Rapport de Stage

Développement d'une API pour une application web de datascience

Fabien ALLEMAND

Année 2023/2024

Stage de deuxième année réalisé à

l'Armée de l'Air et de l'Espace

du 1er juillet au 30 août 2024

en vue de l'obtention des diplômes d'ingénieur

Informatique et réseaux de Télécom Physique Strasbourg

et

Généraliste d'un numérique responsable de Télécom SudParis

Maître de stage : Yohann LE MEUR

Encadrant Télécom Physique Strasbourg : Adlane HABED

Encadrant Télécom SudParis : Emmanuel MONFRINI

Rapport de stage

Développement d'une API pour une application web de datascience

Fabien ALLEMAND

Année 2023/2024

Stage de deuxième année réalisé à

l'Armée de l'Air et de l'Espace

du 1er juillet au 30 août 2024

en vue de l'obtention des diplômes d'ingénieur

Informatique et réseaux de Télécom Physique Strasbourg

et

Généraliste d'un numérique responsable de Télécom SudParis

Fabien ALLEMAND
2 Chemin du Chêne
30700, Saint Quentin la Poterie
06 07 07 63 89
fabien.allemand@etu-unistra.fr
fabien.allemand@telecom-sudparis.eu

Télécom Physique Strasbourg
Pôle API, 300 boulevard Sébastien Brant
67400, Illkirch-Graffenstaden
03 68 85 45 10
tps-accueil@unistra.fr

Télécom SudParis
9 rue Charles Fourier
91000, Evry-Courcouronnes
01 60 76 40 40
infos@telecom-sudparis.eu

Escadron des systèmes d'information opérationnels et de cyberdéfence
Base aérienne 118 « Colonel Rozanoff »
1061 avenue du Colonel Kw Rozanoff
40118, Mont-de-Marsan
yohann.le-meur@intradef.gouv.fr

Maître de stage : Yohann LE MEUR
Encadrant Télécom Physique Strasbourg : Adlane HABED
Encadrant Télécom SudParis : Emmanuel MONFRINI

Remerciements

Ces deux mois de stage au sein de l'Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence (ESIOC) de l'Armée de l'Air et de l'Espace sont une expérience d'une grande richesse marqués par de nombreuses découvertes techniques, managériales et militaires. Je tiens à remercier toutes les personnes qui m'ont accompagné et aidé tout au long du stage et lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes remerciements à mon maître de stage, Yohann LE MEUR, data scientist à l'ESIOC, pour son accueil, le temps passé ensemble et le partage de son expertise.

Je remercie en particulier Auriane GUEDEZ, datat scientist et responsable du projet OMEGA, qui m'a supervisé et avec qui j'ai collaboré tout au long du stage. Son accompagnement et ses explications m'ont permis de mieux appréhender les missions qui m'ont été confiées mais aussi d'acquérir de nouvelles compétences.

Je tiens à souligner le soutien de Maxime FOURCADE, responsable de la platefrome de développement de l'ESIOC, qui a fait en sorte que le stage se déroule au mieux notamment d'un point de vue technique.

Je remercie également Alexandre LE FOLL, étudiant à Télécom Nancy en stage à l'ESIOC pour sa bonne humeur et son esprit d'entre aide.

Acronymes

AAE Armée de l'Air et de l'Espace. iii, 1, 3–5, 12, 15, 16, 27, 31, 32, 37, 41, 50, 51

BA Base Aérienne. ix, 1, 3–5

CCT Cadre de Cohérence Technique. 15, 16
CDC Centre de Détection et de Contrôle. 4
CEAM Centre d'Expériences Aériennes Militaires. 3, 4
CEAM Centre d'Expertise Aérienne Militaire. 4
CEMA Centre d'Essais des Matériels Aéronautiques. 3
CEV Centre d'Essais de Vol. 3
CTAAE Commandement Territorial de l'Armée de l'Air et de l'Espace. 15

DIL Département d'Ingénierie Logicielle. vii, 6, 13, 16–18, 23
DIRISI Direction Interarmées des Réseaux d'Infrastructure et des Systèmes d'Information. 12, 15, 16
DND Département du Numérique et de la Donnée. vii, 1, 5, 6, 16–18, 31, 32
DR Diffusion Restreinte. ix, 13, 17, 19, 32, 46, 48
DRHAAE Direction des Ressources Humaines de l'Armée de l'Air et de l'Espace. 15–18, 24, 25, 27, 49, 51
DSS Data Science Studio. vii, ix, xi, 1, 7–14, 16–29, 31–35, 37, 49–51

ESIOC Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence. iii, vii, ix, xi, 1, 4–7, 12, 15–17, 24, 27, 31, 35, 37, 41, 43, 44, 49, 51

IA Intelligence Artificielle. 7, 12
IDF Ile De France. 23, 49

NP Non Protégé. 12, 13, 19, 32, 33

OMEGA ???. iii, vii, ix, xi, 1, 6, 8, 15–20, 23, 26, 27, 31, 35, 49

PAM Plans Annuels de Mutation. 15–17, 20, 23–25, 27, 49, 50
POC Proof Of Concept. 15–17, 24

REO Référentiel En Organisation. 15, 17, 23, 24, 27, 50
REPAIR REférentiel de Pilotage AIR. vii, 5, 6, 16, 17, 23–25

SI Systèmes d'Information. 12, 16, 17, 31, 51
SIC Systèmes d'Information et de Communication. 1, 3, 37, 39
SITU SITUation (des sous-officiers). 17, 23, 24, 27, 49

Glossaire

API Acronyme de Application Programming Interface (en français : interface de programmation d'application) est une bibliothèque logicielle ou un service web par lequel un logiciel offre des services à d'autres logiciels. Bien que le fonctionnement interne est rarement décrit, la documentation fournie permet d'utiliser correctement une API [27, 2]. Il y a deux API dans le projet OMEGA : une API dite "intermédiaire" en Java (développée par le DIL) et une autre intégrée à Dataiku DSS en Python dont j'avais la responsabilité. ix, xi, 10, 13, 15–20, 22–29, 37, 49, 50

back end Il correspond à la couche logicielle effectuant le traitement des données. De nombreux languages de programmation sont utilisés pour cette couche, entre autres Python et Java. Le back end du projet OMEGA est créé et maintenu par un data scientist du DND dans un projet Dataiku DSS. 13, 16–19, 23, 37

endpoint Dans le cas d'une API sous forme de service web, c'est un chemin d'accès (URL) sur lequel on peut envoyer des requêtes HTTP et attendre une réponse. Chaque endpoint, ou point de terminaison, remplit une fonction unique [10]. L'API Python que j'ai créé possède plus d'une dizaine de endpoints . vii, ix, 18–21, 23–29, 49, 50

entrepôt de données Souvent désigné par le terme anglais data warehouse, c'est une base de données décisionnelle (utilisée pour l'aide à la décision). Il est construit en collectant, ordonnant, journalisant et stockant des informations provenant de plusieurs bases de données opérationnelles. REPAIR est l'entrepôt de données de l'ESIOC. 6, 16, 17

framework Rarement désigné par le terme français cadriciel, il peut être vu comme une bibliothèque logicielle générique et faiblement spécialisée fournissant la structure (d'une partie) d'un logiciel. React est le framework JavaScript utilisé pour le développement front-end du projet OMEGA. 16

front end Produit du développement web frontal (ou front-end development), il correspond à l'interface graphique avec laquelle l'utilisateur peut intéragir. Reposant sur les langages HTML, CSS et JavaScript, il est soumis à de nombreuses contraintes d'ergonomie, d'adaptabilité et d'accessibilité. Le front end du projet OMEGA a été réalisé par un stagiaire et deux sergents du DIL. 16–19, 49

HTTPS L'Hypertext Transfer Protocol Secure est un protocole de communication client serveur sécurisé. Il correspond au protocole de communication HTTP auquel est ajouté une couche de chiffrement utilisant des certificats d'authentification assurant la confidentialité et l'intégrité des données. 16, 20, 25

JSON Le JavaScript Object Notation est un format permettant de stocker des données au format texte. Dérivé de la notation des objets du langage JavaScript, ce format est considéré comme plus lisible (par un humain) que d'autres alternatives comme le format XML. 13, 20, 23–25

REST Défini en 2000, le REpresentational State Transfer est un style d'architecture logicielle pour les services web. Il guarantit l'interopérabilité des ordinateurs reliés à Internet en définissant la méthode de manipulation des ressources web. L'ensemble des ressources web sont associées à une représentation textuelles pouvant subir des modifications suivant des opérations pré-définies. Ces opérations couplées à un protocole sans état permettent d'augmenter la réactivité, la fiabilité et l'extensibilité du système. 15, 16

Table des figures

1.1 Insignes et emblèmes de la BA 118 et de l'ESIOC . . . 5
1.2 Schéma de l'organisation interne de l'ESIOC . . . 5
1.3 Organigramme de l'ESIOC . . . 6

2.1 Flow du projet OMEGA . . . 8
2.2 Interface d'utilisation d'un notebook Python dans Dataiku DSS . . . 8
2.3 Interface de création de librairies Python dans Dataiku DSS . . . 9
2.4 Création d'une librairie Python dans un projet en utilisant l'extension Dataiku DSS de VS Code 9
2.5 Interface de création d'un scenario dans Dataiku DSS . . . 9
2.6 Interface de consultation de dashboard dans Dataiku DSS . . . 10
2.7 Interface de consultation de wiki dans Dataiku DSS . . . 10
2.8 Interface d'utilisation d'une webapp Dash dans Dataiku DSS . . . 11
2.9 Interface d'utilisation d'une webapp Bokeh dans Dataiku DSS . . . 11
2.10 Certifications Dataiku DSS obtenues au cours du stage . . . 12

3.1 Architecture générale du projet OMEGA . . . 16
3.2 Architecture détaillée du projet OMEGA . . . 16
3.3 Cycle de développement selon la méthodologie agile [22] . . . 18
3.4 Sommaire de la section Application Programming Interface (API) du wiki . . . 19
3.5 Interface de paramétrage des endpoints dans Dataiku DSS . . . 20
3.6 Interface de paramétrage des environnements de code dans Dataiku DSS . . . 21
3.7 Interface de programmation des endpoints dans Dataiku DSS . . . 21
3.8 Interface de test des endpoints dans Dataiku DSS . . . 21
3.9 Interface du déployeur d'API Dataiku DSS . . . 22
3.10 Interface du logiciel Postman . . . 22
3.11 Diagramme de flux des endpoints de l'API Python . . . 23
3.12 Diagramme de flux des données . . . 24
3.13 Architecture détaillée du projet OMEGÀ avec transfert de fichiers via un serveur . . . 25
3.14 Diagramme de séquence simplifié des transferts de fichiers . . . 25
3.15 Architecture détaillée du projet OMEGÀ avec gestion de plusieurs sessions . . . 26
3.16 Diagramme de séquence simplifié de la création des instances de projets . . . 26
3.17 Article du wiki documentant un endpoint de l'API Dataiku DSS . . . 28
3.18 Article du wiki documentant la validation des données effectuée dans un endpoint de l'API Dataiku DSS . . . 29
3.19 Article du wiki documentant la préparation des données effectuée dans le flow Dataiku DSS . . . 29
3.20 Article du wiki contenant les suggestions d'améliorations de Dataiku DSS . . . 29

4.1 Flow du projet RECO_CLASSIF . . . 32
4.2 Interface de contrôle des métriques d'un dataset dans Dataiku DSS . . . 33
4.3 Interface de contrôle des checks dans Dataiku DSS . . . 34
4.4 Interface de développement d'une data quality rule dans Dataiku DSS . . . 34
4.5 Interface de contrôle des data quality rules dans Dataiku DSS . . . 34
4.6 Recipe Python dans Dataiku DSS après restructuration, optimisation et documentation du code 35

A.1 Diagrammes de Gantt utilisés pour la plannification et le suivi du stage . . . 42

B.1 Plaquette de présentation de l'ESIOC (recto) . . . 44
B.2 Plaquette de présentation de l'ESIOC (verso) . . . 44

C.1 Arrêté DR factice . . . 46
C.2 Lettre DR factice . . . 47
C.3 Formation DR factice . . . 48
C.4 Menu de cantine factice . . . 48

Table des matières

Introduction **1**

1 Présentation de l'organisme d'accueil **3**
1.1 Armée de l'Air et de l'Espace . . . 3
1.2 Base aérienne 118 « Colonel Rozanoff » . . . 3
1.3 Escadron des systèmes d'informations opérationels et de cyberdéfense . . . 4

2 Dataiku Data Science Studio **7**
2.1 Présentation . . . 7
2.2 Utilisation . . . 7
2.2.1 Fonctionnalités . . . 7
2.2.2 Formations et certifications . . . 12
2.2.3 Configuration à l'ESIOC . . . 12
2.3 Limites et contraintes . . . 13

3 Projet OMEGA **15**
3.1 Description du projet . . . 15
3.1.1 Contexte et motivation . . . 15
3.1.2 Architecture . . . 16
3.2 Organisation au sein de l'ESIOC . . . 17
3.2.1 Organisation des équipes . . . 17
3.2.2 Méthode de développement . . . 17
3.3 Développement de l'API Python . . . 19
3.3.1 Développement d'API dans Dataiku DSS . . . 19
3.3.2 Ebauche de l'API . . . 23
3.3.3 Organisation de l'API . . . 23
3.3.4 Vérification des données . . . 23
3.3.5 Transfert des données . . . 23
3.3.6 Gestion des utilisateurs multiples . . . 25
3.3.7 Démarche de qualité . . . 26
3.4 Écriture d'une documentation . . . 27

4 Projet reconnaissance de timbres de confidentialité **31**
4.1 Description du projet . . . 31
4.1.1 Contexte et motivation . . . 31
4.1.2 Fonctionnement . . . 31
4.2 Améliorations . . . 32
4.2.1 Plannification . . . 32
4.2.2 Gestion des données . . . 33
4.2.3 Autres améliorations . . . 33

Conclusion **37**

A Annexes Communes **41**
A.1 Responsabilité sociétale . . . 41
A.2 Planning . . . 41

B Annexes du Chapitre 1 **43**
B.1 Plaquette de présentation de l'ESIOC . . . 43

C Annexes du Chapitre 4 **45**
C.1 Exemples de documents . . . 45

D Journal de Bord **49**

Résumé - Abstract **51**

Introduction

L'Armée de l'Air et de l'Espace (AAE) utilise de nombreux Systèmes d'Information et de Communication (SIC) numériques. L'Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence (ESIOC), localisé dans la Base Aérienne (BA) 118 de Mont-de-Marsan, a pour mission de produire des logiciels, d'assurer un soutien informatique dans toute l'AAE, de défendre les systèmes numériques et de valoriser les données. Cette dernière mission est composée de l'intégration, du stockage, de la mise à disposition et de la valorisation des données numériques générées quotidiennement. Données de vols des avions de chasse, données relatives à la formation des pilotes ou données personnelles des militaires, les données traitées sont nombreuses, variées et nécessitent des mesures de sécurité adaptées.

Le stage de deux mois que j'ai effectué en tant que civil à l'ESIOC s'inscrit dans les missions de valorisation de la donnée. Intégré à une équipe de développement du Département du Numérique et de la Donnée (DND), j'ai participé au développement de deux projets.

Le premier projet nommé OMEGA est une application d'aide à la décision ayant pour objectif d'améliorer la gestion des mutations des sous-officiers. En analysant les données personnelles des militaires et les postes à pourvoir, un algorithme d'optimisation sous contraintes fournit une proposition de plan de mutations satisfaisant au mieux les besoins de l'AAE et les vœux des sous-officiers. Suite à une étude réalisée par une entreprise privée, OMEGA est en phase de redéveloppement par l'ESIOC. Ce travail permet notamment d'adopter une d'architecture logicielle plus sécurisée. OMEGA est désormais une application web communiquant, à travers plusieurs couches logicielles, avec les serveurs de calculs responsables de l'analyse des données. J'ai développé une interface de communication entre deux couches logicielles et effectué un travail de documentation sur l'ensemble du projet.

De façon subsidiaire et dans une démarche de travail autonaume, j'ai effectué des améliorations sur le projet de reconnaissance de timbres de confidentialité utilisé pour apprécier les capacités des nouveaux outils du DND. Ce travail consiste à assurer le fonctionnement et la qualité du logiciel en optimisant le code, en mettant en place des indicateurs de qualité ainsi qu'en documentant le projet.

Ces deux missions ont nécessité une phase de formation sur les logiciels utilisés pour le développement comme Dataiku Data Science Studio (DSS). D'autres compétences, acquises en école d'ingénieur, se sont avérées utiles par la suite comme les notions d'architecture des applications web, de génie logiciel et la progammation en Python.

Assigné à une mission liant le travail de deux équipes de l'ESIOC, j'ai été emmené à collaborer avec différents professionnels, data scientists ou développeurs. Ce contexte de travail m'a naturellement guidé vers une méthode de développement agile permettant d'apporter des améliorations de façon itératives en fonction du développement des différentes couches logicielles.

Le contexte militaire impose de fortes contraintes de sécurité, en particulier sur les systèmes d'information. Cela implique un respect rigoureux des protocoles de sécurité mais aussi d'assurer la fiabilité des livrables. Tout au long du stage, je me suis appliqué satisfaire la démarche de qualité assurée par l'ESIOC notamment au travers du code informatique produit et d'un important travail de documentation.

Dans un premier temps, le Chapitre 1 décrit la transition de l'AAE vers les SIC numériques, établit un bref historique de la BA de Mont-de-Marsan et présente l'ESIOC, escadron qui m'a acceuilli pour ce stage. Absent de ma formation en école d'ingénieur, la maîtrise du logiciel Dataiku DSS est une réelle plus-value pour l'avenir de ma carrière. Le Chapitre 2 contient mon analyse critique de ce logiciel au travers de mon expérience au cours de ce stage. Le Chapitre 3 présente la majeure partie du travail que j'ai réalisé au cours du stage : suite à une description du projet OMEGA, je présente de façon détaillée le travail réalisé pour répondre aux deux missions qui m'on été confiées sur ce projet. Enfin, le Chapitre 4 décrit brièvement ma participation au projet de reconnaissance de timbres de confidentialité, mission de priorité secondaire qui m'a été assignée.

Chapitre 1

Présentation de l'organisme d'accueil

1.1 Armée de l'Air et de l'Espace

Au début du vingtième siècle, les aéroplanes atteignent un niveau de performance et de fiabilité permettant d'effectuer des vols longs. C'est notamment le cas du Blériot XI qui, piloté par son concepteur Louis Blériot, fut le premier avion à effectuer la traversée de la Manche le 25 juillet 1909. Cette même année, l'armée fraçaise fait l'acquisition de ses premiers aéroplanes donnant naissance à l'aéronautique militaire, une branche de l'armée qui évolue rapidement en particulier lors de la Première Guerre mondiale où l'aviation contribue grandement à la victoire française. Dès 1928, l'aviation militaire vient bouleverser l'organisation des armées françaises avec la création du ministère de l'Air qui conduira finalement à la création et à l'indépendance de l'armée de l'Air en 1934 [17].

Aujour'hui encore, l'armé de l'Air ne cesse d'évoluer et d'affronter de nouveaux défis pour affirmer et maintenir sa souveraineté aérienne. Les évolutions mécaniques des aéronefs sont à l'origine de la monté en puissance de l'armée de l'Air. En 1952, le mystère II est le premier avion français a passer le mur du son [17]. L'année suivante le colonel Constantin Rozanoff est le troisième pilote a franchir le mur du son en vol horizontal aux commandes d'un Mystère IV B de conception française [25]. Pilote au Centre d'Essais des Matériels Aéronautiques (CEMA) (le Centre d'Essais de Vol (CEV) actuel), il deviendra après la seconde guerre mondiale le premier commandant du Centre d'Expériences Aériennes Militaires (CEAM) de la base aérienne 118 de Mont-de-Marsan qui contribue aujourd'hui encore à l'amélioration des avions de chasse [8, 24]. Conséquence directe des capacités croissantes des aéronefs, les frontières de la troisième dimension s'élargissent jusqu'à atteindre la très haute altitude dont la limite basse se situe à dix-huit kilomètres au dessus du sol [15]. Pour cette raison, en 2020, l'armée de l'Air devient l'Armée de l'Air et de l'Espace (AAE) [17]. Le développement de technologies reposant sur les propriétés quantiques ont récemment entraîné l'AAE dans une nouvelle lutte scientifique avec pour objectif de développer des systèmes de communication dont la sécurité est considérée comme sans faille [18]. Les Systèmes d'Information et de Communication (SIC) classiques ne sont pas délaissés pour autant. Que ce soit au sol pour la planification, le contrôle et la surveillance des vols ou dans le ciel avec les logiciels embarqués dans les aéronefs, nombreuses sont les utilisations des SIC dans l'AAE. À ce titre le ministère des armées dépense plus de deux milliards d'euros par an dans l'objectif de maîtriser les données, assurer la souveraineté numérique, créer un socle numérique plus performant, résilient et responsable, accompagner vers de nouveaux usages, relever le défis des ressources humaines et adapter son modèle de gouvernance [5].

1.2 Base aérienne 118 « Colonel Rozanoff »

Dès 1911, l'aviation prend place dans la commune de Mont-de-Marsan sous la forme d'un aérodrome. Situé au centre de l'hippodrome des Grands Pins, ce terrain d'aviation est très prisé des amateurs d'aviation et accueille d'importants meetings aériens [24]. En 1913, la construction d'une annexe militaire à l'hippodrome (utilisée par les élèves pilotes de l'école de Pau) marque le début de l'activité militaire sur la commune [20]. Au cours de la Première Guerre mondiale, ce terrain d'aviation est utilisé par l'armée française comme terrain d'entraînement pour les pilotes avant leur départ pour le front [24].

Quatre ans après la création de l'aéro-club des Landes en 1928, la municipalité déplace terrain d'aviation au sud de l'hippodrome, sur l'emplacement actuel de la BA. Ce terrain d'une centaine d'hectares est inauguré en 1934 mais il rapidement cédé gratuitement en l'Etat en 1938 [24].

Au début de la Seconde Guerre mondiale, le terrain d'aviation montois est un enjeu majeur car il accueille

les élèves de l'école militaire de l'air de Salon-de-Provence. Cependant, dès le 27 juin 1940, la ville est occupée par les Allemands et le terrain d'aviation est réquisitionné. De nombreux travaux sont réalisés en utilisant une main-d'œuvre constituée en partie de prisonniers : Mont-de-Marsan et son terrain d'aviation est le site le plus important du sud-ouest de la France pendant la Seconde Guerre mondiale. Une piste, des ateliers, des hangars et des voies de circulations (encore partiellement visibles aujourd'hui) sont construits pour accommoder les avions de chasse allemands qui décollent pour surveiller la côte Atlantique et bombarder l'Angleterre. En 1943, de nouveaux avions allemands de reconnaissance à long rayon arrivent à Mont-de-Marsan. Ces avions utilisés pour patrouiller au dessus de l'océan Atlantique et repérer les convois et sous-marins alliés représentent un atout majeur. Les alliés décident donc de bombarder la base malgré d'inévitables dégâts collatéraux. Les allemands quittent la base de Mont-de-Marsan en juillet 1944 suite au débarquement en Normandie, quelque semaines avant la libération de la ville, le 21 août 1944 [24].

À la fin de la Seconde Guerre mondiale, l'état major de l'armée de l'Air effectue une grande réorganisation. Le 15 juillet 1945, l'aérodrome de Mont-de-Marsan devient une véritable base aérienne réutilisant les constructions allemandes. La première mission assignée à cette base est d'accueillir le Centre d'Expériences Aériennes Militaires (CEAM) d'Orléans détruit pendant le conflit. Ce nouveau CEAM est commandé par le colonel Constantin Rozanoff (dont la base prendra le nom en 1985). La base continue de s'agrandir, devient la BA 118 en 1948 (avec son insigne homologuée en 1952, Figure 1.1a), accueille le Centre de Détection et de Contrôle (CDC) en 1969 et réponds aux besoins stratégiques de dissuasion nucléaire de 1964 jusqu'en 2011 [24].

