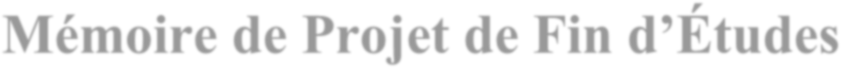




**Master BIG DATA et Internet des Objets**



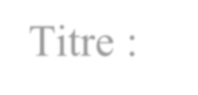
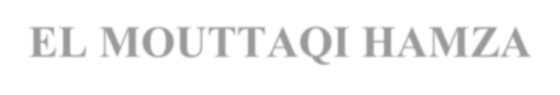


**Mémoire de Projet de Fin d’Études**

##### Présenté en vue de l’obtention du Master de Recherche



**Spécialité : Big Data et Internet des Objets**



  **EL MOUTTAQI HAMZA**  

Titre :

**Refonte et développement de l’IHM Centre de notification pour la gestion des notifications SFR**



Société d’Accueil: *INTELCIA IT SOLUTIONS*

##### Encadré par :

* Pr. Abdelali Zakrani : Professeur à l’ENSAM de Casablanca
* M.Aissam Cheikh : Manager à INTELCIA IT SOLUTIONS

##### Soutenu le : 05/07/2022 devant le jury composé de :

* Pr. Mohamed Moutchou : Professeur à l’ENSAM de Casablanca Président
* Pr. Said Bahassine : Professeur à l’ENSAM de Casablanca Examinateur
* Pr. Abdelali Zakrani : Professeur à l’ENSAM de Casablanca Encadrant

# Avant-Propos

Nom et prénom de l’élève ingénieur stagiaire de l’ENSAM :

##### Mr EL MOUTTAQI HAMZA

Intitulé du travail :

##### Refonte et développement de l’IHM Centre de notification pour la gestion des notifications SFR

Établissement d’accueil :

##### Intelcia IT Solutions, Imm. InTown Bd Zoulikha Naciri, 20270 / Casablanca / Maroc.

Tél : **+212 (0) 6 89 23 55 02**

Coordonnées de l’école :

##### Ecole Nationale Supérieure des arts et métiers 150 Bd du Nil, Casablanca 20670

Nom et prénom de l’encadrant du projet au sein de l’établissement d’accueil :

##### Mr. AISSAM CHEIKH

Nom et prénom du directeur du projet à l’ENSAM :

##### Mr. ZAKRANI ABDELALI

Date de début et de fin de stage :

**Du 13/03/2023 au 13/09/2023**

# Dédicace

“

**Je dédie humblement ce travail avec un amour profond, en invoquant le nom de Dieu, le Tout Miséricordieux, le Très Miséricordieux.**

**À ma très chère mère, qui est ma source d'amour et qui m'entoure de tendresse et d'espoir depuis ma naissance, je ne saurais exprimer suffisamment ma gratitude pour**

**ta présence à mes côtés.**

**À mon cher père, ma force et mon soutien, je te remercie pour tes sacrifices et tes efforts inlassables pour nous offrir le meilleur.**

**À toute ma famille et mes chers ami(e)s, je leur suis reconnaissant(e) pour leur soutien permanent, leurs encouragements et leur appui tout au long de mon parcours**

**universitaire.**

**Je suis également reconnaissant envers mes chers professeurs pour leur serviabilité, leur soutien, leur compréhension et leurs efforts qui méritent mon plus**

**profond respect.**

**Que ce travail soit le reflet de ma gratitude et de mon affection envers tous ceux qui m'ont entouré de leur amour et de leur soutien.**

,,

- **Hamza**

# Remerciement

Au terme de ce travail, j’exprime mes vifs remerciements à toute personne ayant contribué, de près ou de loin, à la réalisation de ce travail.

Je tiens à remercier tout d’abord les professeurs de ENSAM de Casablanca, notamment mes professeurs de l’équipe Master Big data et internet des objets et qui n’ont ménagé aucun effort afin de nous assurer ce parcours et nous transmettre leurs connaissances, leur expérience et leurs valeurs.

Ma profonde gratitude s’adresse à l’entreprise « Intelcia It Solutions » au nom de mon manager AISSAM CHEIKH qui m’a accueilli et qui a dirigé mon projet de fin d’études, pour son engagement permanent, son soutien constant et la confiance totale qu’il m’a accordée.

J’adresse aussi mes profonds remerciements à Monsieur ABDELALI ZAKRANI, mon encadrant pédagogique à la ENSAM Casablanca, pour son encadrement, son assistance, pour tout le temps précieux qu’il m’a octroyé et pour ses conseils qui m’ont été bien utiles pour la rédaction de ce rapport.

Je tiens à remercier également toute l’équipe de travail chez Intelcia It Solutions, Monsieur EL OUARDY ANAS, Monsieur YeYe Nailey, Monsieur Abdelmoughit RABIA, Yassine ZAKARI, pour le temps qu’ils m’ont consacré durant cette période et pour le partage de leur expertise et connaissance. C’est grâce à leur confiance et leur encouragement que j’ai été apte à accomplir mes missions et à surmonter toute difficulté rencontrée tout en gagnant de l’expérience

Par ailleurs, je souhaite adresser mes remerciements aux membres du jury pour leur présence lors de l'évaluation de mon rapport de stage. Leur expertise et leurs commentaires constructifs m'ont permis de prendre du recul sur mon travail et de l'améliorer.

# Résumé

Ce rapport présente le travail que nous avons accompli dans le cadre de notre projet de fin d'études au sein de la société Intelcia IT Solutions. L'objectif principal de notre projet était de contribuer à la refonte du l’IHM de centre de notification, qui gère la configuration des notifications pour la diffusion dans des multiples canaux

Notre participation au projet était axée sur la refonte du IHM CDN, en créant une version conforme aux besoins du client SFR et en corrigeant les problèmes présents dans la version précédente.

Pour assurer le bon déroulement du projet et nous situer dans son contexte, nous avons choisi d'adopter le framework SCRUM, largement utilisée pour la gestion de développement de produits complexes. SCRUM fournit un cadre pour l'application de divers processus et techniques de développement, mettant en évidence l'efficacité des pratiques de gestion et de développement en place, afin de les améliorer continuellement.

Au cours du développement, nous avons effectué des revues de code et des qualifications de la solution pour garantir son bon fonctionnement. Nous avons utilisé un ensemble de technologies et d'outils, tels que les frameworks Spring Boot 3, Angular, Maven, GitLab, Git et Kubernetes.

**Mots clés**: CDN, Angular, GIT, SCRUM, Springboot, GitLab.

# Abstract

This report presents the work we have accomplished as part of our end-of- studies project within Intelcia IT Solutions. The main objective of our project was to contribute to the redesign of the notification center's user interface (UI), which manages notification configuration for broadcasting across multiple channels.

Our involvement in the project was focused on redesigning the CDN UI, creating a version that meets the requirements of the client SFR, and addressing the issues present in the previous version.

To ensure the smooth progress of the project and to provide context, we chose to adopt the SCRUM methodology, widely used for managing the development of complex products. SCRUM provides a framework for implementing various development processes and techniques, highlighting the effectiveness of existing management and development practices in order to continuously improve them.

During the development process, we conducted code reviews and solution qualifications to ensure its proper functioning. We utilized a set of technologies and tools, such as Spring Boot 3, Angular, Maven, GitLab, Git, and Kubernetes.

Keywords: CDN, Angular, Git, SCRUM, Spring Boot, GitLab.

# Liste des abréviations

|  |  |
| --- | --- |
| Abréviation | Désignation |
| B2B | **Business to Business** |
| CDN | **Centre de notification** |
| CI | **Continuous Integration** |
| DMC | **Diffusion multi canaux** |
| HTML | **Hypertext Markup Language** |
| HTTP | **Hypertext Transfer Protocol** |
| IDE | **Integrated Development Environment** |
| IHM | **Interface Homme-Machine** |
| JSON | **JavaScript Object Notation** |
| REST | **Representational State Transfer** |
| SAV | **Service Après-Vente** |
| SSA | **Sous système applicatif** |
| UML | **Unified Modeling Language** |

# Liste des Figures

[FIGURE 1 LOGO INTELCIA 4](#_bookmark13)

[FIGURE 2 PRESENCE A L'INTERNATIONAL 5](#_bookmark14)

[FIGURE 3 HISTOIRE DU GROUPE INTELCIA 6](#_bookmark15)

[FIGURE 4 STRUCTURE DE LA SOCIETE INTELCIA 7](#_bookmark17)

[FIGURE 5 LOGO INTELCIA IT SOLUTIONS 9](#_bookmark20)

[FIGURE 6 ORGANIGRAMME DE POLE IT SOLUTIONS 10](#_bookmark22)

[FIGURE 7 LE POLE D'EXPERTISE IT D'INTELCIA 11](#_bookmark24)

[FIGURE 8 LOGO SFR 11](#_bookmark26)

[FIGURE 9 PRINCIPALES REFERENCES DU GROUPE INTELCIA 12](#_bookmark29)

[FIGURE 10 : PLEIADE SERVICE 14](#_bookmark32)

[FIGURE 11 : ANCIENNE IHM 16](#_bookmark35)

[FIGURE 12 : VOLUME DES NOTIFICATIONS DIFFUSEES 17](#_bookmark37)

[FIGURE 13 CYCLE DE VIE D'UN SPRINT DE LA METHODE SCRUM 19](#_bookmark40)

[FIGURE 14 : DEMARCHE DU REFONTE 22](#_bookmark44)

[FIGURE 15 : EXEMPLE DE JIRA 23](#_bookmark46)

[FIGURE 16 : JIRA DU 3EME SPRINT 23](#_bookmark47)

[FIGURE 17 : JIRA DU 4EME SPRINT 24](#_bookmark48)

[FIGURE 18 : JIRA DU 5EME SPRINT 25](#_bookmark49)

[FIGURE 19 : LOGO UML 27](#_bookmark53)

[FIGURE 20 : DIAGRAMME DE CAS D'UTILISATIONS GENERAL 28](#_bookmark55)

[FIGURE 21 : DIAGRAMME DE CAS D'UTILISATIONS POUR MARKETING 30](#_bookmark56)

[FIGURE 22 : DIAGRAMME DE CLASSE 31](#_bookmark58)

[FIGURE 23 : DIAGRAMME DE SEQUENCE ENVOIE D'UNE NOTIFICATION 32](#_bookmark60)

[FIGURE 24 : DIAGRAMME DE SEQUENCE AJOUTER NOTIFICATION 33](#_bookmark61)

[FIGURE 25:ARCHITECTURE MICRO SERVICE 36](#_bookmark65)

[FIGURE 26 LOGO DE JAVA 17 42](#_bookmark72)

[FIGURE 27 LOGO DE ANGULAR 43](#_bookmark73)

[FIGURE 28 : ARCHITECTURE ANGULAR 44](#_bookmark74)

FIGURE 29 LOGO DE SPRING BOOT 44

[FIGURE 30 LOGO DE KAFKA 46](#_bookmark75)

[FIGURE 31 LOGO DE MAVEN 47](#_bookmark76)

[FIGURE 32 LOGO DE INTELLIJ IDEA 47](#_bookmark77)

[FIGURE 33 LOGO DE POSTMAN 48](#_bookmark78)

[FIGURE 34 LOGO DE GITLAB 48](#_bookmark80)

[FIGURE 35 LOGO DE GIT 49](#_bookmark81)

[FIGURE 36 GITFLOW WORKFLOW 50](#_bookmark82)

[FIGURE 37 : KUBERNETES ET DOCKER 51](#_bookmark83)

[FIGURE 38 L'INTERFACE JIRA 53](#_bookmark85)

FIGURE 39 MICROSOFT TEAMS 54

[FIGURE 40 LOGO DE CONFLUENCE 54](#_bookmark86)

[FIGURE 41 LOGO DE POSTGRESQL 55](#_bookmark88)

[FIGURE 42 : LOGO DE MYSQL 55](#_bookmark89)

[FIGURE 43 : ARCHITECTURE DE CDN BACK END 58](#_bookmark95)

[FIGURE 44 : NEW BACK END 61](#_bookmark97)

[FIGURE 45 : ARCHITECTURE FRONT END 62](#_bookmark99)

[FIGURE 46 NOTIFICATION PAGE 64](#_bookmark102)

[FIGURE 47 : MAPPING INTERFACE 65](#_bookmark103)

[FIGURE 48 : AJOUT BUTTON 65](#_bookmark104)

[FIGURE 49 : PAGE D'AJOUT ET MODIFICATION 66](#_bookmark105)

[FIGURE 50 : PAGE D'AJOUT ET MODIFICATION 2 67](#_bookmark106)

[FIGURE 51 : SUPPRESSION POP-UP 67](#_bookmark107)

[FIGURE 52 : RECHERCHE FORMULAIRE 68](#_bookmark108)

[FIGURE 53 : PAGE VARIABLE 68](#_bookmark110)

[FIGURE 54 POP-UP D’AJOUT ET MODIFICATION 69](#_bookmark111)

[FIGURE 55 : SUPPRESSION VARIABLE 69](#_bookmark112)

[FIGURE 56 : NOTIFICATION D'ESPACE CLIENT 70](#_bookmark115)

[FIGURE 57 : CREATION TEMPLATE 71](#_bookmark117)

[FIGURE 58 : INTERFACE DE SCENARIO 72](#_bookmark118)

[FIGURE 59 : INTERFACE DE DIFFUSION 72](#_bookmark119)

# Liste des tableaux

[TABLEAU 1: DIFFERENCE ENTRE MICROSERVICE ET MONOLITHIQUE 39](#_bookmark68)

# Table des matières

[Avant-Propos II](#_bookmark0)

[Dédicace III](#_bookmark1)

[Remerciement IV](#_bookmark2)

[Résumé V](#_bookmark3)

[Abstract VI](#_bookmark4)

[Liste des abréviations VII](#_bookmark5)

[Liste des Figures VIII](#_bookmark6)

[Liste des tableaux X](#_bookmark7)

[Table des matières XI](#_bookmark8)

[Introduction générale 1](#_bookmark9)

[Chapitre 1 : Contexte générale du projet 3](#_bookmark10)

* 1. [Présentation de l’organisme d’accueil Intelcia 4](#_bookmark11)
     1. [Présentation du groupe Intelcia 4](#_bookmark12)
     2. [Structure de la société 6](#_bookmark16)
     3. [Les valeurs de la société Intelcia 7](#_bookmark18)
  2. [Le pôle IT Solutions 9](#_bookmark19)
     1. [Organigramme du pole IT Solutions 10](#_bookmark21)
     2. [Domaines d’expertise 10](#_bookmark23)
     3. [Partenaire SFR Business 11](#_bookmark25)
     4. [Services de SFR Business 12](#_bookmark27)
     5. [Références du groupe Intelcia 12](#_bookmark28)
  3. [Présentation du Middle Office 13](#_bookmark30)
     1. [Présentation du projet Pléiade Service 13](#_bookmark31)
     2. [Présentation de module notification 14](#_bookmark33)
     3. [Critique de l’existant 15](#_bookmark34)
     4. [Objectifs du projet (module Notification) 16](#_bookmark36)
  4. [Planification du projet 18](#_bookmark38)
     1. [Méthodologie choisie (SCRUM) 18](#_bookmark39)
     2. [Organisation de l’équipe 20](#_bookmark41)
     3. [Le cycle de développement du projet 20](#_bookmark42)
     4. [Planning du projet 21](#_bookmark43)
     5. [Les jiras effectuées 22](#_bookmark45)

[Conclusion 26](#_bookmark50)

[Chapitre 2 : Analyse et Conception de Refonte CDN 27](#_bookmark51)

* 1. [Présentation d’UML 27](#_bookmark52)
  2. [Diagramme des cas d’utilisations 28](#_bookmark54)
  3. [Diagramme de classe 31](#_bookmark57)
  4. [Diagramme de séquence 32](#_bookmark59)

[Chapitre 3 : Architecture technique et technologies utilisées 34](#_bookmark62)

* 1. [Présentation de l’architecture technique de Pléiade Service 34](#_bookmark63)
     1. [L’architecture Micro-Service 35](#_bookmark64)

[Les avantages, les inconvénients d’Architecture micro services 36](#_bookmark66)

[Différences entre les microservices et architectures monolithiques 38](#_bookmark67)

* + 1. [Implémentation de l’architecture micro-service dans l’application](#_bookmark69) [Pléiade 40](#_bookmark69)
  1. [Outils et technologies 42](#_bookmark70)
     1. [Environnements de développement 42](#_bookmark71)
     2. [DevOps 48](#_bookmark79)
     3. [Les Outils de collaboration 53](#_bookmark84)
     4. [Outils de base de données 55](#_bookmark87)

[Conclusion 56](#_bookmark90)

[Chapitre 4 : Mise en œuvre et réalisation du projet CDN 57](#_bookmark91)

[4.1 Structure du projet 58](#_bookmark92)

* 1. [Back-end 58](#_bookmark93)

[Notification module 58](#_bookmark94)

[Notification nouveau back end 61](#_bookmark96)

[Front-end 62](#_bookmark98)

* 1. [Présentation des interfaces du IHM CDN et d’autre interfaces qui](#_bookmark100) [consomme le module notification 64](#_bookmark100)
     1. [Page Notifications 64](#_bookmark101)
     2. [Page Variable 68](#_bookmark109)
     3. [Page Historique 70](#_bookmark113)
     4. [Notification dans l’Espace Client 70](#_bookmark114)
     5. [IHM DMC 71](#_bookmark116)

[Conclusion 73](#_bookmark120)

[Conclusion Générale 74](#_bookmark121)

[Webographie 76](#_bookmark122)

# Introduction générale

De nos jours, chaque nouvelle entreprise aspire à satisfaire ses clients et à améliorer son application afin d'atteindre une version finale de qualité. C'est dans cette optique que SFR s'est lancé dans le projet de refonte de l’interface homme- machine du centre de notification : une solution pour centraliser les différentes communications émises vers les clients. Nous avons mis en place un module de Notification basé sur l'outil DMC (Diffusion Multi Canal) de SFR.

Par rapport à DMC, le module de Notification introduit un canal de notification supplémentaire lié à l'espace client. Les notifications peuvent être envoyées :

* Vers une adresse e-mail
* Vers un numéro fixe
* Vers un numéro de mobile
* Vers l'espace client lui-même

À cet égard, le CDN PLEIADE SERVICE expose :

* Une API générique permettant de déposer des demandes de notifications
* Une interface graphique permettant de déclarer une notification, c'est-à-dire de l'identifier, de spécifier ses paramètres et de la lier au scénario DMC préalablement défini.

Pléiade service est un middle office qui fournit des services génériques et simplifiés à l'ensemble des acteurs du système d'information (SI) du secteur B2B. Il intègre un contrôle des habilitations selon les besoins de nos partenaires, leur permettant ainsi de gérer efficacement leurs opérations et de favoriser la coordination et la collaboration entre les intervenants du SI. Pléiade se concentre sur la simplicité, l'accessibilité et la conformité aux normes de sécurité strictes propres au domaine B2B. Nous sommes un partenaire fiable et innovant, offrant

des solutions adaptées pour soutenir la croissance et le développement durables des activités de nos partenaires.

La mise en œuvre de notre solution nécessite une compréhension approfondie du contexte et des exigences fonctionnelles. Ainsi, une analyse approfondie a été effectuée en amont de ce

Projet de fin d'études afin de saisir pleinement les enjeux liés à la solution envisagée.

Notre stage de fin d'études, réalisé au sein de Intelcia IT Solutions sur une période de six mois (3 mois pour le moment), s'est déroulé dans ce contexte, et nous avons travaillé sur la migration du back end du module notification et la refonte de la partie front-end du CDN, ce qui est un élément clé du projet Pléiade. Ce rapport présente une synthèse du travail réalisé tout au long de ce projet.

Il est composé de quatre chapitres, qui sont les suivants :

***Chapitre 1*** : "Contexte général du projet", nous fournirons une présentation de l'entreprise d'accueil, du sujet du stage, de la problématique abordée et des objectifs visés. De plus, nous expliquerons en détail l'approche adoptée pour garantir le succès du projet et assurer son bon déroulement.

