Table des matières

1 Remerciements

J'ai eu l'occasion de rencontrer et d'interagir avec de nombreuses personnes qui m'ont soutenu de différentes manières durant mes trois mois de stage, et je souhaite exprimer ma gratitude envers elles.

Tout d'abord, je tiens à remercier Guillaume Leboucher, directeur général d'OpenValue et tuteur de stage, pour sa confiance et l'opportunité qu'il m'a offerte de travailler chez OpenValue.

Je tiens également à exprimer ma chaleureuse gratitude envers François Valadier, directeur de l'innovation d'OpenValue, et David Gayou, lead machine learning engineer, pour leur accompagnement et le partage précieux de leurs connaissances.

Mes remerciements s'étendent également à l'équipe IA générative avec laquelle j'ai collaboré tout au long de mon stage : Grégoire Leboucher, Arthur Perigot, Severin Lefébure, Hédi Chaker, Sophiane Barchiche et Anne Herriau.

Enfin, je souhaite adresser mes remerciements à tous les membres d'OpenValue pour leur accueil chaleureux et leur contribution essentielle à la réussite de mon stage.

2 Introduction

2.1 OpenValue

Fondée en 2015, Openvalue a intégré au cours du premier semestre 2021 la Branche Grand Public et Numérique du Groupe La Poste [9]. Cette entreprise se focalise sur le développement de solutions liées au Big Data et à l'intelligence artificielle (IA). De plus, elle engage des réflexions stratégiques et organisationnelles dans ces domaines. Openvalue réunit une communauté d'experts qui élaborent les outils analytiques et prédictifs de demain. Depuis sa création, OpenValue guide ses clients dans la création et la valorisation d'actifs technologiques tels que le machine learning, le deep learning, le traitement automatique du langage naturel et la vision par ordinateur. Reconnue pour sa culture axée sur l'innovation, la mission d'Openvalue est de permettre aux entreprises de réaliser leur plein potentiel grâce aux technologies du Big Data et de l'intelligence artificielle.

2.2 Le sujet de stage

Suite à la récente explosion des IA génératives telles que ChatGPT, DALL-E, Mid-Journey ou encore Llama, l'entreprise OpenValue souhaite rester compétitive. Ainsi, pour explorer ces nouvelles technologies, une équipe dédiée à l'IA générative a été formée en avril 2023. Cette équipe est chargée de développer des cas d'usage orientés métier afin de répondre aux problématiques des métiers présents chez Docapost et chez leurs clients. C'est dans ce contexte que s'inscrit mon stage. Mon objectif était de produire des preuves de concept pour les différents cas d'usage proposés par les cadres de Docapost.

2.3 L'équipe

L'équipe IA générative dans laquelle j'ai pris place était composée de quatres stagiaires chargés du développement des cas d'usages : Grégoire Leboucher, Arthur Perigot, Severin Lefébure et moi-même. Sophiane Barchiche et Hédi Chaker occupaient les rôles de managers au sein de cette équipe. De plus, Guillaume Leboucher et Anne Herriau se chargeaient de suivre nos avancées et de superviser l'orientation de nos recherches.

3 Formation au prompt engineering

Notre première mission au sein de l'équipe IA générative a été de se fromer aux technologies que nous allions utiliser, c'est à dire :

- Python : le language de programation ;
- Langchain : un framework python permettant de communiquer avec les API de nombreuses IA génératives ;
- Streamlit : un framework python permettant de créer facilement des applications web ;
- Le prompt engineering : indispensable pour requeter les IA génératives ;

3.1 Qu'est-ce que le prompt engineering et pourquoi l'apprendre

Le "prompt engineering", désigne le processus de création et d'optimisation de formulations ou d'instructions, appelées "prompts", pour interagir avec les modèles de langage et d'intelligence artificielle, plus souvent appelé "Large Language Model" (LLM). Les LLM sont type de modèle d'IA qui traite et génère donc du texte en utilisant des réseaux de neurones profonds, souvent avec des capacités avancées de compréhension et de production de langage humain. L'objectif principal du "prompt engineering" est d'obtenir des résultats plus précis et pertinents lors de l'utilisation de ces modèles, en fournissant des indications spécifiques pour guider la génération de texte ou les réponses. Ce processus revêt une importance significative dans le développement de solutions exploitant les capacités des LLM.