Actuellement, la BA 118 reste l'une des plus grandes bases aériennes françaises. Sa superficie de 700 hectares lui permet d'accueillir l'une des plus longues pistes de décollage de France (3 600 mètres) et une flotte importante comptant près de 50 avions de chasse de type Rafale. Au quotidien, ce sont 3500 personnes qui travaillent sur la base, répartis dans 66 unités, 55 spécialités et 50 métiers faisant de la BA 118 le premier employeur du département des Landes et l'une des plus grandes structures publiques de la région Nouvelle-Aquitaine [24, 21, 3]. Les activités actuelles de la base s'articulent autour de deux principes : la défense et l'innovation.

La protection, la dissuasion, la projection et l'intervention sont les moyens d'action de la BA 118 pour remplir sa mission de défense des français. La protection de l'espace aérien national est assurée par la surveillance permanente du CDC, le système de défense sol-air SAMP-T « Mamba » et deux avions de chasse toujours prêt au décollage. Bien que la BA 118 ne soit plus responsable des composantes nucléaires aéroportées, elle peut toujours accueillir et appuyer ces forces aériennes en cas de besoin, s'impliquant ainsi dans les missions de dissuasion. La piste de la base montoise offre une grande capacité de chargement et déchargement d'avions de transports requis pour la projection de personnel et de matériel. La 30$^{\text{ème}}$ escadre de chasse (escadre présentant le plus grand nombre d'appareils) ainsi que les aviateurs montois déployés sur d'autres BA sont capables d'intervenir à plusieurs milliers de kilomètres de la France métropolitaine pour des missions de défense aérienne, de frappes ou de reconnaissance [20, 19]. La BA 118 agit également à l'échelle locale en cas de risque majeurs et s'implique dans la lutte anti-terrorisme au travers du plan « Sentinelle ».

Le travail d'innovation mené sur la BA 118 est guidé par les activités de défense. Il consiste à préparer les futurs systèmes de défense : « protéger, dissuader, intervenir demain avec la même efficacité qu'aujourd'hui. » [19] Pour cela, le CEAM utilisé sous le commandement du colonel Constantin Rozanoff pour tester les prises de guerre allemandes et les prototypes français assure toujours des fonctions d'expérimentation et d'innovation. Récemment renommé Centre d'Expertise Aérienne Militaire (CEAM), il réunit désormais plus de 700 experts afin de définir les procédures d'emploi de matériel et analyser les expériences des militaires en relation avec le matériel. De plus, la BA de Mont-de-Marsan s'implique dans l'entraînement de tous les pilotes (chasse, transport, hélicoptère...) français et alliés notamment au travers d'exercices de grande envergures exploitant les grands espaces aériens et champs de tirsair / sol à proximité. Finalement, l'innovation est assurée par la formation militaire ou civile des futures générations de militaires. Ainsi la BA 118 réalise des présentations dans les établissements d'enseignements et des visites de la base pour les jeunes en pré-orientation, propose des cours de préparation pour le brevet d'initiation à l'aéronautique ainsi que des formations en apprentissage ou en stage [20, 19].

1.3 Escadron des systèmes d'informations opérationels et de cyber-défense

Localisé sur la 118, l'Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence (ESIOC) 62.430 est un acteur majeur de la transformation numérique menée par le ministère des Armées [1] [2]. Unité de la Direction des Système d'Information (DSI) de l'AAE, l'escadron surnommé « Marensin » [29] compte deux

1. Une plaquette de présentation de l'ESIOC est disponible dans l'Annexe B.1.
2. Les notions de responsabilité sociétale complétant la présentation de l'ESIOC sont abordées dans l'Annexe A.1.

(a) Insigne de la BA 118

(b) Insigne de l'ESIOC

(c) Emblème de l'ESIOC

FIGURE 1.1 – Insignes et emblèmes de la BA 118 et de l'ESIOC

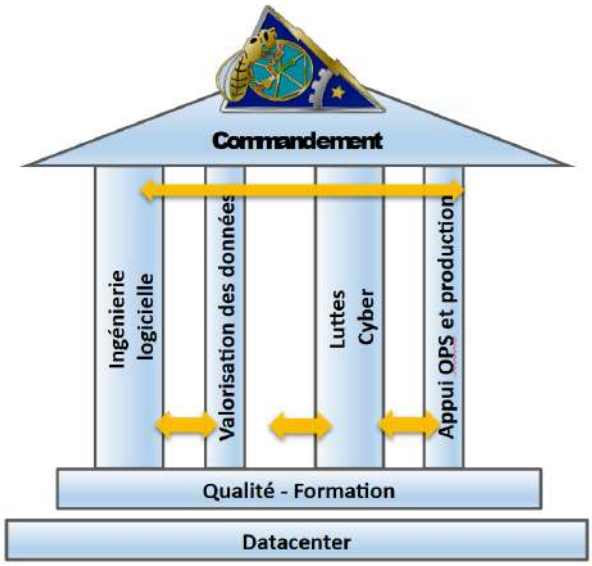


FIGURE 1.2 – Schéma de l'organisation interne de l'ESIOC

cents personnels civil et militaire à Mont-de-Marsan ou engagés en opérations extérieures. Comme le montre la Figures 1.2, les missions de l'ESIOC uniques et contribuant directement aux opérations aériennes s'articulent autour de quatre piliers : produire, appuyer, défendre et valoriser [4].

L'ESIOC est responsable de la production de nombreux logiciels métiers répondant aux besoins spécifiques de l'AAE lors de la préparation des missions ou pour le commandement et le contrôle air. Engageant la sécurité des vols, leur développement nécessite des connaissances en aéronautiques et doit s'adapter aux contraintes matérielles (logciels embarqués) et temporelles.

Plusieurs escadrons bénéficient de l'appui de l'ESIOC en métropole ou en opération extérieure. L'ESIOC assure des fonctions d'administration des systèmes déployés sur les BA à l'échelle nationale, la formation sur les logiciels OTAN et la gestion des terminaux mobiles utilisés par les escadrons de chasse et de transport.

Des attaques dans le cyberespace de l'AAE pourraient avoir de lourdes conséquences, la défense des système de l'air est donc cruciale. L'ESIOC contribue notamment à la lutte informatique défensive et à la cybersurveillance des SI opérationnels air. En cas d'incident, cette unité de pointe possède des capacités d'expertise et de projection dans toute l'AAE.

Aux opérations de gestion de la donnée (intégration, stockage, mise à disposition) s'ajoute désormais la valorisation. Au travers du Département du Numérique et de la Donnée (DND), l'ESIOC est en capacité de traiter de bout en bout la grande variété et le volume en constante augmentation des données produites par l'AAE. Quelque soit le type de données (formation des pilotes, gestion des militaires, préparation ou rapport de vols), leur centralisation, fiabilisation, agrégation et référencement dans une plateforme unique nommée REférentiel de Pilotage AIR (REPAIR) permet d'appliquer les technologies de *big data* et d'intelligence artificielle afin d'en exploiter tout le potentiel.

L'ESIOC accomplit une cinquième mission transversale aux quatre précédentes. Certifié ISO9001 [23, 28], l'escadron Marensin s'engage dans une démarche de qualité dans le développement des logiciels. Cette norme définit les éxigences de qualité devant être adoptées par un organisme. Elle est appliquée au travers de la

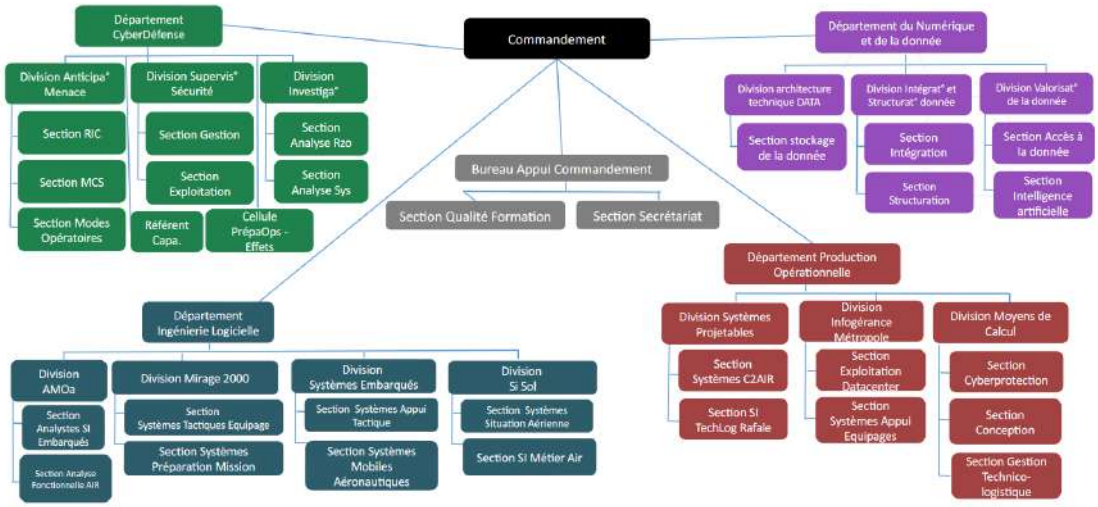


FIGURE 1.3 – Organigramme de l'ESIOC

normalisation des produits, de la satisfaction des exigences métiers, de la traçabilité du développement ainsi que de la documentation et de la qualité du code informatique. La formation et le suivi des clients sur les technologies et solutions proposées témoignent de sa fiabilité.

À titre d'exemple, l'ESIOC est actuellement chargé du développement d'applications mobiles d'aide à la planification de vols de transport, de l'amélioration des logiciels embarqués des avions de chasse Mirage 2000D, d'un logicel de calibration de radar sol et du logiciel de contrôle des vols ARIANE. Le DND est spécifiquement chargé du développement d'outils de visualisation et d'aide à la décision comme les applications web utilisant les données provenant de l'entrepôt de données de REPAIR, le logiciel de personnalisation de la formation des pilotes à l'aide d'analyses prédictives et le projet OMEGA (Chapitre 3). D'autres projets comme un projet de reconnaissance de timbres de confidentialité (Chapitre 4) utilisant la vision par ordinateur ont seulement pour but d'éprouver et valider l'usage de nouveaux outils comme le logiciel développé par Dataiku (Chapitre 2).

Les quatres missions de l'ESIOC (produire, appuyer, défendre et valoriser) sont mises en valeur dans son insigne (Figure 1.1b) et son emblème (Figure 1.1c). La structure interne de l'ESIOC, représentée dans la Figure 1.3, est elle aussi structurée autour des quatres piliers : la production est assurée par le Département d'Ingénierie Logicielle (DIL), les capacités d'appui par le département de production opérationnelle, la défense par le département de cyberdéfense et la valorisation des données par le DND. Les quatre départements collaborent régulièrement entre eux mais aussi avec des entreprises externes fournissant du matériel (avions de chasse, logiciels...) ou des services (accompagnement pour la reprise de projets, voir Chapitre 3).

Chapitre 2

Dataiku Data Science Studio

L'essor de la science des données au cours des dernières années, engendré par l'augmentation de la puissance de calcul et l'accès à une grande quantité de données, s'est accompagné par un large développement de librairies de code massivement utilisées par les développeurs afin de gagner en efficacité. Au cours de mon stage, j'ai pu utiliser un logiciel qui tend à limiter voire supprimer le travail de programmation afin d'améliorer une fois de plus l'efficacité du travail sur les données. Cette section présente ce logiciel, la façon dont je l'ai utilisé pendant mon stage ainsi que les contraintes et limites que j'ai pu rencontrer.

2.1 Présentation

Fondée en 2013 à Paris, la start-up Dataiku adopte une vision innovante de l'Intelligence Artificielle (IA) : « Everyday AI, Extraordinary People » [11]. Dans un monde évoluant rapidement, les entreprises, quelque soit leur taille et leur domaine d'activité, doivent favoriser l'innovation et la collaboration de l'ensemble de leurs employés. Équipes techniques ou dirigeants d'entreprises, tous doivent être en possession des outils pour repenser leur métier au quotidien afin d'obtenir de meilleurs résultats. La start-up française se lance dans la création d'une solution permettant au plus grand nombre de tirer parti des données et de l'IA pour prendre les meilleures décisions. Ainsi l'aide à la décision en toutes circonstances apportée par une IA accessible pour tous permettrait à chaque employé de devenir indispensable [9].

Dataiku Data Science Studio (DSS) est décrit par ces concepteurs comme une plateforme universelle d'IA. C'est un logiciel de science des données collaboratif facilitant le développement, le déploiement et la maintenance d'applications effectuant du traitement de données. Cette application web (logiciel accessible par un navigateur internet) cherche à faciliter et accélérer la valorisation de grandes quantités de données, notamment en réduisant au minimum le travail d'écriture de code.

Comptant désormais plus de 1200 employés dans le monde, Dataiku reste fidèle à son slogan « Your Path to Enterprise AI » en se positionnant sur le marché des entreprises de toutes tailles, en quête d'innovation et dont les besoins, notamment en terme d'IA évoluent rapidement [26]. Depuis la sortie de Dataiku DSS en 2014, l'application n'a cessé d'évoluer afin d'offrir des outils de traitement de données à la fois faciles d'utilisation et performants mais aussi un niveau de stabilité et de sécurité suffisant pour la mise en production d'applications.

2.2 Utilisation

Dataiku DSS est un logiciel qui cherche à faciliter de nombreuses tâches. Ses interfaces (graphique et logicielle) sont donc facile à utiliser mais aussi très complètes et permettent de paramétrer de nombreuses actions et configurations. Cette section présente brièvement les principales fonctionnalités de ce logiciel et les méthodes pour en acquérir la maîtrise. La dernière sous-section décrit la configuration de Dataiku DSS utilisée à l'ESIOC.

2.2.1 Fonctionnalités

L'objectif principal de Dataiku est de rendre accessible la science de données au plus grand nombre. Pour cela, Dataiku DSS propose une interface graphique permettant de gérer les applications les plus simples entièrement en *low code* voire *no code* tout au long de leur cycle de vie (c'est à dire en limitant l'utilisation de langages de programmations). De façon générale, les opérations traditionnellement écrites en lignes de codes sont représentées

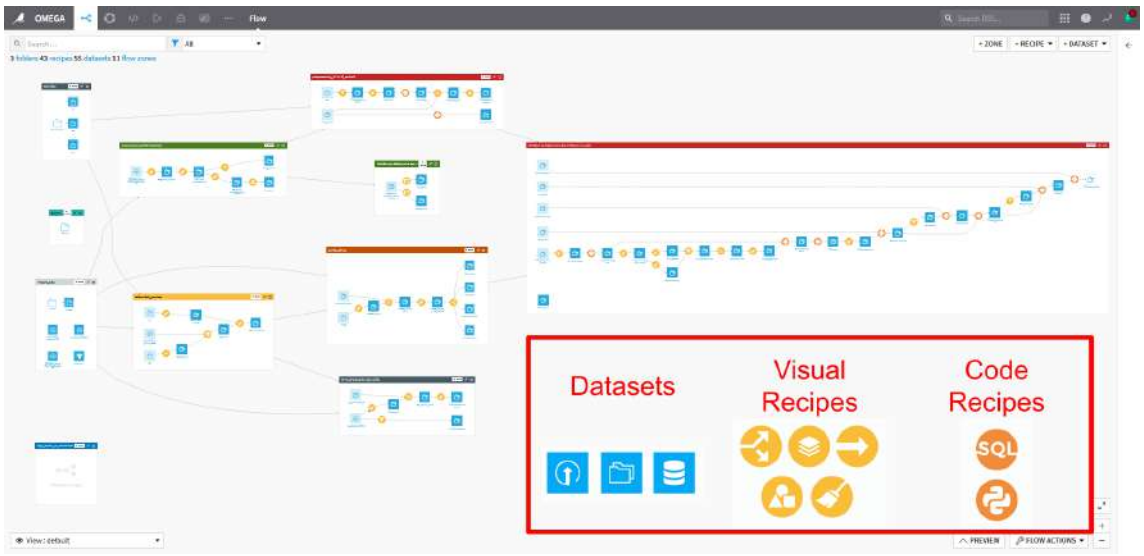


FIGURE 2.1 – Flow du projet OMEGA

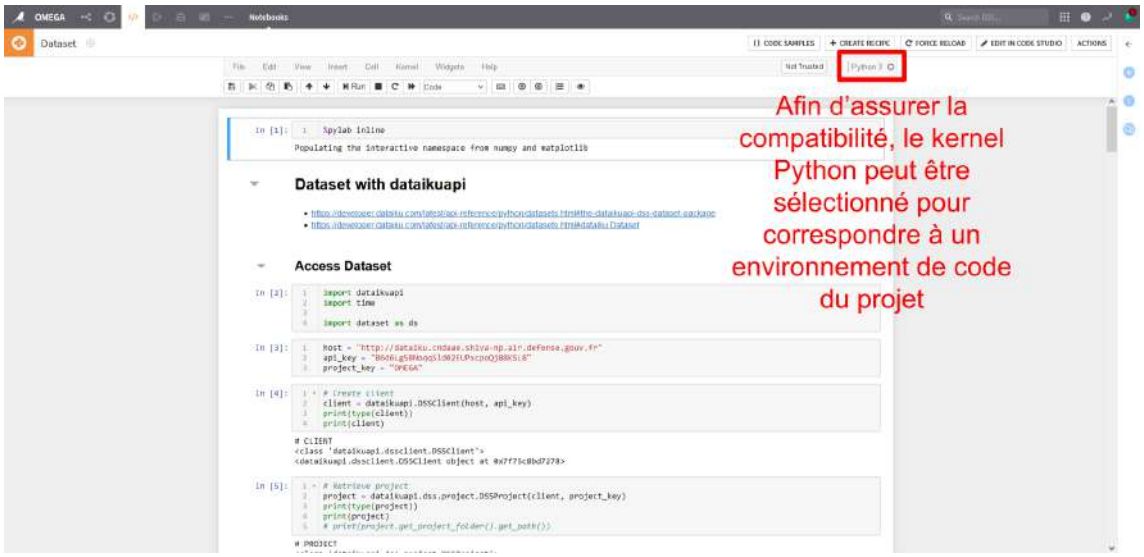


FIGURE 2.2 – Interface d'utilisation d'un notebook Python dans Dataiku DSS

par des blocs d'opération appelés *recipes* dans le *flow* du projet. Un exemple de flow de projet est présenté dans le Figure 2.1.

Néanmoins, l'interface graphique ou les extensions pour divers environnements de développement permettent aussi aux utilisateurs les plus aguéris d'étendre les fonctionnalités de Dataiku DSS pour répondre au mieux à leurs besoins en utilisant les différents langages de programmation (entre autres Python, R, SQL) et les nombreuses connexions à des bases de données [13]. Si besoin, des développeurs peuvent exécuter du code dans des notebooks comme dans la Figure 2.2, créer des recipes exécutant du code Python, R ou SQL ou écrire des librairies Python ou R utilisables partout dans le projet au travers de l'interface présentée dans la Figure 2.3 ou avec les extensions pour IDE comme illustré dans la figure 2.4.

Il est possible de créer des *datasets*, de les visualiser et d'effectuer des traitements ou analyses statistiques en seulement quelques clics. La création de graphiques permettant de visualiser les données statistiques est tout aussi simple. De nombreux traitements sont déjà implémentés comme les traitements type base de données (jointures, groupement...) ou les analyses prédictives (augmentation des données, modèles prédictifs, entrainements et évaluation des modèles). L'automatisation des tâches est aussi simplifiée grâce aux *scenarios* : ensembles d'actions programmés pour s'exécuter selon un ensemble de conditions définies dans l'interface présentée en Figure 2.5.

L'ensemble du logiciel est pensé pour mettre à disposition de l'utilisateur toutes les informations pertinentes au format visuel le plus adapté. L'interface graphique est conçue pour faciliter la lecture du projet : un code

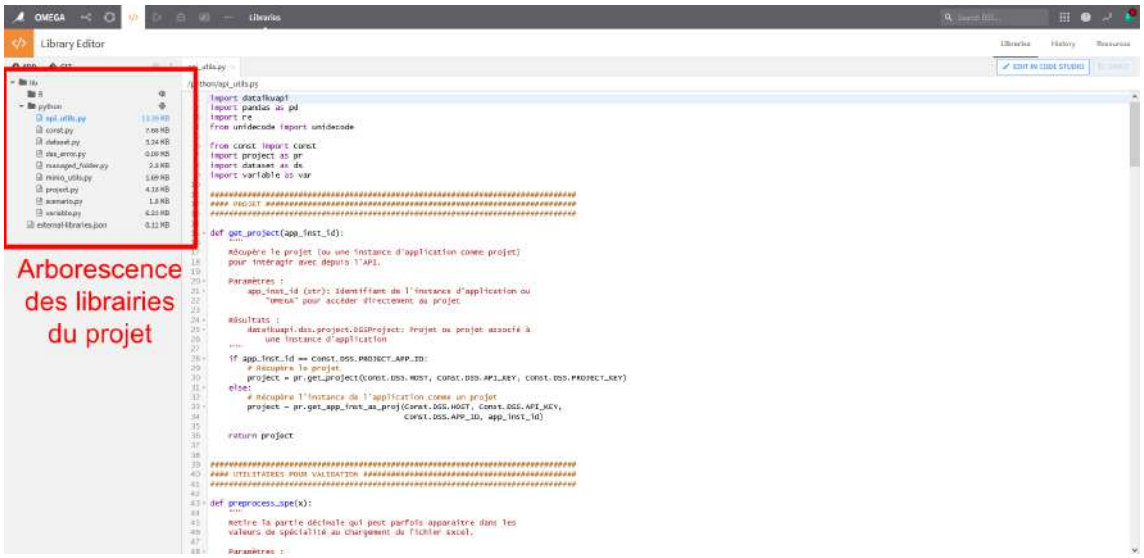


FIGURE 2.3 – Interface de création de librairies Python dans Dataiku DSS

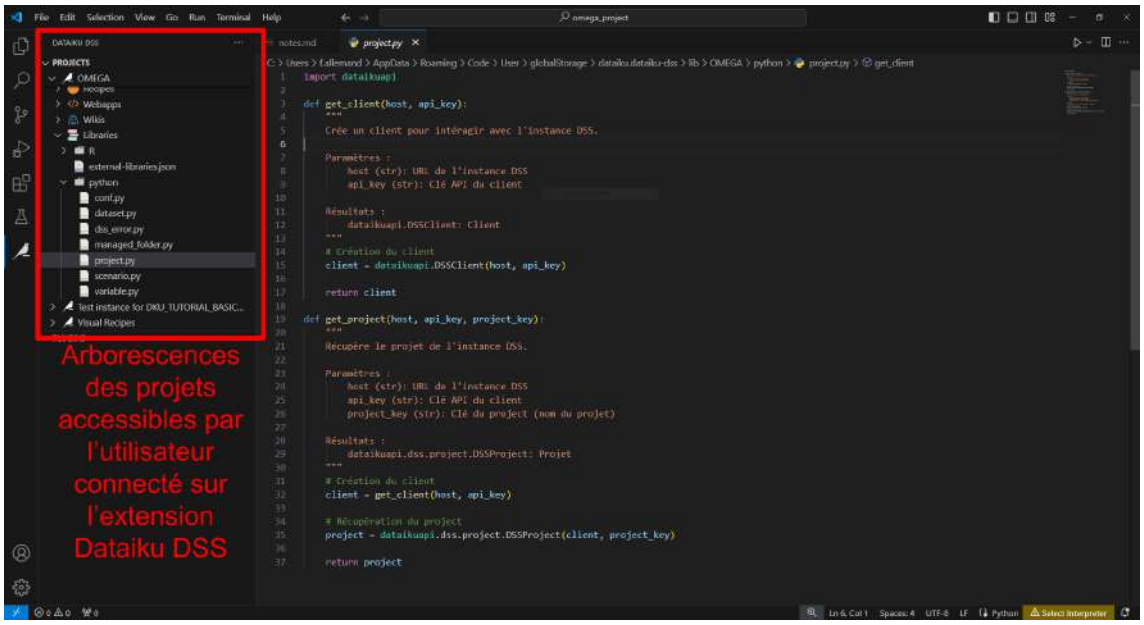


FIGURE 2.4 – Création d'une librairie Python dans un projet en utilisant l'extension Dataiku DSS de VS Code