***Chapitre 2*** : " Analyse et Conception de CDN ", qui décrit en détail le projet Pléiade et plus précisément le module notification, ses objectifs et les besoins fonctionnels que l'application doit satisfaire.

***Chapitre 3*** : " Architecture technique et technologies utilisées ", Nous analyserons en détail l'architecture technique du projet et en donnerons un aperçu détaillé. De plus, nous présentons les outils, les technologies et les méthodologies de développement que nous avons sélectionné et mis en œuvre.

***Chapitre 4*** : " Mise en œuvre et réalisation du projet CDN", où nous présenterons la structure de notre projet et la mise en œuvre des différentes tâches dans le cadre du projet CDN.

# Chapitre 1 : Contexte générale du projet

Ce chapitre vise à fournir un aperçu du contexte global de mon projet de fin d'études. Dans un premier temps, nous présenterons la société « Intelcia IT Solutions », qui est l'organisme d'accueil. Ensuite, nous décrirons le cadre général et les circonstances entourant le développement du projet Pléiade Service. Nous détaillerons également les objectifs que nous devons atteindre, ainsi que l'approche méthodologique que nous avons adoptée pour assurer une gestion efficace de notre projet.

## Présentation de l’organisme d’accueil Intelcia

### Présentation du groupe Intelcia :



*Figure 1 Logo Intelcia*

INTELCIA Group est le premier groupe marocain spécialisé dans les métiers de l’externalisation des services avec une démarche de partenaire à valeur ajoutée. Il intervient dans les secteurs télécoms, médias, assurances, e- commerce et d’autres secteurs.

Créé il y a 20 ans au Maroc et porté par une ambition forte, le Groupe a rapidement étendu sa présence géographique et ses activités pour accompagner ses clients sur l'ensemble de leurs opérations d'externalisation et leur permettre de mieux se concentrer sur leurs enjeux et cœur de métier.

En 2020, INTELCIA poursuit son expansion internationale avec une implantation en Amérique dans les caraïbes et de nouvelles destinations en Europe. Avec plus de 26 580 collaborateurs, le groupe est présent aujourd’hui sur 56 sites à travers 17 pays dont 1 site client en France et 11 sites clients au Portugal et ambitionne d’atteindre 30 000 collaborateurs vers la fin de 2021 avec

une présence plus importante en Europe et de nouvelles implantations aux Etats- Unis. (1)



*Figure 2 Présence à l'international*

Le groupe INTELCIA s’est développé à partir de fusions et acquisitions majeures réalisés depuis 2000 qui peuvent être représentées par la figure ci- dessous :



*Figure 3 Histoire du groupe INTELCIA*

### Structure de la société

A travers ses services, Intelcia permet à ses clients de se concentrer sur leurs enjeux et leur cœur de métier. Elle propose ainsi une offre globale et multisectorielle autour de 4 domaines de solutions :



*Figure 4 Structure de la société Intelcia*

### Les valeurs de la société Intelcia

Ce qui distingue Intelcia, c'est sa singularité et son unité autour de valeurs fortes. Ces principes fondamentaux orientent leurs décisions, leurs actions et leurs comportements, reflétant ainsi leurs croyances et leurs convictions profondes. Ils contribuent à façonner la culture et l'identité d'Intelcia en tant qu'entreprise. Parmi ces valeurs, on retrouve des éléments tels que l'innovation, l'excellence, la responsabilité sociale, l'intégrité et le respect de chaque individu. Elles servent de guide aux employés dans leurs choix et leurs comportements au quotidien, en accord avec la vision et les objectifs de l'entreprise

**Optimisme. Ambition. Audace:** Nos rêves sont notre principal moteur. Notre vision optimiste est résolument tournée vers

l’avenir. Elle nous aide à penser puis à façonner un lendemain meilleur. Nous

croyons en nos capacités. Voilà pourquoi nous n’hésitons psychologiques pour réaliser nos ambitions.

 **Transparence. Respect. Bienveillance:** Notre cultured’entreprise est basée sur le partage.

Générosité et cohésion sont nos leitmotivs. A INTELCIA, nous respectons la différence. C’est ainsi que nous créons des relations riches et durables qui mènent inévitablement vers un succès collectif. Nous veillons également au bien-être de nos collaborateurs de nos clients et de nos partenaires.

**Excellence. Agilité. Inventivité**: Nous avons une responsabilité de résultat. Nous prônons donc la prise

d’initiative et la créativité dans l’exécution de nos missions. Nous sommes des bâtisseurs. Nous nous surpassons au quotidien pour trouver des solutions à tous les obstacles. C’est ce qui nous permet de répondre avec agilité aux exigences et aux besoins de nos client.

## Le pôle IT Solutions



*Figure 5 Logo Intelcia It Solutions*

Le pôle IT Solutions, créé en 2019, est le fruit de regroupement des deux structures IT : ATEXO Maroc et SBS (SFR Business Solutions) Morocco, filiales du groupe Intelcia. Le pôle accumule une expérience de 15 ans comme acteur de référence dans l’infogérance et la digitalisation.

Le pôle INTELCIA IT Solutions a pour vocation de renforcer le portefeuille

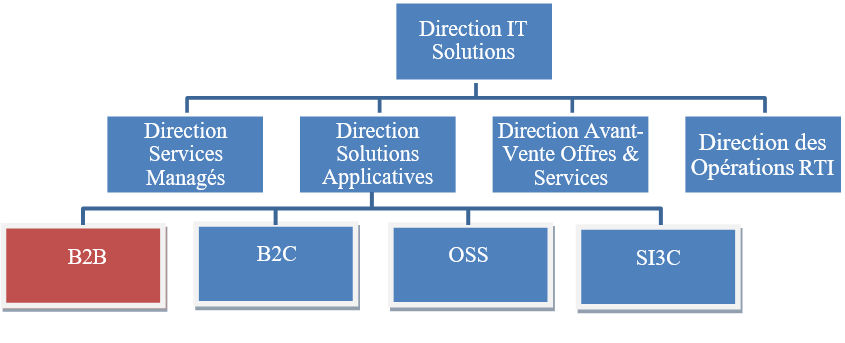
d’expertise du groupe, en ligne avec sa stratégie de diversification entamée il y a quelques années avec le lancement des activités dans le Digital Advertising et le BPO (Business Process outsourcing).

Le pôle INTELCIA IT Solutions s’appuiera pour son lancement sur les expertises des entreprises acquises et regroupera différents services

d’externalisation de solutions SI comme le développement et la maintenance d’applications, les services d’infrastructure, la sécurité, le testing… (2)

### Organigramme du pole IT Solutions

Le pôle IT Solutions est composé de plusieurs direction représentant chacune des Business Unit. La figure ci-dessous présente la structure organisationnelle du pôle :



*Figure 6 Organigramme de pôle IT Solutions*

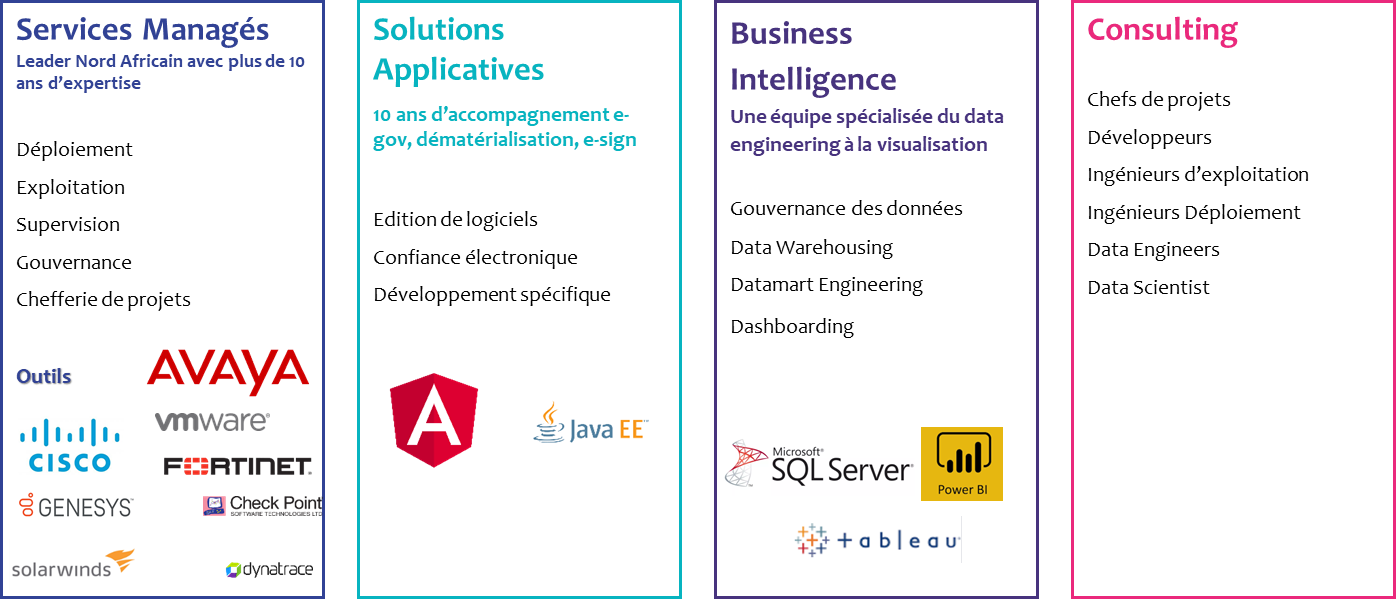
Mon stage PFE a été effectué au sein de la direction des solutions applicatives, particulièrement dans l’unité opérationnelle de B2B, sous l’encadrement du Manager B2B Mr Aissam Chaikh.

### Domaines d’expertise

Intelcia IT Solutions est spécialisée principalement dans quatre pole

d’expertise Services Managés et les solutions applicatives, Business Intelligence et Consulting.

La figure ci-après regroupe l’ensemble des domaines d’expertises d’Intelcia IT Solutions



*Figure 7 Le pôle d'expertise IT d'Intelcia*

### Partenaire SFR Business



*Figure 8 Logo SFR*

La Société Française du Radiotéléphone, SFR, est un opérateur français de télécommunications créé en 1987 par la Compagnie générale des eaux. SFR propose un réseau très haut débit, fixe et mobile, sur le territoire français à

destination des particuliers et des entreprises. L’entreprise réalise également des investissements importants pour le développement de la 5G.

### Services de SFR Business

Les services de SFR Business sont basés principalement sur des offres et solutions IT et infrastructures réseaux, sont présentées ci-contre :

* Les solutions de sécurité Internet
* Les solution d’Internet des Objets
* Les offres de Relation client
* Les solutions WAN et LAN
* Les offres et abonnements de téléphonie fixe et mobile (4G /5G).

### Références du groupe Intelcia

L’activité de Groupe INTELCIA ne se restreint pas au niveau national. En effet, elle est appelée à devenir l’un des bureaux de référence, notamment en Afrique et en Europe, et dispose notamment d’un grand nombre de clients.

Elle compte plus de 100 clients, comme illustré dans la figure ci-après.



*Figure 9 Principales références du groupe INTELCIA*

## Présentation du Middle Office

Le middle office est réparti en 2 grandes équipes :

##### Pléiade service pour la partie synchrone

Délivre des services avec des temps de réponse relativement courts.

##### Artémis service pour la partie Asynchrone

Orchestrateur de commandes, il peut y avoir plusieurs jours de traitement avant la finalisation des opérations.

### Présentation du projet Pléiade Service

Pléiade service est un middle office qui a pour but de fournir des services génériques et simplifiés a l'ensembles des acteurs SI du B2B. En incluant un contrôle des habilitations en fonction des besoins de nos partenaires.

##### Objectif du Pléiade service :

* Fournir des services simples d'utilisations et génériques
* Garantir la confidentialité des données clients en fonctions des différents contextes
* Masquer la complexité du SI à l'ensemble de nos consommateurs
* Centraliser les API exposé aux clients externes de SFR



*Figure 10 : Pléiade Service*

### Présentation de module notification

Le module de notification a été introduit pour rationnaliser/centraliser les différentes communications émises vers nos clients. Il se base sur un outil SFR (également mis à disposition de nos clients): DMC (Diffusion Multi Canal).

Le module de Notification introduit, par rapport à DMC, un canal de notification supplémentaire lié à l'espace client. Une notification peut être émise vers :

* Une adresse mail
* Un numéro fixe
* Un numéro mobile
* L’espace client.

A ce titre le CDN PLEIADE SERVICE expose:

Une API générique permettant de déposer des demandes de notifications

Une interface graphique permettant de déclarer une notification, c'est à dire de l'identifié, d'identifier ses paramètres et de faire le lien avec le scénario DMC qui aura été préalablement défini.

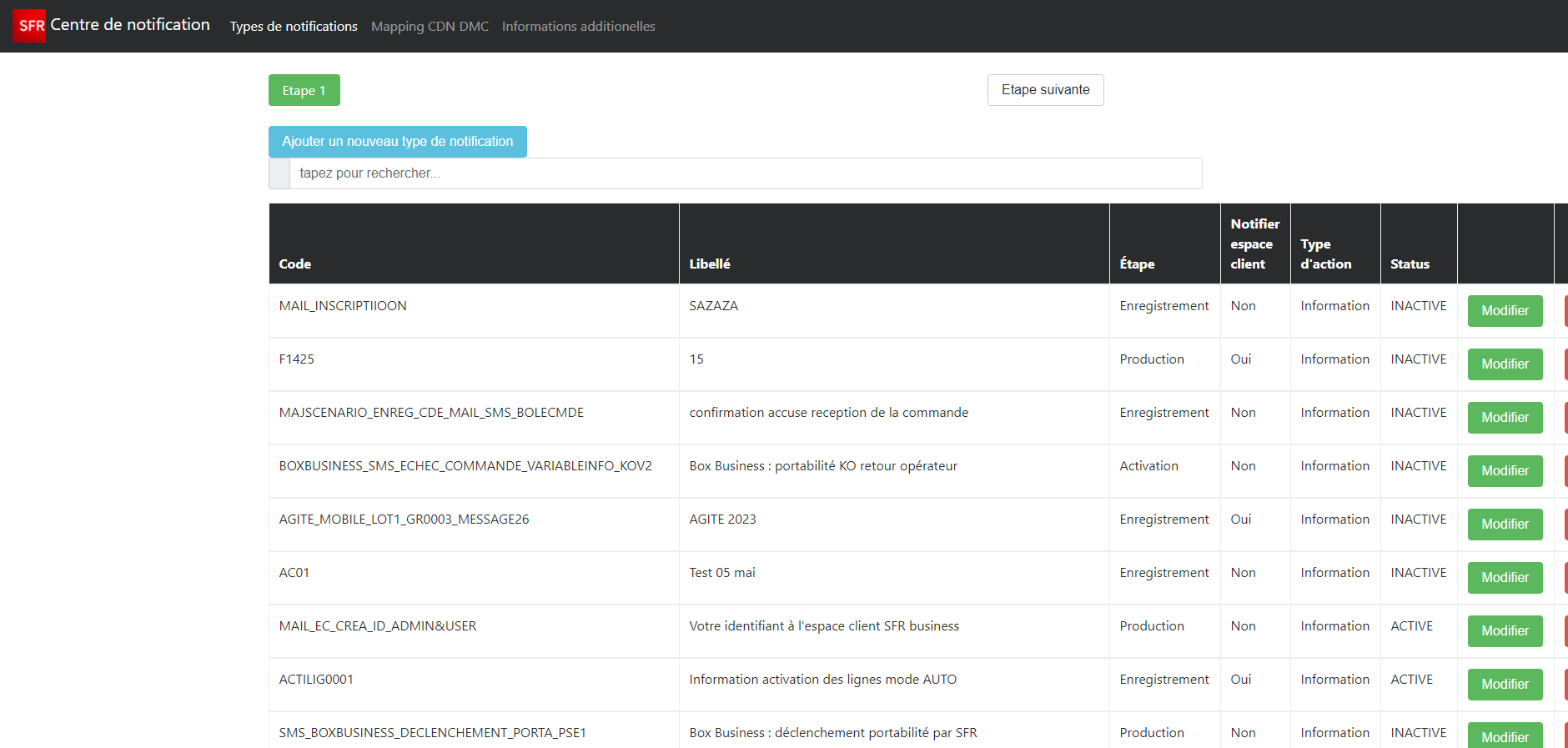
Parmi les utilisateurs du CDN ont trouvé:

* L’espace client B2B
* Les workflows de prise de commande ARTEMIS\_SERVICE et PLEIADE\_CPTL
* Le workflow de déploiement ARPEGE Les types de notification supportés sont:
* Le mail
* Le SMS
* Le message vocal
* Le SMS interactif (3)

### Critique de l’existant

Après l’utilisation de l’ancienne application par les partenaires externes de SFR et les CDP au sein de **SFR**, plusieurs remarques ont été signalées, à savoir :

* + - * **Problème de performance :** plusieurs remarques ont été signalées concernant les problèmes de performance, à savoir le temps de réponse afin d’afficher les données via des web services, mais aussi le temps d’affichage des pages.
      * **Interface de paramétrage:** Mauvais fonctionnement de quelques services.
      * **Plusieurs fonctionnalités manquantes**
      * **Des anciennes versions des technologies**
      * **Design vieillissant**



*Figure 11 : ancienne IHM*

### Objectifs du projet (module Notification)

Le principal objectif de ce projet est d'optimiser le site en procédant à une refonte complète du design et de la structure fonctionnelle de l'application. Cette refonte doit permettre de répondre aux exigences techniques de l'entreprise et d'améliorer les performances de la plateforme. De plus, il est prévu d'intégrer un self-care client en mettant en place une application facile à manipuler.

Cette refonte doit permettre d'améliorer l'expérience utilisateur et de simplifier la gestion des notifications et ces paramétrages.

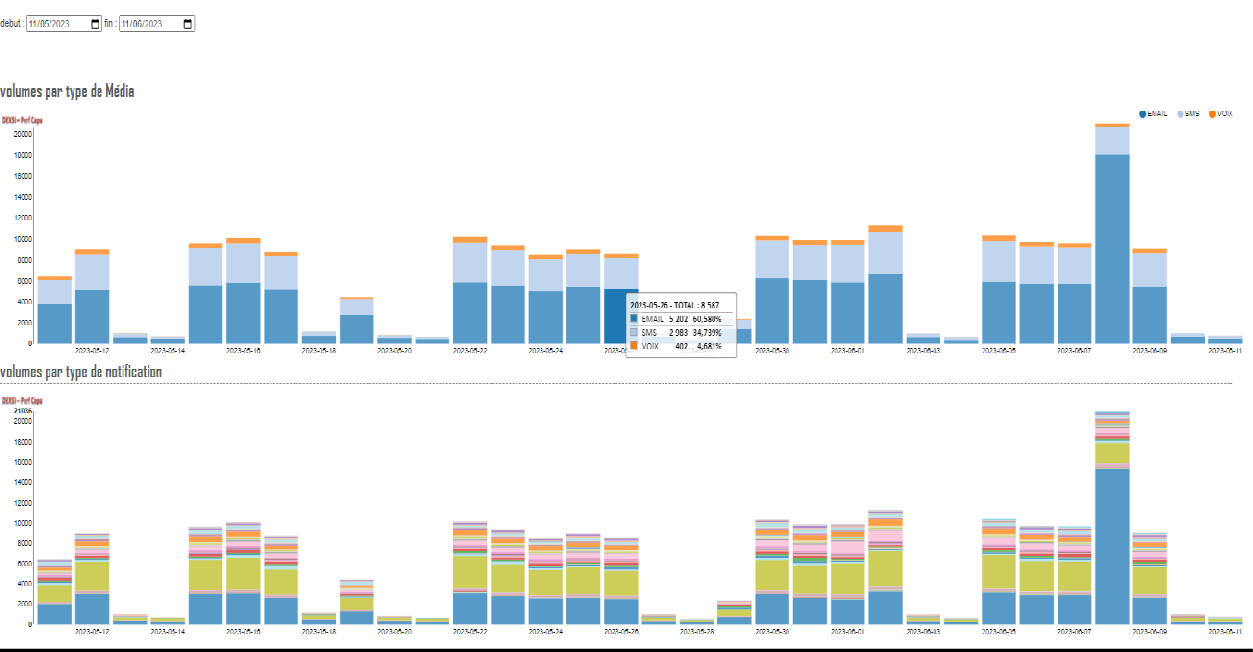
Ainsi, en intégrant cette fonctionnalité et en optimisant la plateforme, l'entreprise pourra répondre aux attentes de ses clients et améliorer sa réputation en matière de service clientèle.

