qui m'a permis d'approfondir mes connaissances en Java et ReactJs. J'ai également eu l'occasion d'apprendre à utiliser un Environnement de Développement Intégré (IDE) et à travailler avec des serveurs et des logiciels middleware.

En outre, j'ai appris à utiliser efficacement les outils de communication à distance et à gérer mon temps et mes tâches de manière autonome.

En somme, mon expérience chez "Maisons et Cités" a été une période d'apprentissage intense et de croissance personnelle et professionnelle. J'ai acquis une compréhension approfondie du fonctionnement d'un service informatique dans une grande entreprise et j'ai développé des compétences qui me seront utiles tout au long de ma carrière.

Mission réalisée

Au cours de mon alternance chez "Maisons et Cités", ma mission principale a été de maintenir une application web existante. Cette application joue un rôle crucial dans la gestion des Diagnostics de Performance Energétique (DPE) de l'entreprise, en recueillant et en envoyant des fichiers PDF et XML à différents endroits, notamment à un Gestionnaire Electronique de Documents (GED) et à l'ERP de l'entreprise, Ikos.

Pour accomplir cette mission, j'ai dû me familiariser avec le langage de programmation utilisé pour développer l'application, à savoir Java avec le Framework Spring et ReactJs. J'ai suivi des cours en ligne et pratiqué en codant des mini-programmes pour renforcer mes compétences.

En plus de maintenir l'application existante, j'ai également eu l'opportunité de créer une API qui permet de contrôler l'intégrité des fichiers XML grâce aux fichiers xsd. Cela a ajouté une couche supplémentaire de sécurité et d'efficacité à l'application, en assurant que les fichiers XML sont corrects et conformes aux normes avant d'être traités.

En somme, ma mission chez "Maisons et Cités" m'a permis de mettre en pratique mes compétences en programmation et en développement web, tout en contribuant à l'amélioration et à l'efficacité des processus de l'entreprise.

Conclusion

En conclusion, mon expérience en tant qu'assistant développeur chez "Maisons et Cités" a été extrêmement enrichissante et instructive. J'ai eu l'opportunité de travailler sur des projets concrets, d'appliquer les compétences techniques que j'ai acquises lors de ma formation et d'en développer de nouvelles. J'ai également eu l'occasion de comprendre le fonctionnement d'un service informatique au sein d'une grande entreprise et d'apprendre à travailler en équipe dans un environnement professionnel.

La maintenance de l'application web existante et la création d'une API pour contrôler l'intégrité des fichiers XML ont été des tâches stimulantes qui m'ont permis de mettre en pratique mes compétences en Java, Spring et ReactJs. Ces expériences m'ont également permis de comprendre l'importance de la sécurité et de l'efficacité dans le développement d'applications.

En outre, j'ai appris à naviguer dans les défis du travail à distance, à utiliser efficacement les outils de communication à distance et à gérer mon temps et mes tâches de manière autonome.

En somme, cette alternance a été une étape importante dans ma formation et mon développement en tant que professionnel de l'informatique. Les compétences et les expériences que j'ai acquises chez "Maisons et Cités" seront inestimables pour ma future carrière. Je suis reconnaissant pour cette opportunité et je suis impatient de continuer à développer mes compétences et à contribuer au domaine de l'informatique.

Annexes

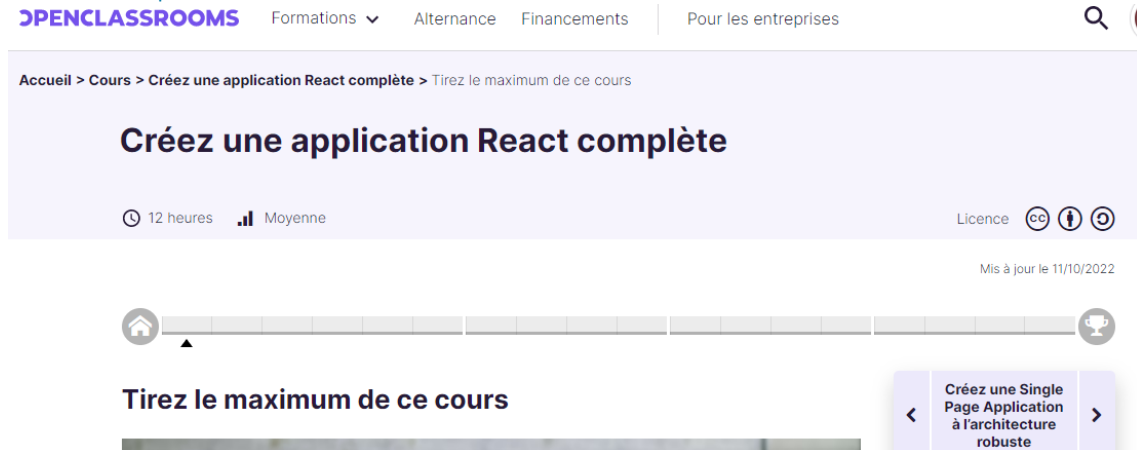
Liste des abréviations et sigles utilisés

1. DIOC : Développement Informatique et Outils Collaboratifs
2. IDE : Environnement de Développement Intégré
3. DPE : Diagnostic de Performance Energétique
4. GED : Gestionnaire Electronique de Documents
5. ERP : Enterprise Resource Planning
6. API : Interface de Programmation d'Applications
7. XML : eXtensible Markup Language
8. PDF : Portable Document Format
9. xsd : XML Schema Definition
10. GitHub : Plateforme de développement logiciel
11. VSCode : Visual Studio Code, un éditeur de code source
12. Spring : Framework de développement pour Java
13. ReactJs : Bibliothèque JavaScript pour la construction d'interfaces utilisateur
14. PostgreSQL : Système de gestion de base de données relationnelle objet
15. CORS : Cross-Origin Resource Sharing, un mécanisme qui permet à de nombreuses ressources (par exemple, les polices, JavaScript, etc.) sur une page web d'être demandées à un autre domaine en dehors du domaine d'origine.
16. AS400 : Système d'exploitation développé par IBM
17. ESB : Enterprise Service Bus, un type de logiciel utilisé pour intégrer des applications d'entreprise
18. Java : Langage de programmation orienté objet
19. HTML : HyperText Markup Language, le langage de balisage standard pour les documents conçus pour être affichés dans un navigateur web
20. CSS/SCSS : Cascading Style Sheets, un langage de feuille de style utilisé pour décrire l'apparence d'un document écrit en HTML ou en XML
21. Maven : Outil de gestion et de compréhension de projet
22. Axios : Bibliothèque JavaScript utilisée pour faire des requêtes HTTP
23. URL : Uniform Resource Locator, une référence à une ressource web
24. FTP : File Transfer Protocol, un protocole de transfert de fichiers

- 25. IXX : Type de fichier spécifique
- 26. GMC : Environnement spécifique dans le contexte de l'entreprise
- 27. FMC : Environnement spécifique dans le contexte de l'entreprise
- 28. WinDev : Atelier de génie logiciel conçu pour développer des applications, principalement orientées données
- 29. db2 : Système de gestion de base de données relationnelle produit par IBM
- 30. Middleware : Logiciel qui se situe entre les systèmes d'exploitation et les applications sur un réseau
- 31. Linux : Famille de systèmes d'exploitation de type Unix
- 32. Blueway : ESB utilisé dans le contexte de l'entreprise
- 33. Xemi : GED utilisé dans le contexte de l'entreprise
- 34. Ikos : ERP utilisé dans le contexte de l'entreprise
- 35. OpenClassroom : Plateforme d'apprentissage en ligne
- 36. YouTube : Plateforme de partage de vidéos en ligne.

Autres annexes pertinentes

Annexe 1 OpenClassroom :



The screenshot shows the OpenClassrooms website interface. At the top, there's a navigation bar with the 'OPENCLASSROOMS' logo and links for 'Formations', 'Alternance', 'Financements', and 'Pour les entreprises'. A search icon is on the right. Below the navigation bar, a breadcrumb trail reads 'Accueil > Cours > Créez une application React complète > Tirez le maximum de ce cours'. The main heading is 'Créez une application React complète'. Below it, there's a clock icon indicating '12 heures' and a bar chart icon labeled 'Moyenne'. To the right, there's a 'Licence' section with icons for CC, BY, and NC. Below this, it says 'Mis à jour le 11/10/2022'. A progress bar is shown with a home icon on the left and a trophy icon on the right. Below the progress bar, the text 'Tirez le maximum de ce cours' is displayed. On the right side, there's a button labeled 'Créez une Single Page Application à l'architecture robuste' with left and right navigation arrows.