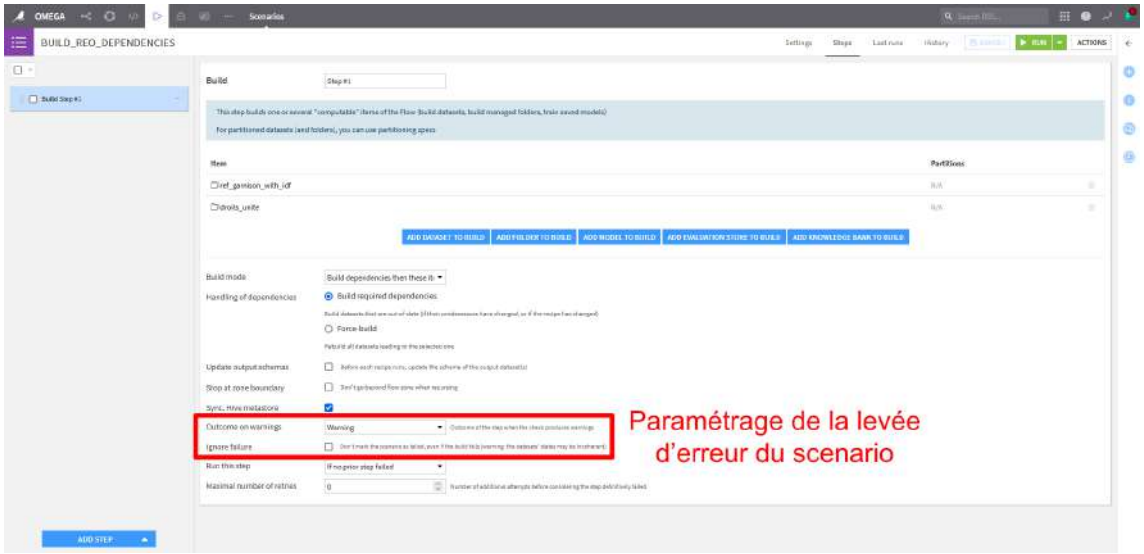


FIGURE 2.5 – Interface de création d'un scenario dans Dataiku DSS

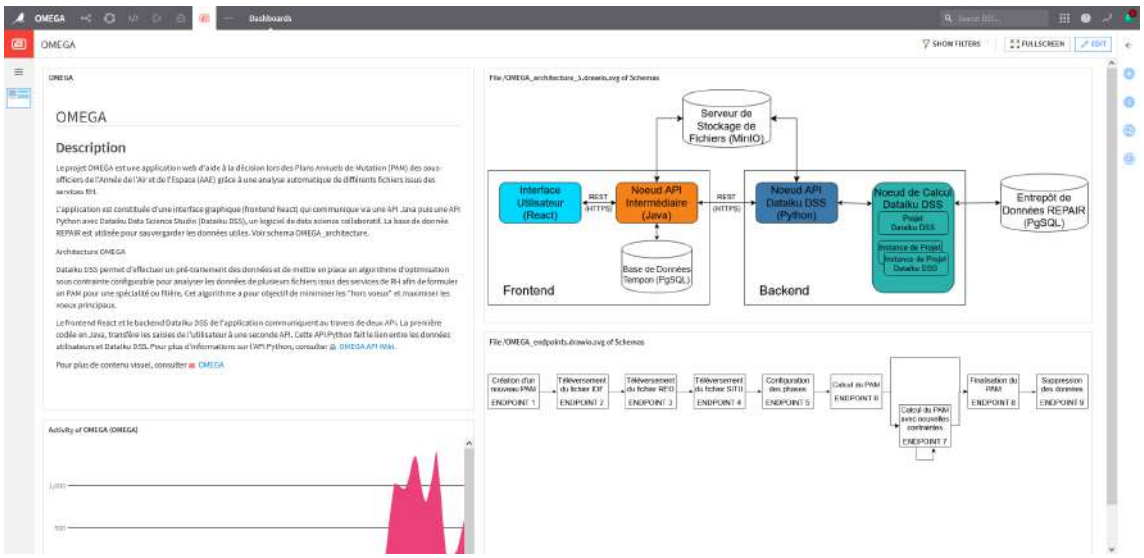


FIGURE 2.6 – Interface de consultation de dashboard dans Dataiku DSS

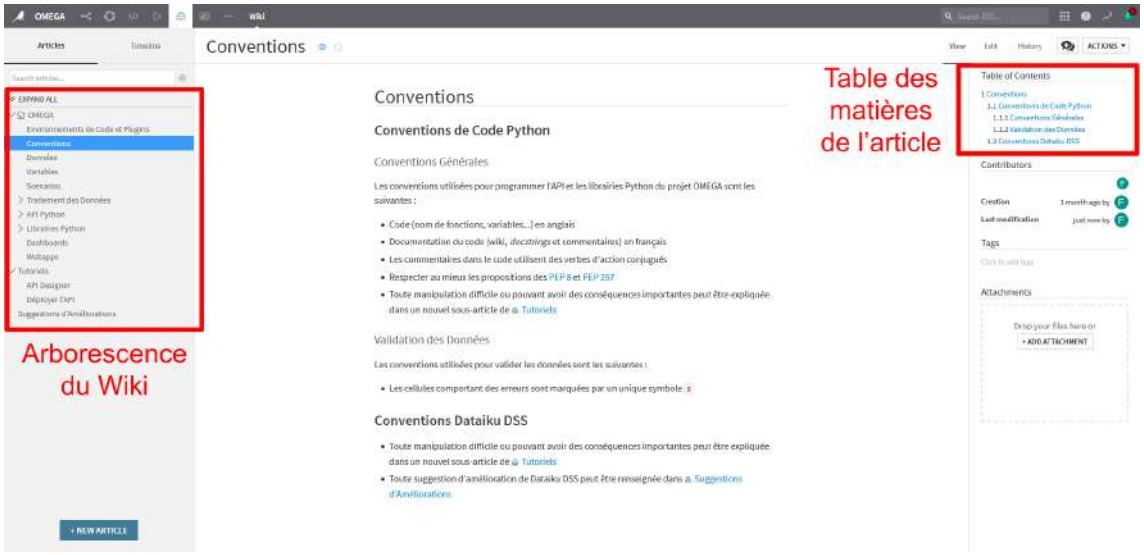


FIGURE 2.7 – Interface de consultation de wiki dans Dataiku DSS

couleur permet de facilement identifier les actions effectuées dans le flow et différentes zones peuvent être créées pour séparer les différentes parties du programme (réception des données, traitement des données...). Les graphiques, les statistiques et les *checks* (Figure 4.3), éventuellement regroupés dans un *dashboard* comme celui montré dans la Figure 2.6, fournissent des indicateurs visuels en temps réels qui permettent de contrôler le projet de bout en bout. L'onglet *wiki* permet d'écrire ou de consulter une documentation au sein même du projet utilisant la syntaxe Markdown et / ou HTML. Comme le montre la Figure 2.7, le document créé est lisible et toujours à disposition. Les utilisateurs peuvent finalement intéragir avec le projet au travers des interfaces graphiques des applications ou *webapps* intégrés dans Dataiku DSS (comme celles que j'ai pu créer, voir Figures 2.8 et 2.9) ou bien au travers d'une autre application communiquant avec le logicel via l'API Python dédiée. Le fonctionnement et le développement d'API pour Dataiku DSS sont détaillés dans la Section 3.3.1.

Dataiku DSS est un logiciel reposant sur les technologies *cloud* afin d'améliorer l'expérience utilisateur (aucune installation locale nécessaire, connexion à distance sur différents postes, calculs décentralisés), la collaboration (projets partagés, documentation commune - wiki) et les performances des applications créées (calculs effectués sur serveur distant, compatibilité avec des bases de données externes). Cela implique que Dataiku DSS doit être installé sur un serveur que ce soit un serveur Dataiku, un serveur dans un centre de données externe (type Amazon Web Service ou Microsoft Azure) ou un serveur entièrement géré en interne, en sachant qu'une instance du programme peut contenir différents projets. L'instance Dataiku DSS ainsi que tout les projets qu'elle contient est entièrement configurée et contrôlée par les administrateurs de l'instance. Ils peuvent notamment gérer les autorisations des autres utilisateurs, c'est à dire les actions qu'ils sont autorisés à effectuer et ce au niveau de

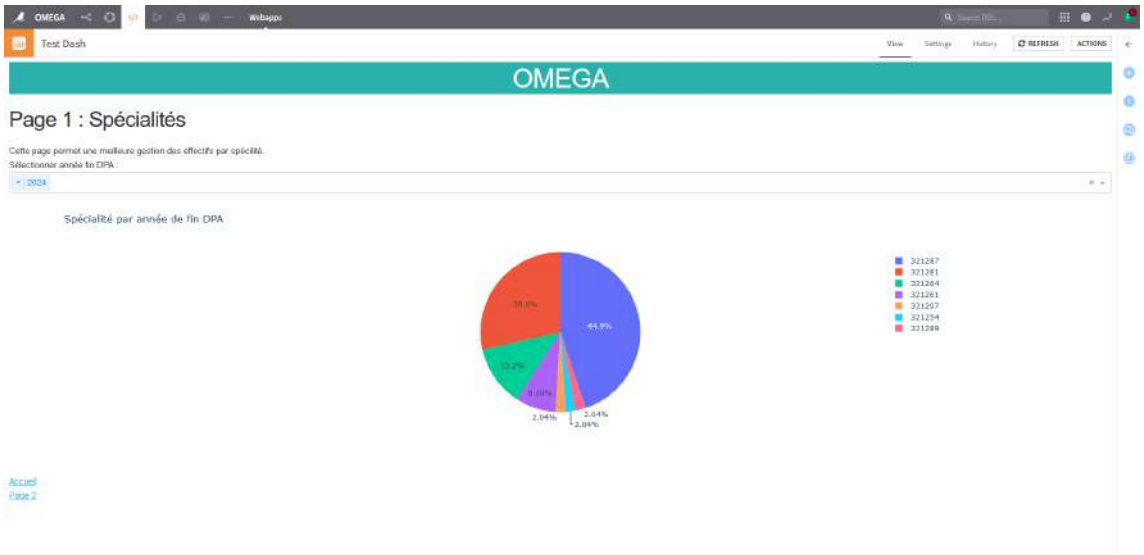


FIGURE 2.8 – Interface d'utilisation d'une webapp Dash dans Dataiku DSS

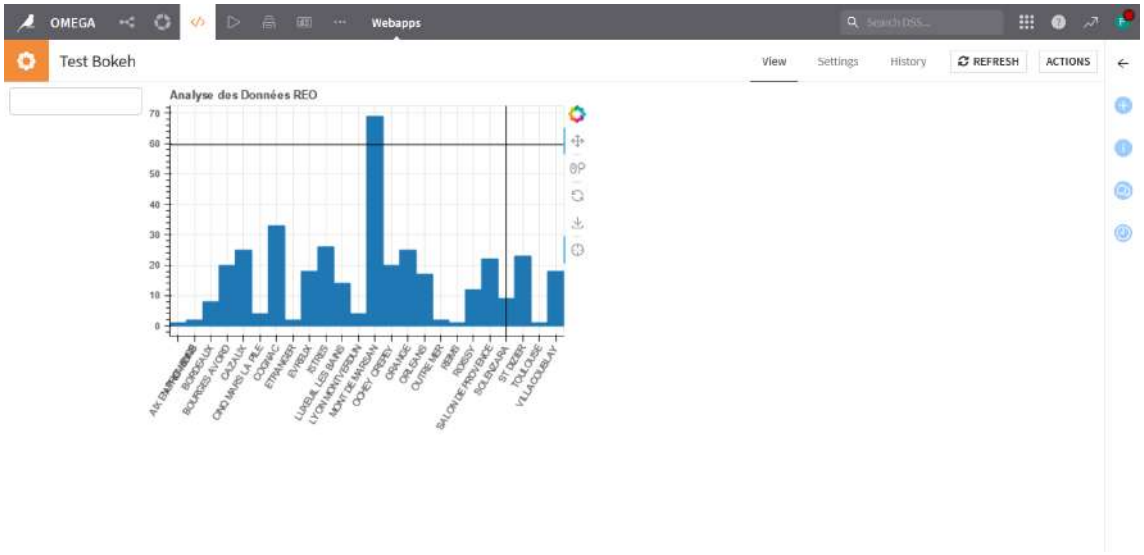


FIGURE 2.9 – Interface d'utilisation d'une webapp Bokeh dans Dataiku DSS

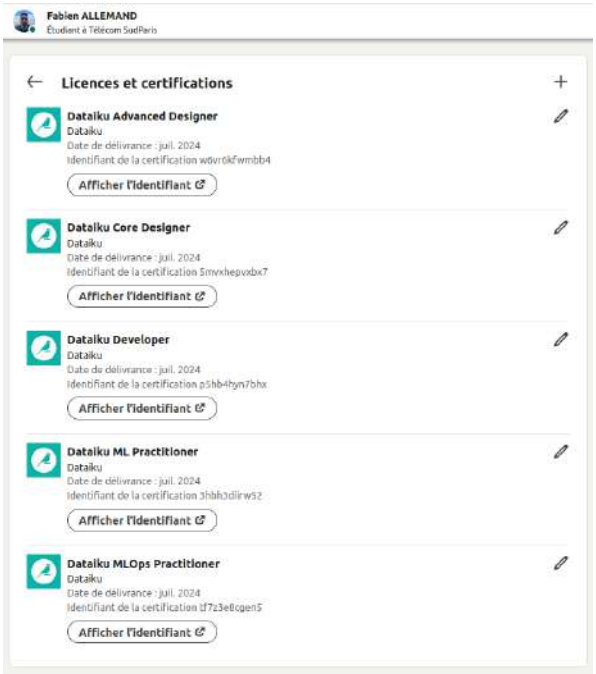


FIGURE 2.10 – Certifications Dataiku DSS obtenues au cours du stage

l'instance Dataiku DSS, des *workspaces* de l'instance ou de chaque projet.

La plus-value de Dataiku DSS est de regrouper dans une seule interface graphique qui se veut agréable et facile d'utilisation, les fonctionnalités de nombreuses librairies allant du traitement des données jusqu'à la création d'applications pour les utilisateurs. S'ajoute à cela de nombreux outils de contrôle et d'analyse mais aussi de gestion de projet tout en laissant une marge de liberté aux développeurs qui peuvent utiliser les zones de code.

2.2.2 Formations et certifications

Dataiku DSS propose de nombreuses fonctionnalités ayant un fort niveau de paramétrabilité. Son interface graphique très complète nécessite donc un temps de formation et de prise en main. Pour cela, de nombreux parcours d'apprentissage sont disponibles sur le site de Dataiku. Ils m'ont permis de découvrir les concepts et fonctionnalités du logiciel grâce aux leçons, de maîtriser l'interface graphique au travers des exercices d'applications et d'évaluer mes compétences lors de questionnaires [12]. Suite à cela, j'ai pu m'inscrire et valider les cinq certifications numériques proposées par Dataiku. Ces certifications sont visibles sur mon profil Linked-In comme le montre la Figure 2.10.

2.2.3 Configuration à l'ESIOC

Au même titre que l'ensemble des SI utilisés par l'AAE, l'utilisation de Dataiku DSS doit respecter les consignes de sécurités imposées par la Direction Interarmées des Réseaux d'Infrastructure et des Systèmes d'Information (DIRISI). C'est actuellement la version 12.6 du logiciel qui est certifiée pour être déployée sur les systèmes de l'AAE. Cette version suffisamment récente introduit de nouveau outils de contrôle de qualité des données ainsi que de grandes améliorations concernant les LLM et l'IA générative.

Il est intéressant de remarquer que Dataiku DSS présente un grand avantage pour des organismes recherchant un haut niveau de sécurité. Accessible depuis un simple navigateur internet, Dataiku DSS ne nécessite aucune autre installation qui nécessiterait une demande et diminurait le niveau de sécurité. Mon poste de travail en tant que stagiaire ne disposait donc pas de moyen d'exécuter du code : bien qu'utilisé dans Dataiku DSS, Python n'était pas installé sur cette machine et je ne disposais pas des droits pour exécuter d'autres programmes.

Au sein de l'AAE, la protection des données est un enjeu majeur. Ainsi les données sont réparties sur différents réseaux avec différents niveaux de sécurité humains (habilitations) et logiciels (isolation des réseaux). On retrouve notamment les réseaux :

— Non Protégé (NP) : Réseau pouvant accueillir les données n'ayant pas de classification particulière et ne nécessitant pas d'habilitation. Ce réseau est relié à internet.

— Diffusion Restreinte (DR) : Réseau pouvant contenir les données restreintes (ou non protégées) dont l'accès nécessite une habilitation. Ce réseau est physiquement isolé d'internet.

Tout au long du stage, j'ai travaillé sur une instance Dataiku DSS installée sur une machine virtuelle d'un serveur connecté au réseau NP. L'accès à l'interface graphique se fait au travers du navigateur web d'un ordinateur connecté à ce même réseau en m'attribuant au plus (lorsque cela fut nécessaire) le statut d'administrateur sur les projets.

L'utilisation du réseau NP au cours de ce stage s'explique par la régulation des habilitations. Les autorisations pour accéder au réseau NP sont longues et ne peuvent être demandées uniquement lorsque la présence dans la base militaire dépasse six mois. Ce stage étant plus court, je ne pouvais pas prétendre à cette habilitation et les données ont dû être déclassifiées. Les données que j'ai utilisées sont des données réelles ayant été anonymisées ou des données factices. Cela permet de développer et tester les applications de façon pertinente sur l'instance NP. À terme, les projets sur lesquels j'ai travaillé seront déployés sur le réseau DR pour traiter de vraies données classifiées DR. J'ai pu observer le transfert et le déploiement des projets sur le réseau DR par le responsable technique de la plateforme afin d'effectuer une première phase de tests en collaboration avec le Département d'Ingénierie Logicielle (DIL).

2.3 Limites et contraintes

En plus des contraintes relatives à l'architecture réseau mentionnée dans la section précédente (phase de développement et de test / production sur deux réseaux isolés), j'ai rencontré de nombreux obstacles liés à Dataiku DSS au cours de mon stage.

Le défaut majeur de ce logiciel provient du fait que c'est un logiciel « boite noire » : l'interface utilisateur est souvent opaque aux opérations réellement exécutées. J'ai notamment été entravé dans mon travail car mes missions nécessitaient l'écriture de code dont l'intégration n'est pas toujours celle attendue. J'ai souvent dû collaborer avec l'architecte de la plateforme afin d'identifier les erreurs, comprendre le fonctionnement caché de Dataiku DSS et corriger mon travail car l'interface utilisateur ne reflétait pas correctement les actions effectuées ou masquait les messages d'erreur ou les logs en particulier aux utilisateurs non administrateurs. En particulier le déploiement d'API et la gestion des connexions type S3 ne fonctionne pas toujours comme on pourrait penser et les messages d'erreur ne sont pas suffisamment précis.

Plus généralement, la gestion des autorisations des utilisateurs est un aspect que j'ai dû étudier en détail car, si des autorisations manquantes masquent des options dans l'interface graphique, elles bloquent aussi certaines fonctionnalités lorsqu'on utilise l'API de développement dataikuapi. La gestion des clefs API (de projet ou personnelles) et de leurs droits se fait au travers de plusieurs menus et n'est pas toujours très précise : certains paramètres sont regroupés sous un même nom et d'autres semblent absents. Cependant, un bon paramétrage des clefs API est nécessaire pour le bon fonctionnement de l'API notamment si on utilise des instances d'application (voir Section 3.3.6).

Les parcours d'apprentissages traitent seulement les cas les plus génériques et les plus simlpes, ceux pour lesquels Dataiku DSS a été conçu. Une utilisation plus avancée du logiciel nécessite plus de recherche et d'expérimentations. Pour cela, Dataiku DSS offre la possibilité de tester du code dans des notebooks et possède une documentation très complète mais qui manque d'exemples et de précision. Cela m'a contraint à repenser mes fonctions d'export de datasets au format JSON car une fonction de l'API ne fonctionne pas sur un nœud de production. Or cette information n'est pas mentionnée dans la documentation. De même, l'utilisation de Dataiku DSS comme back end d'une autre application web est un cas d'usage limite. Certaines fonctionnalités pouvant faciliter cette pratique, telles que la compatibilité avec le *multipart form data*, sont manquantes.

Durant les deux mois de stage j'ai aussi relevé quelques points d'amélioration de l'interface graphique. On trouve tout d'abord des défauts comme l'apparition des messages d'erreur uniquement lorsqu'on annule une opération ou encore l'absence d'alerte de travail concurent dans la même zone qu'un collaborateur dans certaines interfaces (ce mécanisme très utile pour les notebooks et le wiki est absent de l'éditeur d'API). Il y a ensuite de très nombreuses pistes pour améliorer l'expérience utilisateur. Le logiciel fournit une *todo list* pour chaque projet, cependant son utilisation n'est pas facile. Il est impossible de réorganiser les tâches et les zones d'édition de texte ne permettent pas toujours d'accéder à l'ensemble des lignes. À cela s'ajoute la perte de temps et la frustration dûes à au fait que ces zones ne sauvegardent pas les modifications automatiquement malgré un arrêt de l'édition au moindre clic en dehors de celle-ci. De même, les zones d'édition de code comme celle visible dans la Figure 2.3 n'ont pas toutes les fonctionnalités des IDE modernes. La coloration syntaxique du code est limitée et ne dispose d'aucun signalement d'erreur. L'autocomplétion ne fonctionne pas toujours et il n'y a pas d'accès direct à la documentation des fonctions. Enfin, les fonctionnalités avancées d'édition de texte comme les curseurs et selections multiples sont absentes. Les extensions pour IDE externes ne constituent qu'une solution partielle

car elle ne permettent pas de modifier toutes les zones de code : à titre d'exemple, l'extension VS Code donne accès uniquement aux librairies et aux webapps.

De plus, contrairement aux librairies qu'il cherche en partie à remplacer, ce logiciel est payant et n'est pas *open source*. Cela implique qu'il y a beaucoup moins d'utilisateurs et, par extension, qu'il existe peu de forums concernant les problèmes rencontrés. Cet effet est amplifié par le fait que Dataiku DSS fonctionne très bien pour les fonctionnalités de base et donc que les problèmes rencontrés sont souvent des problèmes isolés dûsà une configuration ou un cas d'usage très spécifique. J'ai aussi remarqué que peu d'experts du logiciel participent de façon active sur le forum Dataiku, laissant de nombreuses questions d'utilisateurs sans réponse.