Après avoir analysé l'application existante et rédigé un rapport sur les anomalies qu'elle présente, nous avons procédé à une refonte complète de l'application. Nous avons adopté une nouvelle architecture et utilisé de nouvelles technologies pour la développer. Nous avons également procédé à un refactoring de l'application et optimisé les algorithmes de la partie métier afin de répondre aux besoins de l'entreprise.

Nous avons ajouté plusieurs fonctionnalités à la plateforme existante afin d'améliorer l'expérience utilisateur et répondre aux attentes des clients. Cette refonte a permis de résoudre les problèmes de performance et d'optimisation qui affectaient l'application précédente.

En somme, cette refonte globale a permis de mettre à niveau l'application existante et de la rendre plus performante et plus efficace. Les nouvelles fonctionnalités ajoutées permettent d'améliorer la qualité du service offert aux clients et de répondre aux attentes de l'entreprise en matière de performance et de rentabilité.

##### Volume des notifications diffusées



*Figure 12 : volume des notifications diffusées*

## Planification du projet

La planification est une étape essentielle d'un projet, qui représente le

premier défi pour le gestionnaire de projet ou le maître d'œuvre. L'objectif est de garantir que les résultats souhaités sont atteints et que les attentes des différents groupes et individus impliqués dans le projet sont satisfaites. Pour cela, il est nécessaire d'estimer la progression du projet à travers les différentes phases du cycle de développement.

### Méthodologie choisie (SCRUM)

Dans le cadre du projet, l'adoption d'une approche agile implique la capacité à s'adapter aux changements, qu'ils soient liés aux besoins, aux priorités, aux technologies ou à l'équipe de projet. Les méthodes agiles sont largement utilisées dans le domaine informatique et reposent sur des cycles de développement itératifs et adaptatifs. Elles permettent de répondre de manière efficace aux besoins évolutifs du client en impliquant tous les collaborateurs et le client dans le processus de développement du projet. Cette approche favorise une meilleure satisfaction du client dans des délais serrés tout en renforçant les compétences des collaborateurs, offrant ainsi un avantage concurrentiel aux deux parties.

Parmi les méthodes agiles, SCRUM est actuellement la plus populaire et a été choisie pour mener à bien le projet en cours. Ce choix est motivé par la nécessité d'adaptabilité aux changements, de satisfaction du client et de livraison régulière de versions stables et fonctionnelles d'un sous-ensemble de fonctionnalités du système.

La méthode SCRUM se base sur la découpe du projet en itérations appelées "sprints". Chaque sprint a une durée généralement comprise entre deux semaines et un mois. Avant chaque sprint, les tâches sont estimées en termes de temps et de complexité à l'aide de la méthode PERT à 3 points. Ces estimations permettent de planifier les livraisons et d'estimer les coûts des tâches pour le

client. Les user stories sélectionnées pour un sprint constituent ce que l'on appelle un "sprint backlog" et représentent le produit potentiellement livrable à la fin du sprint. (5)

Le cycle de vie d'un sprint Scrum est illustré dans la figure ci-dessous :



*Figure 13 Cycle de vie d'un sprint de la méthode SCRUM*

1. Le Product Owner (PO) rédige les user stories et les place dans le product Backlog.
2. Le Product Owner priorise ensuite ces User Stories en fonction de leur Business value.
3. L'équipe Scrum se réunit lors de la cérémonie de planification du Sprint pour déterminer les user stories pouvant être traitées pendant le Sprint et la complexité de chacune de ces US. Ensuite, placer le fichier sélectionné dans le sprint Backlog, puis le décomposer en tâches.
4. Le Sprint peut alors commencer pour une itération de 2 à 4 semaines (2 semaines pour ce projet).
5. L’équipe se réunit quotidiennement pour réaliser le Scrum Meeting

(15~30min).

1. À la fin de Sprint, nous avons un produit potentiellement livrable qui fait l’objet d’une démonstration lors de la revue de Sprint. (Une démo faite par un développeur de l’équipe).
2. Le cycle se termine enfin par la rétrospective de Sprint.

L’idée d’utiliser SCRUM dans notre projet s’avère plus efficace que les méthodes traditionnelles où le développeur répond aux besoins exprimés par le chef de projet qui détient le savoir et la prise de décision ce qui n’est pas le cas dans notre projet. Pour toutes ces raisons nous avons trouvé que SCRUM répond à la majorité de nos critères de choix (livrer à temps, avoir une bonne réactivité et une flexibilité aux changements des exigences), en plus, le projet Pléiade est de nature évolutive du fait que les besoins n’ont pas été totalement identifiés dès le début, ce qui nous a orienté vers cette méthode agile.

### Organisation de l’équipe

L'équipe était dirigée par un chef de projet ou Scrum Master (coté

SFR), et un maître d'ouvrage ou Product Owner (coté SFR), et M. Aissam Cheikh était notre superviseur, qui nous a guidés tout au long du projet, et une équipe de sept développeurs expérimentés et seniors.

### Le cycle de développement du projet:

Dans ans le cadre de notre projet Pléiade Service, nous avons adopté une méthode de travail par étapes. Nous avons découpé le projet en plusieurs modules, en les priorisant selon leur importance, puis nous les avons planifiées en sprints de deux semaines. Nous avons ainsi fixé des objectifs à court terme pour éviter les retards et commencer le développement rapidement. Nous avons régulièrement organisé des démonstrations pour présenter le travail accompli au client.

Chaque jour à 9:15, nous avons tenu une réunion de suivi, appelée "Daily

Meeting", qui ne durait pas plus de 15 minutes. Durant cette réunion, chaque membre de l'équipe a répondu aux questions suivantes :

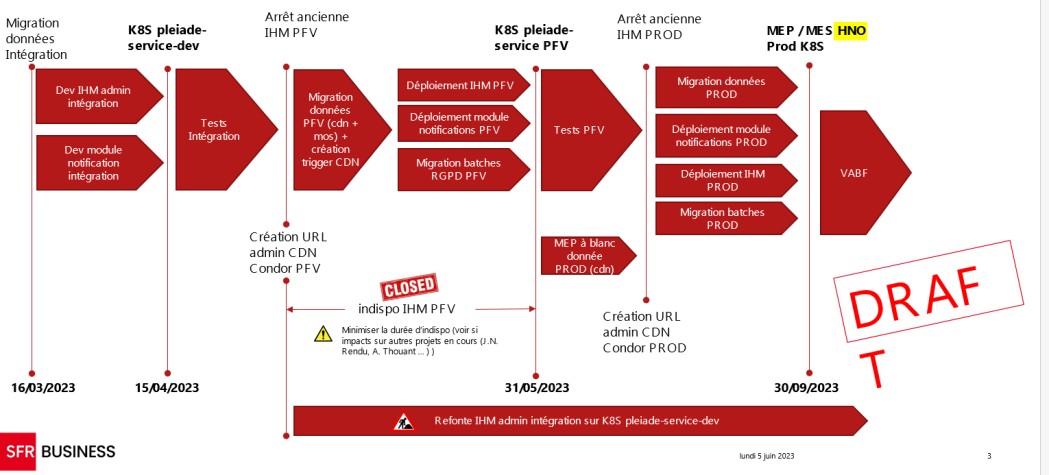
* + - * "Qu'est-ce que j'ai fait hier ?" (DONE).
      * "Qu'est-ce que je vais faire aujourd'hui ?" (TO DO).
      * "Quelles sont les difficultés ou blocages que je rencontre ?" (BLOCK).

Cette réunion était orchestrée par le Scrum Master, qui était également le chef d'équipe. Tous les membres de l'équipe, y compris le Product Owner, y participaient. L'objectif était de faire le point sur l'avancement et les difficultés rencontrées par chacun des membres de l'équipe et d'assurer la synchronisation.

À la fin de chaque sprint, nous avons organisé une réunion pour évaluer ce qui avait fonctionné et ce qui avait moins bien fonctionné. Cette réunion a duré entre 30 minutes et une heure, et un vote a été organisé pour décider des améliorations à apporter. Et on fait le déploiement de ce qui était réalisé

### Planning du projet

Au sein de notre projet Pléiade Service, nous avons mis en place une approche méthodique basée sur des étapes bien définies. Nous avons segmenté le projet en plusieurs modules en attribuant des priorités en fonction de leur importance, puis nous les avons planifiés en sprints de deux semaines. Cette approche nous a permis de fixer des objectifs à court terme, garantissant ainsi la prévention des retards et un démarrage rapide du développement. De plus, nous avons régulièrement organisé des séances de démonstration pour présenter les avancées accomplies au client. Ces démonstrations ont été une occasion précieuse de partager notre travail accompli et d'obtenir des retours pour assurer une adéquation continue avec les attentes du client.

La figure de la page suivante est une modélisation des différentes tâches pour réaliser la refonte :

*Figure 14 : Démarche du refonte*

### Les jiras effectuées :

Durant les trois premiers mois du stage j’ai eu l’opportunité de participer à 6 sprints :

##### Le premier sprint (13 mars – 31 mars) :

Durant ce premier sprint j’ai eu comme mission la documentation du projet pléiade en général, la compréhension des différents modules qui le composent ainsi que les différents types d’utilisateurs et

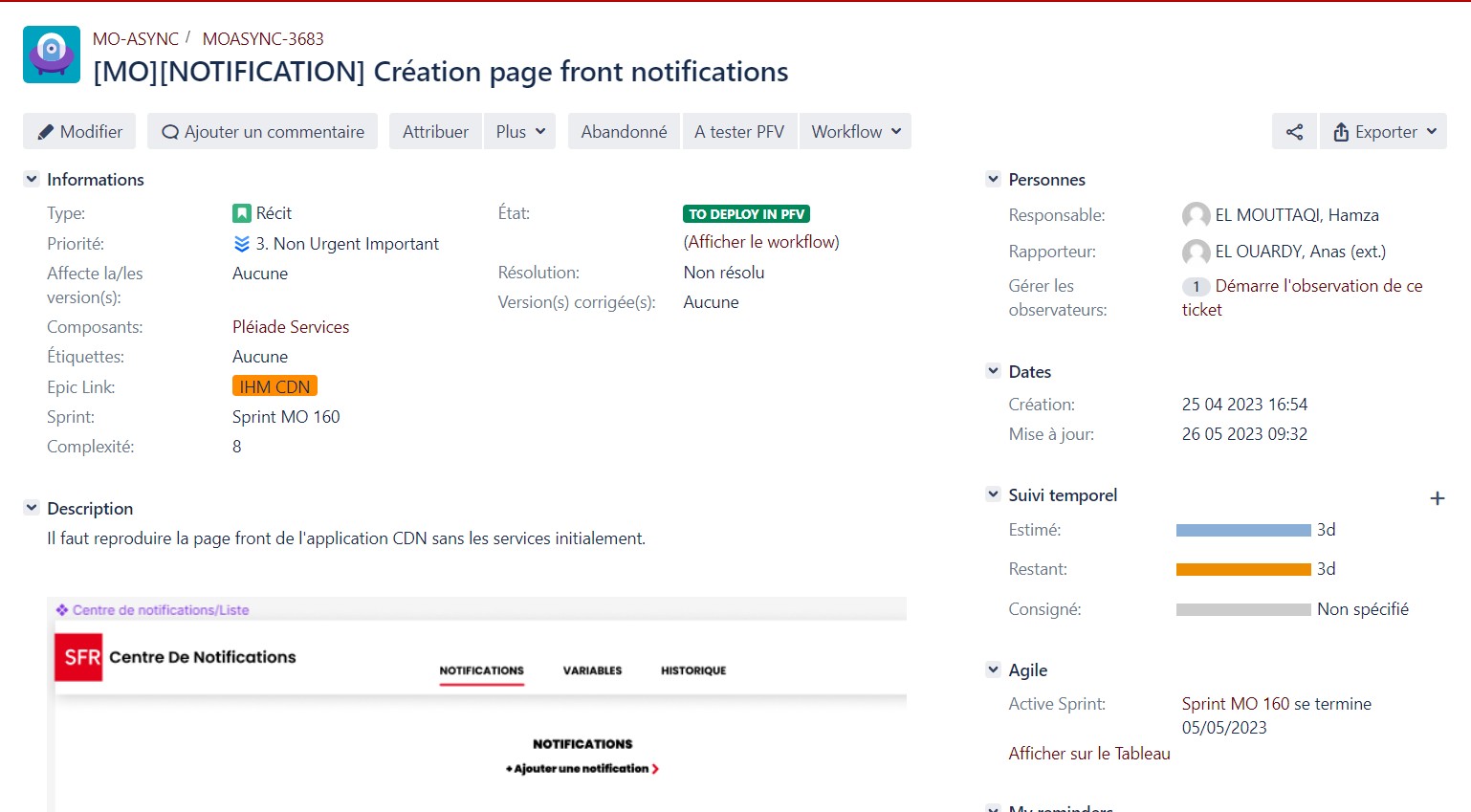
d’authentifications aux applications. J’ai aussi traité des anomalies de faibles complexités.

##### Deuxième sprint (3 avril– 14 avril) :

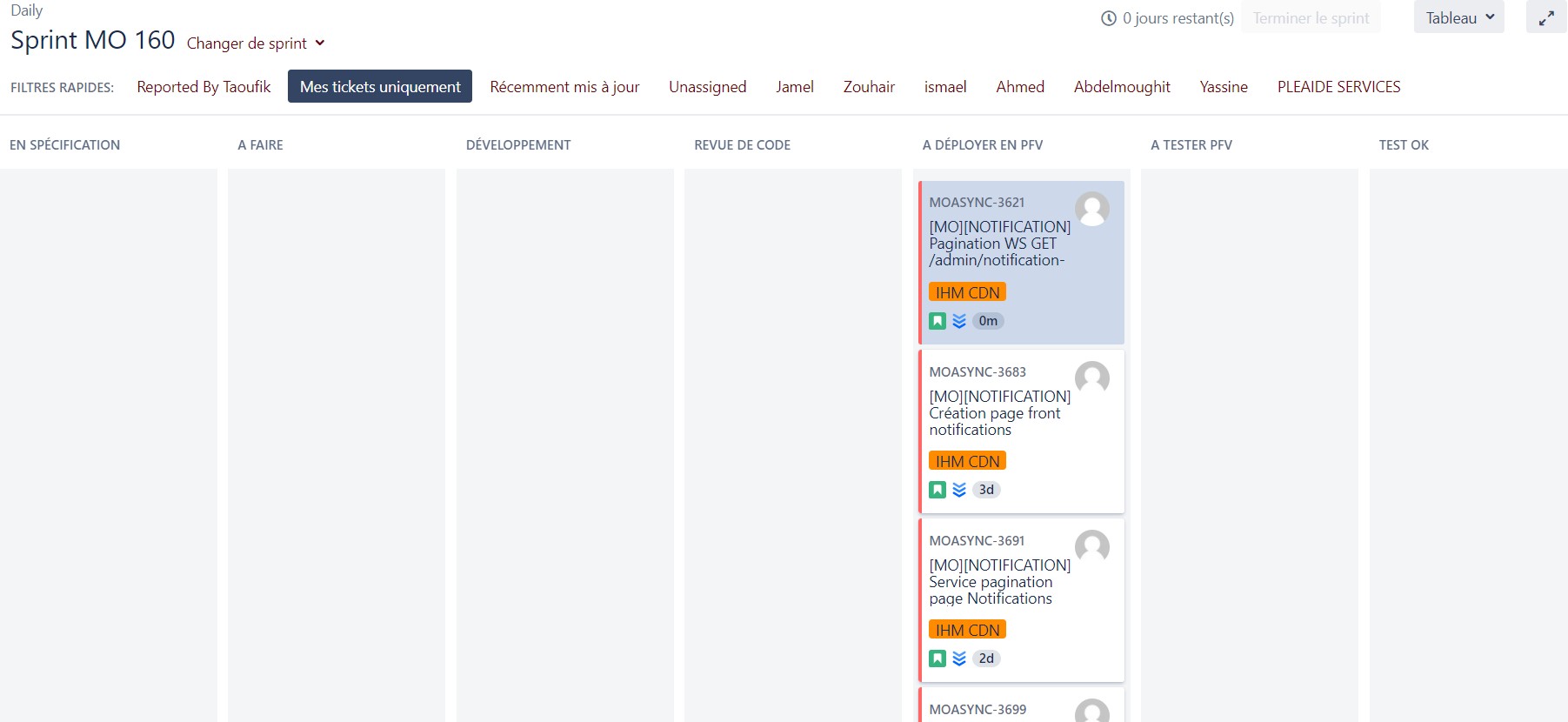
Concernant ce second sprint, j’ai eu pour tâche de migrer le back-end du module notification de java 8 à java 17. J’ai fait de même pour la base de données MySQL vers la base de données PostgreSQL, ainsi que

l’intégration de sonar, qui est un outil pour augmenter la qualité du code.

##### Troisième sprint (17 avril – 28 avril) :

Durant ce sprint, j’ai reçu la maquette de la nouvelle IHM et suite à cela, j’ai commencé les développements et initialisé le projet Angular pour le front end. Ce projet est séparé en trois pages : page notification, page variable et page historique. J’ai ainsi commencé par la page notifications.