Au sein de l'équipe d'IA générative, notre mission était d'exploiter les nouvelles possibilités offertes par ces IA pour résoudre des problématiques spécifiques aux activités de l'entreprise Docapost et aux clients d'OpenValue. Travailler avec ces nouvelles IA impliquait de concevoir des prompts précis afin d'obtenir les résultats désirés.

Une fois ces compétences acquises, j'ai pu trvailler sur mon premier projet au sein d'OpenValue : Créer et dispenser une formation au prompt engineering aux employés et cadres de Docapost.

3.2 Objectifs de la formation

L'objectif de la formation que j'ai conçue et dispensée était de familiariser l'ensemble des employés et cadres de Docapost avec les nouvelles IA génératives afin de :

- Maintenir les membres de l'entreprise informés des dernières technologies émergentes sur le marché ;

- Permettre à tous les membres de l'entreprise de proposer de nouveaux cas d'usage ;
- Clarifier les limites actuelles des IA génératives et mettre en évidence les risques potentiels de leur utilisation ;
- Mettre en avant la première application web que j'ai développée en collaboration avec l'équipe IA générative, afin qu'elle puisse être utilisée par tous.

3.3 Contenu de la formation

C'est une formation d'une durée de deux heures dont voici le plan :

- Comment fonctionne GPT3 et chatGPT ;
- Qu'est-ce que le promt engineering ;
- Vos idées ;
- Comment faire un bon prompt ;
  - Les éléments d'un bon prompt ;
  - Quelques phrases utiles ;
  - Le Few-Shot Learning ;
  - Le Chain of thought ;
  - Quelques astuces supplémentaires ;
  - Comment gérer la température ;
  - Quelques paramètres à connaître ;
  - Les agents
  - Reconnaître une création d'IA
  - Le prompt Hacking et Jailbreaking
- Conclusion ;
- Questions.

La formation s'est déroulée comme suit : Après une introduction au fonctionnement de GPT-3 et ChatGPT, ainsi qu'au concept de prompt engineering, j'ai recueilli les idées des participants à la formation. Ces idées ont été utilisées en direct pour poser des questions à ChatGPT. Ensuite, j'ai approfondi la méthodologie pour créer des prompts efficaces, en expliquant de manière détaillée plusieurs concepts complexes liés au fonctionnement précis des IA génératives textuelles. Après ces explications, nous avons revisité des idées receuillis et suggéré des améliorations pour leurs prompts, afin d'obtenir des résultats encore plus pertinents. Enfin, la formation s'est conclue par une séance de questions-réponses avec les participants.

Pendant cette formation, les exemples ont été abordés à travers l'utilisation du playground mis à disposition par Azure. En passant par ce playground les données sont envoyé à des serveurs situés en Europe de l'Ouest. L'accent a également été mis sur la nécessité pour les membres de Docapost de ne pas partager de données sensibles sur l'instance de chatGPT d'OpenAI dont les serveurs sont aux États-Unis. En effet le groupe La Poste met aujourd'hui tout en oeuvre pour garantir la souveraineté des données. Pour renforcer ce point, nous avons profité de cette formation pour concevoir une application web permettant aux utilisateurs de télévérser leurs PDF et de poser des questions liées à ces documents à un chatbot.

3.4 L'application web développée pour la formation

Afin de décrire l'architecture de l'application web que j'ai développée, il est nécessaire de clarifier quelques concepts. Pour obtenir des réponses d'une IA, j'ai utilisé

le framework Python Langchain. Ce choix a été motivé par le fait que Langchain exploite les technologies les plus récentes. Il bénéficie de mises à jour régulières et propose une documentation exhaustive. De plus, il se révèle très flexible, permettant une variété de requêtes avec des paramètres adaptables. Ainsi, il est possible de solliciter diverses IA, qu'elles soient développées par différentes entreprises ou en open source. Dans cette application, nous avons exclusivement utilisé ChatGPT 3.5 pour les requêtes. Néanmoins, en vue du développement des cas d'usages, j'ai envisagé d'intégrer plusieurs IA, ce qui exigeait une approche générique. Langchain offre une méthode générique pour requêter la multitude d'API disponibles dans ce framework. Un même appel fonctionne pour interagir avec ChatGPT et peut être utilisé pour appeler l'IA de Meta : Llama en ajustant simplement le nom du modèle d'IA sollicité. Cette approche permet ainsi de maintenir une flexibilité générique.