Chapitre 3

Projet OMEGA

La mission principale qui m'a été assignée est la création d'une interface logicielle (API REST Python) pour le projet OMEGA, une application de gestion des mutations des militaires, destinée à la Direction des Ressources Humaines de l'Armée de l'Air et de l'Espace (DRHAAE). Cette section présente le projet dans son ensemble, l'organisation du travail à l'ESIOC ainsi que le travail que j'ai réalisé.

3.1 Description du projet

Cette section est une présentation du projet OMEGA. Une première partie décrit le contexte de développement de ce projet. Une seconde partie aborde le fonctionnement technique du logiciel.

3.1.1 Contexte et motivation

Dans le contexte militaire, la gestion des ressources humaines est un enjeu capital : un militaire doit être assigné à chaque poste créé pour répondre aux besoins de l'armée. Ces besoins étant variables dans le temps et l'espace, de nombreuses contraintes apparaissent et certains militaires doivent s'adapter à de nouvelles missions, évoluer en grade voire être mutés. Depuis la création de l'AAE en 1934, les règles de mutations n'ont cessé d'évoluer pour s'adapter aux besoins et assurer la fiabilité de cette armée. En effet, de nouvelles spécialités et certifications pour les militaires de l'AAE sont régulièrement créées et d'autres disparaissent, de même les unités sont souvent remodelées. Depuis 2023, c'est le Commandement Territorial de l'Armée de l'Air et de l'Espace (CTAAE) qui supervise les besoins financiers, en infrastructures, humains, numériques, de développelent durable et de maîtrise des risques [1]. C'est donc ce nouveau grand commandement de l'AAE qui modifie la règlementation de mutations. Actuellement, l'AAE compte près de 37 000 militaires dont la majorité (environ 30 000) sont des sous-officiers mutés tous les dix ans en moyenne [7]. Chaque année, pendant une période de près de six mois, la DRHAAE effectue des Plans Annuels de Mutation (PAM) qui sont constitués de nombreuses commissions durant lesquelles les données de plusieurs fichiers sont croisées manuellement. L'objectif est d'assigner un sous-officier de la bonne spécialité à chaque poste. Dans une démarche de fidélisation du personnel, les PAM actuels considérent au mieux les souhaits des militaires tout en répondant aux éxigences de l'armée. Cependant, la quantité de données et la charge de travail rendent les résultats des PAM manuels non exempt d'erreur.

Le projet OMEGA est un logiciel d'aide à la décision lors des PAM des sous-officiers de l'AAE grâce à une analyse automatique de différents fichiers des services de ressources humaines. Les deux fichiers principaux utilisés pour un PAM sont des classeurs contenant d'une part les postes à pourvoirs (fichier dit REO pour Référentiel En Organisation) et d'autre part la situation des militaires (fichier SITU). Acronyme signifiant Outil de ???, l'objectif principal de cette application web n'est pas d'automatiser ou d'accélérer les PAM mais seulement d'augmenter leur qualité en traitant au mieux les souhaits des sous-officiers. Le résultat est donc une proposition de PAM qui doit être vérifiée et éventuellement modifiée par la DRHAAE avant d'être acceptée.

Projet initialisement confié à une entreprise ayant répondu à l'Appel à Manifestation d'Intérêt (AMI) en 2019 pour une étude de faisabilité (notamment la création d'une Proof Of Concept (POC)), il montre des premiers résultats satisfaisants pour la DRHAAE qui souhaite effectuer les prochains PAM avec l'aide d'OMEGA. Devant être déployé sur un réseau informatique militaire et traiter des données personnelles, OMEGA doit respecter les contraintes de sécurité imposées par la Direction Interarmées des Réseaux d'Infrastructure et des Systèmes d'Information (DIRISI). Cependant, les technologies Angular et Django utilisées pendant les quatre ans de développement du POC sont incompatibles avec le Cadre de Cohérence Technique (CCT) en vigueur. Le développement du projet OMEGA ne peut donc pas continuer tel quel. La DRHAAE fait alors appel à l'ESIOC,

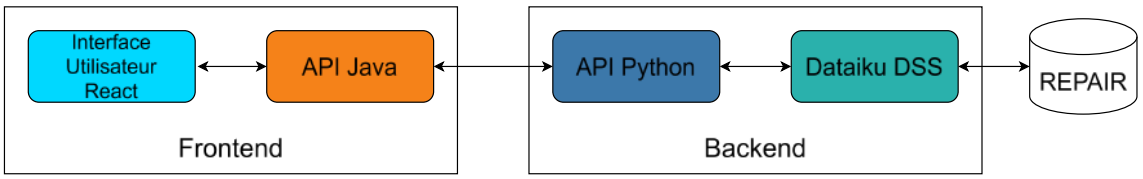


FIGURE 3.1 – Architecture générale du projet OMEGA

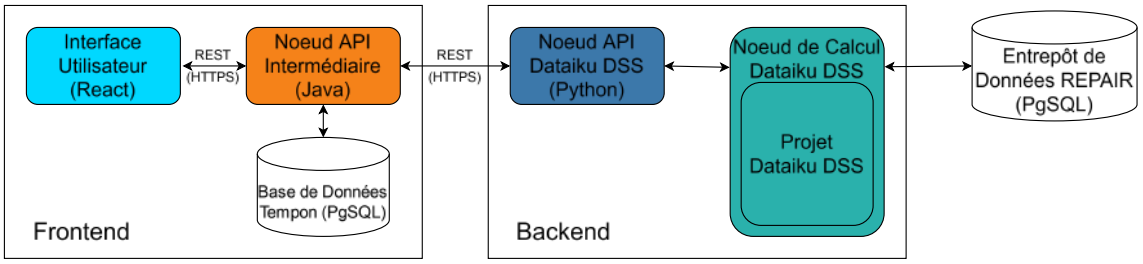


FIGURE 3.2 – Architecture détaillée du projet OMEGA

et plus particulièrement au DND et au DIL, pour une refonte totale du logiciel, respectant les règlementations de la DIRISI.

Le traitement et le stockage des données des militaires par un escadron plutôt qu'un fournisseur de services tiers est gage pour l'AAE de la maîtrise de ses SI. Pour le DND, département récemment créé, c'est aussi une opportunité pour acquérir de l'expérience sur de nouveaux outils et faire rayonner ses compétences dans l'AAE.

La phase de re-développement d'OMEGA par les équipes de l'ESIOC a débuté quelques semaines avant le début de mon stage. Elle s'effectue avec le soutien de l'entreprise ayant créé le POC à la hauteur 10 heures de réunion pour répondre aux questions des développeurs de l'ESIOC. L'entreprise Dataiku participe aussi indirectement au développement : suite à l'acquisition d'une license, les développeurs sont accompagnés par des conseillers pour effectuer une bonne transition vers Dataiku DSS. Une communication constante par mail et des réunions hebdomadaires permettent aux data scientist de l'ESIOC d'assurer le bon développement d'OMEGA sur cette plateforme. Une première version d'OMEGA est attendue début novembre 2024 par la DRHAAE afin de réaliser une commission test avec la méthode traditionnelle (analyse manuelle des tableurs), le logiciel POC et la nouvelle version d'OMEGA.

3.1.2 Architecture

Afin de satisfaire les contraintes de sécurité, l'architecture d'OMEGA est entièrement repensée. On passe d'une application effectuant les calculs en local à une application dite à client léger utilisant un serveur distant pour effectuer les calculs. Il est décidé d'adopter l'architecture présentée dans la Figure 3.1. La nouvelle version de l'application est constituée d'une interface graphique (un front end utilisant le framework React) qui communique via deux API en série (Java et Python) avec Dataiku DSS, un logiciel de data science collaboratif. L'entrepôt de données structurées REPAIR est utilisée pour sauvergarder les données utiles.

Un fonctionnement plus détaillé est représenté dans la Figure 3.2 (ce schéma masque enocre quelques subtilités d'implémentations qui sont expliquées dans la Section 3.3.6). On observe deux zones principales : le front end correspondant à la partie où les données sont seulement transmises ou affichées et le back end, zone de calcul dans laquelle est effectué le traitement des données. Sur la droite, en bout de chaîne, se trouve l'entrepôt de données REPAIR servant à stocker les données les plus importantes comme le résultat du PAM.
L'interface graphique permet à l'utilisateur d'intéragir avec l'application en créant de nouveaux PAM, en déposant des fichiers et en sélectionnant divers paramètres. Toutes ces informations sont transmises au travers d'un lien REST HTTPS à une API intermédiaire.
L'API intermédiaire est nécessaire pour des raisons de sécurité. En effet, la seule API Python de Dataiku DSS ne répond pas aux critères du CCT de la DIRISI. Cette API codée en Java et déployée sur un serveur (machine virtuelle) isolé du reste du projet, est utilisée pour gérer les droits d'utilisateurs (autorisation d'accès) ainsi que pour effectuer le transfert de données entre l'interface graphique et le back end de calcul ; elle n'effectue aucun calcul. Pour cette raison, cette API est classée dans la partie front end du projet et est souvent désignée au sein de des équipes de développement par le terme « back end du front end ».
Un autre lien REST HTTPS établit la connexion entre l'API intermédiaire et l'API du back end de calcul. Cette API programmée en Python et directement intégrée à Dataiku DSS. Cette API communique avec un

projet de l'instance Dataiku DSS qui contient le pré-traitement des données et un algorithme d'optimisation sous contrainte configurable pour analyser les données de plusieurs fichiers issus des services de la DRHAAE et proposer un PAM pour une spécialité ou une filière. Cet algorithme a pour objectif de minimiser les « hors vœux » et maximiser les vœux préférés. L'API Python utilise un lien direct avec l'entrepôt de données PostgreSQL REPAIR pour y enregistrer des données comme celles issues des fichiers REO et SITU soumis par l'utilisateur ou bien les résultats des calculs de PAM. Cet entrepôt utilisé pour unifier les données de différents SI métiers et en extraire des indicateurs bénéficiera directement des données des PAM. Les résultats qui doivent être retransmis à l'utilisateur sont récupérés et renvoyés par l'API Python vers l'API intermédiaire Java.
À ce stade, l'API intermédiaire enregistre ces résultats dans une base de données PostgreSQL tempon indépendante de tout le reste du système afin de satisfaire les requêtes de l'interface graphique.

Les parties du projet nécessitant des compétences de traitement de données (à savoir le back end et l'API Python) ont été confiées au DND, spécialisé dans ce domaine. Les équipes d'experts en engénierie logicielle du DIL s'occupent du reste de l'architecture faisant appel à diverses compétences (programmation frontale, développement d'API sécurisée, gestion de bases de données...).
Stagiaire au DND, le développement de l'API Python et la documentation de la partie Dataiku DSS du projet OMEGA m'ont été attribués comme missions principales. Les sections suivantes détaillent l'organisation du développement d'OMEGA ainsi que ma contribution à ce projet.

3.2 Organisation au sein de l'ESIOC

Dans un contexte de développement où les ressources humaines et temporelles sont limitées l'organisation du travail joue un rôle crucial. Cette section présente tout d'abord l'organisation du développement d'OMEGA dans l'ESIOC puis mon organisation personnelle.

3.2.1 Organisation des équipes

Pendant cette partie du stage, je travaillais sous les ordres d'Auriane GUEDEZ, data scientist et responsable du back end du projet OMEGA. J'étais en collaboration avec le sergent Kevin qui s'occupait du passage en production du back end Dataiku DSS et de la connexion avec l'entrepôt de données uniquement accessible sur le réseau DR (auquel je n'avais pas accès).

Nous avons tous les trois collaboré étroitement avec le DIL responsable du front end React et de son API Java afin de s'accorder sur les différents standards de communication à adopter.

Le responsable technique de la plateforme a aussi joué un rôle crucial au cours des différentes phases de développement, de test et de mise en production du projet OMEGA. J'ai ainsi pu entrevoir (au détour de quelques séances de debuggage) la gestion d'instances Dataiku DSS au niveau des serveurs, des machines virtuelles, des réseaux, des configurations et du contrôle du logiciel.

Tout au long de notre mission, nous avons également pu échanger avec une équipe de data scientists / développeurs de l'entreprise responsable du développement du POC pour faciliter la compréhension du code à transférer sur Dataiku DSS. Cet accompagnement à la hauteur de dix heures de réunion pour répondre à nos question était une précieuse ressource.

Un conseiller de l'entreprise Dataiku nous accompagnait également afin de faciliter la transition vers Dataiku DSS. Cette plateforme ayant été adopté par les équipes de l'ESIOC seulement quelques mois avant le début de mon stage, les échanges avec ce conseiller ont été très utiles pour compléter notre formation sur le logiciel et éclaircir certains détails techniques nécessitant parfois de faire appel à des développeurs de Dataiku DSS.

À la fin de mon stage, j'ai pu transmettre mes connaissances au sergent Lucas afin qu'il puisse poursuivre mon travail dans les meilleures conditions possibles. Maitrisant déjà Dataiku DSS, je lui ai expliqué le fonctionnement d'OMEGA, de son utilité pour la DRHAAE jusqu'au fonctionnement de l'API. J'ai également pris soin de lui montrer toutes les ressources dont il disposait pour travailler efficacement : documentation Dataiku, forums en ligne, documentation et tutoriels écrits au cours du stage.

3.2.2 Méthode de développement

Contrairement à l'équipe de développement lu DIL, les développeurs du DND ne suivent pas à proprement parler la méthodologie scrum (sprints avec gestion des priorités...). De par sa nature, le back end OMEGA ne s'y prête pas puisque son développement s'apparente à de la rétro-ingénierie : il consiste à analyser le code du POC pour le transférer de la meilleure façon sur Dataiku DSS. Le projet a donc simplement été découpé en deux parties (pré-traitement des données et optimisation sous contrainte) traitées l'une après l'autre. Il est important de

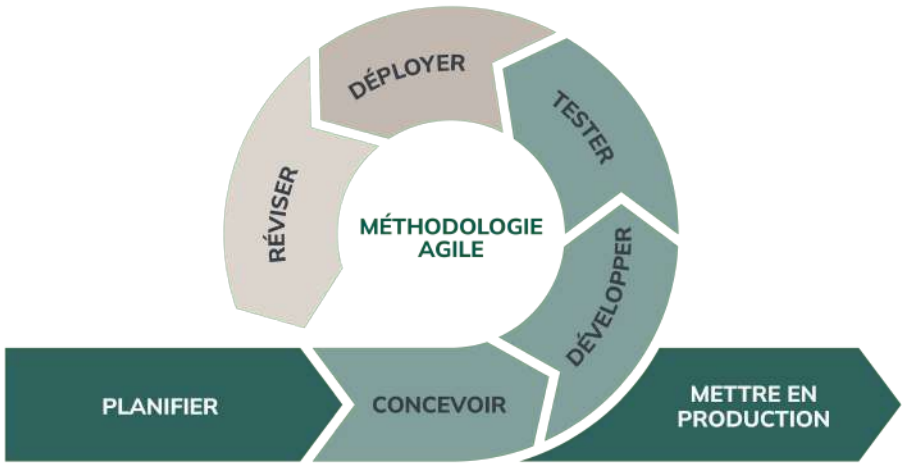


FIGURE 3.3 – Cycle de développement selon la méthodologie agile [22]

mentionner que cette phase de rétro-conception est assistée par une équipe de data scientists et développeurs de l'entreprise ayant répondu à l'AMI afin d'assurer le bon fonctionnement de la version Dataiku DSS de OMEGA. Le développement est donc à la fois guidé par les tests [30] et par les indications des conseillers.

Il est généralement recommandé de créer l'API d'un logiciel une fois ce dernier développé. Cela permet de connaitre précisémment les attentes et les retours des fonctionnalités évitant ainsi de nombreux changements pour adapter l'API au logiciel au fil de son évolution. Cependant, les contraintes de temps ont forcé le DND et le DIL à développer les API en même temps que le développement du front end et du back end de l'application. Étant responsable du développement de l'une des API, j'ai dû adapter ma méthode de travail pour travailler de façon efficace malgré les contraintes. Les évolutions dans la partie calcul entrainant des modifications de l'API, j'ai décidé d'adopter une méthode de développement agile me permettant d'adapter le code produit afin d'assurer la compatibilité de l'API Python avec l'API Java développée par le DIL et le back end du DND.

Plus précisemment, la méthode de développement que j'ai adoptée décrit un cycle qui se décompose en cinq phases, correspondant aux étapes du cycle de développement de la méthode agile présenté dans la Figure 3.3 :

- Une phase de définition et d'analyse des besoins
- Une phase de développement du endpoint
- Une première phase de test dans l'onglet *test queries* de l'API designer
- Le déploiement d'une nouvelle version de l'API
- Une seconde phase de test en condition réelles (avec de vraies données) pour assurer le bon fonctionnement et mesurer les performances

La phase de planification était essentiellement réalisée par la responsable du projet OMEGA en échangeant avec le client (DRHAAE) pour déterminer les fonctionnalités de l'application et avec les autres collaborateurs pour répartir la charge de travail dans le temps et sur les différents collaborateurs.
C'est au cours de l'étape de conception que la responsable me transmet la spécification de l'API et des endpoints en fonction des besoins du projet (nombre de endpoints , fonctionnalités...). À ce stade, il faut déjà que j'anticipe la cohérence entre les besoins et l'implémentation afin de s'accorder sur les spécifications et ainsi éviter les fausses routes.
Pendant la phase de développement qui est essentiellement autonaume, je teste des fragments de codes indépendamment du endpoint dans un Jupyter notebook du projet avant de passer au développement du endpoint dans l'interface utilisée pour gérer, paramétrer et développer les endpoints appelée API designer.
J'utilise l'onglet test queries des endpoints dans l'API designer pour tester chaque fonctionnalité du endpoint, à la façon de test unitaires. Par exemple, pour chaque vérification de donnée devant être effectuée, je crée une requête comportant une erreur afin d'assurer le bon comportement du endpoint.
La phase de déploiement se résume en une simple procédure pour déployer l'API sur le nœud API.
Afin de vérifier le bon fonctionnement de l'API, j'effectue une seconde phase de test après le déploiement. Cela me permet non seulement de tester les endpoints en dehors de Dataiku DSS avec le logiciel Postman (pour s'assurer que l'API était bien accessible sur le nœud API) mais aussi de tester les endpoints avec des données réelles. Ce dernier point est très important pour relever des erreurs d'implémentations, trouver des cas particuliers dans les données ou bien même pour évaluer les performances du système.
La phase de mise en production est plannifiée pour une date ultérieure à celle de la fin de mon stage. Cependant, j'ai pu observer des séances de tests en environnement de production sur le réseau DR.

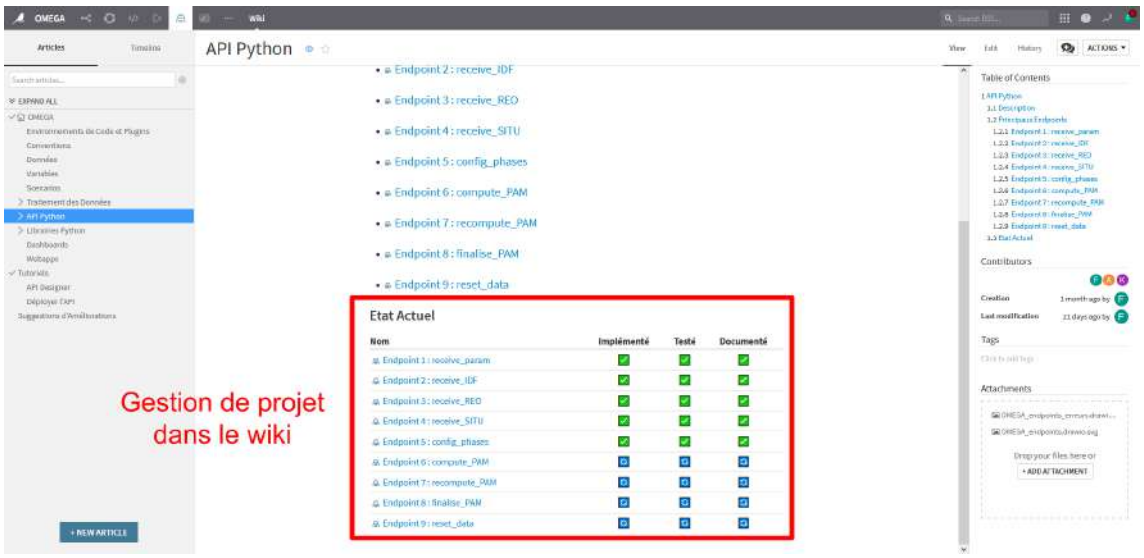


FIGURE 3.4 – Sommaire de la section API du wiki

Une partie du travail de documentation demandé est incorporé dans la méthode agile adoptée puisque j'effectuais la documentation du code en même temps que le développement des endpoints . De même, les zones du wiki concernant les endpoints étaient mise à jour dès que possible pour éviter les erreurs (oublis, imprécisions...). Le reste du wiki était modifié moins régulièrement, lorsque des ajouts étaient nécessaires qu'ils soient demandés par mes collaborateurs ou de ma propre initiative.

En temps normal, les équipes de développement de l'ESIOC utilisent l'outil de gestion agile et de développement logiciel Tuleap. Cependant, en raison du niveau de confidentialité requis pour certains projets, ce logiciel est unqiuement accessible depuis un poste connecté au réseau DR. Afin de collaborer avec moi (ayant seulement accès au réseau NP), nous utilisions les todo lists offertes par l'interface de Dataiku DSS et le wiki comme illustré sur la Figure 3.4.

Le diagramme de Gantt en Figure A.1a de l'Annexe A.2 présente le planning prévisionnel du stage. On y retrouve la phase de formation sur Dataiku DSS, une rapide prise en main du projet OMEGA et les deux tâches principales que je devais réaliser.

3.3 Développement de l'API Python

Comme expliqué précédemment, l'application OMEGA est une application dite à client léger. Elle est donc séparée en deux parties : une interface utilisateur appelée front end qui transmet les saisies de l'utilisateur au back end hébergé sur un serveur qui effectue les calculs et renvoie les résultats. Les résultats sont alors reçus par le front end pour être par exemple affichés à l'utilisateur. En particulier, la réception des données dans Dataiku DSS et l'envoi des résultats vers l'utilisateur est assurée par une API Python dont la responsabilité m'a en partie été confiée.

Pendant les deux mois de développement auxquels j'ai particpé, l'API a subi de nombreuses modifications au fil des versions. Initialement constituée de cinq endpoints très simples, la solution retenue compte désormais neuf endpoints communiquant avec un serveur de fichier et prenant en charge la validation des données. Cette API est désormais accessibles par plusieurs utilisateurs simultanément. Les sections suivantes décrivent les outils de développement d'API dans Dataiku DSS et les étapes clefs dans le développement de l'API Python.