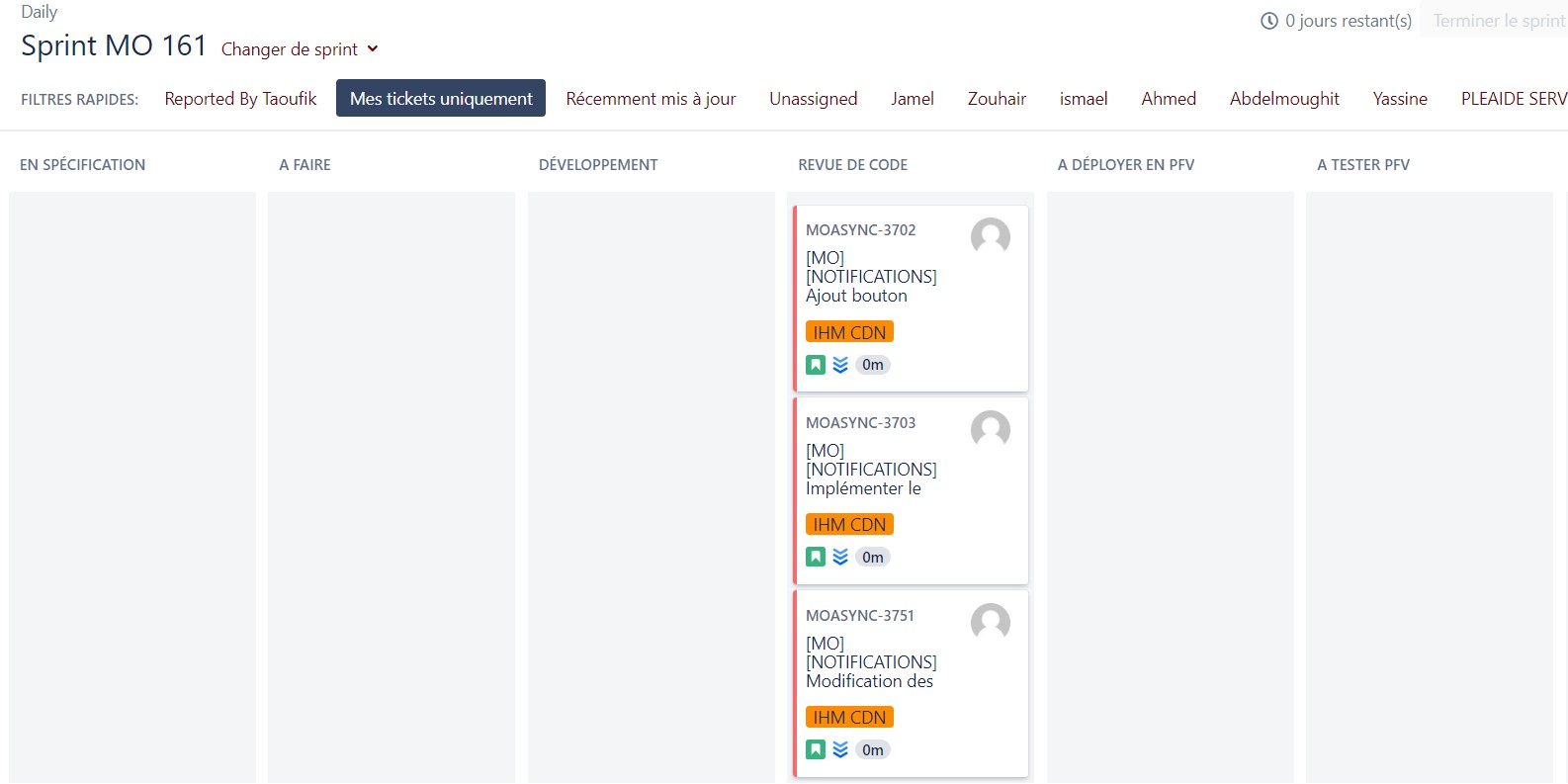
*Figure 15 : exemple de JIRA*



*Figure 16 : Jira du 3éme Sprint*

##### Quatrième sprint (1 mai – 12 mai) :

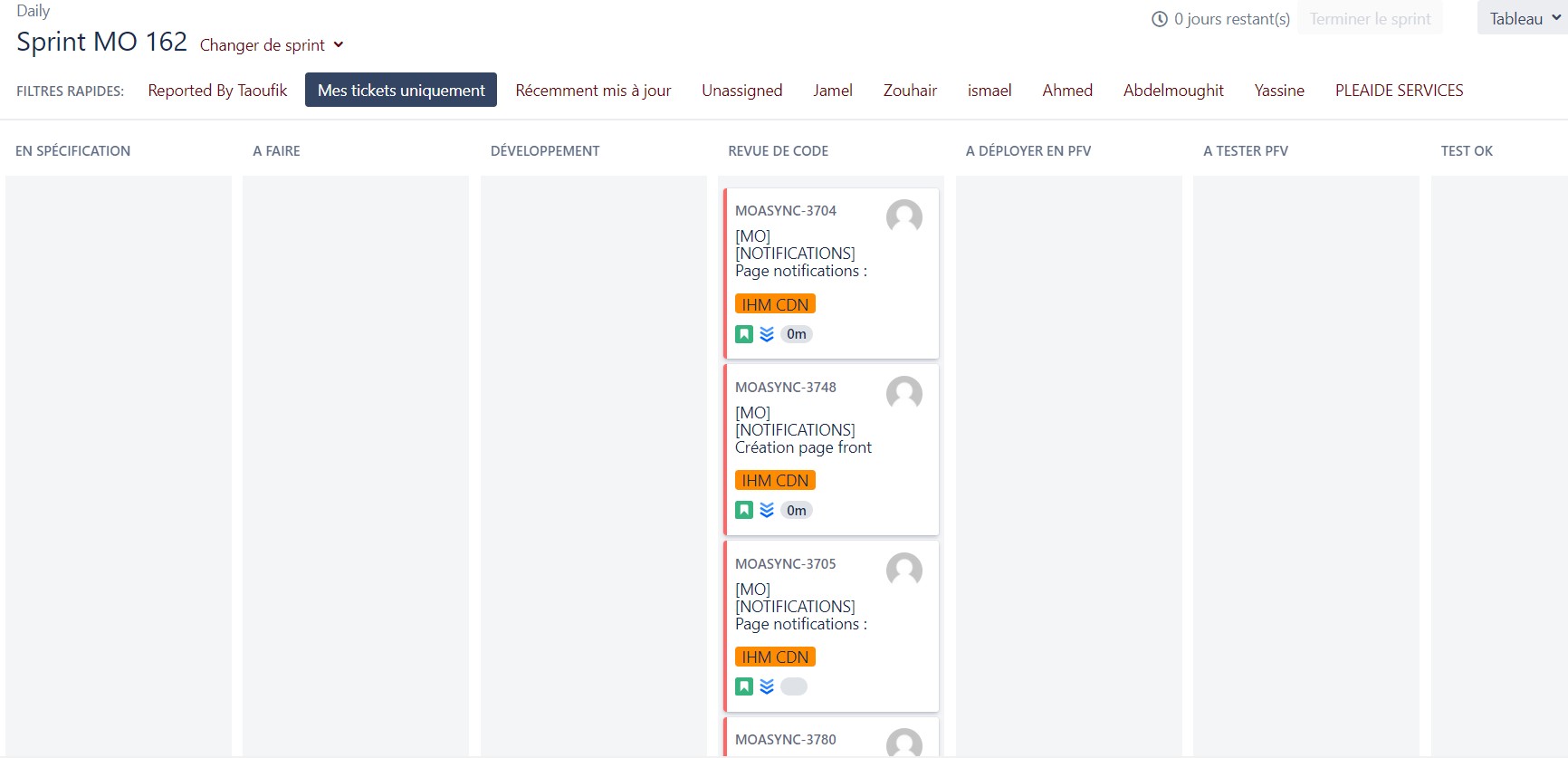
Pendant ce sprint, j’ai continué sur la page notifications en implémentant différents services : ajout, tri, modification, recherche, suppression et pagination.



*Figure 17 : Jira du 4éme Sprint*

##### Cinquième sprint (15 mai – 27 mai) :

Pendant ce sprint, j’ai commencé la deuxième page des variables en implémentant des services de recherche, de pagination et du CRUD. J’ai également fixé différentes anomalies remontées sur la première page.



*Figure 18 : Jira du 5éme sprint*

##### Sixième sprint (29 mai – 9 juin) :

En attendant de clarifier le besoin spécifique pour la page Historique, je suis en train d’étudier les solutions possibles pour un problème de performance rencontré au niveau de la récupération des données (délai long).

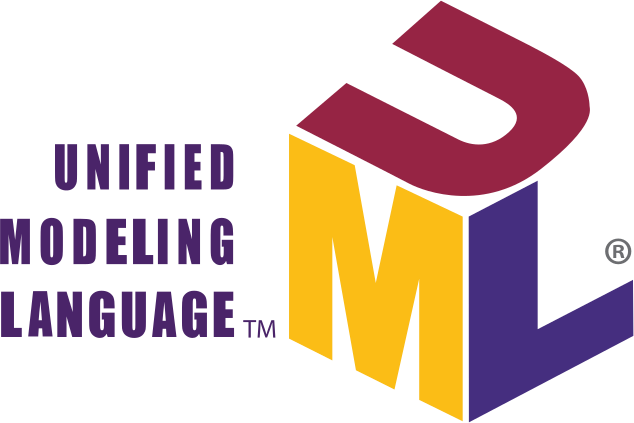
# Conclusion

Dans cette première partie, nous avons introduit le cadre général de mon stage de fin d'études en présentant Intelcia IT Solutions, l'entreprise qui nous accueille. Ensuite, nous avons exposé la méthodologie de développement adoptée pour mener à bien ce projet. Nous avons également détaillé le planning que nous avons suivi pour le développement de la refonte de l’IHM CDN.

Dans le prochain chapitre, nous nous concentrerons sur l'analyse fonctionnelle et conceptuelle de notre projet, en détaillant ses différentes composantes.

# Chapitre 2 : Analyse et Conception de Refonte CDN

## Présentation d’UML :

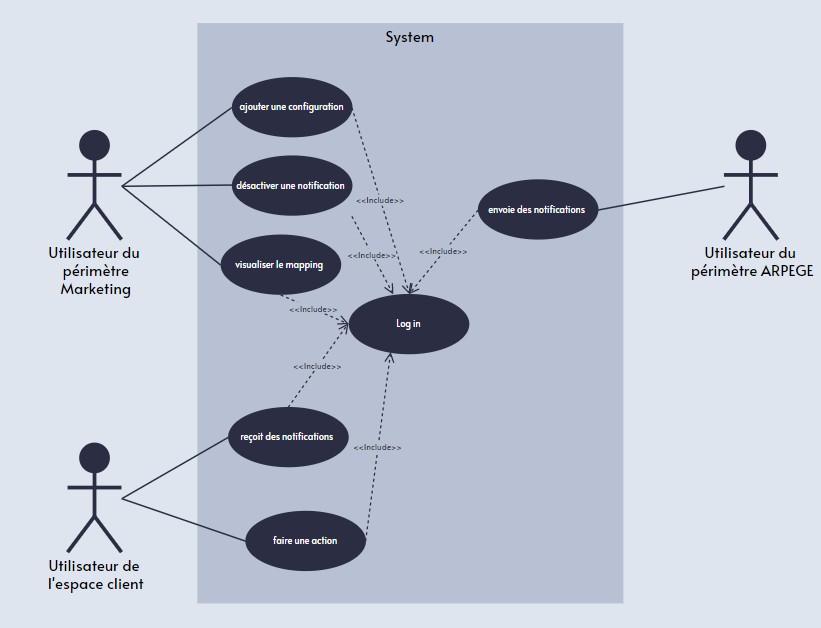


*Figure 19 : Logo UML*

L'UML (Unified Modeling Language ou Langage de modélisation unifiée en français) est un langage graphique de modélisation informatique. Ce langage est désormais la référence en modélisation objet, ou programmation orientée objet. Cette dernière consiste à modéliser des éléments du monde réel (immeuble, ingrédients, personne, logos, organes du corps...) ou virtuel (temps, prix, compétence...) en un ensemble d'entités informatiques appelées « objet ».

L'UML est constitué de diagrammes qui servent à visualiser et décrire la structure et le comportement des objets qui se trouvent dans un système. Il permet de présenter des systèmes logiciels complexes de manière plus simple et compréhensible qu'avec du code informatique. L'UML a des applications dans le développement logiciel, mais aussi dans l'industrie (pour modéliser les flux de processus par exemple), dans l'ingénierie ou le marketing. (4)

## Diagramme des cas d’utilisations



*Figure 20 : diagramme de cas d'utilisations général*

Acteurs :

1. **Utilisateur de périmètre** Marketing : Cet acteur est responsable de la gestion complète de la configuration d'une notification. Il peut effectuer les opérations CRUD (Create, Read, Update, Delete) sur la configuration des notifications. De plus, il peut visualiser le mapping entre les variables de notification et les variables de diffusion.
2. **Utilisateur de périmètre ARPEGE**: c’est la partie responsable de la gestion des commandes, Cet acteur a la capacité d'envoyer une notification après la configuration. Il utilise les configurations fournies par le

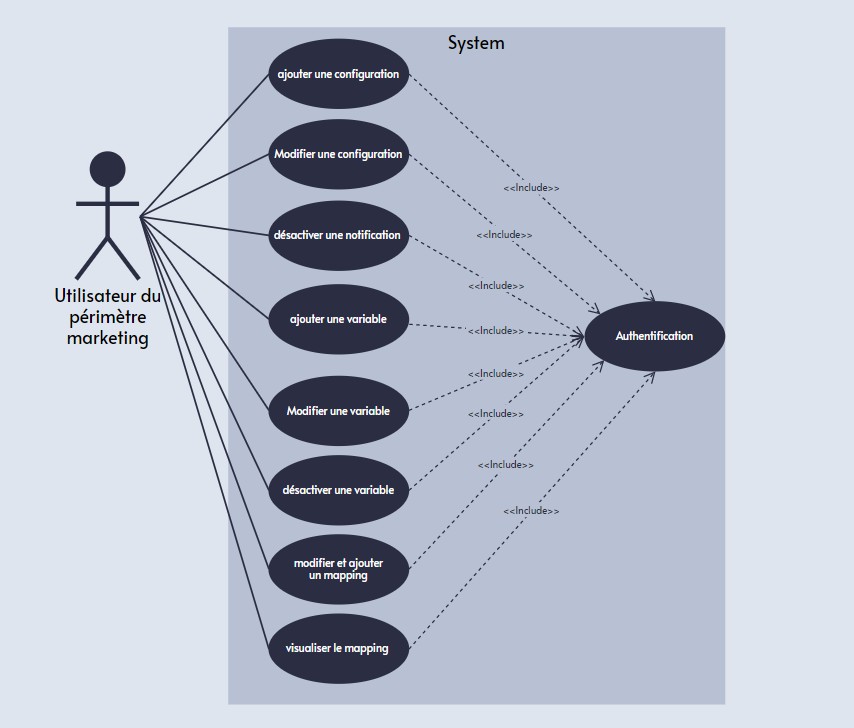
département marketing pour envoyer des notifications aux destinataires appropriés.

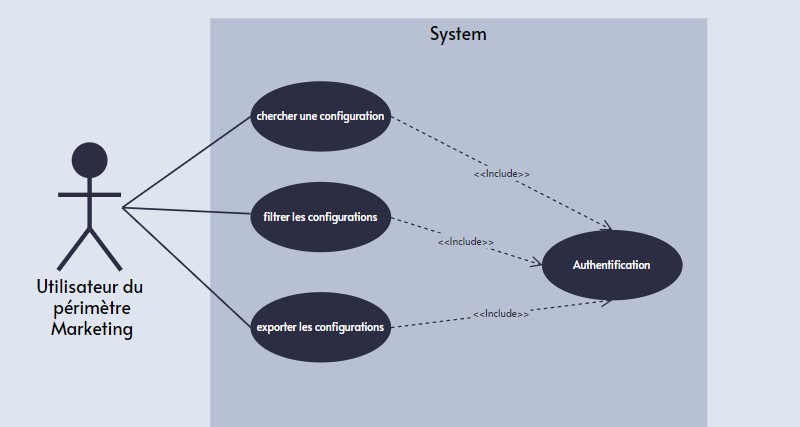
1. **Utilisateur de l’Espace Client** : Cet acteur a la possibilité de visualiser les notifications qui lui sont destinées. Il peut également modifier le statut d'une notification pour le marquer comme "lu".

Cas d'utilisation :

1. Configurer la notification : Le marketing peut créer, lire, mettre à jour et supprimer les configurations de notification. Cela comprend la définition du contenu de la notification, des destinataires, des variables de notification, etc.
2. Visualiser le mapping des variables : Le marketing peut visualiser le mapping entre les variables de notification et les variables de diffusion. Cela permet de comprendre comment les variables seront remplies lors de l'envoi de la notification.
3. Envoyer une notification : Une fois la configuration terminée, ARPEGE peut utiliser les paramètres fournis par le marketing pour envoyer la notification aux destinataires appropriés.
4. Visualiser les notifications : L'espace client peut visualiser les notifications qui lui sont destinées. Cela lui permet de voir les nouvelles notifications non lues.
5. Modifier le statut de la notification : L'espace client peut modifier le statut d'une notification pour la marquer comme "lu". Cela lui permet de gérer les notifications qu'il a déjà consultées.

Ce diagramme des cas d'utilisation met en évidence les interactions et les fonctionnalités principales entre les acteurs du système. Il permet de visualiser les différentes actions et les relations entre les acteurs, facilitant ainsi la compréhension du système et de ses fonctionnalités.



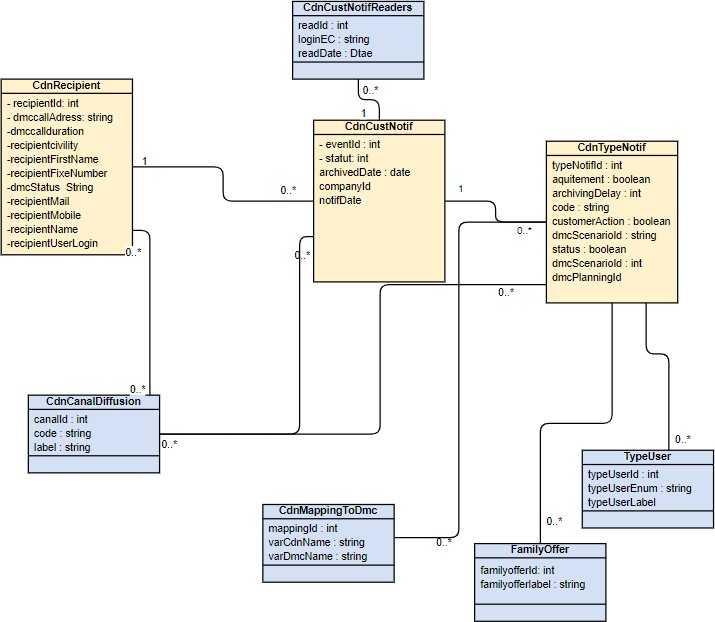


*Figure 21 : diagramme de cas d'utilisations pour marketing*

La figure ci-dessous montre le diagramme de cas d’utilisation pour l’utilisateur de marketing, qui peut effectuer les CRUD du la configuration ou bien le paramétrage des notifications, ainsi que les variables de mapping et aussi les mapping entre DMC et CDN, sans oublier la possibilité de la recherche des notifications et le filtrage et l’export des données

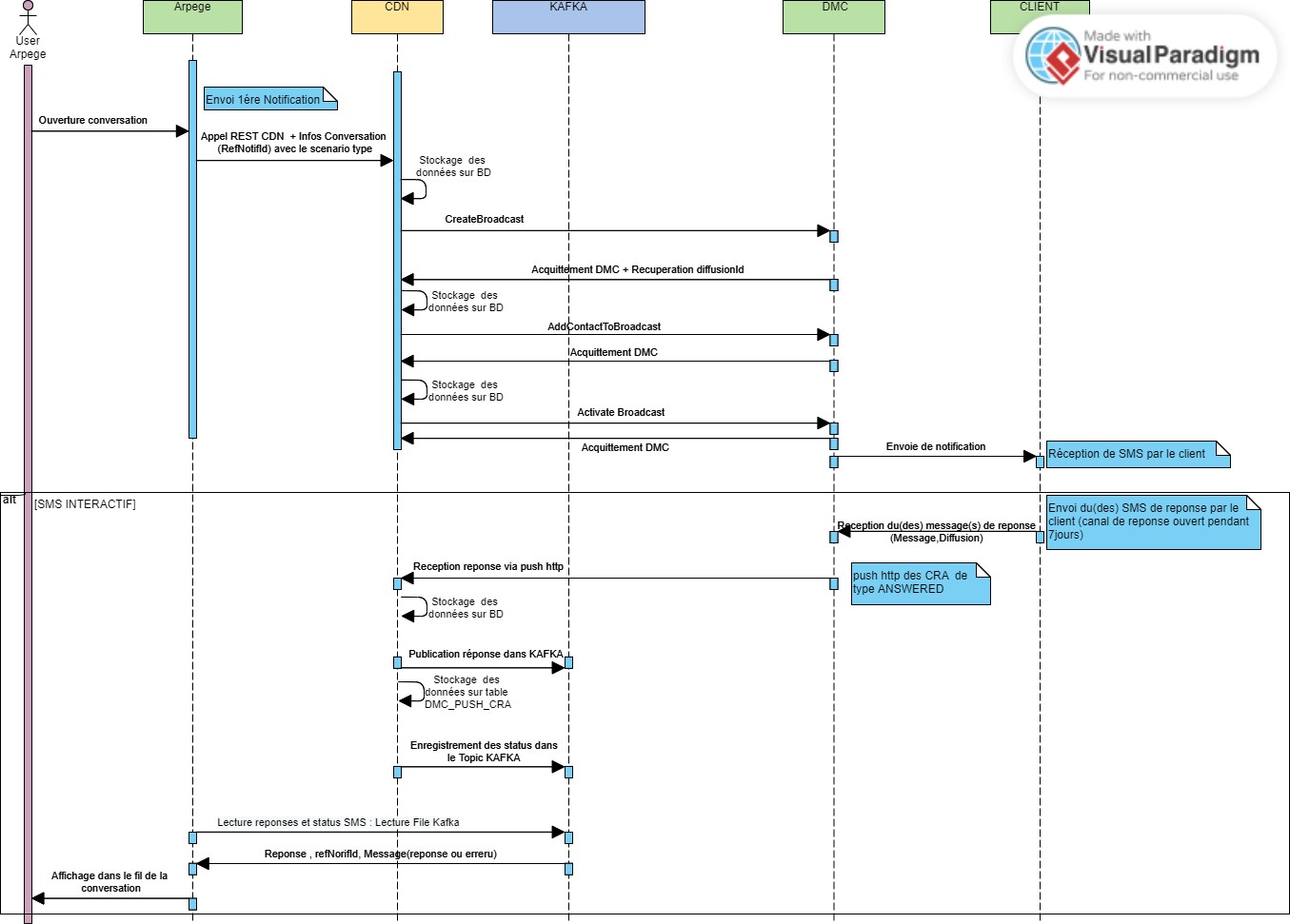
## Diagramme de classe

La figure ci-dessous montre le diagramme de classes établi, pour réaliser les cas d’utilisation de la consultation, recherche et filtrage des configurations, le mapping et les variables. On n’a pas intégré les classes du côté DMC pour ne pas compliqué le diagramme