Annexe 2 DPE :

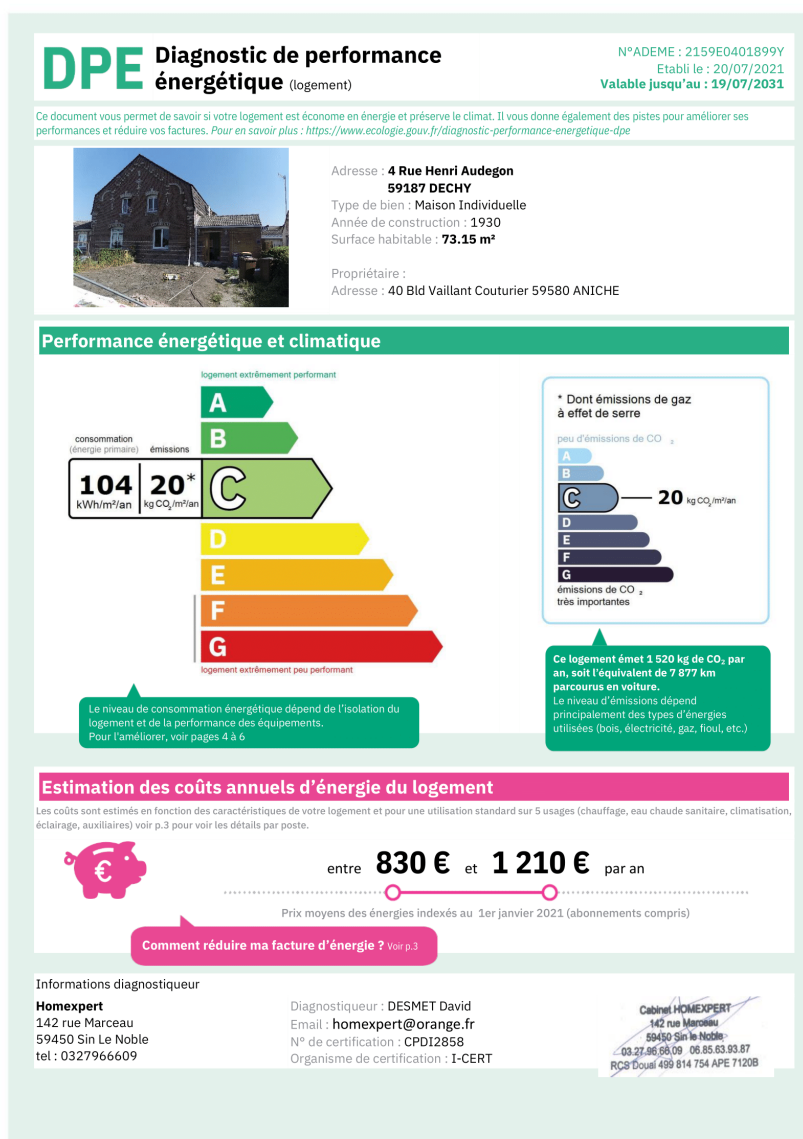
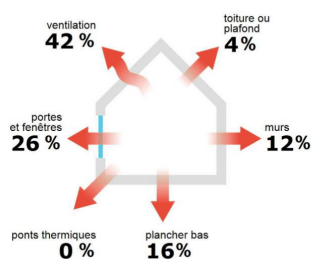


Schéma des déperditions de chaleur



Performance de l'isolation

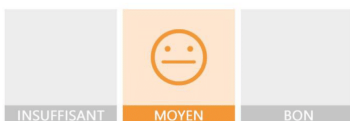


Système de ventilation en place



VMC SF Hygro B après 2012

Confort d'été (hors climatisation)*



Les caractéristiques de votre logement améliorant le confort d'été :



fenêtres équipées de volets extérieurs



toiture isolée

Production d'énergies renouvelables

Ce logement n'est pas encore équipé de systèmes de production d'énergie renouvelable.

Diverses solutions existent :



pompe à chaleur



chauffe-eau thermodynamique



panneaux solaires photovoltaïques



panneaux solaires thermiques



géothermie



réseau de chaleur ou de froid vertueux



chauffage au bois

*Le niveau de confort d'été présenté ici s'appuie uniquement sur les caractéristiques de votre logement (la localisation n'est pas prise en compte).

Rapport technique

Introduction

Ce rapport technique vise à détailler mon travail sur une application web existante au sein de l'entreprise "Maisons et Cités". L'application, développée en Java avec le Framework Spring et ReactJs, est encore en phase de développement et n'est pas encore en production. Ma mission principale a été de maintenir cette application, en résolvant les bugs existants pour assurer sa stabilité.

Récemment, une nouvelle responsabilité m'a été confiée : l'intégration d'une API à l'application. Cette API a pour but de vérifier l'intégrité du contenu des fichiers XML. Cette fonctionnalité est essentielle pour garantir que toutes les données nécessaires sont présentes et correctes, évitant ainsi les problèmes liés à des fichiers incomplets ou à des données manquantes.

Dans ce rapport, je vais décrire en détail les différentes tâches que j'ai réalisées, les problèmes que j'ai rencontrés et comment je les ai résolus, ainsi que les compétences techniques que j'ai acquises et utilisées tout au long de ce projet. Mon objectif est de fournir une vue d'ensemble claire et détaillée de mon travail sur cette application, en mettant l'accent sur les aspects techniques de la maintenance et du développement de l'application.

Développement

Détail des fonctionnalités attendues de votre programme

Le programme sur lequel j'ai travaillé est une application web qui a pour objectif principal de gérer les Diagnostics de Performance Energétique (DPE) de l'entreprise "Maisons et Cités". Les fonctionnalités attendues de ce programme sont les suivantes :

1. Gestion des fichiers DPE : L'application doit être capable de recueillir des fichiers DPE sous forme de paires de fichiers PDF et XML. Ces fichiers contiennent des informations essentielles sur la performance énergétique des bâtiments gérés par l'entreprise.
2. Envoi des fichiers : Une fois les fichiers DPE recueillis, l'application doit être capable de les envoyer à différents endroits. Cela comprend un Gestionnaire Electronique de Documents (GED) et l'ERP de l'entreprise, Ikos.
3. ([Annexe 14 à 16](#)) Contrôle de l'intégrité des fichiers XML : Une fonctionnalité clé de l'application est la capacité de vérifier l'intégrité du contenu des fichiers XML. Cela est réalisé grâce à une API que j'ai intégrée à l'application. Cette API assure que tous les fichiers XML sont complets et que toutes les données nécessaires sont présentes.
4. Maintenance et amélioration de l'application : En tant qu'assistant développeur, une partie de ma mission est de maintenir l'application, de résoudre les bugs existants et d'assurer la stabilité de l'application. Cela implique également de proposer et de mettre en œuvre des améliorations pour augmenter l'efficacité et la performance de l'application.

5. Préparation pour la mise en production : Enfin, l'application doit être préparée pour la mise en production. Cela signifie qu'elle doit être suffisamment stable et efficace pour être utilisée dans un environnement de production réel.

Ces fonctionnalités sont essentielles pour assurer le bon fonctionnement de l'application et pour répondre aux besoins de l'entreprise en matière de gestion des DPE.

L'interface utilisateur ([Annexe 2 à 5](#))

L'interface utilisateur est relativement simple, offrant moins de fonctionnalités qu'une interface administrateur. La première fonctionnalité, et l'une des plus importantes, est la possibilité pour un utilisateur de se connecter avec son compte (email/mot de passe) pour accéder à son tableau de bord.

Fonctionnalités du tableau de bord : ([Annexe 5](#))

Sur le tableau de bord, l'utilisateur doit pouvoir déposer les fichiers XML et PDF requis par "Maisons & Cités". Des contrôles sont effectués avant l'envoi des fichiers sur le serveur, notamment la vérification des types de fichiers (seuls les PDF et les XML sont acceptés), la correspondance des fichiers (le PDF et le XML doivent avoir le même nom, selon une charte de nommage définie par "Maisons & Cités") et le nombre de fichiers (un seul PDF et un seul XML doivent être déposés à la fois pour effectuer les contrôles précédents).