3.3.1 Développement d'API dans Dataiku DSS

De même qu'il est possible d'intéragir avec une instance et des projets Dataiku DSS au travers de l'interface graphique, il est parfois utile d'utiliser un programme pour agir sur l'instance ou le projet. En effet, cela permet d'automatiser des tâches ou de faire le lien entre deux programmes. On utilise pour cela une interface logicielle appelée API. Ces API peuvent être des librairies logicielles comme les deux API Python (`dataiku` et `dataikuapi`) qui facilitent les intéractions programmées avec l'application. Ces API implémentent des fonctionnalités élémentaires. À titre d'exemple, elles permettent de se connecter à une instance Dataiku DSS, sélectionner un projet puis un scenario dans ce projet et exécuter ce dernier. Ces fonctionnalités permettent à d'autres développeurs d'implémenter des fonctionnalités plus avancées répondant à leurs besoins spécifiques. En particulier, cela peut

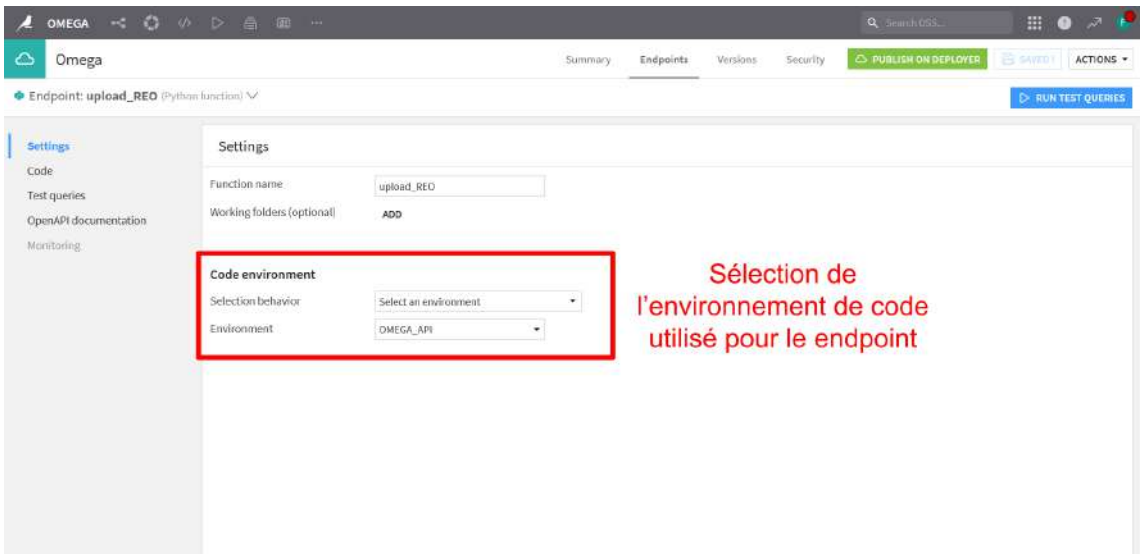


FIGURE 3.5 – Interface de paramétrage des endpoints dans Dataiku DSS

être fait dans une API agissant comme un service web. Ce deuxième type d'API permet à un utilisateur externe d'agir sur un logiciel au travers de requêtes web. Ces requêtes sont des requêtes HTTP / HTTPS envoyées sur une adresse URL précise. Dans la suite, ces URL seront appelés endpoints .

L'API qu'il m'a été demandé de réaliser fait partie du second type d'API. Recevant des requêtes en provenance de l'API intermédiaire, cette API permet d'exécuter des actions dans un projet OMEGA. Voici un cas d'usage simplifié démontrant l'utilité de l'API. En suivant le schéma de la Figure 3.1 de gauche à droite, un utilisateur peut saisir la spécialité de sous-officier à traiter pour une commission de PAM dans l'interface utilisateur. Cette information est transmise (à travers une requête HTTPS) à l'API intermédiaire qui envoie à son tour une requête HTTPS contenant la spécialité sur le endpoint dédié. L'API Python reçoit la requête et, grâce au code associé au endpoint, peut paramétrer le projet pour la spécialité en question.

Dataiku DSS possède une interface graphique dédiée au développement d'API Python. Cette interface très complète permet de gérer, paramétrer, programmer et tester chaque endpoint de l'API. L'interface de Dataiku DSS offre la possibilité de développer plusieurs API dans un même projet. Dans le cas d'OMEGA, une seule API nommée Omega est nécessaire. L'interfcae permet d'ajouter, renommer ou supprimer des endpoints dans l'API. Chaque nouveau enpoint doit être configuré. La Figure 3.5 montre l'interface prévue à cet effet. Elle permet notamment de définir le nom de la fonction Python définissant le comportement du endpoint et l'environnement de code à utiliser. La Figure 3.6 correspond à l'interface de paramétrage d'un environnement de code, ici, celui utilisé pour l'API d'OMEGA nécessitant plusieurs bibliothèques Python externes.
La Figure 3.7 montre l'interface d'édition du code d'un endpoint. Un endpoint correspond à une fonction Python (dont le nom est renseigné dans les paramètres du endpoint). Cette fonction Python peut prendre plusieurs arguments en entrée et renvoyer plusieurs valeurs. Il est possible dans cette fonction de faire appel à d'autres fonctions développées dans la zone de code du endpoint, dans les librairies internes au projet ou dans les librairies installées dans l'environnement de code. Il est aussi possible de lever des exceptions.
L'onglet test queries illustré dans la Figure 3.8, permet de tester le fonctionnement du endpoint en créant des requêtes dont le corps contient le nom et les valeurs des arguments de la fonction Python du endpoint au format JSON. Le résultat renvoyé par le endpoint est aussi une requête dont le corps contient les valeurs de retour de la fonction Python au format JSON.
À ce stade, l'API n'est pas déployée. C'est à dire qu'elle n'est pas accessible depuis le serveur web qui doit l'acceuillir. Suivant les motivations de la start-up Dataiku, DSS permet de déployer et maintenir facilement l'API grâce à l'API deployer dont l'interface est présenté en Figure 3.9. Déployer une nouvelle version de l'API nécessite moins de dix clics.
Finalement, l'API étant déployée elle peut à nouveau être testée en utilisant les outils intégrés à Dataiku DSS (onglet de test dans l'API deployer) ou, pour plus de fiabilité et de flexibilité, en utilisant un navigateur web ou un logiciel dédié comme Postman. Comme le montre la Figure 3.10, ce logicel présente de nombreux avantages.

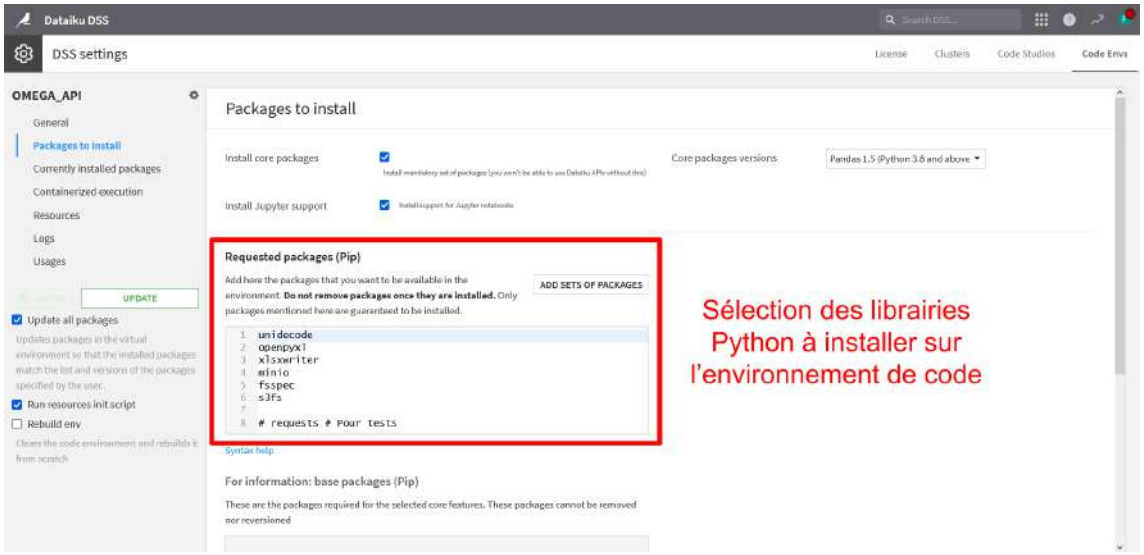


FIGURE 3.6 – Interface de paramétrage des environnements de code dans Dataiku DSS

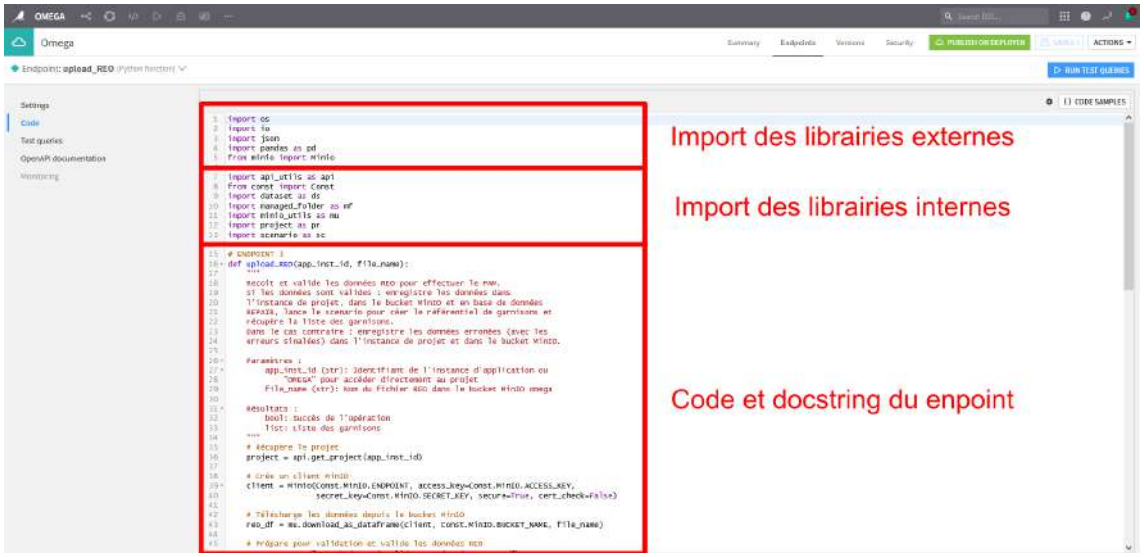


FIGURE 3.7 – Interface de programmation des endpoints dans Dataiku DSS

FIGURE 3.8 – Interface de test des endpoints dans Dataiku DSS

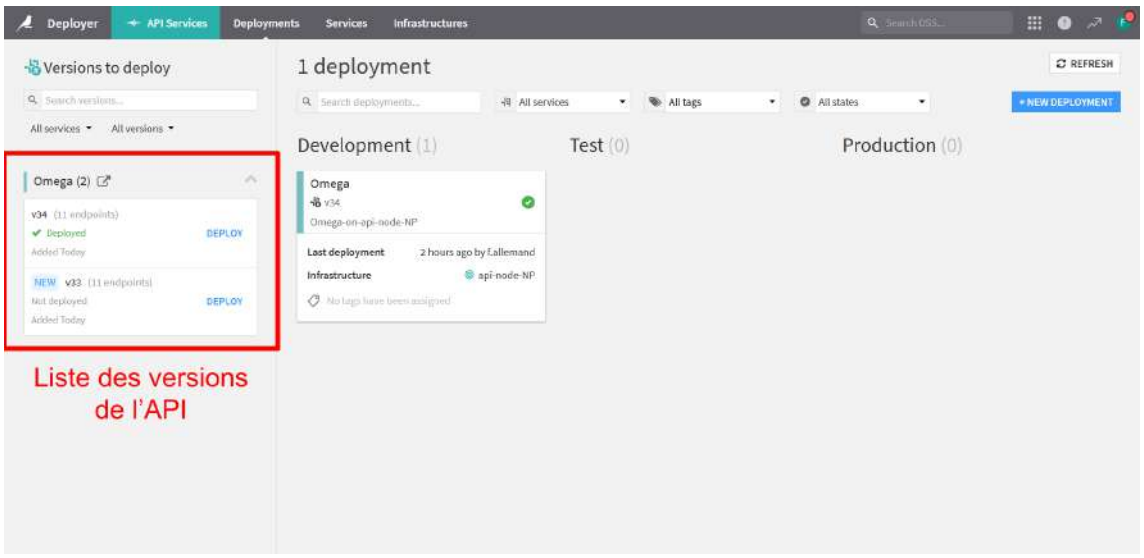


FIGURE 3.9 – Interface du déployeur d'API Dataiku DSS

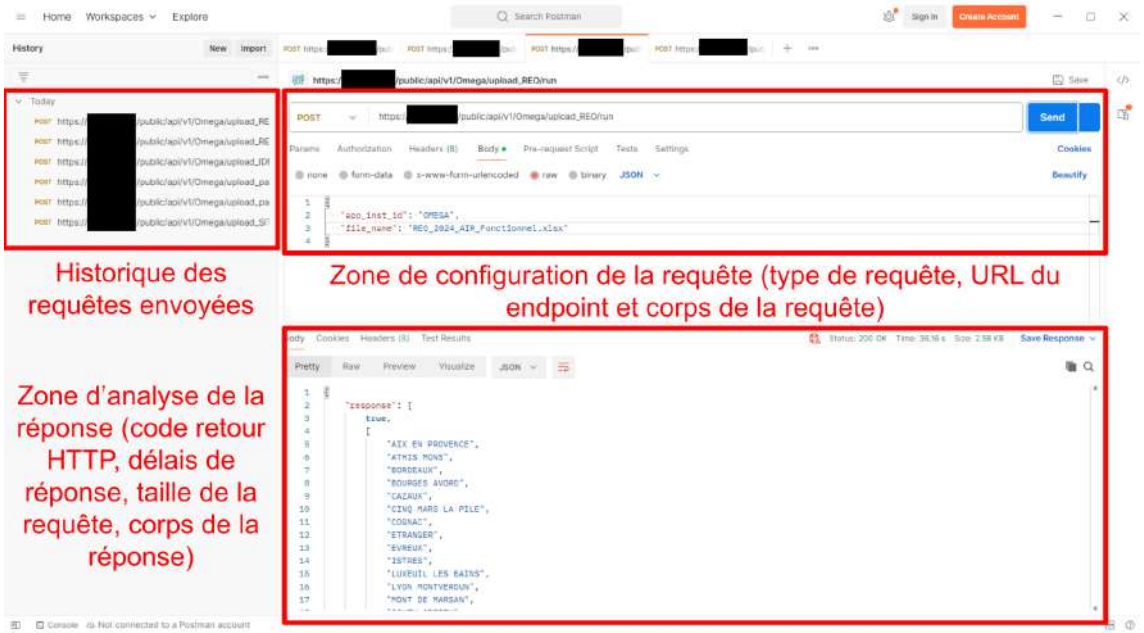


FIGURE 3.10 – Interface du logiciel Postman

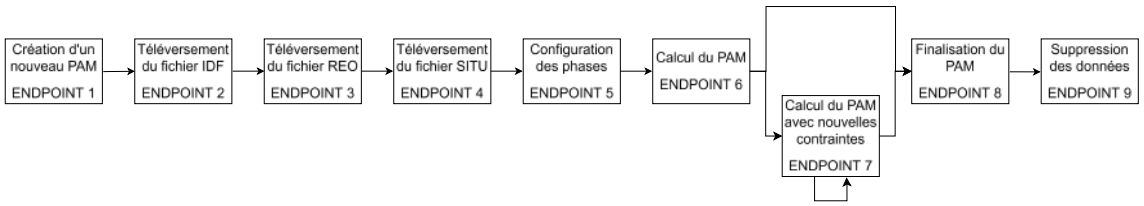


FIGURE 3.11 – Diagramme de flux des endpoints de l'API Python

3.3.2 Ebauche de l'API

Bien qu'au tout début du stage les spécificités des endpoints n'étaient pas encore définies, il fallait que je prépare tout pour être opérationnel le moment venu. Cela impliquait de me former sur les outils du développement d'API, à savoir : l'API designer, les librairies internes et les librairies Dataiku dataiku et dataikuapi. Il fallait aussi commencer à implémenter des fonctions nécessaires dans la librairie interne pour intéragir avec un projet Dataiku DSS et ses objets. Par exemple, j'ai implémenté une fonction permettant de lister les variables d'un projet Dataiku DSS dans la librairie interne puis j'ai créé le endpoint get_variables qui utilisait cette fonction pour renvoyer à un utilisateur les variables. Par la suite, ce endpoint me servait de endpoint témoin pour vérifier le bon déploiement de l'API. Les toutes premières versions de l'API n'était donc qu'un ensemble de endpoints permettant de tester les fonctionnalités développées dans la librairies internes.

3.3.3 Organisation de l'API

Le début du développement de OMEGA a révélé la nécessité d'au moins quatre endpoints : réception des données, calcul du PAM, re-calcul du PAM, finalisation du PAM. J'ai alors créé ces endpoints ou tout du moins leur structure puisque leur comportement précis était en grande partie encore inconnu.

Initialement constituée de quatre endpoints , la solution retenue compte désormais neuf endpoints dont le fonctionnement est présenté dans la Figure 3.11. Ce diagramme de flux montre l'ordre dans lequel les endpoints de l'API Python doivent être appelés. Cette évolution est apparue suite à une réunion avec le DIL (responsable de l'API intermédiaire). À ce stade du projet, les classeurs de l'utilisateur étaient convertis au format JSON (le format multipart form data n'étant pas supporté par Dataiku DSS) pour être directement reçus comme argument de fonction de endpoint. Cela se traduisait donc par des requêtes dont le corps était très volumineux. Les ressources des machines virtuelles et des réseaux étant limitées, le téléversement des fichiers ne pouvait donc pas se faire simultanément. La solution choisie fut de diviser le endpoint de réception des données en autant de endpoints que de fichiers téléversés par l'utilisateur (à savoir les fichiers IDF, REO et). Cette décision est aussi cohérente avec l'interface graphique de l'utilisateur.

3.3.4 Vérification des données

Le comportement de ces trois endpoints a rapidement passé de simple réception de fichier avec stockage dans la VM du projet et en base de donnée REPAIR (avec éventuellement le lancement d'un scénario) en une phase de vérification des données. Les phases de validation des données IDF, REO et SITU ont été transférées du flow de l'application aux endpoints de réception des données car il faut signaler à l'utilisateur, dans le cas de fichiers non conformes, où sont les erreurs. Il a été décidé de colorer les cellules erronées en rouge dans l'interface graphique. La solution la plus simple dans un premire temps était de renvoyer l'ensemble du tableau avec les cellules erronées marquées par le caractère "$" (puisque le format JSON utilisé pour échanger les données ne supporte pas de coloration) et de ne sauvegarder les données que si elles sont valides en effectuant la validation des données dans la fonction du endpoint.

La validation des données est toujours séparée en deux phases : une phase de préparation pour vérifier la cohérence avec le schéma attendu, sélectionner des données utiles pour et uniformiser les valeurs (par exemple, uniformisation des noms de villes : majuscules, sans accent, "SAINT" transformé en "ST"...) suivie d'une phase de test pour détecter des valeurs manquantes, des valeurs non conformes ou des doublons incompatibles (duplicata d'identifiant unique de militaire). Le traitement des données dans le back end du logiciel est schématisé par un diagramme de flux dans la Figure 3.12.

3.3.5 Transfert des données

La gestion du transfert des données a nécessité beaucoup d'effort. Une première étape de réflexion était nécessaire afin de savoir si le réseau permettrait le transfert de requêtes aussi volumineuses. La configuration du réseau et un simple test nous ont confirmé la compatibilité.

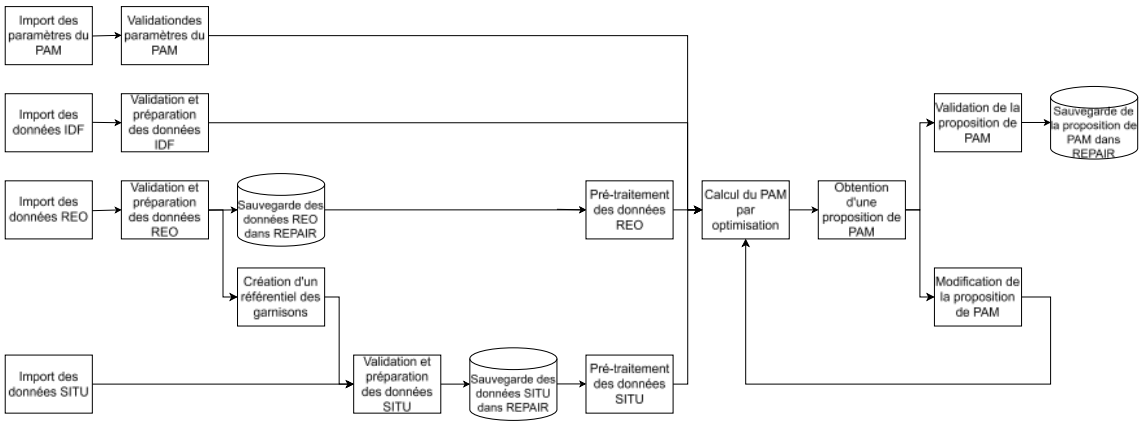


FIGURE 3.12 – Diagramme de flux des données

Nom du fichier	Description	Nombre d'attributs	Taille	Traitement	Temps de traitement
REO_2024_AIR_10.json	Données REO fournies par Wavestone	128	8870 Ko	Toutes les colonnes	3,820 s
REO_2024_AIR_10_min.json	Données REO minimales pour le calcul du PAM	15	1264 Ko	Toutes les colonnes	590 ms
REO_2024_AIR_10.json	Données REO fournies par Wavestone	128	8870 Ko	Colonnes utiles au calcul du PAM	2,031 s
REO_2024_AIR_10_min.json	Données REO minimales pour le calcul du PAM	15	1264 Ko	Colonnes utiles au calcul du PAM	554 ms
REO_2024_AIR_ESIOC_10.json	Données REO fournies par DRHAAE	49	6275 Ko	Colonnes utiles au calcul du PAM	1,513 s

TABLE 3.1 – Résultats des tests de traitement des données REO

Le fonctionnement de l'API a du être modifié une fois de plus suite aux premiers tests de communication entre l'API intermédiaire et l'API Dataiku DSS. Ce dernier ne supportant pas le transfert de fichier selon le protocole multipart form data, les données doivent être transférées au endpoint au format JSON. Cependant, la taille de la requête était trop importante pour être traitée par la machine virtuelle Dataiku DSS.

Nous avons effectué les tests avec les données contenues dans l'exemple de fichier REO. Ce fichier contenant 128 attributs pour près de 30000 postes resencés peut être utilisé pour créer des requêtes atteignant une taille de 100 Mo. Suite à plusieurs crash et après avoir testé avec plusieurs tailles de requêtes (environ 10 %, 25 % et 50 % du fichier REO de test), les ressources de la VM ont été revues à la hausse : 6 cœurs CPU au lieu de 4 et 8 Go de RAM au lieu de 2 Go, le responsable technique de la platefrome à aussi supprimé plus de 100 Go de fichiers de log, d'environnement de code et de projets non-utilisés pour ne pas saturer l'espace de stockage. Avant cette manipulation, seule la requête avec 10 % du fichier REO (2351 individûspour 6,33 Mo) fonctionnait.

Même si le traitement des données fonctionnait de bout en bout, le délais entre l'entrée des données et la réception des données corrigées dans l'API Java était bien trop élevé pour une utilisation convenable. Nous avons mesuré des temps d'attente côté utilisateurs de plus de deux minutes. J'ai réalisé un travail de rétro-ingénierie seul puis en collaboration avec un développeur REPAIR afin de limiter la taille du fichier REO. J'ai dans un premier temps supprimé tous les attributs non utilisés pour calculer le PAM afin d'obtenir les requêtes minimales. Les requêtes contenant alors seulement 15 attributs pesaient 14,25 % de la requête initiale diminuant drastiquement le temps de traitement. J'ai ensuite procédé à une optimisation au niveau du code Python afin de ne travailler dans Dataiku DSS (API et flow) qu'avec les données utiles pour le PAM. Cette optimisation permet de gagner plus d'une seconde lorsque les 128 attributs sont présents. Sur un fichier de petite taille avec peu d'attributs le gain de performance est négligeable. Les résultats de mes tests sont présentés dans la Table 3.1.