*Figure 22 : Diagramme de classe*

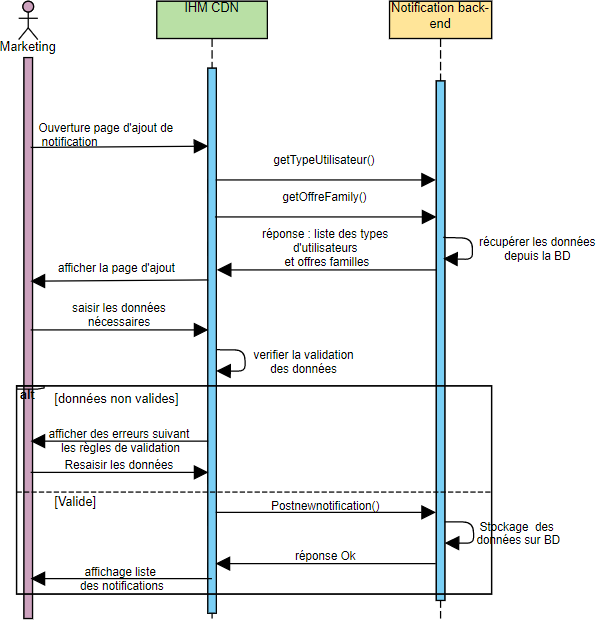
## Diagramme de séquence

La figure ci-dessous montre le diagramme de séquence système illustrant l’envoie d’une notification depuis la première étape jusqu’à la réception. En traitant le cas de SMS interactif

*Figure 23 : diagramme de séquence envoie d'une notification*

La figure ci-dessous montre le diagramme de séquence système illustrant l’ajout d’un paramétrage d’une notification en passant par la configuration des

mapping entre les variables du centre de notification et les variables de diffusion en traitant les cas de validation



*Figure 24 : diagramme de séquence ajouter notification*

# Chapitre 3 : Architecture technique et technologies utilisées

La réalisation de la refonte nécessite la maîtrise de différentes technologies et le choix approprié des méthodes tant au niveau technique que fonctionnel. Cette approche garantit un développement efficace et de qualité, en utilisant les outils et les pratiques les mieux adaptés à chaque aspect du projet.

Ainsi, en combinant une expertise technique solide avec une approche fonctionnelle optimale, les objectifs du projet peuvent être atteints avec succès.

Ce chapitre met en lumière les spécifications techniques de l'application Pléiade en général et le module notification en particulier ainsi que les différentes technologies utilisées pour sa réalisation.

## Présentation de l’architecture technique de Pléiade Service

Notre projet se distingue par la mise en œuvre d'une architecture de micro service innovante, où nous avons divisé notre application en 24 modules distincts (notifications, user, traces, ...). Chaque module représente une fonction spécifique et fonctionne indépendamment, mais s'intègre parfaitement avec d'autres modules pour former une application complète. Cette approche nous a donné flexibilité et évolutivité puisque chaque module peut être développé, testé et déployé indépendamment. En utilisant des technologies de conteneur comme Docker et des outils d'instrumentation comme Kubernetes, nous avons pu créer un environnement de déploiement flexible et hautement évolutif. En divisant les fonctionnalités de notre application entre ces 24 modules, nous avons optimisé la gestion des erreurs, puisque la défaillance d'un module n'affecte pas le fonctionnement global de l'application. Cette approche nous a également permis

de développer de nouvelles fonctionnalités plus rapidement et plus efficacement, garantissant la stabilité et les performances de l'ensemble de notre application.

### L’architecture Micro-Service

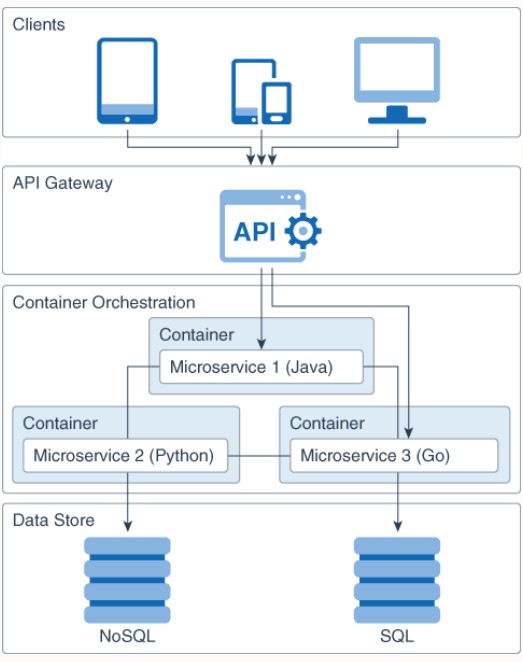
L’architecture microservices est une approche de conception logicielle qui transforme la façon dont les applications sont intégrées et déployées.

Contrairement à l’architecture monolithique traditionnelle, qui regroupe toutes les fonctionnalités dans un seul bloc de code, l’architecture microservices divise une application en un ensemble de services et d’outils plus petits et indépendants.

Ces services sont des modules autonomes qui se concentrent sur des tâches spécifiques au sein de l’application. Chaque microservice est responsable d’une fonctionnalité distincte et communique avec les autres services via des protocoles légers,tels que les APIs ou des messages asynchrones. Cette conception modulaire permet une meilleure flexibilité et évolutivité de

l’application.

Les principes clés de l’architecture microservices reposent sur la décomposition des fonctionnalités en services distincts et la mise en œuvre de l’autonomie et de la responsabilité individuelle de chacun d’entre eux. Chaque service peut être développé, déployé et mis à l’échelle indépendamment, ce qui permet aux équipes de développement de travailler de manière agile et d’introduire des changements plus rapidement. (6)



*Figure 25:architecture Micro Service*

#### Les avantages, les inconvénients d’Architecture micro services:

* Les avantages :
  + **Scalabilité et flexibilité :** Les architectures microservices permettent de mettre à l'échelle chaque microservice indépendamment en fonction des besoins spécifiques. Cela offre une grande flexibilité pour gérer les pics de charge et optimiser l'utilisation des ressources, tout en minimisant les coûts.
  + **Déploiement indépendant :** Chaque microservice peut être développé, testé et déployé de manière autonome. Cela permet aux équipes de développement de travailler de manière plus agile, en apportant des mises à jour et des améliorations à des parties spécifiques de l'application sans perturber l'ensemble du système.
  + **Résilience et isolation des pannes :** En cas de défaillance d'un microservice, les autres parties de l'application peuvent continuer à fonctionner normalement. Cela permet d'isoler les pannes et d'améliorer la résilience globale de l'application.
  + **Facilité de développement et de maintenance :** En découpant l'application en modules indépendants, le développement et la maintenance deviennent plus faciles. Chaque équipe peut se concentrer sur un microservice spécifique, ce qui facilite la compréhension du code, les tests et les modifications ultérieures.
  + **Technologie hétérogène :** Les architectures microservices permettent l'utilisation de différentes technologies et langages de programmation pour chaque microservice, en fonction de leurs besoins spécifiques. Cela offre une plus grande liberté de choix technologique et permet d'exploiter au mieux les avantages de chaque outil ou framework.
  + **Évolutivité et innovation :** Les architectures microservices favorisent l'innovation en permettant le développement rapide de nouvelles fonctionnalités. Chaque microservice peut être développé de manière autonome, ce qui accélère les cycles de développement et permet d'expérimenter de nouvelles idées plus facilement.
* Les inconvénients :
  + Complexité de la gestion : l'architecture des micro services peut introduire de la complexité dans la gestion de plusieurs micro services. L'organisation des relations et des dépendances entre les différents modules peut devenir complexe, nécessitant une planification et une gestion minutieuses pour assurer le bon fonctionnement de l'ensemble du système.
  + Surcoût de communication : Etant donné que les microservices communiquent généralement via des interfaces de programmation d'applications (API), la communication interservices peut entraîner une latence supplémentaire et une surcharge du réseau. Des appels réseau fréquents peuvent augmenter la complexité et le coût de la gestion des performances et de la sécurité des communications de microservices.
  + Cohérence des données : La gestion de la cohérence des données peut devenir plus complexe dans un environnement basé sur des microservices. Lorsque plusieurs microservices interagissent pour traiter une transaction, maintenir la cohérence des données peut nécessiter des mécanismes supplémentaires tels que la compensation des transactions, la synchronisation asynchrone ou les événements.

#### Différences entre les microservices et architectures monolithiques :

Avant de vous lancer dans la conception d'applications utilisant l'architecture microservices, il est important de comprendre les différences entre cette approche et l'architecture monolithique traditionnelle.

Dans une architecture monolithique, toute l'application est construite en une seule unité, où la logique métier et les fonctionnalités sont regroupées ensemble. L'accent est mis sur la résolution des besoins opérationnels et la mise en œuvre de la logique métier au sein de cette unité monolithique.

En revanche, dans l'architecture des microservices, la logique métier est organisée en plusieurs services indépendants, qui sont conçus pour être couplés de manière lâche. Chaque microservice se concentre sur une fonctionnalité spécifique de l'application et peut être développé, testé, déployé et mis à l'échelle de manière indépendante.

Le tableau suivant résume les différences entre les microservices et les architectures monolithiques.

|  |  |  |
| --- | --- | --- |
| Caractéristique | Architecture des microservices | Architecture monolithique |
| Conception de l'unité | L'application consiste en des services à couplage libre. Chaque service prend en charge une seule tâche métier. | Toute l'application est conçue, développée et déployée en une seule unité. |
| Réutilisation de la fonctionnalité | Les microservices définissent des API qui exposent leurs fonctionnalités à n'importe quel client. Les clients pourraient même être d'autres applications. | La possibilité de réutiliser les fonctionnalités entre les applications est limitée. |
| Communication au sein de l'application | Pour communiquer entre eux, les microservices d'une application utilisent le | Les procédures internes (appels de fonctions) facilitent la communication |

|  |  |  |
| --- | --- | --- |
|  | modèle de communication requête-réponse. L'implémentation typique utilise des appels API REST basés sur le protocole HTTP. | entre les composants de l'application. Il n'est pas nécessaire de limiter le nombre d'appels de procédure interne. |
| Flexibilité technologique | Chaque microservice peut être développé à l'aide d'un langage et d'un cadre de programmation qui conviennent le mieux au problème que le microservice est conçu pour résoudre. | Habituellement, l'ensemble de la demande est rédigé en un seul langage de programmation. |
| Gestion des données | Décentralisé : chaque microservice peut utiliser sa propre base de données. | Centralisé : toute l'application utilise une ou plusieurs bases de données. |
| Déploiement | Chaque microservice est déployé indépendamment, sans affecter les autres micros*T*e*a*r*b*v*l*i*e*c*au*es*1:*d*D*e*if*l*fé*'a*re*p*n*p*c*l*e*i*e*c*n*a*t*t*r*i*e*o*m*n*i*.*croservice et monolit* | Tout changement, aussi petit soit-il, nécessite le redéploiement et le  *hiq*r*u*e*e*démarrage de l'ensemble de l'application. |
| Maintainability | Les microservices sont simples, ciblés et indépendants. L'application est donc plus facile à entretenir. | À mesure que la portée de l'application augmente, le maintien du code devient plus complexe. |
| Résilience | La fonctionnalité de l'application est répartie entre plusieurs services. Si un microservice échoue, la fonctionnalité offerte par les autres microservices continue d'être disponible. | Une défaillance dans n'importe quel composant pourrait affecter la disponibilité de l'ensemble de l'application. |
| Evolutivité | Chaque microservice peut être échelonné indépendamment des autres services. | Toute l'application doit être mise à l'échelle, même si l'exigence commerciale ne concerne que certaines parties de l'application. |

### Implémentation de l’architecture micro-service dans l’application Pléiade

**Découpage des fonctionnalités en microservices :** Dans notre projet, nous avons découpé les fonctionnalités de notre application en différents microservices, en mettant l'accent sur l'indépendance et la responsabilité spécifique de chaque microservice. Par exemple, nous avons un microservice dédié à la gestion des utilisateurs, un autre pour la gestion des notifications, un troisième pour la gestion des alertes, et ainsi de suite.

Cette approche modulaire a considérablement facilité le développement, le test et la maintenance de chaque fonctionnalité. Chaque microservice a son propre codebase, ce qui nous permet de travailler de manière plus agile et efficace.

**Communication et intégration :** Dans notre projet, nous avons mis en place une architecture microservices où les microservices sont déployés sur GitLab et Kubernetes. La communication et l'intégration entre les microservices sont réalisées en utilisant différentes technologies en fonction des besoins spécifiques.

Pour la communication asynchrone entre les microservices, nous utilisons une approche basée sur REST API. Chaque microservice expose des points de terminaison API RESTful qui permettent aux autres microservices de les appeler de manière asynchrone. Cette communication asynchrone permet une meilleure résilience du système en évitant les blocages ou les dépendances strictes entre les microservices. Par exemple, lorsque le microservice de traitement des commandes envoie une notification de livraison à un autre microservice, il le fait de manière asynchrone via une requête REST API, sans attendre de réponse immédiate.

En ce qui concerne la communication synchrone entre les microservices, nous avons adopté l'utilisation de Kafka.

**Gestion des données :** nous avons adopté une approche polyglotte en matière de gestion des données, en utilisant à la fois PostgreSQL et MongoDB.

Nous sommes en train de migrer notre base de données principale de MySQL vers PostgreSQL. Cette migration est motivée par les avantages offerts par PostgreSQL en termes de performances, de scalabilité et de capacités avancées. PostgreSQL est réputé pour sa prise en charge étendue des requêtes complexes, sa robustesse et sa compatibilité avec des volumes de données importants. Cette transition vers PostgreSQL nous permettra d'améliorer l'efficacité de nos opérations de base de données et d'optimiser les performances globales de notre application.

En parallèle, nous utilisons également MongoDB pour des besoins spécifiques de stockage de données non relationnelles. MongoDB est une base de données NoSQL populaire, qui offre une grande flexibilité pour le stockage de données non structurées ou semi-structurées, ainsi que des fonctionnalités avancées pour la mise à l'échelle horizontale. Nous avons identifié des cas d'utilisation précis dans notre projet où MongoDB se révèle particulièrement adapté, comme la gestion de données semi-structurées ou la mise en place de fonctionnalités de recherche avancées.

## Outils et technologies

Le choix des technologies dans un projet joue un rôle déterminant dans la qualité du travail réalisé. En sélectionnant les bonnes technologies, on peut rendre le processus de développement plus agréable, améliorer l'efficacité et la productivité de l'équipe, favoriser la collaboration et la communication, et assurer la maintenance et l'évolutivité du projet à long terme. Les bonnes technologies offrent des outils puissants, des fonctionnalités avancées et une facilité d'intégration des nouvelles exigences. Il est donc essentiel de faire des choix éclairés en tenant compte des besoins spécifiques du projet et des compétences de l'équipe.

### Environnements de développement:

Le choix d'environnements de développement appropriés est essentiel pour assurer un développement efficace et de haute qualité. Les environnements de développement fournissent les outils, les bibliothèques et les fonctionnalités nécessaires pour créer, tester et déployer des applications

##### Java 17



*Figure 26 Logo de Java 17*

Java 17 est un langage de programmation et une plateforme informatique largement utilisés pour le développement de diverses applications, allant des applications de bureau et mobiles aux systèmes d'entreprise et aux appareils embarqués. Il est un choix populaire parmi les développeurs en raison de sa simplicité, de son indépendance de plateforme, de sa robustesse et de ses vastes bibliothèques et frameworks (7).

Java est un langage de programmation mature et largement utilisé, et chaque nouvelle version de Java vise à maintenir la stabilité et la compatibilité avec les versions précédentes. Cela signifie que les applications développées avec des versions antérieures de Java peuvent généralement être facilement migrées vers Java 17, sans avoir à effectuer de modifications majeures dans le code existant et aussi les nouvelles versions de Java incluent souvent des améliorations de performance, telles que des optimisations du compilateur, une meilleure gestion de la mémoire et des algorithmes plus efficaces. L'utilisation de Java 17 peut donc permettre d'obtenir des performances accrues par rapport aux versions antérieures de Java.

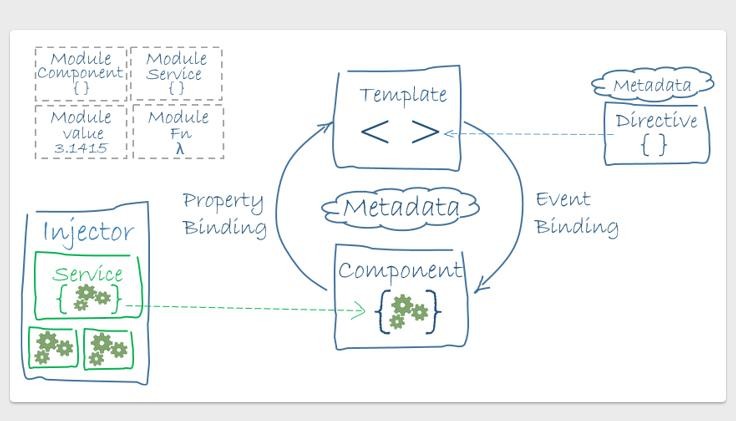
##### Angular v15



*Figure 27 Logo de Angular*

Angular est un framework pour clients, open source, basé sur TypeScript et codirigé par l'équipe du projet « Angular » chez Google ainsi que par une communauté de particuliers et de sociétés. Angular est une réécriture complète d'AngularJS, cadriciel construit par la même équipe. Il permet la création d’applications Web et plus particulièrement d'applications Web monopages : des applications Web accessibles via une page Web unique qui permet de fluidifier l’expérience utilisateur et d’éviter les chargements de pages à chaque nouvelle action. Le framework est basé sur une architecture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités. Un type d’architecture qui a largement fait ses preuves et qui

permet une forte maintenabilité et une amélioration du travail collaboratif (8)



*Figure 28 : architecture angular*

##### Spring boot 3.0



*Figure 29 Logo de Spring boot*

Spring est un framework de développement d'applications Java qui facilite la création d'applications d'entreprise robustes et évolutives. Spring est basé sur le paradigme d'inversion de contrôle (IoC) et encourage une conception modulaire et flexible grâce à l'injection de dépendances.