Ensuite, l'utilisateur doit pouvoir voir un historique des fichiers qu'il a déposés lui-même. Enfin, un utilisateur doit pouvoir envoyer une demande concernant l'ajout ou la suppression d'un compte de son entreprise. Cette demande est ensuite visible côté administrateur pour validation ou refus.

Fonctionnalités annexes : ([Annexe 2 à 4](#) et [8](#))

En plus du tableau de bord et du système de connexion, il y a un système de réinitialisation du mot de passe qui permet à l'utilisateur de définir un nouveau mot de passe via un lien reçu par mail et un token unique généré pour cela. Ensuite, lors de l'accès à une page, il faut vérifier que l'utilisateur est bien connecté et qu'il possède le bon rôle lui permettant d'accéder à ladite page. Si ce n'est pas le cas, il est redirigé sur une page d'erreur 403 (accès interdit). Enfin, si un utilisateur tente d'accéder à une page qui n'existe pas, il doit être redirigé sur une page d'erreur 404 personnalisée plutôt que sur la page d'erreur standard des navigateurs.

L'interface administrateur ([Annexe 6 et 7](#))

L'interface administrateur comprend principalement des fonctionnalités de gestion des utilisateurs. Tout comme l'interface utilisateur, celle-ci comprend le même système de connexion avec un courriel et un mot de passe qui permettent d'accéder au tableau de bord administrateur.

Fonctionnalités du tableau de bord : ([Annexe 6](#))

Sur le tableau de bord, l'administrateur doit pouvoir avoir un aperçu de l'ensemble des sociétés qui peuvent déposer des fichiers sur l'interface administrateur, avec la possibilité d'accéder aux détails de cette entreprise. Il doit également pouvoir créer de nouvelles entreprises si "Maisons & Cités" fait appel à d'autres prestataires. De plus, le tableau de bord affiche la liste des demandes faites par les utilisateurs, avec un système de validation ou de refus des demandes. En cas de validation, cela entraînera la création d'un nouvel utilisateur ou la suppression, selon le choix sur la demande.

Fonctionnalité de la page de détails entreprise ([Annexe 7](#)) :

Sur cette page, l'administrateur doit pouvoir voir tous les fichiers déposés par les utilisateurs de l'entreprise en question. Il doit également avoir un aperçu de la liste des utilisateurs avec leurs informations. De plus, il doit pouvoir ajouter ou supprimer des utilisateurs sans passer par les demandes, par exemple lors de l'ajout d'une entreprise, pour pouvoir ajouter le premier utilisateur.

Fonctionnalités annexes :

Au niveau des fonctionnalités annexes, l'interface administrateur doit posséder les mêmes que l'interface utilisateur.

Outils collaboratifs utilisés

Au cours de mon alternance chez "Maisons et Cités", j'ai utilisé plusieurs outils collaboratifs pour faciliter la communication, la coordination et le développement du projet. Ces outils ont joué un rôle crucial dans l'efficacité de mon travail et la réussite du projet. Voici les principaux outils collaboratifs que j'ai utilisés :

1. GitHub : GitHub est un service de gestion de versions basé sur Git qui facilite le travail en équipe sur un même projet. J'ai utilisé GitHub pour partager le code de l'application, suivre les modifications apportées par moi-même ou par d'autres membres de l'équipe, et gérer les versions de l'application. GitHub a également servi de plateforme pour la revue de code et la gestion des problèmes et des bugs.
2. VSCode : Visual Studio Code (VSCode) est un éditeur de code source développé par Microsoft. J'ai utilisé VSCode pour écrire et modifier le code de l'application. VSCode offre de nombreuses fonctionnalités qui facilitent le développement, comme la coloration syntaxique, l'auto-complétion de code, et l'intégration avec Git et GitHub.
3. Suite Office : J'ai utilisé plusieurs applications de la suite Office de Microsoft pour diverses tâches liées à la gestion du projet. Par exemple, j'ai utilisé Word pour rédiger des documents et des rapports, Excel pour gérer les données et créer des tableaux et des graphiques, et Outlook pour la communication par e-mail avec les autres membres de l'équipe et avec les autres parties prenantes du projet.

Ces outils collaboratifs ont grandement facilité mon travail et ont contribué à la réussite du projet. Ils ont permis une communication efficace, une coordination fluide et un développement de qualité de l'application.

Descriptif de la démarche adoptée

La démarche adoptée pour la réalisation de mes missions au sein de "Maisons et Cités" a été structurée et méthodique, afin d'assurer l'efficacité et la qualité du travail. Voici les principales étapes de cette démarche :

1. Compréhension du projet : La première étape a été de comprendre en profondeur l'objectif du projet, les fonctionnalités attendues de l'application et les technologies utilisées. Cela a impliqué des discussions avec mon tuteur et les autres membres de l'équipe, ainsi que l'étude de la documentation existante.
2. Formation aux technologies nécessaires : Pour pouvoir travailler efficacement sur l'application, j'ai dû me former aux technologies utilisées, notamment Java avec le Framework Spring et ReactJs. J'ai suivi des cours en ligne et pratiqué par moi-même pour acquérir les compétences nécessaires.
3. Maintenance de l'application existante : Une grande partie de mon travail a consisté à maintenir l'application existante, en résolvant les bugs et en améliorant la stabilité de l'application.
4. Développement de nouvelles fonctionnalités : J'ai également été chargé de développer de nouvelles fonctionnalités pour l'application, comme l'intégration d'une API pour vérifier l'intégrité des fichiers XML.
5. Test et validation : Chaque nouvelle fonctionnalité ou modification apportée à l'application a été soigneusement testée pour s'assurer qu'elle fonctionne correctement et qu'elle ne cause pas de problèmes avec les autres parties de l'application.

6. Collaboration et communication : Tout au long de ce processus, j'ai utilisé divers outils collaboratifs pour communiquer avec les autres membres de l'équipe, partager le code et coordonner le travail.

Cette démarche structurée m'a permis de travailler efficacement sur l'application et de contribuer à son amélioration et à son développement.

Cas d'utilisation

Les cas d'utilisation de l'application que j'ai maintenue et améliorée au cours de mon alternance chez "Maisons et Cités" sont principalement liés à deux groupes d'utilisateurs : les entreprises prestataires et le personnel de "Maisons et Cités".

1. Entreprises prestataires : Ces entreprises sont employées par "Maisons et Cités" pour effectuer les diagnostics de performance énergétique (DPE). Elles utilisent l'application pour envoyer les fichiers (PDF et XML) correspondant à chaque diagnostic réalisé. Ces fichiers sont ensuite traités par l'application, qui vérifie leur intégrité et les envoie aux destinations appropriées.
2. Personnel de "Maisons et Cités" : Bien que le personnel de "Maisons et Cités" n'utilise pas directement l'application, ils en sont les bénéficiaires indirects. En effet, les fichiers envoyés par les entreprises prestataires via l'application sont utilisés pour alimenter l'ERP (Ikos) de "Maisons et Cités". Cela permet au personnel de disposer de toutes les données nécessaires pour leur travail, comme les informations sur les performances énergétiques des bâtiments gérés par "Maisons et Cités".

Ces cas d'utilisation montrent l'importance de l'application dans le processus de gestion des diagnostics de performance énergétique chez "Maisons et Cités". Ils soulignent également la

nécessité d'une maintenance et d'une amélioration continues de l'application pour garantir son bon fonctionnement et la satisfaction de ses utilisateurs.

Descriptif des méthodes importantes

L'API que j'ai intégrée à l'application lors de mon alternance chez "Maisons et Cités" a une fonctionnalité clé : vérifier l'intégrité du contenu des fichiers XML. Cette fonctionnalité est essentielle pour garantir que les données reçues sont complètes et correctes, ce qui est crucial pour le bon fonctionnement de l'application et pour la qualité des informations fournies à l'ERP (Ikos) de "Maisons et Cités".