Voici les quelques fonctionnalités du framework Langchain utilisées dans cette application web :

- Mise en place d'un prompt système : il s'agit d'une directive puissante transmise en tant que paramètre à chaque appel de l'API. Cette approche permet de spécifier clairement le comportement souhaité pour le bot créé.
- Mise en place d'un historique : cette fonctionnalité permet de fournir plusieurs messages en tant que paramètre lors d'un appel. De cette manière, les k derniers messages sont pris en compte dans la réponse générée par l'IA.
- Mise en place d'une chaîne : cette fonctionnalité permet de connecter les appels, ce qui permet de passer plusieurs nouveaux arguments à l'appel de l'IA tout en leur attribuant une priorité.

Afin de spécialiser le bot dans la fourniture de réponses en fonction d'un document, voici la méthodologie que j'ai suivie :

- Téléchargement d'un document ;
- Division du document en plusieurs sections de taille prédéfinie ;
- Calcul de l'embedding pour chaque section ;
- Calcul de l'embedding pour la question posée par l'utilisateur ;
- Comparaison de l'embedding de la question avec les embeddings des sections ;
- Sélection des k sections dont les embeddings sont les plus proches de celui de la question ;
- Appel à l'API en transmettant les k sections précédemment déterminées ainsi que la question de l'utilisateur en tant que contexte, puis récupération de la réponse de l'API.

Une section de document est appelée "chunk". Pour chaque chunk, un recouvrement appelé "overlap" est appliqué afin de garantir une représentation chronologique précise du texte, si nécessaire.

L'architecture de l'application est la suivante et est présentée en figure 1 de l'annexe :

- config.yml : fichier définissant toutes les constantes de l'application
  - IA générative utilisée ;
  - Modèle de cette IA utilisée s'il y en a plusieurs ;
  - Nom du bot ;
  - Nombre de chunks utilisés pour répondre à la question ;
  - Taille de l'historique ;
  - Prompt système.

— Vault.py : fichier contenant les clés d'accès à API d'Azure, cachées par un secret.
— Ui.py : fichier définissant les éléments de style global ;
— theme.css : fichier style pour l'application web appelée dans la fonction définis dans Ui.py ;
— PDFLoader : classe permettant de nettoyer un string de tout bruit potentiel et permettant de convertir un PDF en document exploitable par nos requêtes API, avec de l'OCR si les PDF ne sont pas numériques ;
— PDFbot.py : classe principale mettant en place l'application web et dans laquelle est déféni le requêtage aux API.

Un visuel de l'application est disponible en figure 4 de l'annexe.

3.5 Résultats de la formation

La formation s'est déroulée à distance via Microsoft Teams et a rassemblé plus de 500 participants. Par la suite, cette formation a été utilisée par François Delavier lors de la réunion annuelle du comité d'exécution de Docapost, ainsi qu'au sein du comité d'exécution élargi, afin de former les instances supérieures de l'entreprise à ces nouvelles technologies.

Un visuel de l'application web en utilisation est disponible en figure 4 de l'annexe.

Suite à cette présentaion, j'étais donc prêt à démarrer mon travail sur les cas d'usages.

4 Développement des cas d'usages

4.1 Spécification des cas d'usages proposés

Cinq cas d'usages ont été proposés à l'équipe IA générative :
— Créer des fichiers structurés JSON à partir de données textuelles ;
— Créer un assistant médical ;
— Créer un outil de réponse automatique aux appels d'offres ;
— Créer un outil d'aide aux audits concernant le numérique responsable ;

L'équipe étant composée de quatre stagiaires, nous avons réparti entre nous ces quatre cas d'usage. J'ai choisi de travailler sur le premier cas d'usage : la création de fichiers structurés XML ou JSON à partir de données textuelles.