La version 3.0 de Spring, sortie en décembre 2009, a introduit plusieurs nouvelles fonctionnalités et améliorations par rapport à la version 2.0. Voici quelques différences clés entre Spring 3.0 et Spring 2.0 :

Configuration basée sur les annotations : Dans Spring 2.0, la configuration se faisait principalement à l'aide de fichiers XML. Cependant, dans Spring 3.0, une plus grande importance a été accordée à la configuration basée sur les annotations. Cela permet de définir les dépendances, les intercepteurs, les gestionnaires d'événements, etc., directement dans les classes avec des annotations spécifiques, réduisant ainsi la nécessité de configurations XML lourdes.

Spring Expression Language (SpEL) : Spring 3.0 a introduit le langage d'expression Spring (SpEL), qui permet d'évaluer des expressions complexes dans les configurations XML ou les annotations. Cela facilite la manipulation et l'accès aux valeurs des propriétés, aux variables, aux appels de méthode, etc., dans le cadre de la configuration Spring.

Améliorations de Spring MVC : Le framework Spring MVC a également été amélioré dans Spring 3.0, avec l'introduction de nouvelles fonctionnalités telles que la prise en charge des annotations pour les contrôleurs, la validation, la conversion de données et l'ajout de la prise en charge de RESTful web services.

Améliorations de la gestion des transactions : Spring 3.0 a apporté des améliorations significatives dans la gestion des transactions, notamment avec l'introduction d'une API de programmation déclarative pour les transactions, simplifiant ainsi la configuration et l'utilisation des transactions dans les applications.

Globalement, Spring 3.0 a permis une meilleure intégration avec les dernières technologies Java EE, une configuration plus simplifiée grâce aux annotations, une gestion des transactions améliorée et des améliorations dans le framework Spring MVC. Ces changements ont contribué à rendre le développement d'applications avec Spring plus efficace, flexible et en phase avec les évolutions du paysage technologique Java.

##### Kafka



*Figure 30 Logo de Kafka*

Apache Kafka est une plateforme de streaming distribuée, initialement développée par LinkedIn et maintenant gérée par la fondation Apache. Il est conçu pour gérer des flux massifs de données en temps réel, avec une grande échelle, une haute disponibilité et une faible latence

Kafka fonctionne selon le principe de journal de messages persistant, où les données sont publiées par des producteurs dans des topics, puis consommées par des consommateurs intéressés par ces topics spécifiques. Il garantit la persistance des données sur une période définie, ce qui permet aux consommateurs de récupérer et de traiter les données passées, même en cas de panne ou de reprise après une panne.

L'avantage de Kafka réside dans sa capacité à gérer de grands volumes de données et à garantir une faible latence, ce qui le rend adapté aux environnements à haute performance. Il permet également une intégration flexible avec différentes technologies et frameworks, ce qui facilite l'échange de données entre nos applications existantes.

En choisissant Kafka, nous avons opté pour une solution fiable, évolutive et tolérante aux pannes, qui nous permet de gérer efficacement le flux de données entre nos applications et notre SSA. Cela nous aide à assurer une communication fluide et sécurisée, tout en garantissant la cohérence et la disponibilité des données. (9)

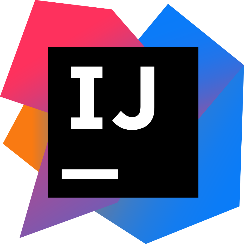
##### Maven



*Figure 31 Logo de Maven*

Maven est un outil de gestion de projet puissant qui favorise une approche cohérente et ordonnée du développement logiciel. Il offre une infrastructure solide pour le développement de projets de toutes tailles, tout en promouvant les meilleures pratiques de développement et en fournissant des fonctionnalités prêtes à l'emploi pour la gestion des dépendances, la compilation, les tests et bien plus encore (10).

##### IntelliJ IDEA



*Figure 32 Logo de IntelliJ IDEA*

IntelliJ IDEA est un environnement de développement intégré (IDE) qui facilite la création d'applications pour les professionnels de la technologie. Avec ses fonctionnalités avancées et son interface utilisateur ergonomique, IntelliJ IDEA vous permet de coder plus rapidement et avec une meilleure précision. Le soutien intégré pour les technologies de pointe et les frameworks populaires vous aide à rester productif et offre une expérience de développement sans heurts (11).

##### POSTMAN



*Figure 33 Logo de Postman*

Postman est un outil de développement d'API collaboratif qui simplifie le processus de création, de test, de documentation et de partage d'API. Il permet aux développeurs de créer des requêtes HTTP, d'envoyer des requêtes vers des endpoints spécifiques, de visualiser les réponses et de gérer les différentes variables et environnements nécessaires aux tests d'API. Postman offre une interface conviviale et des fonctionnalités avancées telles que la création de collections d'API, les tests automatisés, la génération de documentation et la collaboration en équipe (12).

### DevOps

DevOps favorise une culture de collaboration et de communication entre les équipes techniques, ce qui permet de briser les silos organisationnels et de favoriser la responsabilité partagée. Il met l'accent sur l'automatisation des processus de développement, de test et de déploiement, ainsi que sur l'utilisation d'outils et de technologies pour soutenir ces processus.

##### GitLab CI



*Figure 34 Logo de GitLab*

GitLab est une plateforme de développement de logiciels complète qui permet aux équipes de collaborer de manière efficace et de fournir rapidement des produits de haute qualité. Reposant sur Git, GitLab fournit une gestion du code source, un suivi des problèmes, un suivi des demandes de fusion, des intégrations continues et bien plus encore, le tout dans une seule application cohérente et facile à utiliser (13).

GitLab CI est une solution d'intégration continue intégrée au système de gestion de code source GitLab. Il s'agit d'un outil qui permet d'automatiser le processus d'intégration, de test et de déploiement des applications logicielles.

Avec GitLab CI, vous pouvez définir des pipelines d'intégration continue pour votre projet, qui sont des ensembles de tâches et de configurations exécutées automatiquement à chaque fois que vous effectuez des modifications sur votre code source. Ces pipelines peuvent inclure des étapes telles que la compilation du code, l'exécution de tests automatiques, la génération de rapports, le déploiement vers des environnements de développement ou de production, et bien plus encore.

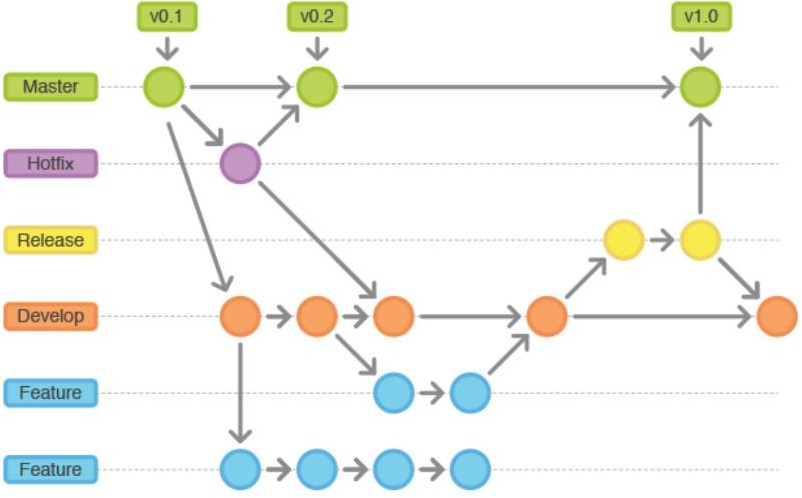
##### Git



*Figure 35 Logo de Git*

Git est un système de contrôle de version open-source distribué, conçu pour gérer efficacement les projets de toutes tailles. Il est rapide, évolutif et offre une gestion flexible des branches. Git facilite la collaboration entre les développeurs, permettant à chacun de travailler sur une copie locale du projet et de fusionner facilement les modifications apportées par les autres membres de l'équipe. Avec Git, vous pouvez suivre l'historique des modifications, revenir à des versions précédentes du code et résoudre les conflits de manière efficace. (14)

##### Git Flow



*Figure 36 Gitflow Workflow*

Le Gitflow Workflow est un modèle de flux de travail Git qui définit une structure de branches stricte basée sur la gestion des versions du projet. Il offre un cadre solide pour la gestion de projets de plus grande envergure.

Dans le cas de l'application Pléiade, l'utilisation du Gitflow Workflow répond à la nécessité d'organiser et de planifier le cycle de publication du projet. Ce flux de travail n'introduit pas de nouveaux concepts ou de nouvelles commandes, mais se concentre sur les branches et leur interaction. Il attribue des rôles spécifiques à chaque branche et définit quand et comment elles doivent interagir entre elles. En plus des branches de fonctionnalités, il utilise des branches distinctes pour la préparation, la maintenance et la version des releases.

L'utilisation du Gitflow Workflow présente plusieurs avantages pour le projet, notamment la possibilité de bénéficier des fonctionnalités des branches de fonctionnalités telles que les pull requests, les expérimentations isolées et une collaboration plus efficace.

La branche "pfv" est créée à partir de la branche "master" et contient l'historique complet du projet.

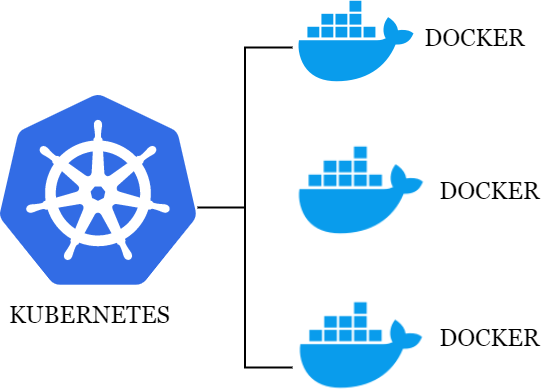
Chaque nouvelle fonctionnalité est développée dans sa propre branche "feature", qui a pour branche parente la branche "pfv". Une fois la fonctionnalité complétée, elle est fusionnée avec la branche "pfv".

Les branches de maintenance ou "prod" sont utilisées pour apporter rapidement des correctifs aux versions en production. Les branches "prod" sont similaires aux branches "release" et "feature", mais elles sont créées à partir de la branche "master" au lieu de la branche "pfv". On utilise ces branches pour le déploiement en production sur kubernetes

La branche "release" est créée à partir de la branche "pfv". Une fois terminée, elle est fusionnée avec les branches "pfv" et "master".

L'environnement de test (recette) utilise la branche "develop" qui est mise à jour quotidiennement. L’environnement de production utilise la branche "prod".

##### Kubernetes et Docker



*Figure 37 : Kubernetes et Docker*

Dans le cadre de notre projet, nous avons utilisé Kubernetes et Docker pour le déploiement et la gestion de notre application.

Docker est une plateforme open-source populaire de virtualisation légère basée sur des conteneurs. Il permet d'emballer une application et toutes ses

dépendances dans un conteneur autonome, garantissant ainsi que l'application s'exécutera de manière cohérente et isolée sur n'importe quel environnement. Docker nous a offert une solution portable et reproductible pour le packaging de notre application, simplifiant ainsi le déploiement et l'exécution de nos microservices.

Kubernetes, quant à lui, est un système d'orchestration de conteneurs open- source qui permet de gérer et de coordonner efficacement les conteneurs Docker dans un environnement distribué. Il offre des fonctionnalités avancées telles que le déploiement automatisé, la mise à l'échelle dynamique, la gestion des ressources, l'équilibrage de charge et la haute disponibilité. Kubernetes a été notre choix pour orchestrer nos conteneurs Docker, assurant une gestion robuste et évolutive de nos microservices.

En utilisant Docker, nous avons pu encapsuler nos microservices, leurs dépendances et la configuration associée dans des conteneurs Docker. Cela nous a permis de garantir que notre application fonctionnerait de manière cohérente et prévisible, indépendamment de l'environnement d'exécution.

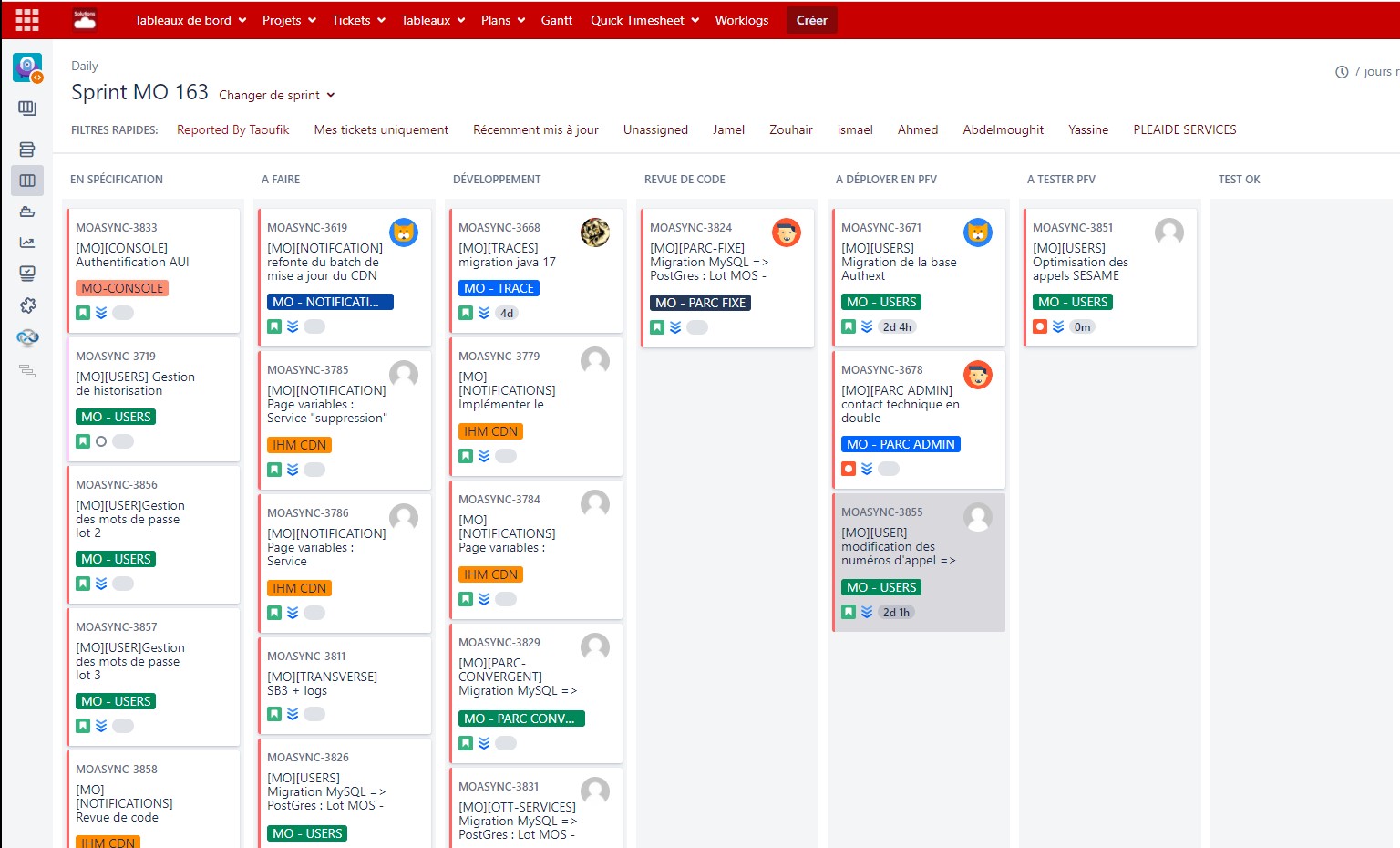
Kubernetes a ensuite pris en charge le déploiement, la gestion et la mise à l'échelle de ces conteneurs Docker. Nous avons créé des pods Kubernetes pour encapsuler et gérer nos conteneurs, assurant ainsi une exécution fiable et équilibrée de nos microservices. Kubernetes nous a également permis de gérer les ressources, de surveiller l'état de nos conteneurs, de gérer les mises à jour sans temps d'arrêt et de garantir la haute disponibilité de notre application.

L'utilisation conjointe de Docker et de Kubernetes nous a offert une solution complète pour le déploiement et la gestion de notre application. Docker nous a permis de créer des conteneurs autonomes et portables, tandis que Kubernetes nous a fourni les outils nécessaires pour orchestrer, équilibrer la charge et gérer efficacement ces conteneurs dans un environnement distribué.

En résumé, l'utilisation de Docker et Kubernetes a été essentielle pour simplifier le déploiement, améliorer la portabilité, garantir la gestion des ressources et assurer la haute disponibilité de notre application. Ces deux technologies complémentaires nous ont permis de tirer parti des avantages de la virtualisation légère et de l'orchestration avancée des conteneurs, contribuant ainsi au succès de notre projet.

### Les Outils de collaboration

##### JIRA



*Figure 38 L'interface Jira*

JIRA est une solution qui permet de gérer le processus de développement de logiciels en offrant des fonctions sociales qui facilitent les échanges en temps réel entre toutes les parties concernées : développeurs, équipes techniques, utilisateurs professionnels et clients. JIRA propose des fonctionnalités et des outils performants à chaque phase du cycle de vie de développement.

##### Microsoft Teams



*Figure 39 Microsoft Teams*

Microsoft Teams une application de collaboration conçue pour le travail hybride afin que vous et votre équipe restiez informés, organisés et connectés, le tout au même endroit.

Nous l'utilisons principalement pour garder un canal de communication avec l'équipe à Paris, pour avoir des réunions quotidiennes et/ou pour poser des questions si nous avions un blocage.

##### Confluence



*Figure 40 Logo de Confluence*

Confluence est un outil de collaboration qui permet aux équipes de travailler ensemble pour créer, organiser et discuter de leur travail dans un espace centralisé et accessible à tous. Que vous ayez besoin de créer une documentation de projet, de partager des connaissances internes, de collaborer sur des spécifications de produit ou de suivre les progrès de votre équipe, Confluence vous permet de le faire de manière transparente et efficace (15).

### Outils de base de données :

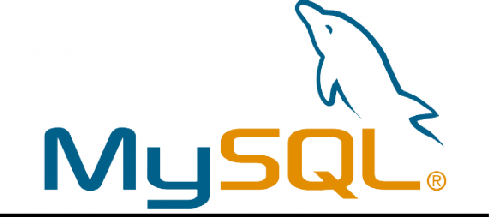
##### PostgreSQL



*Figure 41 Logo de PostgreSQL*

PostgreSQL est un système de gestion de base de données relationnelle (SGBDR) open-source, puissant et hautement extensible. Il offre une grande fiabilité, une intégrité des données élevée et des fonctionnalités avancées pour la gestion et l'analyse des données. PostgreSQL est reconnu pour sa conformité aux normes SQL et sa capacité à gérer des charges de travail complexes et volumineuses (16).

##### MySQL



*Figure 42 : Logo de MySQL*

MySQL est un serveur de bases de données relationnelles SQL développé dans un souci de performances élevées en lecture, ce qui signifie qu'il est davantage orienté vers le service de données déjà en place que vers celui de mises à jour fréquentes et fortement sécurisées. Il est multithread et multi- utilisateur.

C'est un logiciel open source, développé sous double licence selon qu'il est distribué avec un produit libre ou avec un produit propriétaire. Dans ce dernier cas, la licence est payante, sinon c'est la licence publique générale GNU (GPL) qui s'applique. (17)

# Conclusion

Dans ce chapitre, nous avons abordé l'étude technique du projet en nous concentrant sur les choix architecturaux que nous avons faits pour mener à bien ce travail. Nous avons détaillé les différentes architectures que nous avons sélectionnées et expliqué les raisons de ces choix. Nous avons également analysé les avantages et les inconvénients de chaque architecture afin de prendre des décisions éclairées.

Le chapitre suivant se concentrera sur la présentation des interfaces que nous avons développées tout au long du projet, ainsi que sur la phase de déploiement où nous expliquerons comment nous avons mis en place notre solution.