Malgré les résultats de mes test, il n'était pas envisageable de conserver uniquement les 15 attributs nécessaires au calcul du PAM. Il était important de trouver un compromis afin de conserver les attributs potentiellement utilisés par REPAIR [1]. En prenant exemple sur un fichier REO transmis par la DRHAAE et en supprimant tous les champs inutiles ajoutés par l'entreprise responsable du POC dans les fichiers de test, nous avons réduit le nombre d'attributs à 49. Ce compromis permet de gagner près d'une demie seconde sur un fichier de petite taille comme le montre la dernière ligne de la Table 3.1.

Au delà de mettre en valeur des zones d'améliorations des fichiers de tests et de l'implémentation, cette analyse a montré que le goulot d'étranglement du système venait de l'API Java. En effet, afin de transmettre les données à l'API Python, l'API Java doit lire le fichier tableur téléversé par l'utilisateur, le convertir au format JSON puis l'envoyer. Nous avons tout d'abord pensé à transmettre les données sous forme d'octets (afin d'éviter

1. L'équipe REPAIR propose le développement d'outils d'analyse et de visualisation de données. L'ensemble des données REO et SITU (et non seulement celles requises pour le calcul du PAM) consistent en une source de données fiable et intéressantes

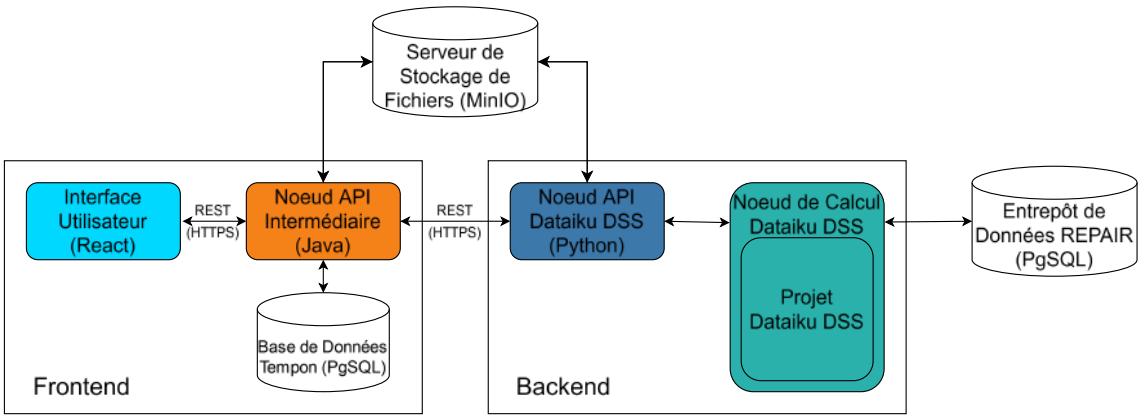


FIGURE 3.13 – Architecture détaillée du projet OMEGÀ avec transfert de fichiers via un serveur

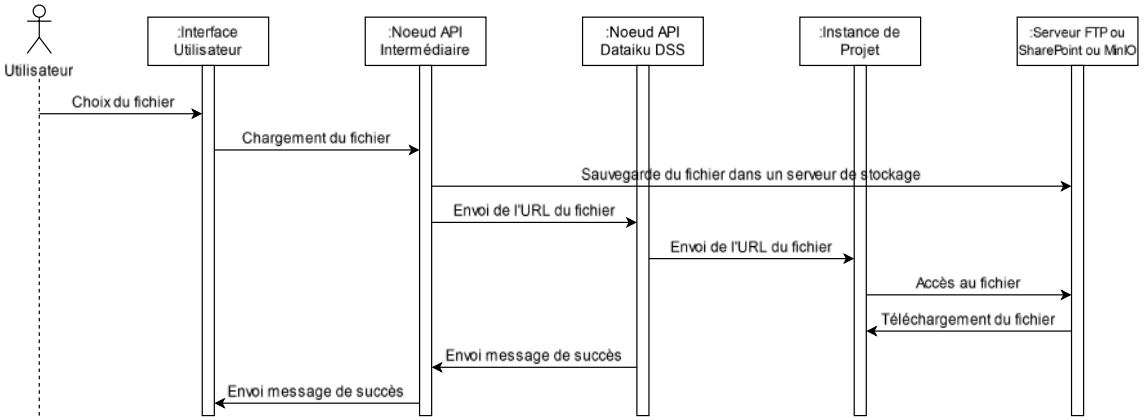


FIGURE 3.14 – Diagramme de séquence simplifié des transferts de fichiers

la lecture et la conversion) mais la solution retenue est d'utiliser un serveur de stockage pour échanger les fichiers comme illustré dans la Figure 3.13. Ce serveur de stockage permet de d'éviter tout traitement de donnée dans l'API grâce à l'utilisation d'un protocole adapté au transferts de fichiers. Les grands volumes de données ne circulent plus sur le lien HTTPS entre les deux API mais par les connections S3 du serveuur de stockage de fichiers. La Figure 3.14 est un diagramme de séquence simplifié détaillant les intéractions entre les différentes entités. Seul l'URL du fichier sur le serveur de fichier est transmis entre les deux API. Plusieurs options ont été étudiées : serveur *File Transfer Protocol* (FTP), le Microsoft SharePoint intégré à l'instance Dataiku DSS ou le MinIO associé à l'instance Dataiku DSS. Le serveur FTP a été abandonné au profit des deux autres solutions qu'il fallait donc tester et comparer pour choisir la meilleure. La priorité a été mise sur MinIO afin d'avoir un produit fonctionnel pour la première recette prévue pour le mois de novembre. Par manque de temps, l'option SharePoint n'a pas encore été testée.

Ce changement remet en cause la structure de l'API. Les fonctions de validation des données étaient jusqu'alors effectuées par l'API car les données arrivaient par le endpoint au format JSON et devaient être transfromées pour être intégrées au flow. Or, les données sont désormais stockées sur un serveur, le rôle de l'API serait plutôt d'ordonner au logiciel d'aller récupérer les données et d'effectuer le traitement dont il a besoin (par exemple avec un recipe Python dans le flow). Cela serait d'autant plus cohérent que la VM du nœud API est moins puissante que la VM dédiée au projet.

3.3.6 Gestion des utilisateurs multiples

La DRHAAE pouvant effectuer plusieurs PAM au même moment, l'API a du être adaptée pour fonctionner en cas d'utilisations simultanées. En effet, l'API intéragissant avec le projet, deux utilisateurs auraient pu téléverser leurs données à la suite et le premier obtenir les résultats du PAM non plus sur ses données mais sur celles du second. Dès le début du développement, nous avions anticipé cette problématique en ajoutant un identifiant de base de données retransmis à l'API intermédiaire et réutilisé par la suite pour utiliser les bonnes données provenant de REPAIR. Cependant, pour des raisons techniques, il a été décidé de créer les datasets dans le projets en partir des fichiers téléversés dans un *managed folder* et non plus directement de la base de données. Il est impossible de déposer un fichier dans le managed folder (avec un nom identifiant unique) et de créer

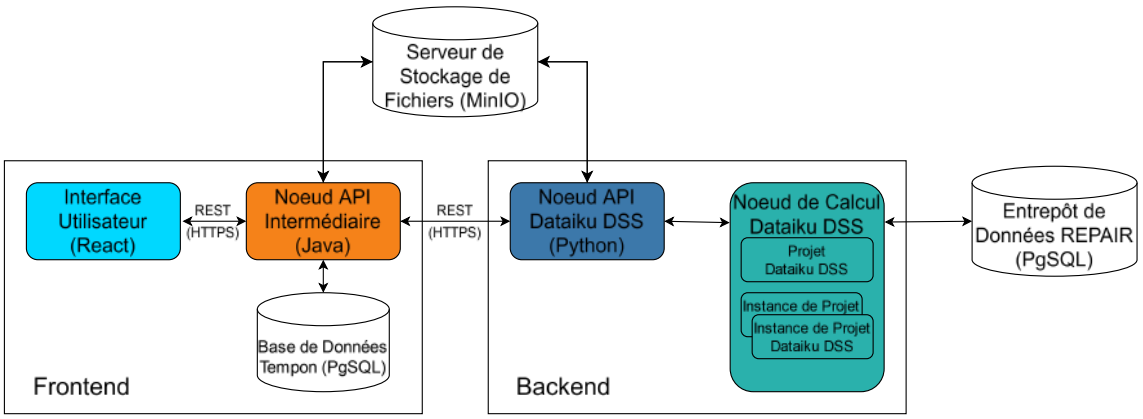


FIGURE 3.15 – Architecture détaillée du projet OMEGÀ avec gestion de plusieurs sessions

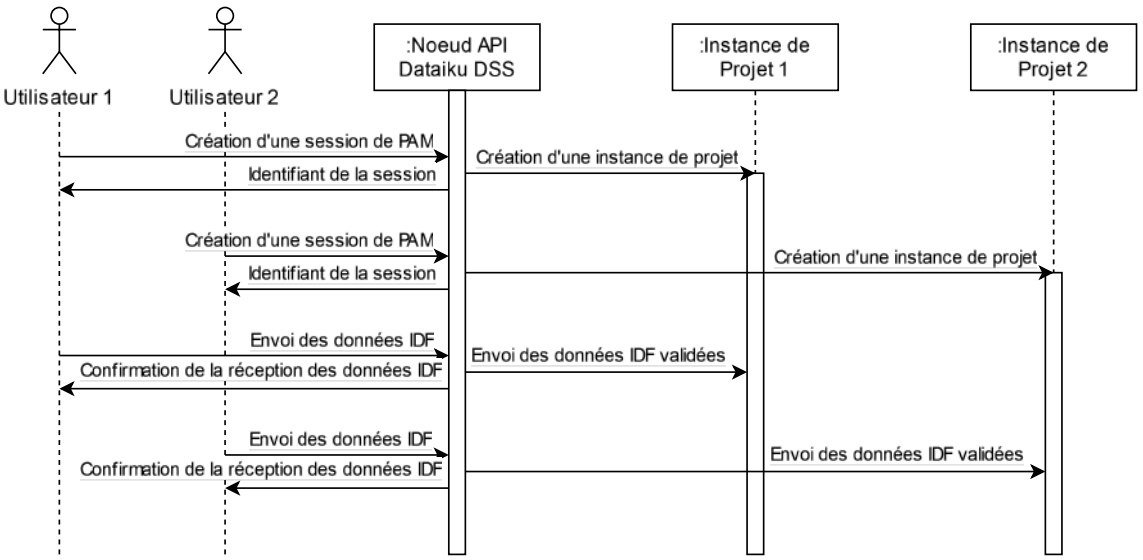


FIGURE 3.16 – Diagramme de séquence simplifié de la création des instances de projets

un nouveau dataset dans le flow au besoin. Le conseiller Dataiku qui nous accompagnait sur ce projet nous a recommandé d'utiliser des *applications* comme instances de projet. Cette méthode permet de créer une copie du projet entièrement indépendante comme illustré dans la Figure 3.15. L'identifiant de base de données est donc devenu une variable de projet et a été remplacé dans les endpoints par l'identifiant de l'instance de projet afin que l'API communique toujours avec la bonne instance de projet. Le fonctionnement de l'API avec la gestion de multiples utilisateurs est illustré dans le digramme de séquence simplifié de la Figure 3.16. On remarque qu'une nouvelle instance d'application est créée pour chaque utilisateur et éviter les accès concurents indésirés. Tout au long de sa session, chaque utilisateur communique uniquement avec l'instance d'application qui lui est associée grâce à un identifiant unique.

3.3.7 Démarche de qualité

Tout au long de la phase de développement de l'API, j'ai porté une attention particulière à la lisibilité et la compréhension du code. Pour cela, j'ai pris l'initiative de proposer des standards de programmation et de les rendre disponibles à toute l'équipe dans la documentation globale du projet, c'est à dire, dans le wiki du projet comme illustré dans la Figure 2.7. Les conventions utilisées pour programmer l'API et les librairies Python du projet OMEGA sont les suivantes :

- Code (nom de fonctions, variables...) en anglais
- Documentation du code (wiki, *docstrings* et commentaires) en français
- Les commentaires dans le code utilisent des verbes d'action conjugués
- Respecter au mieux les propositions des PEP 8 et PEP 257 [16, 14]

J'ai également écrit un tutoriel concernant le développement d'API dans Dataiku DSS. Ces mesures viennent compléter le travail de documentation qui m'a été demandé et qui est présenté dans la section suivante.

3.4 Écriture d'une documentation

Comme de nombreux projets informatiques, OMEGA est voué à évoluer au fil du temps : les règles de mutation des militaires de l'AAE changent régulièrement et les responsables des PAM à la DRHAAE sont susceptibles de demander des modifications ou améliorations au logiciel fourni. Le projet OMEGA est donc destiné à passer entre les mains de plusieurs développeurs pendant plusieurs années rendant la documentation du programme obligatoire.

Les chefs de projet ont été d'autant plus rigoureux sur ce point pour deux raisons. La première découle de la certification ISO9001 acquise par l'ESIOC. Dans le cadre de la qualité des produits et services fournis, la documentation joue un rôle majeur. Ensuite, le redéveloppement du projet OMEGA a nécessité l'analyse et la compréhension de plusieurs milliers de lignes de code issues de plusieurs développeurs successifs qui avaient construit par dessus le travail précédent sans supprimer le code mort et en ajoutant très peu de commentaires. Dans un contexte souffrant déjà des contraintes temporelles, la mauvaise qualité de ce code a fait perdre beaucoup de temps aux équipes de l'ESIOC. Apprenant de cette expérience, il m'a été demandé de tout mettre en place pour que mon successeur puisse poursuivre mon travail dans les plus brefs délais pour la phase de test de septembre et jusqu'au premier livrable prévu pour début novembre.

Il m'a donc été demandé de créer une documentation complète du projet OMEGA au niveau du code comme mentionné précédemment mais aussi dans le wiki du projet. En suivant le modèle de documentation décrit dans les parcours d'apprentissage, j'ai pu créer une documentation décrivant le projet dans son ensemble, la configuration de Dataiku DSS pour ce projet, les conventions adoptées (convention de code et de documentation), le traitement des données reçues par l'application, l'API et les librairies internes.

Ainsi ce wiki contient une documentation complète des fonctions endpoints avec les types des arguments, des exemples d'entrées et de sorties des fonctions et une explication du travail effectué comme on peut le voir dans les Figures 3.17 et 3.18. L'ensemble du traitement des données réalisé dans le flow du projet est lui aussi entièrement documenté comme le montre la Figure 3.19. J'ai pris l'initiative d'ajouter deux tutoriels traitant le développement et le déploiement d'API Python dans Dataiku DSS afin que des développeurs non formés sur Dataiku DSS puissent rapidement effectuer des modifications sur l'API sans nécessairement passer par les parcours d'apprentissages assez long à suivre.

De façon générale, j'ai essayé de rendre la documentation la plus complète et la plus compréhensible possible afin de répondre au mieux aux attentes d'un nouveau développeur sur le projet. Afin de rendre la documentation plus visuelle, j'ai créé plusieurs schémas décrivant le fonctionnement d'OMEGA comme celui visible sur le dashboard dans la Figure 2.6. J'ai aussi pris le temps d'insérer et maintenir à jour une grande quantité d'exemples dans la documentation. Ces exemples permettent d'illustrer de façon très précise des concepts parfois difficiles à comprendre par écrit. Ce sont notamment les exemples de requêtes entrantes et sortantes de endpoint visibles sur la Figure 3.17.

Pour compléter ma formation sur Dataiku DSS, j'ai créé deux webapps sur Dataiku DSS l'une basée sur Dash (Figure 2.8) et la seconde sur Bokeh (Figure 2.9), les deux technologies de webapp Python compatibles avec Dataiku DSS. Ces deux applications présentent toutes les deux des graphiques intéractifs pour les données REO ou SITU et peuvent toute deux être utilisées comme telles ou comme exemple si une webapp doit être développée pour OMEGA, complétant ainsi les exemples documentant ce projet.

Dataiku DSS étant un logiciel ambitieux qui ne cesse d'évoluer et venant tout juste d'être adopté par les équipes de l'ESIOC. Pendant mon stage, j'ai réalisé une activité de bug tracking / reporting et d'expérimentation. Cela consiste à repérer les bugs ou les défauts de Dataiku DSS et de les documenter dans un fichier. Cela permet, dans un premier temps, de partager mon expérience aux autres développeurs et responsables de la plateforme Datiku DSS à l'ESIOC. Dans un second temps, il est possible d'effectuer des bugs reports sur le forum Dataiku ou de transmettre ces retours directement au conseiller Dataiku. Cette zone collaborative de suggestions d'améliorations de Dataiku, présentée dans la Figure 3.20, DSS s'inscrit dans une démarche de participation à l'amélioration du logiciel en tant qu'utilisateur.

Comme le montre la Figure A.1b de l'Annexe A.2, ce travail de documentation a commencé dès le début de ma contribution sur le projet. Parfois chronophage, le maintien de la documentation s'effectue tout au long de la durée de vie du logiciel afin d'en assurer la pérennité.

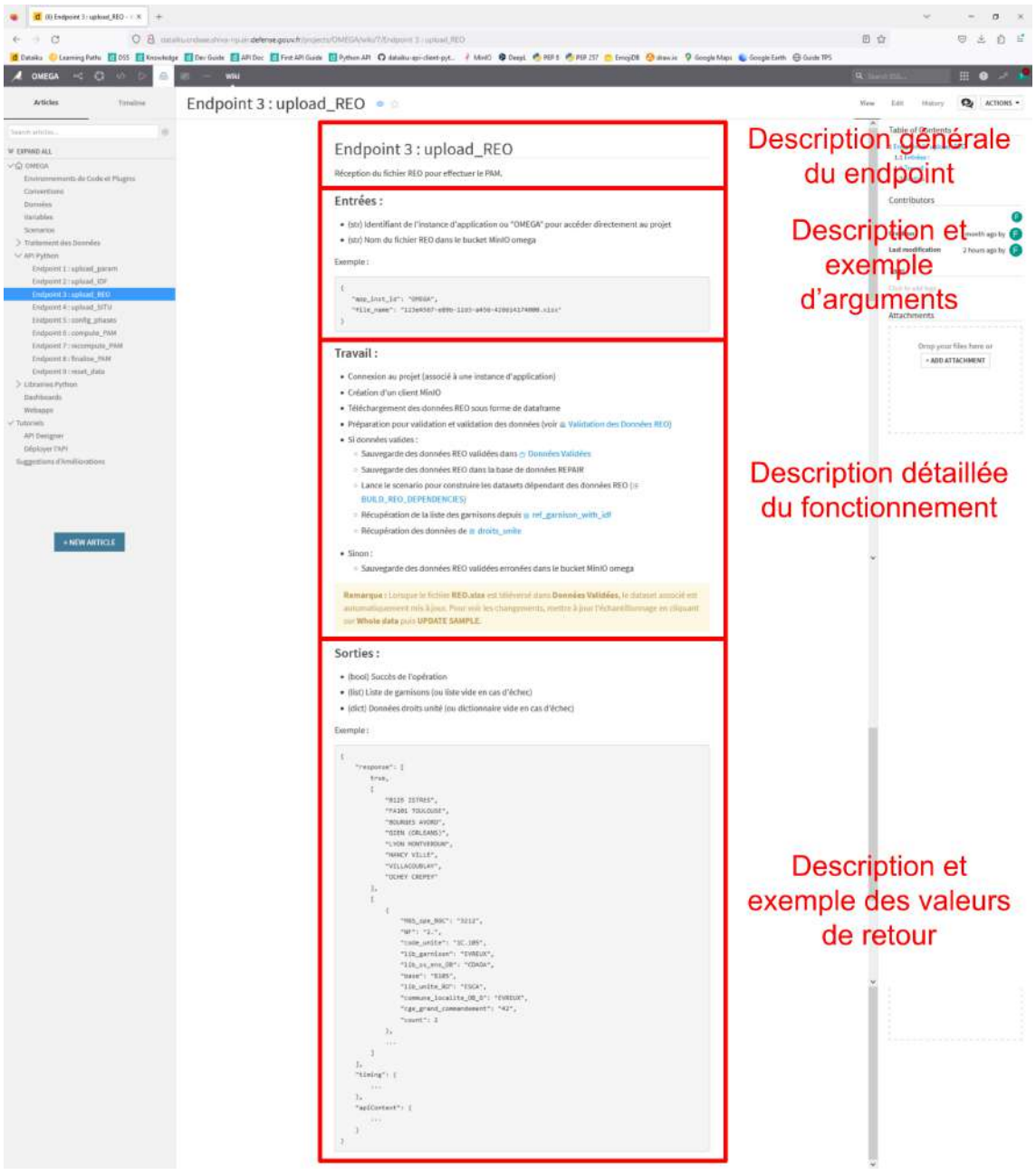


FIGURE 3.17 – Article du wiki documentant un endpoint de l'API Dataiku DSS

FIGURE 3.18 – Article du wiki documentant la validation des données effectuée dans un endpoint de l'API Dataiku DSS

FIGURE 3.19 – Article du wiki documentant la préparation des données effectuée dans le flow Dataiku DSS

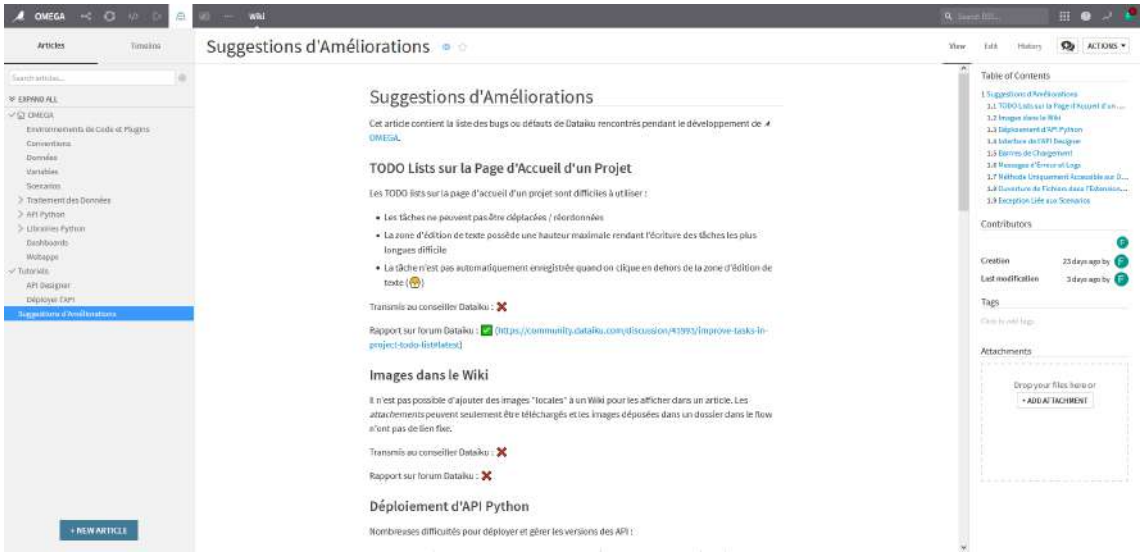


FIGURE 3.20 – Article du wiki contenant les suggestions d'améliorations de Dataiku DSS

Chapitre 4

Projet reconnaissance de timbres de confidentialité

L'AAE dispose de ressources limitées. Les ressources humaines, particulièrement précieuses, sont régies de façon à maximiser leur efficacité. Ainsi, la plupart des développeur de l'ESIOC sont assignés sur plusieurs projets avec différents niveaux de priorité. Lors de ce stage, une mission secondaire m'a également été attribuée. Considérée comme de priorité inférieure au travail sur OMEGA, elle consistait à effectuer quelques améliorations sur un projet expérimental du DND. Cette section présente le projet et comment j'ai pu y contribuer.

4.1 Description du projet

Divisée en deux sous-sections couvrant d'une part le contexte et d'autre part les aspects techniques, cette section a pour objectif de présenter le projet de reconnaissance de timbres de confidentialité.