Ce cas d'usage a été initié par les services juridiques de Docapost dans le but de rationaliser la gestion des contrats. En effet, en tant qu'entreprise, Docapost hérite de nombreux modèles de contrats provenant de différentes entreprises, ce qui présente des défis. Du fait de la variété des formats de ces documents, il n'est pas possible d'automatiser facilement l'extraction d'informations. Néanmoins, ces contrats contiennent généralement des informations similaires en fonction du type traité. Le traitement manuel de ces contrats s'avère coûteux, justifiant ainsi l'intérêt pour un outil permettant d'extraire les informations pertinentes d'un contrat et de les structurer en vue de les rendre utilisables dans un tableau de bord interactif dynamique. Cet outil serait utile aussi bien pour Docapost que pour ses clients.

Pour répondre à ce use case, j'ai mené une étude approfondie des spécifications de ce cas d'usage. Ces spécifications sont présentées dans le tableau ci-dessous 1.

Table 1 – Spécification des cas d'usages

Fonctionnalités		
N°	**Intitulés**	**Critères**
Créer des fichiers structurés à partir de données textuelles		
N°1	Extraire l'information	L'outil développé doit être capable d'extraire l'information d'un document pdf.
N°2	Structurer l'information	L'outil doit être capable de structuré l'information extraite en fonction de critères donnés.
N°3	Télécharger l'information	L'utilisateur doit être en capacité de télécharger le fichier structuré produit.
N°4	Questionner les documents	L'utilisateur doit être en capacité de poser des questions au bot et d'obtenir une réponse croisant potentiellement les sources si plusieurs documents sont chargés.
N°5	Sources des réponses	L'utilisateur doit avoir accès aux sources des documents pour une réponse donnée.
N°6	Simplicité d'usage	L'application doit être utilisable par des utilisateurs non développeurs.

4.2 Architecture de l'application web

Afin de simplifier le développement de la preuve de concept, ce cas d'usage est séparé en deux applications web :

- structureBot : permettant de structurer l'information d'un document, il traite les documents un par un (répondant ainsi aux sprécification 1, 2, 3, 5 et 6) ;
- knowledgeBot : permettant de poser des questions à une base d'un ou plusieurs documents structurés (répondant ainsi aux sprécification 4, 5 et 6).

L'application structureBot développée reprend la même architecture que l'architecture présentée en partie 3.4 et en figure 1. Cependant les classes ont été étoffées. Les fichiers confg.yml, vault.oy, ui.py et theme.css ne voient que leur constante rédéfinis ou ont quelques paramètres ajustés. PDFLoader.py et structureBot.y diffèrent légèrement plus :

- PDFLoader.py : j'ajoute une fonction savePDF afin de pouvoir enregistrer les pdf des documents et les afficher à l'utilisateur dans l'application ;
- strucutureBot : j'ajoute une fonction downloadFile permettant à un utilisateur de télécharger un fichier, une fonction getAnswerAndStructure permettant de précharger des prompt type pour un type de document donné afin d'avoir la réponse structuée sans passer pas le chatBot, ainsi que plusieurs modifications du streamlit pour une éxpérience utilisateur améliorée.

Cette nouvelle architecture est présentée en figure 2 de l'annexe. Des visuels de l'application sont présenté en figure 5, 6 et 7 de l'annexe.

L'application knoledgeBot développée reprend également la même architecture mais quelques classes ont été modifiées. Les fichiers confg.yml, vault.oy, ui.py et theme.css ne voient que leurs constantes rédéfinies ou ont quelques paramètres ajustés. PDFLoader.py et structureBot.y diffèrent légèrement plus :

- PDFLoader.py : la foction getDocumentFromPDF est remplacé par vectori-

zer, cette fonction permet de traiter les documents PDF ainsi que les documents structurés (txt, json et xml) ;
— nkowledgeBot : modification du streamlit pour une expérience utilisateur améliorée.

Cette nouvelle architecture est présentée en figure 3 de l'annexe. Des visuels de l'application sont présentés en figure 8 et 9 de l'annexe.