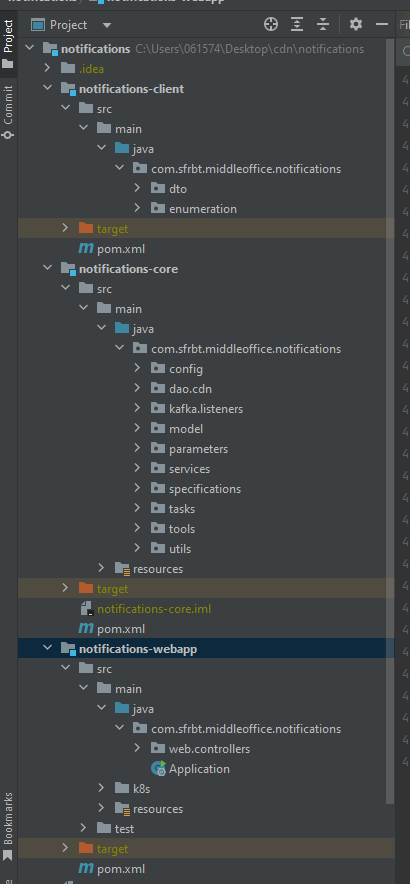
# Chapitre 4 : Mise en œuvre et réalisation du projet CDN

Dans ce chapitre, nous aborderons la phase de mise en œuvre de l'application, où nous décrirons en détail la structure du projet et la méthodologie que nous avons suivie tout au long de sa réalisation. Nous mettrons en évidence certaines fonctionnalités clés en présentant des captures d'écran illustrant leur utilisation. Vous découvrirez ainsi concrètement comment l'application a été développée et comment elle répond aux besoins identifiés. Nous vous présenterons également les différentes étapes de la démarche suivie, en mettant en évidence les choix et les décisions pris tout au long du processus de développement.

Nous ferons une séparation entre le back-end et le front-end pour sécuriser la bonne explication du projet.

## 4.1 Structure du projet

### Back-end

Notification module :

*Figure 43 : architecture de CDN back end*

Dans l'architecture Spring Boot, il est courant de nommer les modules en fonction de leur rôle et de leurs responsabilités spécifiques. Dans votre cas, vous pouvez appeler les modules de la manière suivante :

**Module\_client :** Ce module peut être nommé "ClientService" ou "ClientModule". Il est responsable de la gestion des fonctionnalités liées aux clients, telles que la création, la lecture, la mise à jour et la suppression de données client. Il peut également inclure des fonctionnalités spécifiques liées aux clients, telles que l'authentification et l'autorisation des utilisateurs.

**Dossier Dto :** est un ensemble de classes ou de structures de données utilisées pour transférer des informations de manière efficace et structurée entre différentes parties d'une application.

**Enumération :** contient les différents enum

**Module\_webapp :** Ce module peut être nommé "WebApp" ou "WebModule". Il est responsable de la partie interface utilisateur de l'application. Il inclut les contrôleurs, les vues et les ressources liées à l'interface utilisateur. Ce module expose les fonctionnalités de l'application via des API REST ou des pages web pour l'interaction avec les utilisateurs.

**Controllers** : où les fichiers de contrôleurs sont regroupés. Les contrôleurs sont des composants responsables de la gestion des requêtes HTTP et de la coordination des actions à effectuer en réponse à ces requêtes. Ils jouent un rôle essentiel dans l'architecture d'une application web, car ils reçoivent les demandes des utilisateurs, traitent les données nécessaires et renvoient les réponses appropriées.

**Module\_core :** Ce module peut être nommé "Core" ou "CoreModule". Il contient le cœur de l'application, les services métier, les entités, les opérations de persistance et les règles de validation. Ce module représente la logique métier centrale de l'application et fournit les fonctionnalités de base utilisées par d'autres modules.

projet

**Config :** contient la configuration du Kafka et la configuration générale du

**DAO :** "Data Access Object" ou on a les différents repository , pour séparer

la logique métier de l'accès aux données.

**Kafka listeners :** est un composant utilisé dans les applications qui se connectent à un cluster Apache Kafka pour consommer des messages à partir des topics Kafka**.** Dans une architecture basée sur Kafka, les applications peuvent publier des messages sur des topics spécifiques, et les écouteurs Kafka permettent de consommer ces messages de manière asynchrone. Les écouteurs sont responsables d'écouter les partitions des topics Kafka auxquels ils sont abonnés, de recevoir les messages et de les traiter.

**Spécifications :** contient les critères de recherche utiliser sur JPA spécification

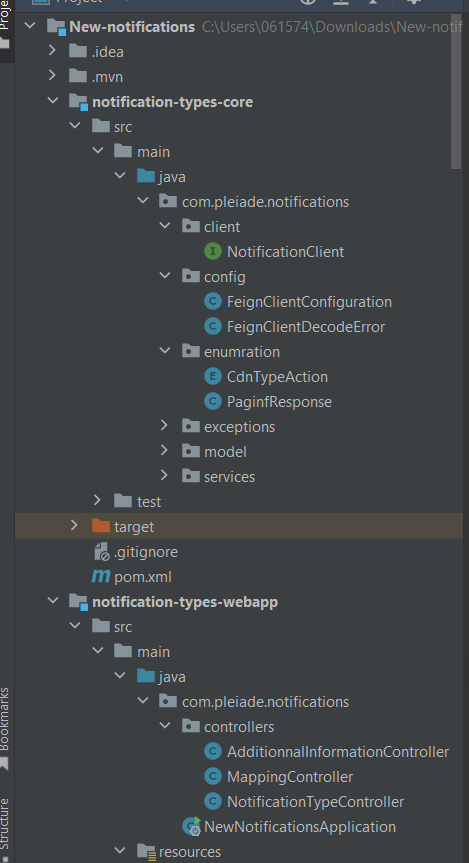
**Model :** les tables de la base de données suivant le diagramme de classe.

**Services :** Ce dossier contient les classes de services qui encapsulent la logique métier de l'application et effectuent les opérations nécessaires sur les entités.

En nommant les sous-modules de cette manière, vous indiquez clairement leur rôle et leur responsabilité dans l'architecture globale de l'application. Cela facilite la compréhension et la maintenance du code, ainsi que la collaboration au sein de l'équipe de développement.

L’intérêts de cette architecture c’est la capacité de la réutilisation des différents sous-module dans d’autre module. Par exemple si on veut réutiliser les classes de la partie core dans un autre module, on doit juste importer le sous module core en forme JAR

Notification nouveau back end:

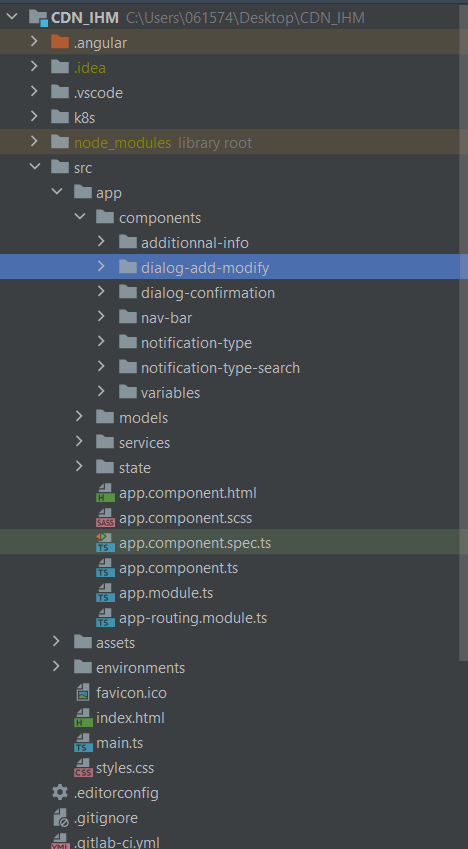


*Figure 44 : new back end*

Pour garder les services qui sont utilisés et ne pas perturber la performance de quelque API en ajoutant des paramètres ou bien changer un traitement, on a décidé de travailler sur un back end qui consomme les services de l’ancien back

end en utilisant ‘FEIGNCLIENT’ et ajouter les services qu’on veut sur le nouveau back-end en attendant le déploiement final en production.

Front-end



*Figure 45 : architecture front end*

L'architecture en composants est un paradigme de conception où une application est divisée en composants autonomes et réutilisables. Chaque composant est responsable de son propre rendu, de sa logique et de sa gestion des

données. Les composants interagissent entre eux pour former une application fonctionnelle.

**Dossier Model**: Le dossier Model est destiné à contenir les classes et les interfaces qui définissent la structure des données de notre application. Cela peut inclure des modèles de données, des entités ou des interfaces liées à l'API. Les fichiers présents dans ce dossier fournissent une représentation claire des données utilisées par l'application.

**Dossier Service**: Le dossier Service est responsable de la logique métier de notre application. Il contient les services Angular qui interagissent avec les différentes sources de données, telles que les appels d'API, la gestion de l'état de l'application, etc. Les services fournissent des fonctionnalités réutilisables et facilitent la communication entre les composants et les modèles de données.

**Dossier Components**: Le dossier Components abrite les composants Angular réutilisables et les vues de notre application. Les composants sont les blocs de construction de l'interface utilisateur et peuvent être réutilisés dans différentes parties de l'application. Ils encapsulent la logique spécifique à l'interface utilisateur et interagissent avec les services pour récupérer et afficher les données.

**Dossier K8s (Kubernetes)**: Le dossier K8s est utilisé pour la configuration de Kubernetes, une plateforme open-source de gestion et d'orchestration de conteneurs.

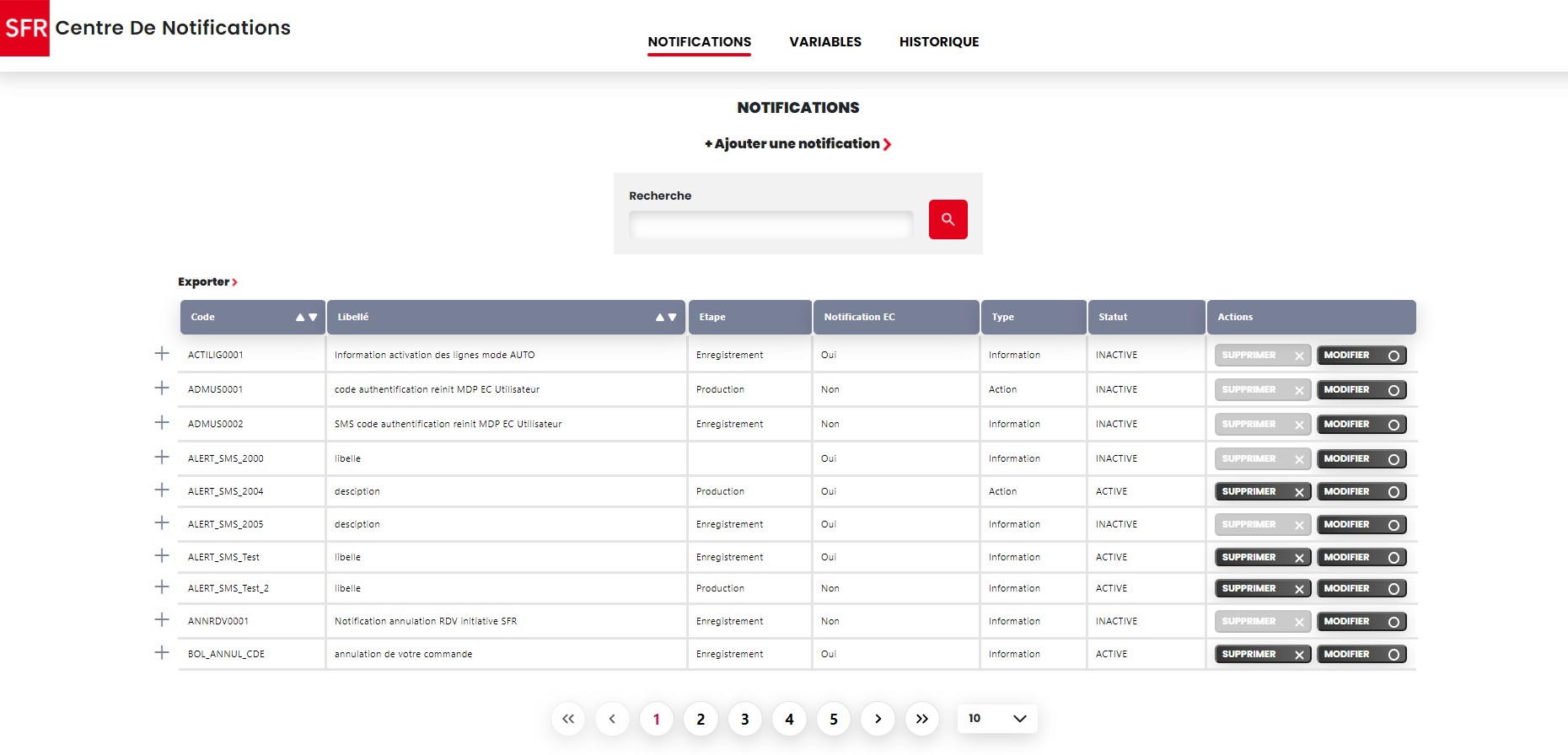
## Présentation des interfaces du IHM CDN et d’autre interfaces qui consomme le module notification :

### Page Notifications :

Nous nous retrouvons ici dans la page d'accueil de notre centre de notifications. On peut ajouter, modifier et supprimer une notification.

On peut aussi rechercher une notification par son code, afficher son détail (mapping des variables CDN-DMC), trier par ordre ainsi qu'exporter toutes les notifications existantes sous formats xlsx.

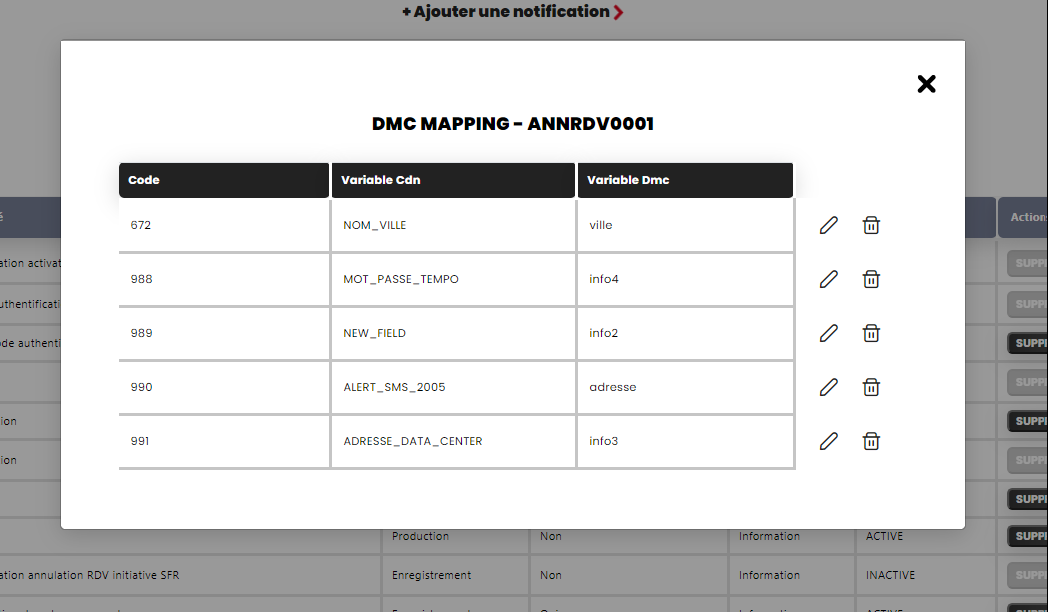
Aussi en peut appliquer plusieurs filtres suivant la statut type ou bien l’étape avec la pagination.



*Figure 46 Notification Page*

On peut également voir le mapping de chaque notification en cliquant sur l'onglet <+> à gauche du code de la notification.

Nous aurons ainsi la pop-up suivante :

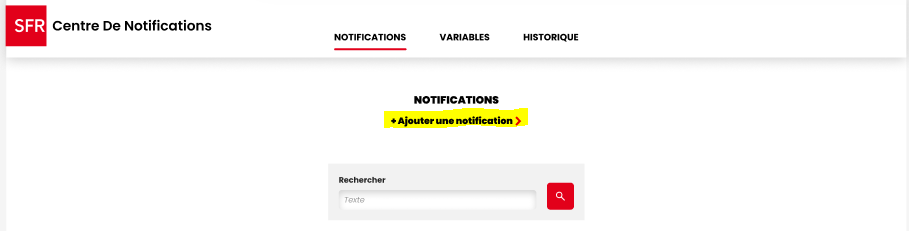


*Figure 47 : mapping interface*

##### A partir de cette pop-up, on peut ajouter ou supprimer un mapping.

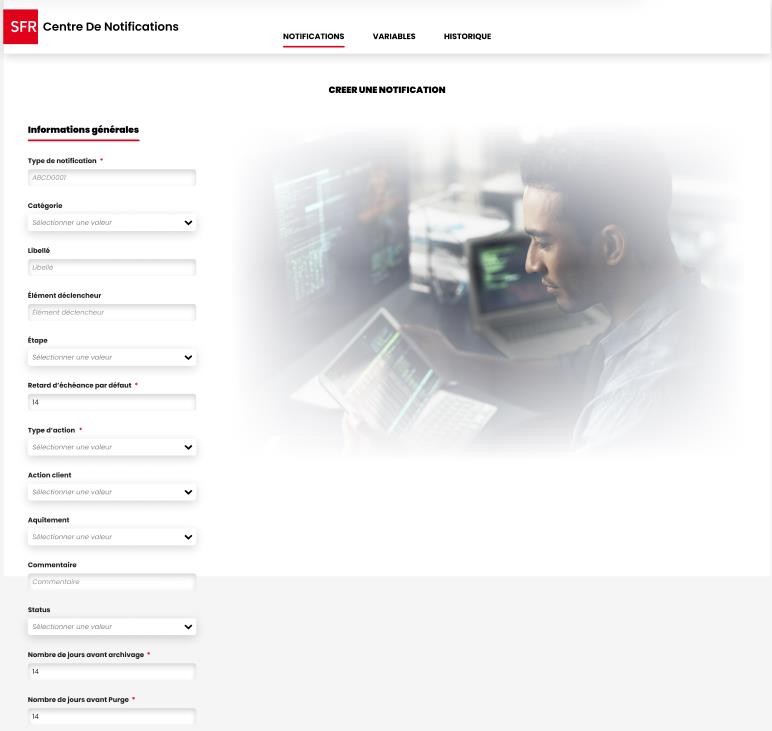
**Ajout et Modification**

Pour ajouter une notification, on clique sur le bouton "Ajouter une notification" de la page d'accueil.