Voici une explication détaillée de cette fonctionnalité :

1. Réception des fichiers XML : L'API reçoit les fichiers XML envoyés par les entreprises prestataires qui réalisent les diagnostics de performance énergétique (DPE). Ces fichiers contiennent les données détaillées de chaque diagnostic.
2. Vérification de l'intégrité : Une fois qu'un fichier XML est reçu, l'API procède à la vérification de son intégrité. Cela signifie qu'elle vérifie que le fichier est complet, c'est-à-dire qu'il contient toutes les informations requises, et que ces informations sont correctes et cohérentes. Cette vérification est réalisée en comparant le contenu du fichier XML à un schéma XML (fichier XSD) qui définit la structure attendue du fichier et les types de données de chaque élément.
3. Rapport d'erreurs : Si l'API détecte une erreur ou une incohérence dans le fichier XML, elle génère un rapport d'erreurs qui est renvoyé à l'entreprise prestataire. Cela permet à l'entreprise de corriger les erreurs et de renvoyer le fichier.

4. Transmission des fichiers valides : Si le fichier XML passe la vérification d'intégrité, l'API le transmet à l'application, qui l'utilise pour alimenter l'ERP de "Maisons et Cités".

Cette fonctionnalité de l'API contribue à améliorer la qualité des données reçues par l'application et par l'ERP de "Maisons et Cités", ce qui est essentiel pour le bon fonctionnement de ces systèmes et pour la précision des informations sur les performances énergétiques des bâtiments gérés par "Maisons et Cités".

Descriptions des données et structures des données

La description des données concerne les informations contenues dans les fichiers XML qui sont traités par l'application. Ces fichiers sont générés par les entreprises prestataires qui réalisent les diagnostics de performance énergétique (DPE) pour "Maisons et Cités". Voici une description détaillée de ces données :

1. Informations sur le diagnostic : Chaque fichier XML contient des informations détaillées sur le diagnostic de performance énergétique réalisé. Cela comprend des données telles que la date du diagnostic, le type de bâtiment diagnostiqué (par exemple, résidentiel ou commercial), la localisation du bâtiment, et le nom de l'entreprise prestataire qui a réalisé le diagnostic.
2. Résultats du diagnostic : Le fichier XML contient également les résultats du diagnostic de performance énergétique. Cela comprend des informations telles que la consommation d'énergie du bâtiment, son émission de gaz à effet de serre, et sa classification énergétique (par exemple, A, B, C, etc.).
3. Recommandations : Enfin, le fichier XML peut également contenir des recommandations pour améliorer la performance énergétique du bâtiment. Cela peut

inclure des suggestions pour des travaux de rénovation, l'installation de systèmes d'énergie renouvelable, ou des changements dans les comportements d'utilisation de l'énergie.

Ces données sont essentielles pour "Maisons et Cités", car elles permettent à l'entreprise de comprendre la performance énergétique de ses bâtiments et d'identifier les opportunités pour améliorer leur efficacité énergétique. Elles sont également utilisées pour alimenter l'ERP de l'entreprise, ce qui facilite la gestion des informations sur les performances énergétiques et leur utilisation pour la prise de décisions.

Descriptif des algorithmes utilisés

Langages et Frameworks utilisés et fonctionnement

Pour réaliser ce projet, l'ancien alternant a utilisé différents langages en fonction de la couche sur laquelle il travaillait. Pour le frontend, il a utilisé le Framework ReactJS, composé de Javascript, HTML et CSS/SCSS. Pour le backend, il a utilisé le Framework Spring, basé sur Java. Ce Framework utilise également Maven pour l'exécution de certaines commandes, comme l'exécution des tests ou la construction de l'application.

Pour faire communiquer les différentes couches, il y a différents procédés selon les couches. Pour faire interagir le frontend avec le backend, des Webservices sont utilisés. Pour utiliser ces Webservices depuis l'application React, il faut utiliser la librairie Axios qui permet l'utilisation de Webservices via des requêtes GET, POST, DELETE, PUT, etc. Pour faire interagir le backend avec la base de données, on utilise une des fonctionnalités de Spring qui permet d'interroger une base de données directement depuis l'application Spring.

Base de données

Pour ce projet, on utilise une base de données PostgreSQL. Ce type de base de données était déjà utilisé par certaines applications WinDev chez "Maisons & Cités" en plus des bases de données db2 présentes pour les AS400.

L'API

Annexe 14 XmlValidator

Dans un premier temps j'ai créé une nouvelle classe nommée « XmlValidator » qui fournit une méthode statique validateXmlFile pour valider un fichier XML par rapport à un schéma XML (XSD). Voici ce que fait chaque partie du code :

1. Il importe les classes nécessaires pour la manipulation de documents XML, la validation de schémas XML et la gestion des exceptions.
2. La méthode validateXmlFile prend en entrée un fichier XML (File xmlFile).
3. Il crée une instance de DocumentBuilderFactory et utilise cette fabrique pour créer une instance de DocumentBuilder.
4. Le DocumentBuilder est utilisé pour analyser le fichier XML et créer une instance de Document.
5. Il récupère l'élément racine du document XML et extrait la valeur de l'élément "enum_version".

6. Il construit le nom du fichier XSD correspondant à la version extraite de l'élément "enum_version". Il suppose que le fichier XSD se trouve dans le répertoire "xsd" et suit le format "DPEv{version}.xsd".
7. Il tente de récupérer l'URL du fichier XSD à partir du chargeur de classes. Si le fichier XSD n'est pas trouvé, il lance une IllegalArgumentException indiquant que la version n'est pas supportée.
8. Il crée une instance de SchemaFactory et utilise cette fabrique pour créer une instance de Schema à partir du fichier XSD.
9. Il crée un Validator à partir du Schema et tente de valider le fichier XML par rapport au schéma. Si la validation échoue, il renvoie un message d'erreur indiquant que le fichier XML est incorrect lors de sa comparaison avec le XSD.
10. Si la validation réussit, il renvoie null, indiquant qu'il n'y a pas d'erreur.

En résumé, ce code est utilisé pour valider un fichier XML par rapport à un schéma XSD spécifique, déterminé par la valeur de l'élément "enum_version" dans le fichier XML.

Annexe 15 FileController

J'ai dû modifier un controller afin qu'il traite une liste de fichiers XML (files), les valide en utilisant un schéma XSD, et les stocke si la validation réussit. Voici ce que fait chaque partie du code :

1. Il crée une instance de `GitLabXsdFetcher` et appelle la méthode `fetchXsdFilesFromGitLab()`. Cette méthode est probablement utilisée pour récupérer les dernières versions des fichiers XSD à partir d'un dépôt GitLab.
2. Pour chaque fichier dans la liste `files`, il fait les opérations suivantes :
 - a. Il crée un fichier temporaire et transfère le contenu du fichier multipart dans ce fichier temporaire.
 - b. Il valide le fichier temporaire en utilisant la méthode `validateXmlFile` de la classe `XmlValidator`. Si la validation échoue, il renvoie une réponse HTTP avec le statut `BAD_REQUEST` et un message d'erreur.
 - c. Si la validation réussit, il stocke le fichier en utilisant le service `fileStorageService`.
 - d. Il ajoute le nom du fichier original à la liste `fileNames`.
 - e. Il sauvegarde l'action d'upload et le nom du fichier dans une base de données en utilisant le service `fileService`.

En résumé, ce code traite une liste de fichiers XML, les valide en utilisant un schéma XSD, et les stocke si la validation réussit. Si la validation échoue pour un fichier, il renvoie une réponse HTTP avec le statut `BAD_REQUEST` et un message d'erreur.

Cette classe `GitLabXsdFetcher` contient une méthode `fetchXsdFilesFromGitLab()` qui récupère des fichiers XSD à partir d'un dépôt GitLab. Voici ce que fait chaque partie du code :

1. Il définit l'URL de l'API GitLab et un token d'accès pour interagir avec l'API.
2. Il envoie une requête GET à l'API GitLab en utilisant le token d'accès. La réponse est attendue au format JSON.
3. Si le statut de la réponse est 200 (ce qui signifie que la requête a réussi), il procède à l'extraction des fichiers XSD.
4. Il récupère le chemin absolu du dossier "xsd/" dans le classpath.
5. Il parcourt chaque élément dans le corps de la réponse. Chaque élément représente un fichier dans le dépôt GitLab.
6. Pour chaque fichier, si le nom du fichier se termine par ".xsd" (ce qui signifie qu'il s'agit d'un fichier XSD), il récupère l'URL du fichier.
7. Il ouvre un flux d'entrée à partir de l'URL du fichier et copie le contenu du flux dans un fichier local dans le dossier "xsd/". Si un fichier avec le même nom existe déjà, il est remplacé.