4.3 Comparaison des différentes IA génératives textuelles

Après avoir réalisé nos premiers essais avec ChatGPT, disponible grâce à la licence Microsoft Azure de l'entreprise, notre mission consistait à tester plusieurs autres IA afin de comparer leurs performances. Pour sélectionner les IA à tester, je me suis basé sur le classement proposé par Hugging Face [4]. Hugging Face est une plateforme open-source spécialisée dans le traitement du langage naturel (NLP), offrant une interface de programmation (API) pour accéder à une variété de modèles pré-entraînés. Cette plateforme simplifie le processus d'expérimentation puisque les modèles sont déjà entraînés et prêts à l'emploi. En plus de l'API, Hugging Face propose des outils pour la gestion des données et des modèles, ainsi que pour la création et l'entraînement de modèles personnalisés.

Cette start-up franco-américaine fournit également un classement des IA génératives textuelles open source basé sur quatre benchmarks disponibles dans l'implémentation de Hailey Schoelkopl [5] sur GitHub. Le leaderboard évolue constamment. Au début de mon stage, l'IA Falcon, développée par l'Institut d'Innovation Technologique (TII) à Abu Dhabi et publiée sous licence Apache 2.0 [3], dominait le classement. Une information intéressante, bien que non publique, est que le TII a en réalité fait appel à l'entreprise française LightOn pour créer ce LLM. Nous avons ainsi établi un contact avec eux pour obtenir une licence du LLM Mini, une version dérivée de Falcon [6].

Parallèlement, nous avons également testé Claude2, le modèle d'Anthropic [2], ainsi que différentes variantes de Llama, le modèle de Meta [8]. Finalement, nous avons rapidement mis fin à ces deux derniers tests, car Claude2 était un modèle privé, difficile à évaluer sans l'achat d'une licence. De même, nos tests sur Llama ont été arrêtés rapidement, Llama2 ayant été publié et surpassant nettement les anciennes versions de LLM que nous avions testées [7].

Mes efforts se sont donc concentrés sur les trois modèles suivants : ChatGPT 3.5, Mini et Llama2. J'aurais souhaité inclure GPT-4 et Claude2 dans mes investigations, mais malheureusement, ces modèles étaient actuellement trop coûteux.

Malheureusement, il n'existe pas de classement global regroupant à la fois des IA open source et des IA issues d'entreprises privées sous licence commerciale. J'ai donc mis en place une petite échelle de mesure spécifique à mon cas d'usage. Cette évaluation était assez simple : j'ai utilisé treize contrats pour lesquels j'ai structuré les informations en utilisant chacune des IA retenues pour le test. Ensuite, mes collègues ont classé les résultats de chaque IA, du plus pertinent au moins pertinent. Un total de seize collègues ont participé à cette évaluation, et voici les résultats obtenus (voir Tableau 3).

Table 2 – Comparaison des différents LLM

Modèle	**nombre de 1ère position (%)**
ChatGPT 3.5	32%
Mini	11%
Llama2	57%

Cette métrique comporte plusieurs défauts :
- Petite taille de l'échantillon : seulement treize contrats ;
- Manque de diversité : seulement seize collègues ;
- Evaluation subjective : pas d'explication des choix, seize personnes de la même entreprise ;

J'ai donc enrichi cette métrique avec des données recensées sur Internet. Pour ce faire, je me suis basé sur Falcon pour comparer Llama2 et Mini, et j'ai consulté divers articles pour établir une comparaison entre Llama2 et ChatGPT [10].

Finalement, la métrique mise en place complétée par mes recherches confirmait la tendance suivante :
- Llama2 ;
- ChatGPT 3.5 ;
- Mini.

Llama2 est donc le candidat retenu pour tester plus en profondeur le cas d'usage.

4.4 Déploiement de Llama2

Pour utiliser Llama2, il était nécessaire de le déployer sur les serveurs Azure utilisés par l'entreprise. Ensuite, l'objectif était de configurer le serveur pour accueillir Llama2. Pour cela, j'ai suivi un tutoriel disponible sur YouTube [11]. Une attention particulière a été accordée à cette étape, car le modèle Llama2 que nous utilisions (celui avec 70 milliards de paramètres) est très volumineux et requiert une importante puissance de calcul, nécessitant au minimum 80 Go de VRAM [1]. Par conséquent, il a été nécessaire de déployer des serveurs Azure à $30 l'heure.