4.1.1 Contexte et motivation

Au sein de l'AAE, la protection des données est un enjeu majeur. Ainsi chaque document se voit assigné un niveau de protection indiquant qui peut y avoir accès et où la donnée peut être stockée. Ce niveau de protection est signalé par un tempon (manuel ou numérique) et éventuellment un filigramme sur le document. Cependant, ce niveau de protection est souvent géré manuellement pouvant conduire à des erreurs. Le projet de reconnaissance de timbre de confidentialité surnommé RECO_CLASSIF a pour objectif de créer un outil numérique permettant de détecter les erreurs de gestion des données classifiées.

Entièrement créé sur la plateforme Dataiku DSS, RECO_CLASSIF est un logiciel utilisant l'apprentissage machine pour analyser le contenu de dossiers et identifier les mentions de niveau de protection sur les documents quel que soit leur format. Cette classification des documents par intelligence artificielle permet de repérer automatiquement et régulièrement (en utilisant les capacités d'automatisation de Dataiku DSS) les fichiers protégés sauvegardés au mauvais endroit.

Actuellement considéré comme projet expérimental afin d'apprécier les capacités et limites de Dataiku DSS sur des tâches d'apprentissage machine et de vision par ordinateur, RECO_CLASSIF pourrait à terme être utilisé comme outil interne à l'ESIOC. De plus, si son fonctionnement est satisfaisant, un déploiement à plus grande échelle pourrait être envisagé. Une fois déployé, ce logiciel contribuerait activement à la sécurité des SI de l'AAE.

4.1.2 Fonctionnement

Développé sur la platefrome Dataiku DSS afin d'en évaluer les performances, RECO_CLASSIF suit l'organisation classique d'un projet d'apprentissage machine. Comme le montre la Figure 4.1, il s'articule autour des deux étapes principales que sont le pré-traitement des données et l'entraînement des modèles. Ce projet servant d'expérimentation, il a la particularité de contenir trois zones d'entraînements afin de comparer différentes méthodes : l'utilisation des modèles pré-implémentés dans Dataiku DSS et l'utilisation de modèles personnalisés avec et sans *transfer learning*. Les capacités de mise en production offertes par Dataiku DSS expliquent la présence d'une zone d'automatisation.

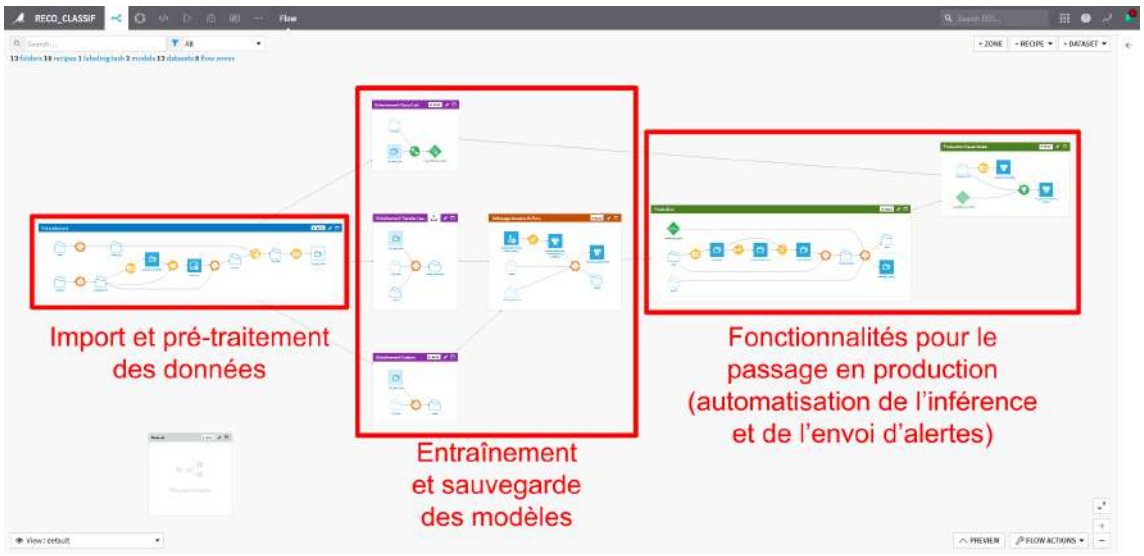


FIGURE 4.1 – Flow du projet RECO_CLASSIF

Pour des raisons de sécurité, les données classifiées utilisées pour l'apprentissage machine se sont pas des données réelles. Ces données ont dû être crées spécifiquement pour ce projet en utilisant les exemples de documents utilisés par l'administration de l'AAE. Ces données peuvent être des lettres, des arrêtés ou des formations internes. De même, on retrouve une grande variété de documents non classifiés comme des menus de cantine. Il est important de remarquer qu'en plus de la diversité de documents, il y a une diversité dans le marquage des documents. Certains documents sont marqués par un tempon manuel alors que d'autres sont entièrement numériques. La position des timbres varie aussi d'un document à l'autre et certains défauts ont été ajoutés consciemment, en particulier sur les documents numériques (timbre partiellement masqué par une image). Quelques exemples de documents sont disponibles dans l'Annexe C.1.

Ce projet ayant majoritairement été développé par un data scientist du DND, il m'a été demandé de parfaire son travail suite à son départ. Ma contribution à ce projet est détaillée dans la section suivante.

4.2 Améliorations

Cette section présente dans un premier temps la phase de plannification en amont dû travail d'amélioration qui est décrit dans un second temps.

4.2.1 Plannification

Le data scientist responsable du projet dit RECO_CLASSIF m'a présenté son travail avant son départ, c'est à dire à la moitié de mon stage, le 31 juillet, comme indiqué dans le journal de bord de l'Annexe D. Nous avons discuté et établi une liste des potentielles améliorations à apporter à ce projet. Le stâches se regroupent en trois catégories comme suit :

- Données :
  - Vérifier le bon fonctionnement de la transformation en images des documents Libre Office ou PDF
  - Vérifier le bon fonctionnement du regroupement des données
  - Equilibrer classes
- Entraînements :
  - Faire varier la taille des images
  - Faire varier la taille des *batches*
  - Utiliser le réseau EfficientNet
  - Faire varier le nombre et la taille des couches denses
- Passage en production :
  - Passage en production sur un nœud *automation*
  - Essayer de lancer une inférence manuellement
  - Envoyer des alertes (mail)

Ce projet a du être transféré de l'instance DR à l'instance NP de Dataiku DSS afin que je puisse y accéder. Les données utilisées étant des données factices (c'est à dire des faux documents officiels avec timbres de confidentialité créés pour cet usage), ce transfert ne pose pas de problème au niveau de la sécurité. Cependant,

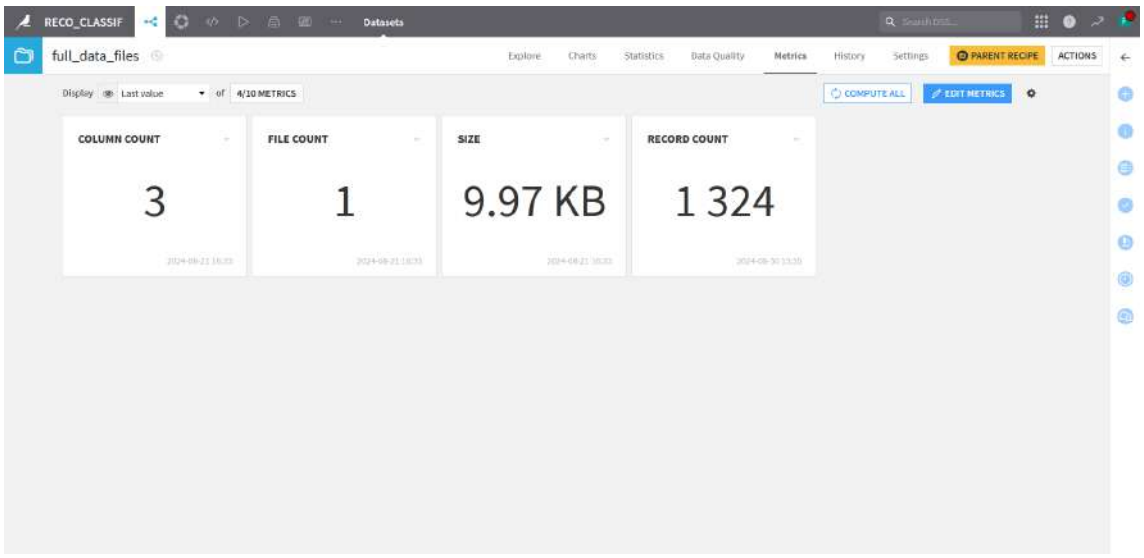


FIGURE 4.2 – Interface de contrôle des métriques d'un dataset dans Dataiku DSS

j'ai dû collaborer avec le responsable technique de la plateforme afin de créer et configurer correctement les environnements de code nécessaires. Les sous-sections suivantes décrivent les améliorations que j'ai apportées à ce projet.

4.2.2 Gestion des données

Les données étant la clef de l'apprentissage machine, je me suis tout d'abord concentré sur les tâches les concernant. Cela impliquait notamment de vérifier le bon fonctionnement de la pipeline de pré-traitement des données puis de mettre en place des tests pour contrôler la création des datasets.

Je me suis tout d'abord concentré sur la vérification du pré-traitement des données. Pour cela, j'ai vérifié le bon fonctionnement de Libre Office sur la machine virtuelle de l'instance Dataiku DSS. Pour cela, j'ai excécuté la recipe (en partie visible sur la Figure 4.6) permettant de transformer les fichiers d'origine numérique (traitement de texte, diaporama, PDF et images) en images et j'ai vérifié manuellement que les fichiers semblaient bien convertis.

Dans un second temps, j'ai ajouté différents checks et *data quality rules* afin de contrôler la création des datasets à partir des fichiers convertis. Ces fonctionnalités de Dataiku DSS permettent de contrôler en temps réel le bon déroulement des opérations dans le flow. En effet, il est possible de calculer différentes métriques sur les datasets comme l'illustre la Figure 4.2. Il est ensuite possible de mettre en place des vérifications appelées checks (pré-implémentées ou personnalisées) permettant de contrôler automatiquement le projet en fonction du résultat de la vérification (par exemple, ré-entraîner le modèle si la métrique est trop faible) et d'envoyer des alertes. Les data quality rules, introduites dans les versions les plus récentes du logiciel, offrent une analyse sur le long terme de la qualité des données. Ces règles (pré-implémentées ou personnalisées comme dans la Figure 4.4) permettent de suivre l'évolution de la qualité des données au cours du temps et à différents niveaux (projet, instance de projet, dataset...).
Dans une démarche d'expérimentation j'ai créé un check et une data quality rule permettant de contrôler la formation du dataset contenant l'ensemble des données, à savoir les données au format papier scannées et les fichiers numériques. Ainsi, lors de l'exécution de la recipe créant le dataset, des messages d'alerte apparaissent si le nombre total de fichiers est incorrect, c'est à dire si le nombre de fichiers n'est pas égal au nombre de fichiers scannés et numériques. Les résultats de ces deux outils de contrôles sont visibles dans les Figures 4.3 et 4.5.

4.2.3 Autres améliorations

Je n'ai malheureusement pas pu effectuer les tâches relatives aux entraînements car la machine virtuelle de l'instance Dataiku DSS NP n'est pas assez performante. En effet, cette machine virtuelle n'a pas de GPU, composant permettant d'accélérer l'entraînement des modèles. En effet, sans les capacités de calcul parallèle de ce composant, chaque entraînement aurait duré plusieurs jours et aurait monopolisé une grande parties des ressources de la VM partagée avec les autres développeurs, dégradant l'expérience d'utilisation (latence et lenteurs de calcul).

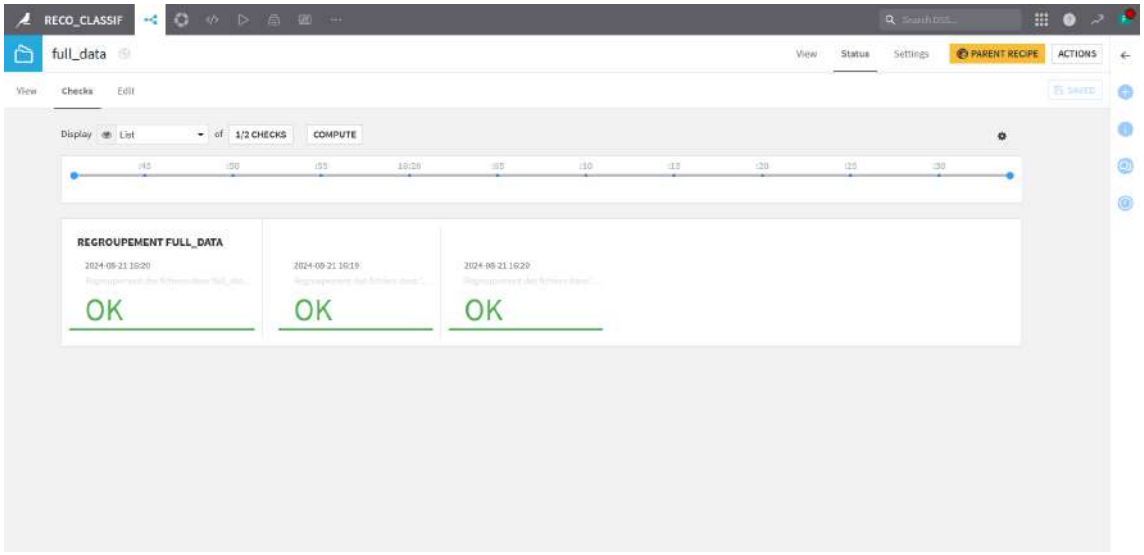


FIGURE 4.3 – Interface de contrôle des checks dans Dataiku DSS

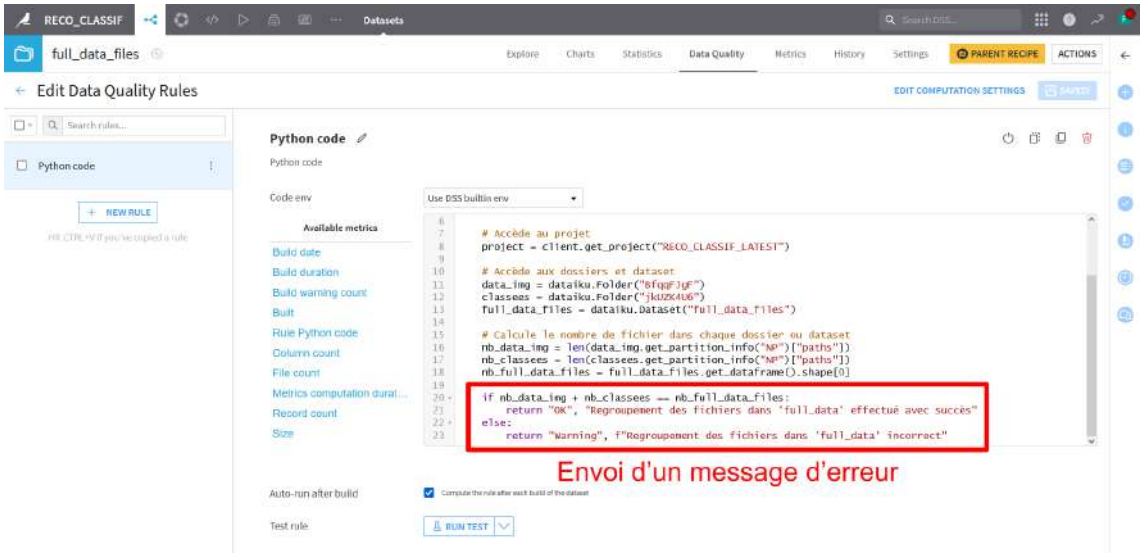


FIGURE 4.4 – Interface de développement d'une data quality rule dans Dataiku DSS

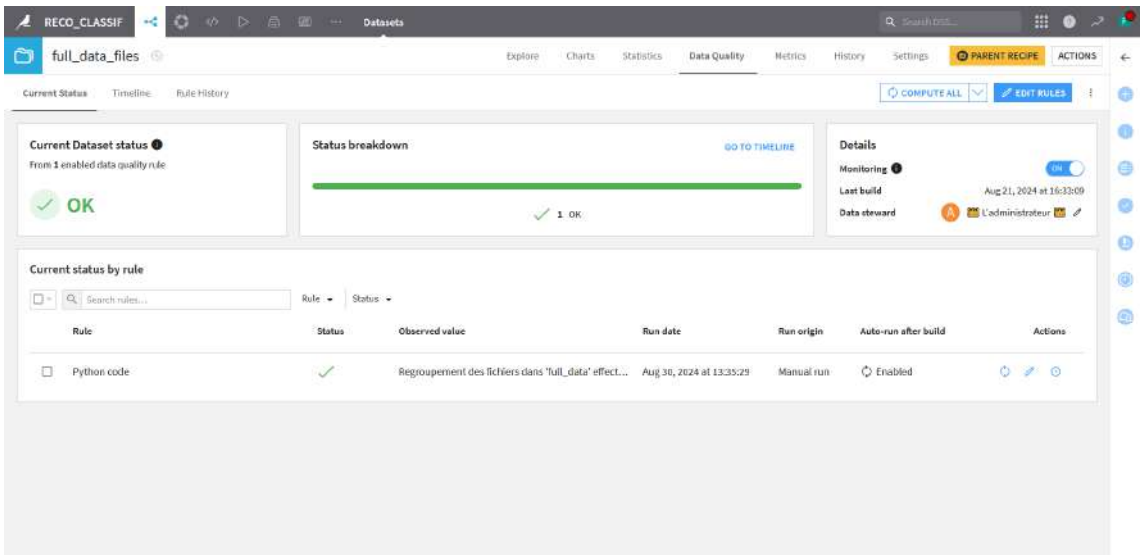


FIGURE 4.5 – Interface de contrôle des data quality rules dans Dataiku DSS

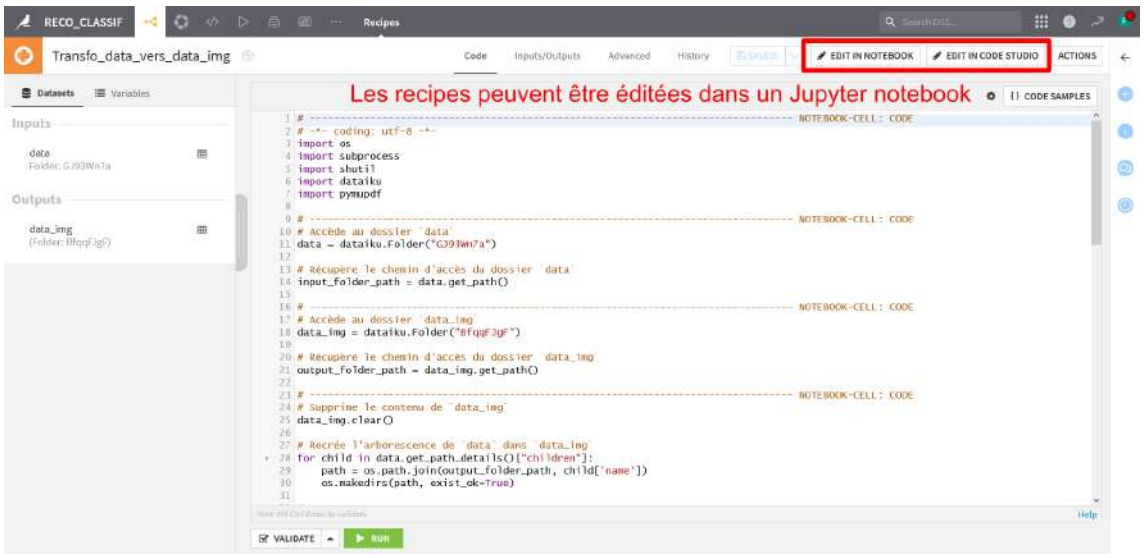


FIGURE 4.6 – Recipe Python dans Dataiku DSS après restructuration, optimisation et documentation du code

Afin de toujours maintenir le niveau de qualité requis à l'ESIOC, j'ai pris l'initiative de restructurer et commenter le code des recipe Python afin de suivre un standard (en l'ocurrence les mêmes règles que pour le projet OMEGA, décrites à la section 3.3.7). La Figure 4.6 montre le code d'une recipe Python après ce travail. J'ai aussi ajouté un wiki au projet. Cette documentation de plusieurs pages contient une description générale du projet ainsi que le fonctionnement des différents éléments du flow.

Conclusion

Réalisé au sein d'un escadron de l'Armée de l'Air et de l'Espace (AAE) spécialisé dans les Systèmes d'Information et de Communication (SIC), ce stage d'une durée de deux mois m'a permis d'assurer les fonctions d'un développeur - data scientist.
La majorité de mon temps de travail a été occupé par la création d'une API Python, interface logicielle de communication entre l'interface utilisateur, le back end Dataiku Data Science Studio (DSS) et les bases de données contituant une application web d'aide à la décision. Destinée à assurer la qualité des affectations des militaires selon leurs vœux, cette application traite les données personnelles des sous-officiers.
L'ESIOC mène aussi des projets expérimentaux destinés à se former et tester les nouveaux logiciels. Le projet de reconnaissance de timbre de confidentialité (qui pourrait être déployé à terme) auquel j'ai apporté quelques améliorations (vérifications automatiques, qualité de code, documentations) éprouve les capacités de Dataiku DSS sur les tâches d'apprentissage machine et de vision par ordinateur.

Travailler dans un contexte militaire implique le respect de règles de sécurité spécifiques. Ces règles s'appliquent dès l'accès au matériel et coditionnent le développement informatique. À ces règles s'ajoutent les critères de la certification ISO9001 acquise par l'ESIOC que j'ai dû respecter tout au long du stage afin d'assurer la qualité des logiciels fournis.

Afin de mener a bien les missions qui m'ont été confiées, j'ai dû me former sur la plateforme de science de données collaborative Dataiku DSS. C'est au travers de cette plateforme que j'ai utilisé des compétences acquises en école d'ingénieur telles que le développement d'applications web, les bonnes pratiques de génie logiciel et la programmation Python. Plus généralement, il m'a fallu faire preuve de capacité de rétro-ingénierie et de compréhension de code.

Travailler dans un escadron de l'AAE offre un cadre de travail varié. Pendant ce stage, j'ai pu travailler en autonomie notamment pour rechercher et développer des solutions techniques ce qui m'a emmené à organiser mon travail, faire preuve d'esprit critique et prendre des initiatives. D'autres phases se sont déroulées en équipe en collaborant avec des militaires ou des civils. Ces temps de travail nécessitaient de bonnes capacités de communication.

Finalement, ce stage effectué dans un contexte où la sécurité joue un rôle important m'a permis d'utiliser des compétences acquises au préalable et d'en acquérir de nouvelles (notamment la maîtrise de Dataiku DSS) en travaillant sur le développement d'un logiciel de support à la décision dont la mise en production pour plusieurs années est prévue pour octobre.

Bibliographie

[1] Cmdt territorial - Armée de l'air et de l'espace. *Le CTAAE en 2 minutes*. URL : https://www.youtube.com/watch?v=ET4QmCSuFzw&ab_channel=Cmdtterritorial-Arm%C3%A9edel%E2%80%99airetdel%E2%80%99espace. (consulté : 30/08/2024).

[2] api.gouv.fr. *Qu'est-ce qu'une API ?* URL : https://api.gouv.fr/guides/api-definition. (consulté : 30/08/2024).

[3] Ministère des Armées. *Base aérienne 118 Mont-de-Marsan*. URL : https://devenir-aviateur.fr/venir/implantations/base-aerienne/base-aerienne-118-mont-de-marsan. (consulté : 30/08/2024).