En résumé, cette méthode récupère tous les fichiers XSD à partir d'un dépôt GitLab spécifique et les stocke localement dans le dossier "xsd/".

Les missions

Le projet de gestion des diagnostics, pour le moment, l'application permet uniquement de recueillir les diagnostics DPE. Cependant, ce ne sont pas les seuls diagnostics qu'il faudra recueillir. Pour cela, je devrai effectuer une évolution de l'application. Cela signifie qu'il faudra intervenir sur l'application elle-même, dans la partie frontend et backend, modifier la base de données PostgreSQL et faire des changements dans les services liés (Blueway, qui est un Enterprise Service Bus, et Xemi, qui est un gestionnaire électronique de documents). Différents bugs sont apparus, empêchant la livraison de l'application à temps. J'ai dû résoudre ces bugs avant d'envisager de faire des améliorations afin de recueillir d'autres diagnostics. En première amélioration, il faut intégrer une API qui vérifie l'intégrité des fichiers XML. Dans la rubrique suivante, je vais vous expliquer les différents problèmes rencontrés.

Les bugs que j'ai résolus

Le premier obstacle était une erreur d'un service Blueway ([Annexe 9 à 13](#)). J'ai donc débogué ce service et j'ai constaté que les fichiers étaient déposés sur deux serveurs (serveur backend) mais que les chemins d'arborescence des fichiers étaient différents entre les deux. J'ai donc modifié l'arborescence des fichiers d'un serveur pour la rendre identique à celle du second grâce aux commandes de navigation Linux.

```
cd (change directory)
cp chemin (vers le répertoire dont le chemin absolu est donné)
cd .. (répertoire parent)
cd ~ (répertoire de base)
cd - (répertoire précédent)
cd / (répertoire racine)
mkdir (créer un répertoire)
mkdir répertoire
rm (remove, efface!!!)
rm -R (enlèvement récursif!!!)
rm fichier
rm -i fichier (interactivement, avec demande de confirmation)
rm -f fichier (avec force, sans demande de confirmation)
```

`rm -r` fichier (avec récursivité, avec les sous répertoires)

`rm -rf` dossier (supprime le répertoire et tout son contenu, sans confirmation)

Ensuite, nous avons eu un problème de redondance des serveurs. Le fonctionnement de l'application repose sur trois serveurs. Un premier qui sert à stocker le frontend (l'affichage du site web) et les deux derniers qui stockent le backend (partie du code qui fait les contrôles, l'envoi vers la base de données, l'analyse des documents, etc.). Le deuxième serveur n'était pas configuré, ce qui générait une erreur CORS. Pour résoudre cela, j'ai dû dans un premier temps identifier la cause de ce bug. J'ai donc lancé l'application en local, puis sur l'environnement de test, puis sur l'environnement de recette. Le bug n'était pas présent, j'ai donc redéployé l'application sur le serveur de production, mais l'erreur était toujours présente. J'ai donc déduit que cela ne venait pas du code. Par la suite, avec un administrateur réseau, nous avons forcé l'application à pointer sur un serveur backend et là, le problème était résolu. Mais il fallait à tout prix avoir les deux serveurs backend pour la redondance. J'ai donc fait une note de mes besoins pour l'application et je l'ai transmise à l'administrateur réseau qui s'est chargé de régler le problème avec le serveur.

([Annexe 11](#) - [12](#)) Enfin, nous avons découvert plusieurs bugs liés aux services développés sur l'ESB Blueway. Lorsque le traitement Blueway était configuré pour fonctionner sur l'organisme GMC, les fichiers PDF des DPE étaient déposés sur l'organisme FMC dans la GED XEMI (GMC et FMC sont des environnements). Le fichier qui permet de définir l'environnement et de contenir des données interprétables pour la GED, car un PDF ne contient pas des données interprétables pour la GED, est un fichier .IXX. Il contenait une valeur brute "EX-PDIC01" si je configurais l'environnement sur FMC et "EXPDIC05" si je configurais l'environnement sur GMC. J'ai donc décidé de créer une variable d'environnement afin de pouvoir changer plus facilement celle-ci et donc cette variable d'environnement est définie sur EXPDIC05 pour GMC.

Nous avons également découvert un autre bug. Suite à des modifications précédentes, la date n'était plus transmise dans XEMI. Il a donc fallu récupérer la date et la mettre dans le format correct pour que le gestionnaire électronique de documents puisse l'interpréter.

Une autre difficulté est apparue. Il existe plusieurs versions de DPE. Les DPE changent si le logement est dit neuf ou non, donc les DPE de logement neuf n'ont pas été intégrés dans IKOS. Pour pallier ce problème, il y a une valeur dans le fichier .xml qui désigne si le fichier traite un logement neuf ou non, cette valeur est appelée « enum_model_dp_id » si elle est égale à un, c'est un logement ancien si sa valeur est deux, c'est un logement neuf. Donc, dans le service « DPE_XML_TO_BD » nous devons créer une condition IF afin d'ajuster le code selon les versions enum.

Dû à des versions différentes de DPE, certains fichiers XML sont différents. Donc, certains noms de variable diffèrent. Pour contrer ça, j'ai donc ajouté une condition IF qui va vérifier si la variable existe sinon elle pointerait sur l'autre.

En conclusion, le projet de gestion des diagnostics a été une expérience enrichissante, malgré les défis rencontrés. Les bugs ont été résolus et l'application est maintenant prête pour l'intégration de nouvelles fonctionnalités, notamment l'ajout d'autres types de diagnostics et l'intégration d'une API pour vérifier l'intégrité des fichiers XML.

Protocoles utilisés

Dans le cadre de ce projet, plusieurs protocoles ont été utilisés pour assurer la communication entre les différentes parties de l'application et pour garantir la sécurité des données.

1. HTTP/HTTPS: Ces protocoles ont été utilisés pour la communication entre le client (navigateur web) et le serveur. HTTP (Hypertext Transfer Protocol) est le protocole utilisé pour le transfert de données sur le web. Cependant, pour des raisons de sécurité, nous avons principalement utilisé HTTPS (HTTP Secure), qui est la version

sécurisée de HTTP. HTTPS chiffre les données envoyées entre le client et le serveur, ce qui empêche les attaques de type "man-in-the-middle".

2. FTP/SFTP: Le protocole FTP (File Transfer Protocol) a été utilisé pour le transfert de fichiers entre les différents serveurs. Tout comme pour HTTP, nous avons privilégié l'utilisation de SFTP (SSH File Transfer Protocol), qui est une version sécurisée de FTP, pour garantir la confidentialité et l'intégrité des fichiers transférés.
3. SSH: SSH (Secure Shell) est un protocole qui permet d'établir une connexion sécurisée entre un client et un serveur. Il a été utilisé pour l'administration à distance des serveurs, permettant d'exécuter des commandes et de gérer les fichiers sur les serveurs à distance.
4. SQL: SQL (Structured Query Language) est le langage utilisé pour communiquer avec la base de données PostgreSQL. Il permet de créer, de lire, de mettre à jour et de supprimer des données dans la base de données.
5. REST: REST (Representational State Transfer) est un style d'architecture pour les services web. Il a été utilisé pour la communication entre le frontend et le backend de l'application. Les services web RESTful utilisent les méthodes HTTP standard (GET, POST, PUT, DELETE, etc.) pour effectuer des opérations sur les ressources.

Ces protocoles ont joué un rôle crucial dans le bon fonctionnement de l'application, en assurant une communication fluide et sécurisée entre les différentes parties de l'application.

Aspects liés à la sécurité (si applicable)

La sécurité est un aspect crucial de toute application, en particulier lorsqu'elle traite des données sensibles. Dans le cadre de ce projet, plusieurs mesures ont été prises pour garantir la sécurité des données et des communications.