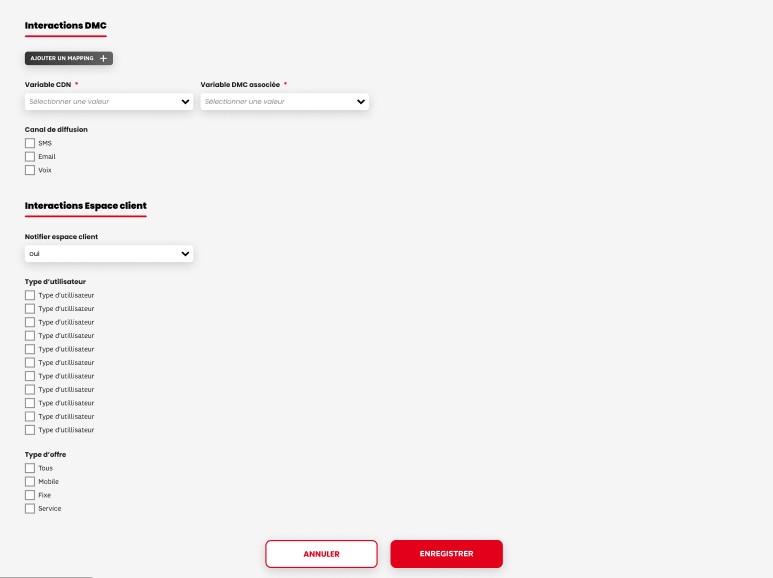
La page suivant s'affiche :

*Figure 48 : Ajout button*



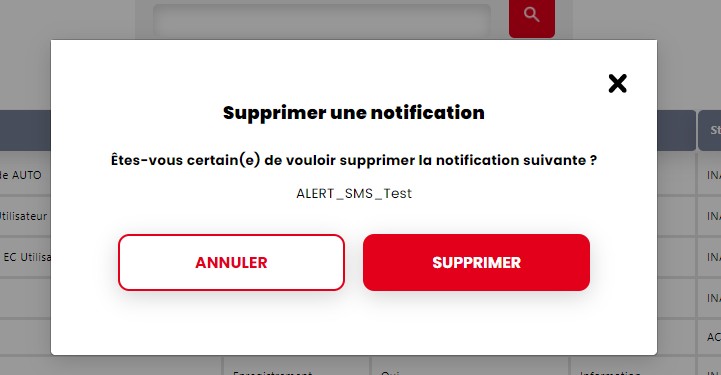
*Figure 49 : page d'ajout et modification*

On applique plusieurs validateurs de requis de nombre de caractère ou bien d’unicité



*Figure 50 : page d'ajout et modification 2*

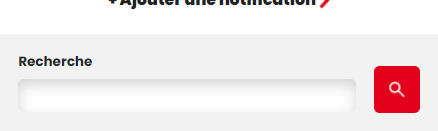
##### Suppression de notification :

On n’a pas une suppression totale de la notification car elle peut être utiliser dans un autre service donc la suppression change le statut du configuration- notification vers Inactive

*Figure 51 : suppression pop-up*

##### Recherche service

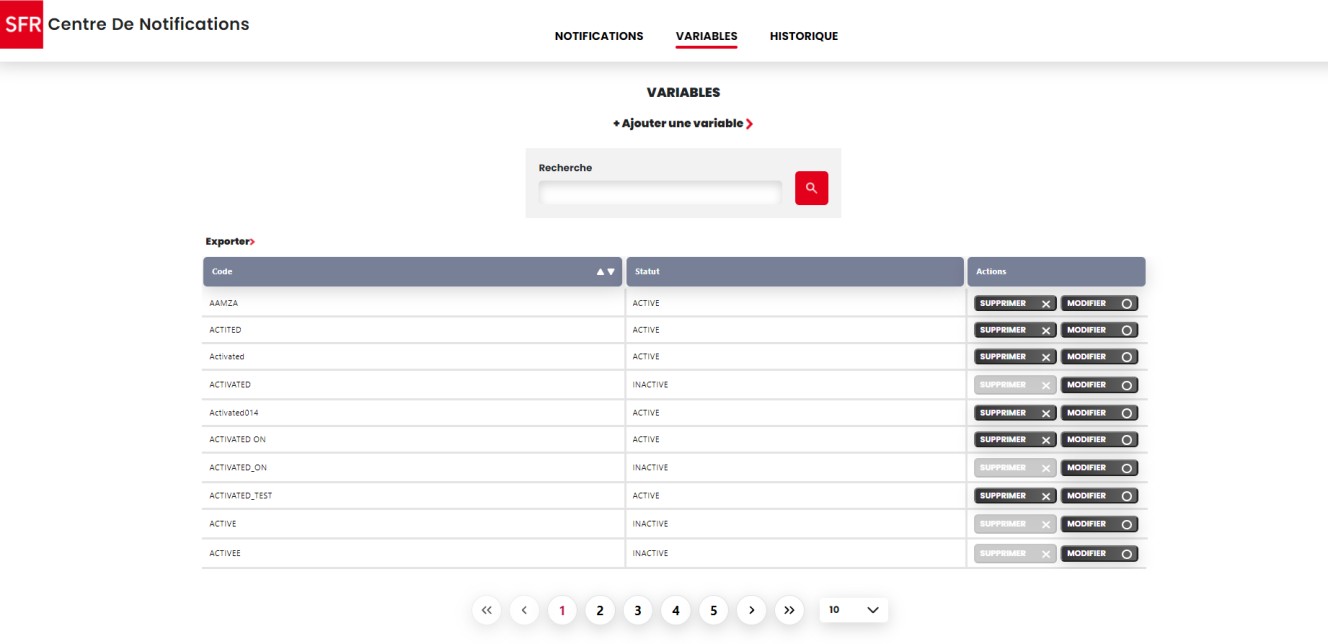
Service de recherche globale et générique en back-end et en front-end, on cherche le code suivant la page active



*Figure 52 : recherche formulaire*

### Page Variable :

Nous nous retrouvons ici dans la page variable de notre centre de notifications. On peut ajouter, modifier, supprimer et rechercher une variable.



*Figure 53 : page variable*

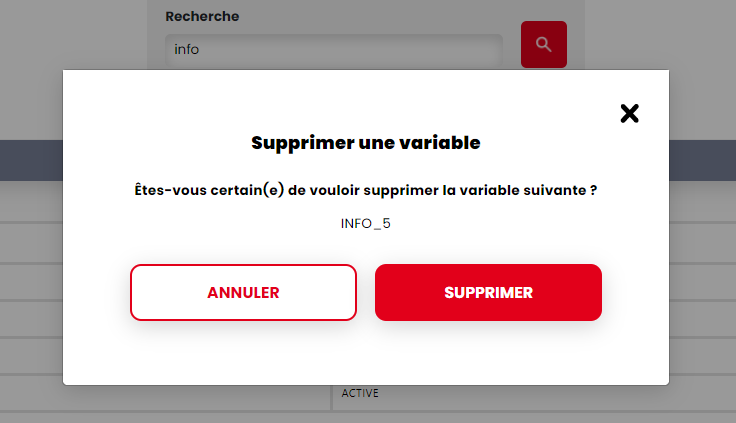
##### Ajout et Modification :

Pop-Up pour ajouter ou bien modifier une variable suivant des critères d’unicité



*Figure 54 Pop-up d’ajout et modification*

##### Suppression d’une variable CDN



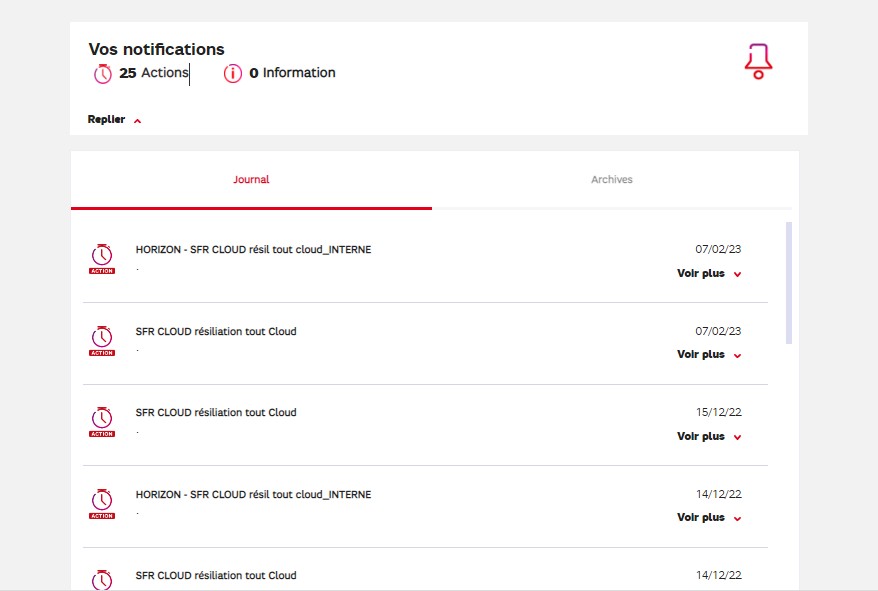
*Figure 55 : suppression variable*

Comme déjà mentionner sur notification, la même chose ici sur variable, on ne Supprime pas totalement la variable.

### Page Historique

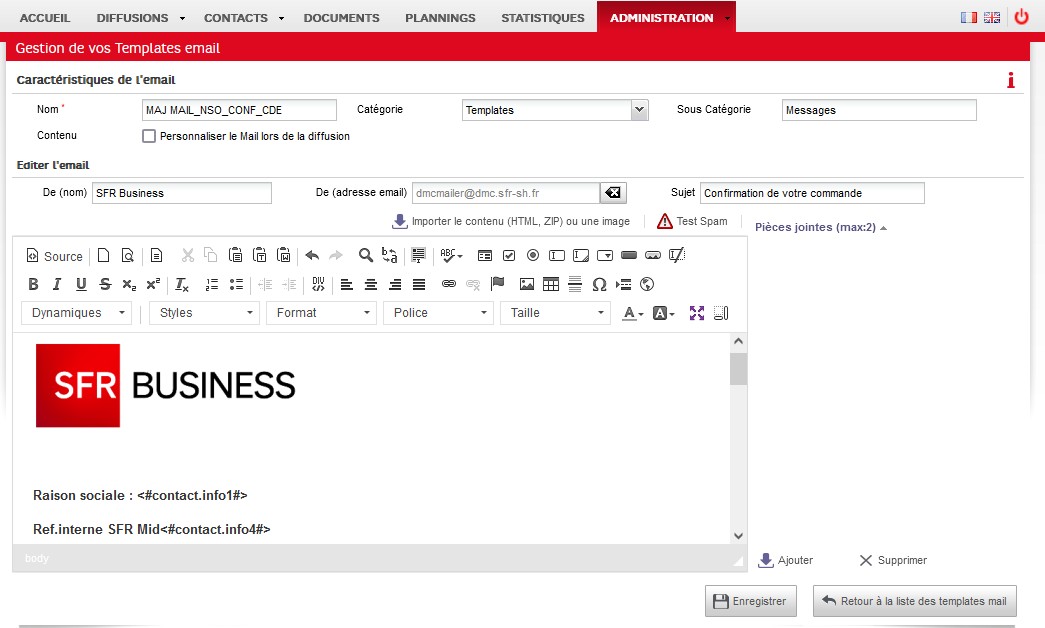
La page présente la liste des notifications qui ont été diffusées en mentionnant la date de diffusion, les récepteurs et d’autres informations qui seront utiles pour l’espace marketing. Cette page va avoir plusieurs critères de filtrage et de regroupement suivant le type de notification, le canal de diffusion et suivant le récepteur. Tout cela en récupération des données de la part de la connexion avec DMC.

### Notification dans l’Espace Client :

Ci-dessous une fenêtre montrant les notifications dans l’Espace Client (autre consommateur du module Notification) :

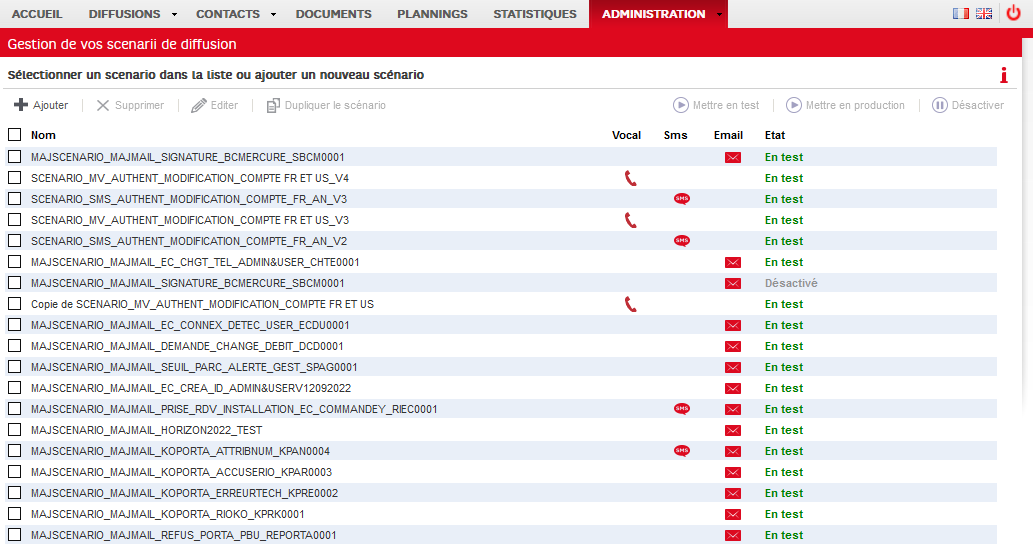
*Figure 56 : notification d'espace client*

### IHM DMC :

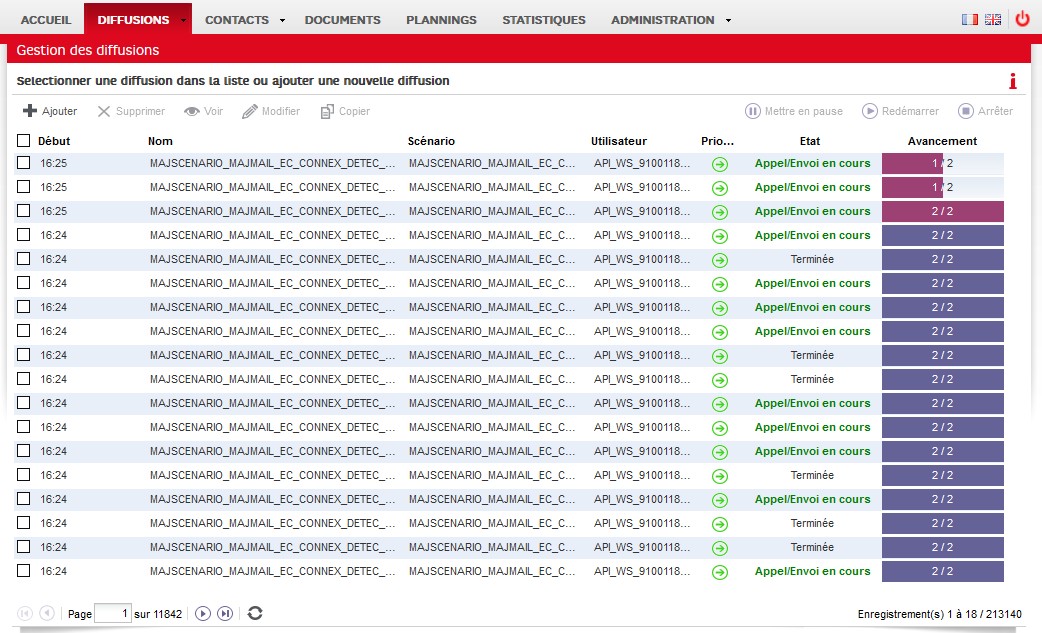
L’IHM dédiée à la configuration et à la préparation des Template qui seront diffusés vers les récepteurs qu’on a déjà configurés au niveau de l’IHM CDN. En premier lieu, on crée notre Template qui sera diffusé avec des variables qu’on appelle variable DMC, puis on paramètre la notification sur l’IHM CDN.

*Figure 57 : création Template*

Après le paramétrage et le mapping sur DMC, notre module notification offre un service qui donne la possibilité de déclencher la création d’une diffusion vers les récepteurs via les canaux de diffusion sélectionnés, et on configure la diffusion depuis cette interface :



*Figure 58 : Interface de scenario*

On peut lancer et observer l’état de diffusion depuis l’interface ci-dessous. En entrant vers chaque ligne, on peut voir le statut de la notification si elle a bien été diffusée ou s’il y a un problème avec des informations du récepteur, qui sont des informations confidentielles qu’on ne peut pas montrer via une capture d’écran.

*Figure 59 : Interface de diffusion*

# Conclusion

En conclusion de ce chapitre, nous avons examiné la mise en œuvre et la réalisation du projet de Centre de Notification (CDN). Notre objectif était de développer un système de notification efficace et fiable pour permettre la diffusion d'informations importantes à un large éventail d'utilisateurs.

Nous avons mis en place une architecture robuste pour le back-end du CDN, en utilisant des microservices pour séparer les différentes fonctionnalités du système. Cette approche modulaire nous a permis de développer et de gérer chaque aspect du système de manière indépendante, offrant ainsi une flexibilité et une évolutivité accrues.

Du côté du front-end, nous avons conçu une interface utilisateur intuitive et conviviale pour permettre aux utilisateurs de gérer leurs préférences de notification et de recevoir les informations pertinentes. Nous avons accordé une attention particulière à l'expérience utilisateur en créant des interfaces claires et réactives, facilitant ainsi la navigation et l'interaction avec le système.

# Conclusion Générale

Notre projet de fin d'études, mené au sein de Intelcia IT SOLUTIONS, a abouti à une refonte complète de l'application "CDN IHM". L'objectif principal de cette application est de gérer les notifications, les variables et la configuration des paramétrages entre le centre de notification et l'application de diffusion multi-canaux. Nous avons débuté par une étude approfondie de la solution existante afin de mettre en évidence ses limites. Ces dernières étaient principalement liées à l'architecture en place ainsi qu'aux choix technologiques effectués pour la partie backend et frontend. Par la suite, notre responsabilité a été d'établir l'architecture de la nouvelle solution, en prenant en compte la collecte des besoins, la conception et la réalisation.

Grâce à cette refonte, nous avons réussi à résoudre les problèmes techniques liés au framework backend et frontend. Le choix de l'Angular 15, Java 17 et Spring Boot 3.0 en tant que frameworks backend et frontend a permis de remédier aux problèmes liés aux temps de réponse ainsi qu'aux aspects fonctionnels de l'application.

À l'heure actuelle, notre solution répond quasiment à tous les objectifs énoncés au début de notre stage. En effet, nous avons pu développer tant les fonctionnalités que les exigences non fonctionnelles identifiées dès le départ. À ce stade, nous avons réalisé environ 70% du projet, nous reste encore à finaliser la page historique prévue pour le premier août.

En plus de notre travail de refonte de l'application "CDN IHM", j'ai également été chargé de traiter les anomalies sur différents modules de l'application existante. Cela comprenait l'identification, l'analyse et la résolution des problèmes rencontrés, afin d'assurer le bon fonctionnement de l'application dans son état actuel.

En parallèle, j'ai également participé aux activités de maintenance de l'application, en veillant à ce que les mises à jour et les corrections nécessaires soient effectuées régulièrement. Cela comprenait la gestion des correctifs, la surveillance des performances et la prise en compte des retours utilisateurs pour garantir une expérience utilisateur optimale.

Grâce à mon implication dans la résolution des anomalies et dans la maintenance de l'application, nous avons pu améliorer la stabilité et la fiabilité de l'ensemble du système, offrant ainsi une meilleure expérience aux utilisateurs finaux.

# Webographie

1. (s.d.). Récupéré sur https://[www.intelcia.com/fr/it-solutions](http://www.intelcia.com/fr/it-solutions)
2. (s.d.). Récupéré sur https://[www.intelcia.com/fr/le-groupe](http://www.intelcia.com/fr/le-groupe)
3. (s.d.). Récupéré sur https://confluence-jmsp.private.sfr.com
4. (s.d.). Récupéré sur https://marjory.io/architecture-microservices-guide- complet
5. (s.d.). Récupéré sur https://confluence-jmsp.private.sfr.com
6. (s.d.). Récupéré sur [https://medium.com/@anoukzouaoui/m%C3%A9thode-agile-](https://medium.com/%40anoukzouaoui/m%C3%A9thode-agile-) a2d65e319ce6
7. (s.d.). Récupéré sur https://docs.oracle.com/en/java/javase/17/index.html
8. (s.d.). Récupéré sur https://fr.wikipedia.org/wiki/Angular
9. (s.d.). Récupéré sur https://kafka.apache.org/ 10.(s.d.). Récupéré sur https://maven.apache.org/ 11.(s.d.). Récupéré sur https://[www.jetbrains.com/idea/](http://www.jetbrains.com/idea/)

12.(s.d.). Récupéré sur https://[www.postman.com/product/what-is-postman](http://www.postman.com/product/what-is-postman) 13.(s.d.). Récupéré sur https://about.gitlab.com/

14.(s.d.). Récupéré sur https://git-scm.com/

15.(s.d.). Récupéré sur https://[www.atlassian.com/fr/software/confluence](http://www.atlassian.com/fr/software/confluence) 16.(s.d.). Récupéré sur https://[www.postgresql.org/](http://www.postgresql.org/)

17.(s.d.). Récupéré sur https://fr.wikipedia.org/wiki/MySQL