[4] Ministère des Armées. *Direction générale du numérique : l'escadron des systèmes d'information opérationnels et de cyberdéfense (ESIOC)*. URL : https://www.defense.gouv.fr/dgnum/employeurs-numeriques-du-ministere-armees/lescadron-systemes-dinformation-operationnels-cyberdefense-esioc. (consulté : 30/08/2024).

[5] Ministère des Armées. *Direction générale du numérique : le numérique et les SIC*. URL : https://www.defense.gouv.fr/dgnum/numerique-sic. (consulté : 30/08/2024).

[6] Ministère des Armées. *Responsabilité sociale*. URL : https://www.defense.gouv.fr/sga/nous-connaitre/responsabilite-sociale. (consulté : 30/08/2024).

[7] Ministère des Armées. *Site du recrutement de l'armée de l'air et de l'espace : QUESTIONS FREQUENTES*. URL : https://academy.dataiku.com/page/learning-paths. (consulté : 30/08/2024).

[8] Dassault Aviation. *Constantin « Kostia » Rozanoff*. URL : https://www.dassault-aviation.com/fr/passion/histoire/biographies/constantin-kostia-rozanoff/. (consulté : 30/08/2024).

[9] Dataiku. *About Dataiku*. URL : https://www.dataiku.com/company/. (consulté : 30/08/2024).

[10] Dataiku. *API Deployment*. URL : https://academy.dataiku.com/api-deployment?next=%2Fapi-deployment%2F1993373. (consulté : 30/08/2024).

[11] Dataiku. *Everyday AI, Extraordinary People*. URL : https://www.dataiku.com/everyday-ai/. (consulté : 30/08/2024).

[12] Dataiku. *Learning paths*. URL : https://academy.dataiku.com/page/learning-paths. (consulté : 30/08/2024).

[13] Dataiku. *Supported connections*. URL : https://doc.dataiku.com/dss/latest/connecting/connections.html. (consulté : 30/08/2024).

[14] Guido van Rossum David Goodger. *PEP 257 – Docstring Conventions*. URL : https://peps.python.org/pep-0257/. (consulté : 30/08/2024).

[15] Antoine Frêlon. « Nouvelles frontières : la très haute alitude ». In : *Air Actualités* Juin 2024.770 (2024), p. 32-33.

[16] Alyssa Coghlan Guido van Rossum Barry Warsaw. *PEP 8 – Style Guide for Python Code*. URL : https://peps.python.org/pep-0008/. (consulté : 30/08/2024).

[17] Lieutenant Thomas Hory. « Il était une fois... l'Armée de l'Air et de l'Espace ». In : *Air Actualités* Juin 2024.770 (2024), p. 34-45.

[18] Lieutenant Thomas Hory. « Prospective : plongée au cœurs des technologies quantiques ». In : *Air Actualités* Mai 2024.769 (2024), p. 20-21.

[19] *La base aérienne 118 « Colonel ROZANOFF » située dans le département des Landes à Mont-de-Marsan, forte de de plus de 3500 personnels, met en œuvre les systèmes d'armes les plus modernes, participe au quotidien à la protection des français et contribue à l'amélioration permanente de nos capacités de défense.* URL : https://www.reservistes.defense.gouv.fr/document-poste/6968. (consulté : 30/08/2024).

[20] Préfet de Landes. *Historique et missions*. URL : https://www.landes.gouv.fr/Services-de-l-Etat/Defense/Delegation-militaire-departementale/La-base-aerienne-118/Historique-et-missions. (consulté : 30/08/2024).

[21] Préfet de Landes. *La base aérienne 118*. URL : https://www.landes.gouv.fr/Services-de-l-Etat/Defense/Delegation-militaire-departementale/La-base-aerienne-118. (consulté : 30/08/2024).

[22] Pierre-Jean Lajoie MICHAUD. *Le développement logiciel agile : tout ce que vous devez savoir*. URL : https://www.nexapp.ca/blogue/developpement-logiciel-agile. (consulté : 30/08/2024).
[23] Organisation internationale de NORMALISATION. *ISO 9001 :2015*. URL : https://www.iso.org/fr/standard/62085.html. (consulté : 30/08/2024).
[24] WIKIPEDIA. *Base aérienne 118 Mont-de-Marsan*. URL : https://fr.wikipedia.org/wiki/Base_a%C3%A9rienne_118_Mont-de-Marsan. (consulté : 30/08/2024).
[25] WIKIPEDIA. *Constantin Rozanoff*. URL : https://fr.wikipedia.org/wiki/Constantin_Rozanoff. (consulté : 30/08/2024).
[26] WIKIPEDIA. *Dataiku*. URL : https://fr.wikipedia.org/wiki/Dataiku. (consulté : 30/08/2024).
[27] WIKIPEDIA. *Interface de programmation*. URL : https://fr.wikipedia.org/wiki/Interface_de_programmation. (consulté : 30/08/2024).
[28] WIKIPEDIA. *ISO 9001*. URL : https://fr.wikipedia.org/wiki/ISO_9001. (consulté : 30/08/2024).
[29] WIKIPEDIA. *Marensin*. URL : https://fr.wikipedia.org/wiki/Marensin. (consulté : 30/08/2024).
[30] WIKIPEDIA. *Test driven development*. URL : https://fr.wikipedia.org/wiki/Test_driven_development. (consulté : 30/08/2024).

Annexe A

Annexes Communes

A.1 Responsabilité sociétale

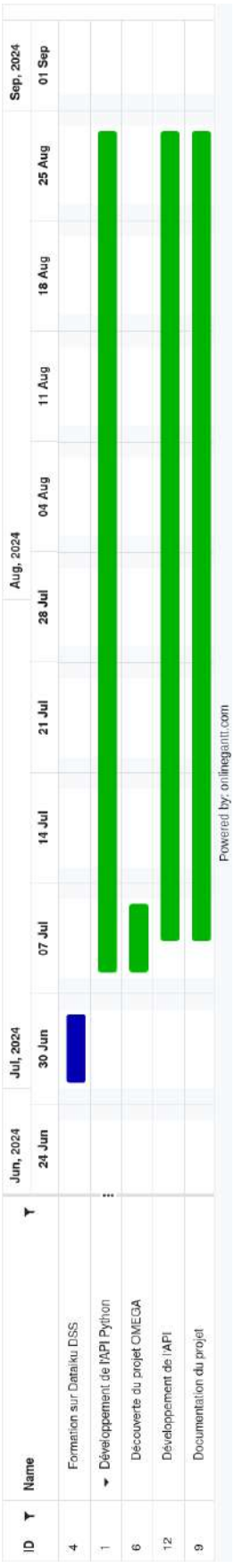
Les missions de l'Armée de l'Air et de l'Espace (AAE) s'inscrivent essentiellement dans le 16ème objectif de développement durable de l'ONU « intitulé paix, justice et institutions efficaces ». L'AAE doit avant tout assurer la sécurité des territoires français au travers de sa gouvernance aérienne. Cependant, en tant qu'employeur, le ministère des Armées est porteur de plusieurs politiques de responsabilité sociétale. Malgré les fortes contraintes et l'engagement exigeant des militaires, il assure l'égalité professionnelle entre les femmes et les hommes, la diversité, l'inclusion ainsi qu'une vie privée épanouie grâce au plan famille et la charte du temps [6].

Actuellement l'Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence (ESIOC) développe un logiciel de personnalisation de formations des pilotes et une application web d'aide à la décision pour la gestion des mutations des sous-officiers. Cet escadron est donc en accord avec les objectifs durables de qualité d'éducation et réduction des inégalités.

À l'échelle locale, la base aérienne 118 de Mont-de-Marsan investit pour protéger l'environnement : les espaces extérieurs sont aménagés et entretenus de façon à favoriser la biodiversité, les structures sont adaptées aux déplacements en mobilités douces et des services de transports en commun circulent dans la base. Le mess (cantine militaire) dispose d'une politique anti-gaspillage et favorise les produits locaux. Ainsi, pendant mes journées de stage, je laissais ma voiture sur le parking non goudronné de l'ESIOC et je prenais le bus pour rejoindre le mess où je pouvais profiter de yaourts produits dans la région.

Il convient cependant de remarquer que sans le contexte militaire, la sécurité et la fiabilité sont les principaux objectifs. Le développement durable n'est donc pas prioritaires dans l'AAE.

A.2 Planning



(a) Planification prévisionnelle du stage

(b) Déroulement du stage

FIGURE A.1 – Diagrammes de Gantt utilisés pour la plannification et le suivi du stage

Annexe B

Annexes du Chapitre 1

B.1 Plaquette de présentation de l'ESIOC

FIGURE B.1 – Plaquette de présentation de l'ESIOC (recto)

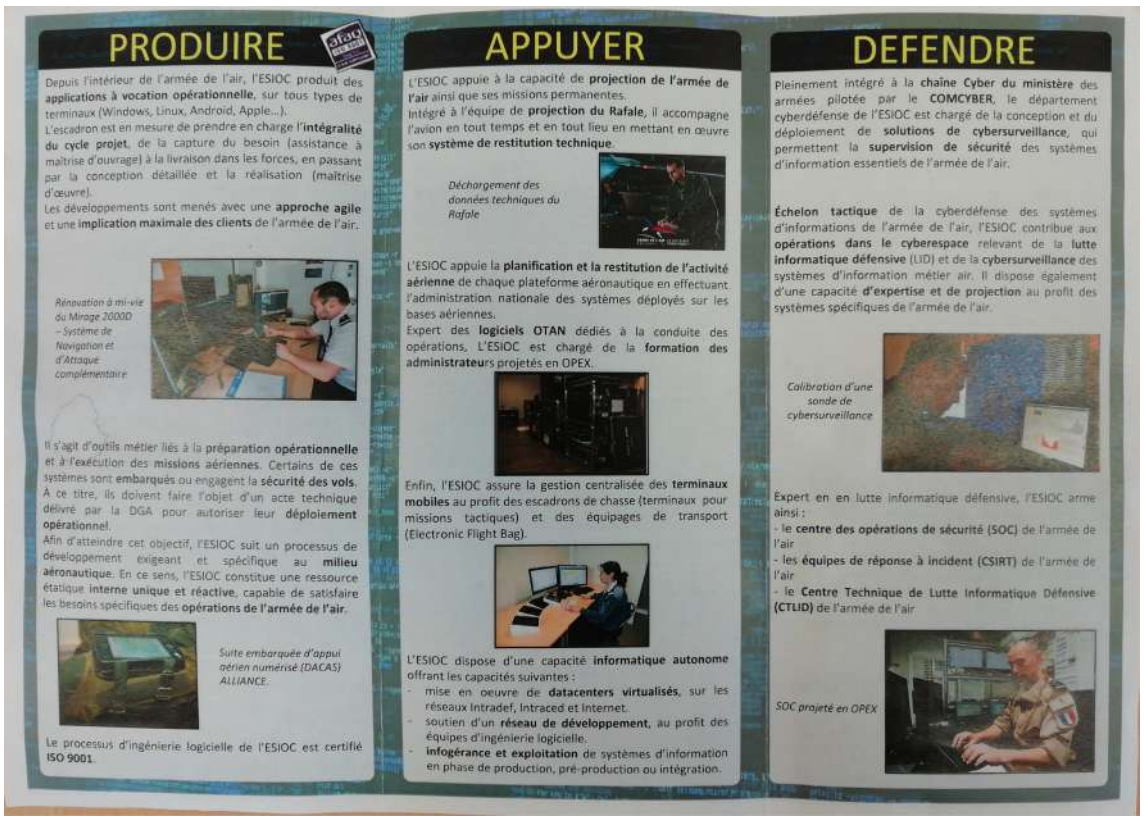


FIGURE B.2 – Plaquette de présentation de l'ESIOC (verso)

Annexe C

Annexes du Chapitre 4

C.1 Exemples de documents

DIFFUSION RESTREINTE

MINISTÈRE
DES ARMÉES
Liberté
Égalité
Fraternité

Marine nationale
CECMED
Niveau échelon de division ou formation

Toulon, le (date)
N°XX/CECMED/XXX/XXX/DR

ARRÊTÉ
(portant ou fixant) (objet de l'arrêté sans sigle ou abréviation)

ANNEXE : exemple d'annexe.
T. ABROGÉ : exemple de texte à abroger.

La ministre des Armées,

Vu le/la (exemple de référence n° XXX/TIMBRE/TIMBRE du 01 mars 1999) ;

Vu le/la (exemple de référence n° XXX/TIMBRE/TIMBRE du 22 mai 2001).

Arrête :

Article 1er - (**titre le cas échéant**)

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna.

Article 2 - (**titre le cas échéant**)

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna.

Article 3 - (**titre le cas échéant**)

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, elen sila lùmenn' omentielvo eros quis urna.

BCRM TOULON – CECMED/Division ou bureau
BP 900 – 83800 Toulon Cedex 9
courriel professionnel ou fonctionnel
Dossier suivi par : grade Nom

FIGURE C.1 – Arrêté DR factice

MINISTÈRE
DES ARMÉES
Liberté
Égalité
Fraternité

Marine nationale
CECMED
Niveau échelon de division ou formation

Toulon, le (date)

(formule de politesse manuscrite)

Corps de texte pour un préambule rédigé à la forme personnelle.

Suspendisse dui purus, scelerisque at, vulputate vitae, pretium mattis, nunc. Mauris eget neque at sem venenatis eleifend. Ut nonummy.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, elen sila lùmenn' omentielvo magna eros quis urna.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin pharetra nonummy pede. Mauris et orci.

Aenean nec lorem. In porttitor. Donec laoreet nonummy augue.

Aenean nec lorem. In porttitor. Donec laoreet nonummy augue.

Suspendisse dui purus, scelerisque at, vulputate vitae, pretium mattis, nunc. Mauris eget neque at sem venenatis eleifend. Ut nonummy.

(formule de politesse manuscrite type « je vous prie de croire… »)

Le (grade Prénom Nom)
(fonction),

DESTINATAIRE :
- [adresse postale du destinataire avec la formule « Monsieur/Madame (fonction) »]

BCRM TOULON – CECMED/Division ou bureau
BP 900 – 83800 Toulon Cedex 9
courriel professionnel ou fonctionnel
Dossier suivi par : grade Nom

FIGURE C.2 – Lettre DR factice

FIGURE C.3 – Formation DR factice

SEMAINE DU 25 AU 31 MARS 2024

	Lundi	Mardi	Mercredi	Jeudi	Vendredi	Samedi	Dimanche
Menu du jour	Paleron de bœuf Bolognaise	Pavé de colin Côte de porc	Cordon bleu de volaille Émincés de bœuf ananas/poivrons	Sauté de canard à l'orange Pavé de saumon à la crème	Sauté de porc à l'ananas Steak de bœuf	Filet de poulet basquaise	Osso bucco de veau
	Penne rigate bio Duo de carottes	Frites Brocolis bio	Purée aux 3 légumes Ratatouille	Pdt vapeur Duo de courgettes	Riz Poêlée brocolis/ champignons	Blé Piperade de légumes	Pdt grenaille Haricots verts
Grillades							
Végétarien	Tortellinis ricotta épinards	Tarte aux courgettes/ tomates/chèvre	Œufs brouillés bio	Lasagnes aux légumes du sud	Cappelletis aux 4 fromages		
Pizza		Crousti fromages		Margherita			
Dîner	Brandade de morue Salade verte	Pave de dinde Tandoori Nouilles chinoises Brunoise de légumes	Ventrèche grillée Pdt Duchesse Tomates à la provençale	Boulettes de bœuf sauce tomate Semoule bio Gratin de chou-fleur	Pavé de poisson à la provençale Petit épeautre Julienne de légumes	Steak haché façon bouchère Petits pois / carottes	Travers de porc sauce barbecue Tagliatelles Poêlée Tex-Mex

FIGURE C.4 – Menu de cantine factice

Annexe D

Journal de Bord

Semaine du 01/07/2024
— Parcours d'accueil à l'ESIOC
— Présentation du projet OMEGA et attribution d'une mission
— Réunion en présentiel avec les représentants Wavestone
— Journée de l'aviateur : cérémonie, présentation des équipes de développement des logiciels embarqués dans le Mirage 2000, démonstration de maître-chien
— Réception de l'ordinateur et accès à la session
— Début de ma formation sur Dataiku DSS : parcours d'apprentissage *Core Designer* et *Developer*
Semaine du 08/07/2024
— Poursuite de ma formation Dataiku DSS : parcours d'apprentissage *Developer* et *MLOps practitioner*
— Début du développement de l'API : prise en main librairies Python dataiku et dataikuapi, création d'une ébauche des premiers endpoints , découverte des *test queries*, debugging et configurations des endpoints
— Début de la documentation : écriture des premiers articles du Wiki pour les endpoints
— Réunion avec le conseiller Dataiku : questions sur les bonnes pratiques du développement d'API Python dans Dataiku DSS
Semaine du 15/07/2024
— Poursuite de ma formation Dataiku DSS : certifications *Core Designer*, *Advanced Designer*, *Developer* et *ML Practitioner*
— Réunion avec Auriane et Jonathan : réorganisation des endpoints afin d'améliorer la cohérence avec le front end
— Poursuite du développement de l'API : création, modification ou suppression des endpoints selon les décisions prises lors des réunions avec les autres équipes
— Poursuite de la documentation : écriture des articles du Wiki pour le projet OMEGA (présentation du projet, configurations, variables, dashboards...)
— Réunion avec le conseiller Dataiku : questions sur les bonnes pratiques du développement d'API Python dans Dataiku DSS
— Prise en main du logiciel Postman
Semaine du 22/07/2024
— Poursuite de ma formation Dataiku DSS : parcours d'apprentissage *ML Basics* et *Advanced Designer* ; certification *MLOps Practitioner*
— Poursuite du développement de l'API : premiers essais de déploiement de l'API, mise à jour des endpoints (vérification des données REO, SITU et IDF)
— Poursuite de la documentation : création de schémas, mise à jour du dashboard principal, mise à jour des articles du Wiki
— Réunion avec le conseiller Dataiku : questions sur les *credentials* et la gestion de PAM simultanés
Semaine du 29/07/2024
— Poursuite de ma formation Dataiku DSS : créations de webapps pour le projet OMEGA, création d'un petit projet d'apprentissage machine sur des images
— Poursuite du développement de l'API : mise à jour des endpoints (modifications suite à la réunion avec Wavestone et DRHAAE), déploiements, tests et corrections des endpoints en utilisant Postman et des requêtes correspondant aux cas d'usages (création de requêtes à partir des fichiers de test)
— Poursuite de la documentation : mise à jour des articles du Wiki
— Réunion avec Pierre-Jordi : présentation et prise en main du projet de reconnaissance de timbres
Semaine du 05/08/2024

— Création de requêtes test (différents formats et tailles pour tester les performances du système)
— Poursuite du développement de l'API : ajout de la gestion de PAM simultanés, créations de endpoints tests qui forcent le téléversement des fichiers dans le projet (en supprimant les lignes erronnées) et débuggage de la publication de l'API
— Poursuite de la documentation : mise à jour des articles du Wiki
— Test de la machine virtuelle pour le projet de reconnaissance de timbres

Semaine du 12/08/2024

— Poursuite du développement de l'API : premiers tests de communication avec Jonathan (téléversement du fichier REO)
— Rétro-engineeering pour retrouver les données contenues dans le REO utiles pour le PAM
— Création de requêtes test (réduction des données contenues dans le fichier REO)
— Poursuite de la documentation : mise à jour des articles du Wiki
— Relecture et documentation de code Python pour le projet de reconnaissance de timbres

Semaine du 19/08/2024

— Poursuite du développement de l'API : étude de faisabilité pour l'utilisation d'un serveur de fichiers, modification de l'API pour utiliser un serveur de fichiers (MinIO)
— Poursuite de la documentation : mise à jour des articles du Wiki
— Relecture et documentation de code Python pour le projet de reconnaissance de timbres
— Ecriture du rapport de stage

Semaine du 26/08/2024

— Poursuite du développement de l'API : modification de l'API pour utiliser un serveur de fichiers (MinIO), tests de communication avec Jonathan (téléversement du fichier REO), modifications de l'API suite aux tests
— Poursuite de la documentation : mise à jour des articles du Wiki
— Etude pour la création de nouveaux fichiers d'entraînement afin d'équilibrer les classes (demander à un expert métier comment doivent être placés les timbres, exemple : Diffusion Restreinte / Confidentiel Personnel, aucun exemple au format numérique)
— Ecriture et validation du rapport de stage
— Rencontre avec la Direction de la Maintenance Aéronautique (DMAé) et échanges sur l'utilisation de Dataiku DSS au sein de l'AAE

Résumé - Abstract

Développement d'une API pour une application web de datascience

L'Armée de l'Air et de l'Espace (AAE) produit des données variées dont le volume est en constante augmentation. Equipé de matériel de pointe comme la plateforme collaborative de science des données Dataiku Data Science Studio (DSS), l'Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence (ESIOC) est responsable de l'intégration, du stockage et de la mise à disposition des données mais aussi des traitements de données massives et du développement d'agents intelligents.

J'ai participé au développement d'un logiciel d'aide à la décision pour la mutation des sous-officiers utilisant les données des SI de la Direction des Ressources Humaines de l'Armée de l'Air et de l'Espace (DRHAAE). Affecté au sein de l'équipe responsable du développement de la partie calcul de cette application web multi-couche, j'ai développé une interface logicielle assurant la communication entre la partie calcul implémentée dans Dataiku DSS et les autres couches logicielles (interface utilisateur et bases de données).

J'ai aussi apporté des améliorations à un projet de classification de timbres de confidentialité mettant à l'épreuve les capacités de Dataiku DSS pour les tâches d'apprentissage machine et de vision par oridnateur.

Tout au long du stage, j'ai dû respecter des consignes de sécurité strictes, propres à l'AAE, réglementant l'accès au matériel, le développement informatique et la gestion des données. Afin de satisfaire les critères de la certification ISO9001 de l'ESIOC, j'ai porté une attention particulière à la qualité du code et au travail de documentation (documentation du code, création d'un wiki et de tutoriels).

Ce stage m'a permis d'acquérir une maîtrise du logiciel Dataiku DSS. Le contexte de développement nécessitait des capacités de rétro-ingénierie et de compréhension de code. J'ai mis en pratique de nombreuses compétences acquises en école d'ingénieur, en particulier dans les domaines du développement d'application web, du génie logiciel et de la programmation Python.

Development of an API for a data science web application

The Armée de l'Air et de l'Espace (AAE) produces an ever-increasing volume and variety of data. Equipped with leading-edge equipment such as the Dataiku collaborative data science platform, the Escadron des Systèmes d'Information Opérationnels et de Cyberdéfence (ESIOC) is responsible for integrating, storing and making data available, as well as for processing massive data and developing intelligent agents.

I took part in the development of a decision support software for the transfer of non-commissioned officers using data from the information systems of the AAE's human resources department. Assigned to the team responsible for developing the back end of this multi-layer web application, I developed a software interface ensuring communication between the back end part implemented in Dataiku DSS and the other software layers (user interface and databases).

I also made improvements to a project to classify confidentiality stamps, testing Dataiku DSS's capabilities for machine learning and computer vision tasks.

Throughout my internship, I had to comply with strict security instructions, specific to the AAE, governing access to equipment, IT development and data management. In order to meet ESIOC's ISO9001 certification criteria, I paid particular attention to the quality of the code and the documentation work (code documentation, creation of a wiki and tutorials).

This internship enabled me to gain a thorough understanding of the Dataiku DSS software. The development context required reverse engineering and code understanding skills. I put into practice many of the skills I had acquired at engineering school, particularly in the areas of web application development, software engineering and Python programming.