1. HTTPS: Comme mentionné précédemment, HTTPS a été utilisé pour sécuriser la communication entre le client et le serveur. Il garantit que les données sont chiffrées lors de leur transfert, empêchant ainsi leur interception par des tiers malveillants.
2. SFTP: Pour le transfert de fichiers entre les serveurs, nous avons utilisé SFTP, qui assure le chiffrement des données lors du transfert.
3. Authentification et autorisation: L'application dispose d'un système d'authentification qui nécessite un nom d'utilisateur et un mot de passe pour accéder à l'interface utilisateur ou à l'interface administrateur. De plus, un système d'autorisation a été mis en place pour garantir que chaque utilisateur n'a accès qu'aux ressources qui lui sont autorisées.
4. Gestion des mots de passe: Les mots de passe des utilisateurs sont stockés de manière sécurisée. Ils sont hashés avant d'être stockés dans la base de données, ce qui signifie qu'il est pratiquement impossible de les retrouver à partir du hash. De plus, un système de réinitialisation du mot de passe est en place, permettant aux utilisateurs de définir un nouveau mot de passe via un lien reçu par mail et un token unique généré pour cela.
5. Validation des entrées: Pour prévenir les attaques par injection SQL, toutes les entrées de l'utilisateur sont validées avant d'être utilisées dans une requête SQL. Cela empêche les attaquants d'injecter du code malveillant dans les requêtes.
6. Pare-feu: Un pare-feu a été mis en place pour protéger l'application contre les attaques non autorisées. Il surveille et contrôle le trafic entrant et sortant selon des règles de sécurité prédéfinies.

7. Mises à jour régulières: Les serveurs et les logiciels utilisés pour l'application sont régulièrement mis à jour pour garantir qu'ils sont protégés contre les dernières vulnérabilités connues.

Ces mesures de sécurité contribuent à garantir que l'application reste sécurisée et que les données des utilisateurs sont protégées.

Conclusion

Au cours de cette alternance chez Maisons et Cités, j'ai eu l'opportunité de travailler sur un projet d'application web complexe, qui m'a permis d'acquérir et de renforcer mes compétences techniques et professionnelles. J'ai pu me familiariser avec des technologies telles que Java, Spring, ReactJS, PostgreSQL, et des outils collaboratifs comme GitHub, VSCode et la suite Office. J'ai également eu l'occasion de comprendre et de mettre en pratique des concepts importants de la sécurité informatique.

La maintenance et l'amélioration de l'application, notamment par l'intégration d'une API pour vérifier l'intégrité des fichiers XML, ont été des défis stimulants qui m'ont permis de développer mon sens de la résolution de problèmes et ma capacité à travailler de manière autonome tout en faisant partie d'une équipe.

En outre, j'ai pu comprendre le fonctionnement d'une grande organisation comme Maisons et Cités, et comment le service informatique soutient les objectifs de l'entreprise. J'ai également appris à naviguer dans le milieu professionnel, à communiquer efficacement avec mes collègues et à gérer mon temps et mes responsabilités.

En conclusion, cette expérience a été extrêmement enrichissante et formatrice. Elle m'a permis de mettre en pratique les connaissances acquises lors de ma formation en Licence Pro

DIOC, et de me préparer à entrer dans le monde professionnel en tant que développeur. Je suis impatient de continuer à apprendre et à développer mes compétences dans mes futures expériences professionnelles.

Annexes

Liste des abréviations et sigles utilisés

API : Application Programming Interface

DPE : Diagnostic de Performance Énergétique

GED : Gestionnaire Électronique de Documents

ERP : Enterprise Resource Planning

XML : eXtensible Markup Language

PDF : Portable Document Format

VSCode : Visual Studio Code

ESB : Enterprise Service Bus

SQL : Structured Query Language

SSH : Secure Shell

FTP/SFTP : File Transfer Protocol/SSH File Transfer Protocol

HTTP/HTTPS : Hypertext Transfer Protocol/HTTP Secure

CORS : Cross-Origin Resource Sharing

REST : Representational State Transfer

AS400 : Système d'exploitation et serveur d'IBM

XEMI : Gestionnaire électronique de documents

GMC et FMC : Environnements de l'application

IXX : Type de fichier pour la GED

XSD : XML Schema Definition

SCSS : Sassy CSS

ReactJS : Bibliothèque JavaScript pour construire des interfaces utilisateur

Spring : Framework de développement d'applications Java

Java : Langage de programmation orienté objet

PostgreSQL : Système de gestion de base de données

Rappel du formalisme

Dans le cadre de ce rapport technique, il est important de rappeler certains formalismes et conventions utilisés tout au long du projet. Ces formalismes sont essentiels pour comprendre le fonctionnement de l'application, les technologies utilisées et les méthodes de développement et de test.

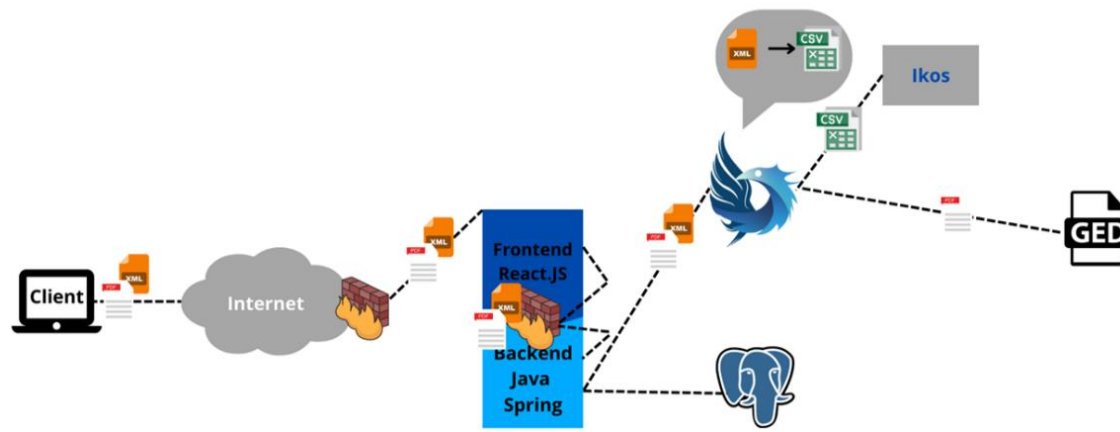
1. Langages de programmation : L'application a été développée en utilisant deux langages de programmation principaux : Java pour le backend et JavaScript pour le frontend. Java est un langage de programmation orienté objet largement utilisé pour le développement d'applications d'entreprise. JavaScript, en combinaison avec le framework ReactJS, a été utilisé pour créer l'interface utilisateur de l'application.
2. Frameworks : Le développement de l'application a été facilité par l'utilisation de deux frameworks principaux : Spring pour le backend et ReactJS pour le frontend. Spring est un framework Java qui fournit un modèle de programmation complet et une configuration par défaut pour les applications d'entreprise. ReactJS est une bibliothèque JavaScript pour la construction d'interfaces utilisateur.

3. Base de données : PostgreSQL a été utilisé comme système de gestion de base de données. C'est un système de base de données relationnelle open source puissant et extensible.
4. Protocoles de communication : L'application utilise plusieurs protocoles de communication pour assurer le transfert de données entre le client et le serveur, et entre les différents serveurs. Ces protocoles incluent HTTP/HTTPS pour la communication client-serveur, FTP/SFTP pour le transfert de fichiers entre les serveurs, et SQL pour la communication avec la base de données.
5. Méthodes de test : Les tests ont joué un rôle crucial dans le développement de l'application. Les tests unitaires ont été réalisés en utilisant JUnit, un framework de test pour Java. Les tests d'intégration et de système ont également été effectués pour s'assurer que l'application fonctionne correctement dans son ensemble.
6. Outils de versioning : Git a été utilisé comme système de contrôle de version pour le code source de l'application. Cela a permis de suivre les modifications apportées au code, de gérer les différentes versions de l'application et de faciliter la collaboration entre les membres de l'équipe.
7. Outils collaboratifs : Plusieurs outils collaboratifs ont été utilisés tout au long du projet, notamment GitHub pour le partage de code et la revue de code, VSCode pour l'édition de code, et la suite Office pour la rédaction de documents et la communication.

Ces formalismes et conventions ont été essentiels pour le développement, la maintenance et l'amélioration de l'application. Ils ont permis une communication efficace, une coordination fluide et un développement de qualité de l'application.