4.5 Résultats

Le cas d'usage relatif à la structuration de données textuelles avait été soumis par les services juridiques de Docapost. Cependant, c'est avec l'entreprise Oblige, cliente de Docapost, que ce cas d'usage a suscité le plus d'intérêt. En effet, lors de la visite de Stanislas de Pelichy, directeur commercial d'Oblige, dans nos locaux pour des discussions avec Docapost, j'ai remarqué que leur solution traitait ce cas d'usage, mais en utilisant des IA non génératives. À travers nos échanges, j'ai réussi à les persuader de l'intérêt de ma proposition, qui consistait à exploiter des IA génératives pour éliminer la nécessité de la phase d'entraînement.

En conséquence, j'ai pu rencontrer Edouard Chantrier, Directeur des Solutions chez Oblige. Nous avons convenu de mettre en place une métrique pour comparer les performances de leur modèle sur six contrats par rapport aux résultats obtenus par mon modèle sur les mêmes six contrats. La métrique était relativement simple : pour

chacun des six contrats, nous avions des résultats attendus en tant qu'étiquettes. Il s'agissait donc de fournir ces six contrats en paramètres aux solutions, puis de compter le nombre de résultats concordant avec les étiquettes. Il y avait au total 50 étiquettes. Nous avons ensuite calculé l'accuracy du modèle en utilisant la formule : $accuracy = \frac{True_{positive}+True_{negative}}{Total}$, soit le pourcentages d'élément bien étiquettés.

Oblige est une société dont l'expertise est de structurer l'information de document non strucutrée. L'objectif de cette comparaison était donc d'évaluer le potentiel de l'IA générative pour leur solution. Un cas interessant pour eux serait que l'IA générative soit capable de structuré l'information de document sur lesquelles ils n'ont pas d'exemple et ne peuvent pas faire apprendre leur modèle par exemple. En aucun cas l'objectif était de remplacer leur modèle par un modèle d'IA générative.

Les résultats obtenus sont présentés dans le tableau 3.

Table 3 – Comparaison solution Oblige et solution OpenValue

Solution	**Accuracy(%)**
Oblige	94%
OpenValue	88%

Avec la solution que j'ai développée, j'atteins donc un taux d'exactitude de 88%. Il est important de noter que la solution que j'ai élaborée présente encore certaines faiblesses, notamment lorsqu'il s'agit de contrats longs. En effet, sur les 50 étiquettes, je constate 6 erreurs, à savoir :

- Une erreur de compréhension ;
- 5 informations non trouvées dans le contrat long.

contre, côté Oblige :

- trois erreurs de compréhension ;

Ces résultats ont grandement suscité l'intérêt d'Oblige. L'IA générative est capable de structuré l'information sans apprentissage à effectuer de leur côté. Il est alors fortement probable que le cas d'usage soit intégré en production dès septembre, avec la perspective de compléter la solution déjà proposée par l'entreprise Oblige.

5 Conclusion

Le stage que j'ai eu la chance d'effectuer au sein de l'entreprise OpenValue a été une expérience extrêmement enrichissante, tant sur le plan personnel que professionnel.

Sur le plan professionnel, mes missions au sein de l'équipe IA générative m'ont offert une opportunité concrète de contribuer à l'activité de l'entreprise. J'ai pu découvrir le travail en équipe au sein d'une entreprise et relever le défi de créer de la valeur dans un projet d'innovation d'entreprise. J'ai dû me familiariser avec des technologies que je ne maîtrisais pas telle que le prompt engineering ou encore les framework Langchain et streamlit, et concevoir des applications novatrices à partir de ces technologies.

Sur le plan personnel, ce stage m'a permis de découvrir mon intérêt prononcé pour les domaines de l'intelligence artificielle et de la recherche de solutions innovantes. J'ai particulièrement apprécié ma mission sur la mise en place d'une formation au prompt engineering, au cours de laquelle j'ai pu exprimer davantage d'initiative et de créativité. Analyser une technologie, concevoir une nouvelle application, présenter la solution et convaincre un public de plus de 500 personnes étaient autant de défis que j'ai aimé. J'ai également beaucoup apprécié travailler en équipe sur des sujets très techniquement similaires mais ayant des clients et des objectifs différents aux miens. Cet environnement a clairement favorisé l'entraide, tout en nous laissant une grande autonomie quant à notre rythme et notre vision de la solution.