Autres annexes pertinentes

Annexe 1 schéma organisation de l'application :



Annexe 2 page login :



Bienvenue sur l'interface d'intégrations des « Diagnostic DPE »

Veuillez vous connecter :

Identifiant

Mot de passe

[Connexion](#)

[Mot de passe oublié](#)

Annexe 3 Mot de passe oublié:



Mot de passe oublié

✉ Email / Courriel

✓ Envoyer

Annexe 4 réinitialiser le mot de passe :



Nouveau mot de passe

🔒 •••••••••• 🔍

🔒 Confirmer le mot de passe


Confirmer

Critères à respecter :

- ✓ Au moins 12 caractères
- ✗ Au moins 1 majuscule
- ✓ Au moins 1 minuscule
- ✗ Au moins 1 caractère spéciales parmi - _ & # /
- ✗ Au moins 1 chiffre

Annexe 5 dashboard utilisateur :

Déconnexion



Bonjour, Remi.David@maisonsetcites.fr

Liste des Fichiers déposés :

Nom du document	Type	Numéro Adresse	Numéro UO	Date
DPE_IND_132507_2262E0486374D_20220311	pdf	2262E0486374D	132507	2022-05-04
DPE_IND_132507_2262E0486374D_20220311	xml	2262E0486374D	132507	2022-05-04

1 à 2 sur 2 < > Page 1 sur 1 >


Demander d'ajout ou suppression

Ajouter un diagnostic

Glisser déposer ou cliquer pour ajouter un pdf et un xml

Annexe 6 dashboard administrateur :

Déconnexion



Bonjour, Remi.David@maisonsetcites.fr

Liste des Fournisseurs :

Entreprise	Dernière mise à jour	Nombre de Diag
Apave		0
Exim		0
Socotec		0
Alto Diagnostic		0
AC Environnement		0
ETI Diag		0
Maisons et Cités	2022-05-04	4

1 à 7 sur 7 < > Page 1 sur 1 >

Ajouter un fournisseur

Liste des demandes d'ajout/suppression utilisateur :

Type de demande	Demande	État de la demande
Ajout	Voir la demande	Validé
Suppression	Voir la demande	Validé
Ajout	Voir la demande	En attente Valider Refuser

1 à 3 sur 3 < > Page 1 sur 1 >

Annexe 7 affichage des fournisseurs :

[Retour](#)
[Déconnexion](#)

Bonjour, Remi.David@maisonsetcites.fr

Maisons et Cités

Liste des comptes utilisateurs :

Prénom NOM	Téléphone	Email	Action
Rémi DAVID	0123456789	Remi.David@maisonsetcites.fr	Supprimer ?

[Ajouter un utilisateur](#)

Liste des Fichiers déposés :

Prénom NOM	Nom du document	Type	Adresse	Date	UG
Rémi DAVID	DPE_IND_132507_2262E0486374D_2...	pdf	2262E0486374D	2022-05-04	132507
Rémi DAVID	DPE_IND_132507_2262E0486374D_2...	xml	2262E0486374D	2022-05-04	132507
Ancien Utilisateur	DPE_IND_132507_2262E0486374D_2...	pdf	2262E0486374D	2022-04-28	132507
Ancien Utilisateur	DPE_IND_132507_2262E0486374D_2...	xml	2262E0486374D	2022-04-28	132507

1 à 4 sur 4 Page 1 sur 1

Annexe 8 page forbidden err403 :



403 - Vous ne passerez pas

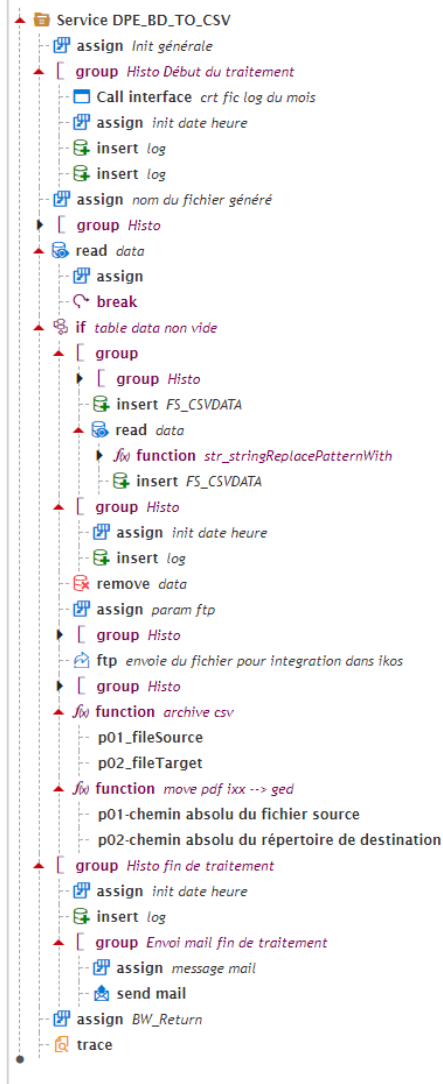
Uh oh, Gandalf bloque le passage !
 Peut-être essayez-vous d'accéder à une page sur laquelle
 vous n'avez pas les droits?

[Retour à la page de connexion](#)

Annexe 9 service Blueway d'extraction des données des xml :



Annexe 10 Service Blueway de génération du csv :



Annexe 11 Extrait du code ixm :

Formule

```

=<?xml version="1.0" encoding="ISO-8859-1" standalone="yes">
<XDOC>
  <Indexation>
    <Document Final="false" Type="" +BW_FTP_IKOS_DIAG.Type EXPDIC-"" Department="D01"
    NumberOfPages="" +nb_page+"" DocumentNumber="222350001" CreationUser="XDOCADM">
      <Description>Diagnostic DPE - UG " +num_ug+""</Description>
    </Document>
    <Keywords>
      <Add Type="GNDO" >EXPDIC</Add>
      <Add Type="CTNHP" >+hp+</Add>
      <Add Type="DTGEN" >+dateFormatted+</Add>
      <Add Type="NUMUG" >+num_ug+</Add>
    </Keywords>
    <Folders>
      <Add FolderId="ENV" +hp+"" CheckExistence="">
        <CreateOrUpdateFolder State="A" Date="" +str_currentDateFormat("yyyyMMdd")+"">
          <Description1>I4142DPE</Description1>
          <Description2>I4143"+str_upperCase(ville)+"</Description2>
          <Description3>I4A0F</Description3>
          <Description4>ENV-Environnement</Description4>
          <SearchText1>ENV"+hp+""</SearchText1>
        </CreateOrUpdateFolder>
      </Add>
    </Folders>
  </Indexation>
</XDOC>

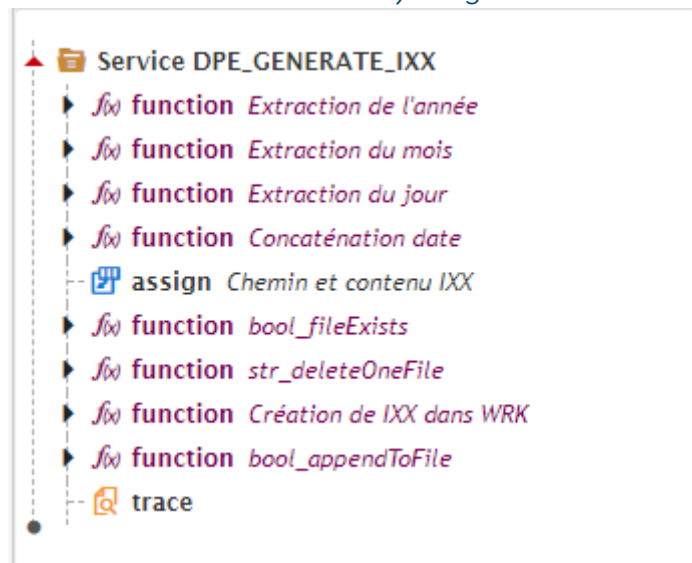
```

Analyser

Valider

Annuler

Annexe 12 Service Blueway de génération du ixm :



Annexe 13 variables d'environnement :

Service DPE_VML_TO_SD - 5 * Service DPE_GENERATE_JOX - 2 *

Conte

Type de l'objet : Variable d'environnement

Nom : BW_FTP_MCS_DIAG

Désignation :

Catégorie : Global

Nom	Crypté	Type	developpement	recette	production
FTP_SRV_URL	<input type="checkbox"/>	Alphanumeric	DEVIL	DEVIL	SOA-INT
FTP_SRV_REP	<input type="checkbox"/>	Alphanumeric	CONFIDENTIAL	CONFIDENTIAL	CONFIDENTIAL
FTP_SRV_USER	<input type="checkbox"/>	Alphanumeric	CONFIDENTIAL	CONFIDENTIAL	CONFIDENTIAL
FTP_SRV_PWD	<input type="checkbox"/>	Alphanumeric	CONFIDENTIAL	CONFIDENTIAL	CONFIDENTIAL
BW_LOCAL_PATH_CSV	<input type="checkbox"/>	Alphanumeric	CONFIDENTIAL	CONFIDENTIAL	CONFIDENTIAL
BW_LOCAL_TRAITES	<input type="checkbox"/>	Alphanumeric	CONFIDENTIAL	CONFIDENTIAL	CONFIDENTIAL
BW_GED	<input type="checkbox"/>	Alphanumeric	CONFIDENTIAL	CONFIDENTIAL	CONFIDENTIAL
Type_EXPDOC	<input type="checkbox"/>	Alphanumeric	EXPDOC	EXPDOC	EXPDOC

Annexe 14 XmlValidator

```

1  package fr.maisonsetcites.integrationdiag.utils;
2
3  import org.w3c.dom.Document;
4  import org.w3c.dom.Element;
5  import org.xml.sax.SAXException;
6
7  import javax.xml.XMLConstants;
8  import javax.xml.parsers.DocumentBuilder;
9  import javax.xml.parsers.DocumentBuilderFactory;
10 import javax.xml.parsers.ParserConfigurationException;
11 import javax.xml.transform.stream.StreamSource;
12 import javax.xml.validation.Schema;
13 import javax.xml.validation.SchemaFactory;
14 import javax.xml.validation.Validator;
15 import java.io.File;
16 import java.io.IOException;
17 import java.net.URL;
18
19 public class XmlValidator {
20
21     public static String validateXmlFile(File xmlFile) throws ParserConfigurationException, IOException, SAXException {
22         DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newInstance();
23         DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBuilder();
24         Document document = documentBuilder.parse(xmlFile);
25         Element rootElement = document.getDocumentElement();
26         String enumVersion = rootElement.getElementsByTagName("name:enum_version").item(0).getTextContent();
27
28         String xsdFileName = "xsd/DPEv" + enumVersion + ".xsd";
29         URL xsdFileUrl = XmlValidator.class.getClassLoader().getResource(xsdFileName);
30
31         if (xsdFileUrl == null) {
32             throw new IllegalArgumentException("Version non supportée : " + enumVersion);
33         }
34
35         File xsdFile = new File(xsdFileUrl.getFile());
36         SchemaFactory schemaFactory = SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);
37         Schema schema = schemaFactory.newSchema(xsdFile);
38         Validator validator = schema.newValidator();
39
40         try {
41             validator.validate(new StreamSource(xmlFile));
42         } catch (SAXException e) {
43             return "Erreur : le fichier " + xmlFile.getName()
44                 + " est incorrect lors de sa comparaison avec le XSD. Veuillez revoir sa structure.";
45         }
46
47         return null;
48     }
49 }
50

```

Annexe 15 FileController

```
import lombok.AllArgsConstructor;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.multipart.MultipartFile;

import javax.servlet.http.HttpServletRequest;
import java.util.*;
import java.io.File;

import static org.springframework.http.HttpHeaders.AUTHORIZATION;

@RestController
@RequestMapping("/api/v1/files")
@AllArgsConstructor
@CrossOrigin(origins = "http://localhost:3000")
// Définition des API et fonctions
public class FileController {

    private final FileStorageService fileStorageService;
    private final FileService fileService;

    @PostMapping("/user/uploadDPE")
    // fonction d'ajout de fichier
    public ResponseEntity<FileResponse> uploadDpe(@RequestParam("file") MultipartFile[] files,
        HttpServletRequest request) {
        try {
            List<String> fileNames = new ArrayList<>();

            // On récupère le token présent dans le header de la requête
            String authorizationHeader = request.getHeader(AUTHORIZATION);
            String token = authorizationHeader.substring("Bearer ".length());
            // On décrypte le token pour en extraire les informations
            Algorithm algorithm = Algorithm.HMAC256("secret".getBytes());
            JWTVerifier verifier = JWT.require(algorithm).build();
            DecodedJWT decodedJWT = verifier.verify(token);
            // On récupère le nom d'utilisateur de celui qui a envoyé la requête
            String username = decodedJWT.getSubject();

            // Créez une instance de GitLabXsdFetcher et appelez fetchXsdFilesFromGitLab()
            GitLabXsdFetcher gitLabXsdFetcher = new GitLabXsdFetcher();
            gitLabXsdFetcher.fetchXsdFilesFromGitLab();

            // Pour chaque fichier...
            for (MultipartFile file : files) {
                // Valider le fichier XML
                File tempFile = File.createTempFile(prefix:"temp", suffix:".xml");
                file.transferTo(tempFile);
                String validationResult = XmlValidator.validateXmlFile(tempFile);
                if (validationResult != null) {
                    // Si la validation échoue, retournez le message d'erreur
                    return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(new FileResponse(validationResult));
                }

                // ... on stocke le fichier
                fileStorageService.storeFile(file);
                // ... on ajoute le fichier dans le tableau à retourner dans la réponse de la
                // requête
                fileNames.add(file.getOriginalFilename());
                // ... on sauvegarde l'action et le fichier dans la bdd
                fileService.saveUploadDpe(username, Objects.requireNonNull(file.getOriginalFilename()));
            }

            String message = "Uploaded the files successfully: " + fileNames;

            return ResponseEntity.status(HttpStatus.OK).body(new FileResponse(message));
        } catch (Exception e) {
            String message = "Fail to upload files!";
            return ResponseEntity.status(HttpStatus.EXPECTATION_FAILED).body(new FileResponse(message));
        }
    }

    @GetMapping("/admin/getAllDataByFournisseur/{name}")
    public List<Map<String, String>> getAllDataByFournisseur(@PathVariable("name") String name) {
        return fileService.getAllDataByFournisseur(name);
    }

    @GetMapping("/user/getDataByUsername/{email}")
    public List<fr.maisonsetcites.integratiodiag.file.File> getDataByUsername(@PathVariable("email") String email) {
```

Annexe 16 GitLabXsdFetcher

```

1  package fr.maisonsetcites.integrationdiag.utils;
2
3  import kong.unirest.HttpResponse;
4  import kong.unirest.JsonNode;
5  import kong.unirest.Unirest;
6  import org.json.JSONObject;
7  import org.springframework.core.io.ClassPathResource;
8
9  import java.io.IOException;
10 import java.io.InputStream;
11 import java.net.URL;
12 import java.nio.file.Files;
13 import java.nio.file.Paths;
14 import java.nio.file.StandardCopyOption;
15
16 public class GitLabXsdFetcher {
17
18     public void fetchXsdFilesFromGitLab() throws IOException {
19         String gitLabApiUrl = "https://gitlab.com/api/v4/projects/observatoire-dpe%2Fobservatoire-dpe/repository/tree?ref=main";
20         String gitLabToken = " "; // Veuillez remplacer ceci par votre propre token d'accès
21                                     // GitLab
22
23         HttpResponse<JsonNode> response = Unirest.get(gitLabApiUrl)
24             .header(name: "Private-Token", gitLabToken)
25             .asJson();
26
27         if (response.getStatus() == 200) {
28             String xsdFolderPath = new ClassPathResource("xsd/").getFile().getAbsolutePath();
29             response.getBody().getArray().forEach(item -> {
30                 JSONObject itemJson = (JSONObject) item;
31                 String fileName = itemJson.getString("name");
32                 if (fileName.endsWith(suffix: ".xsd")) {
33                     String fileUrl = itemJson.getString("web_url").replace(target: "blob", replacement: "raw");
34
35                     try (InputStream inputStream = new URL(fileUrl).openStream()) {
36                         Files.copy(inputStream, Paths.get(xsdFolderPath, fileName),
37                             StandardCopyOption.REPLACE_EXISTING);
38                     } catch (IOException e) {
39                         e.printStackTrace();
40                     }
41                 }
42             });
43         }
44     }
45 }

```