Un autre aspect de ce stage que j'ai particulièrement apprécié a été l'implication de mon tuteur de stage, Guillaume Leboucher. En effet, j'ai eu la chance d'assister à plusieurs réunions stratégiques et politiques au sein du groupe La Poste. J'ai également pu participer pendant trois jours au salon VivaTech, en plus d'assister à divers autres événements et réunions. Ce stage m'a offert l'opportunité d'élargir considérablement mon réseau professionnel et de développer une vision précise de la politique mise en place au sein des grands groupes.

En fin de compte, ces dix semaines passés à OpenValue m'ont permis de mieux cerner le poste vers lequel je souhaite m'orienter à l'avenir et de dessiner plus clairement mon plan de carrière.

6 Annexes

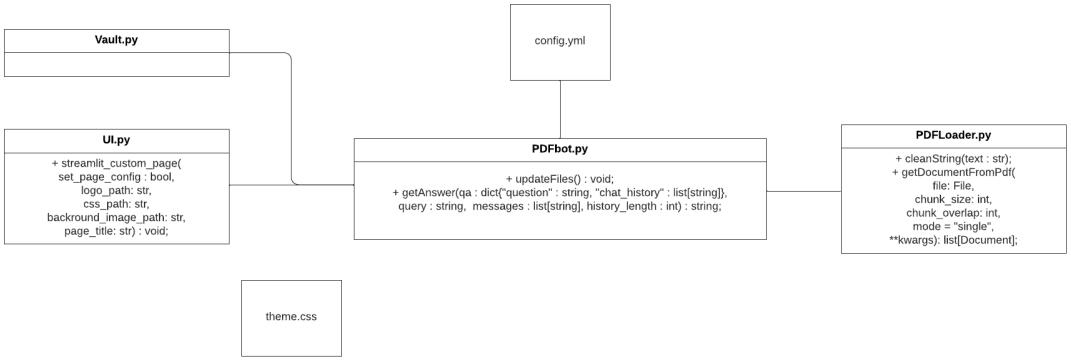

Figure 1 – Diagramme UML de l'application web pour la formation

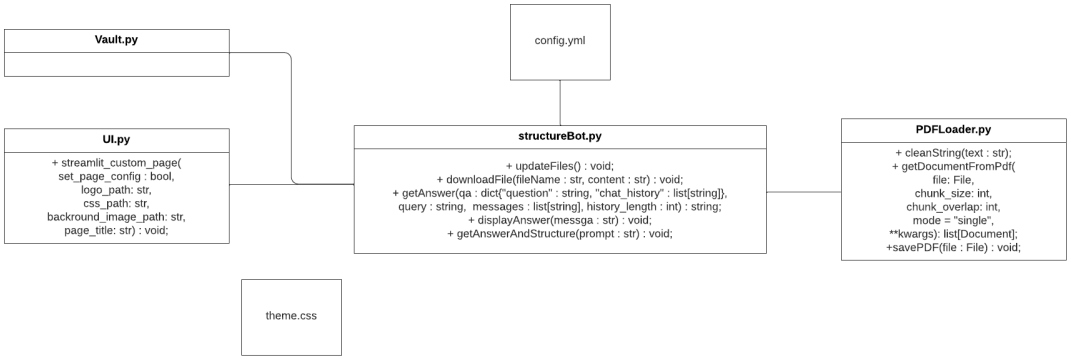

FIGURE 2 – Diagramme UML de l'application web pour structurer les fichiers

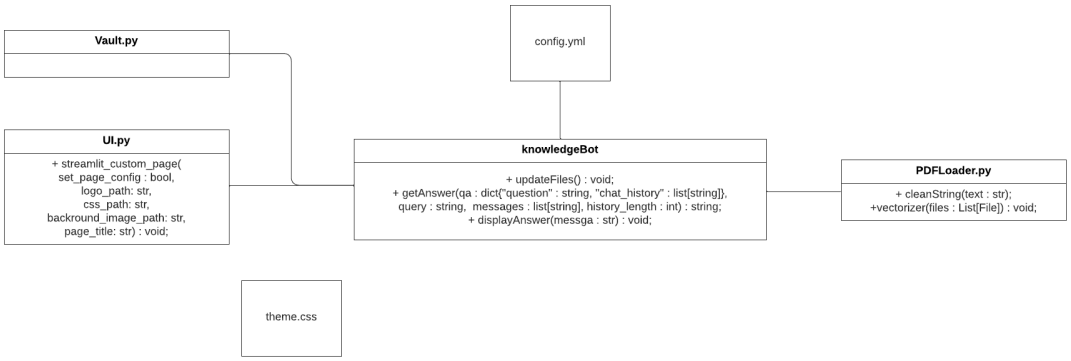

FIGURE 3 – Diagramme UML de l'application web pour questionner les documents structurés

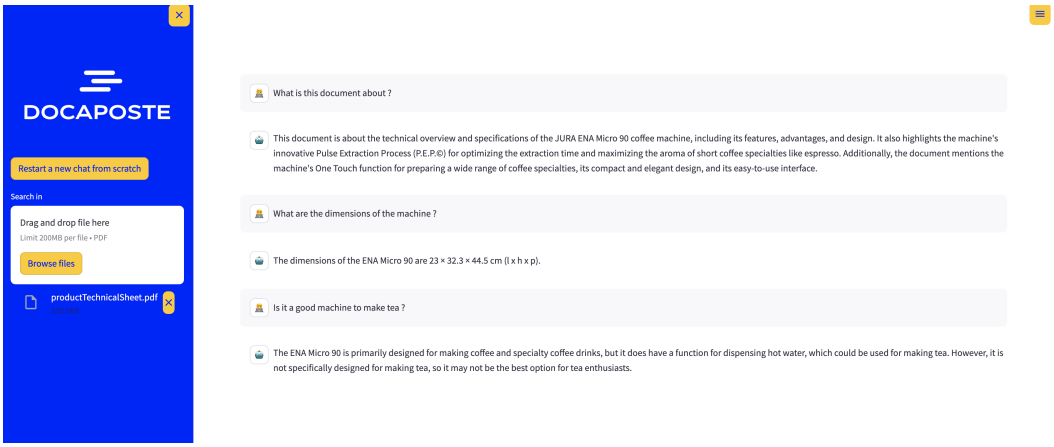

FIGURE 4 – Illustration de l'application web réalisée dans le cadre de la formation

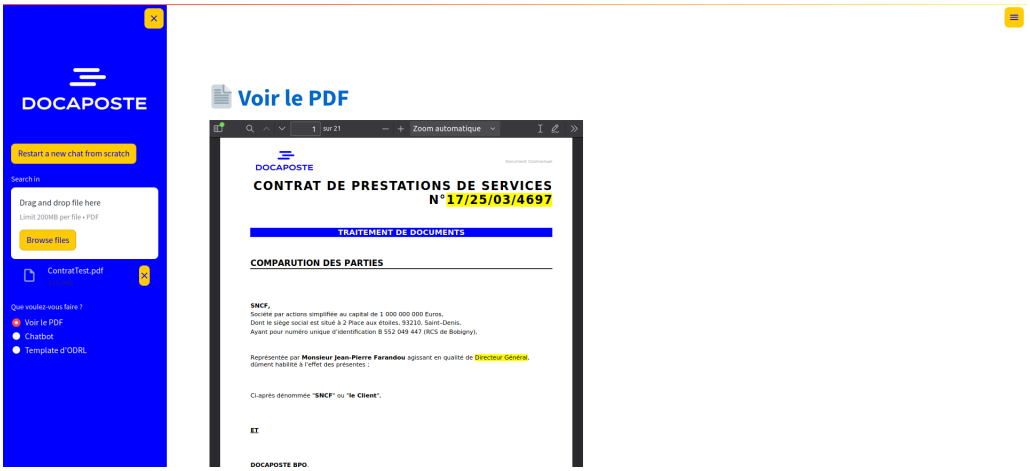

FIGURE 5 – Illustration de l'application web permettant de structurer les fichiers onglet 1

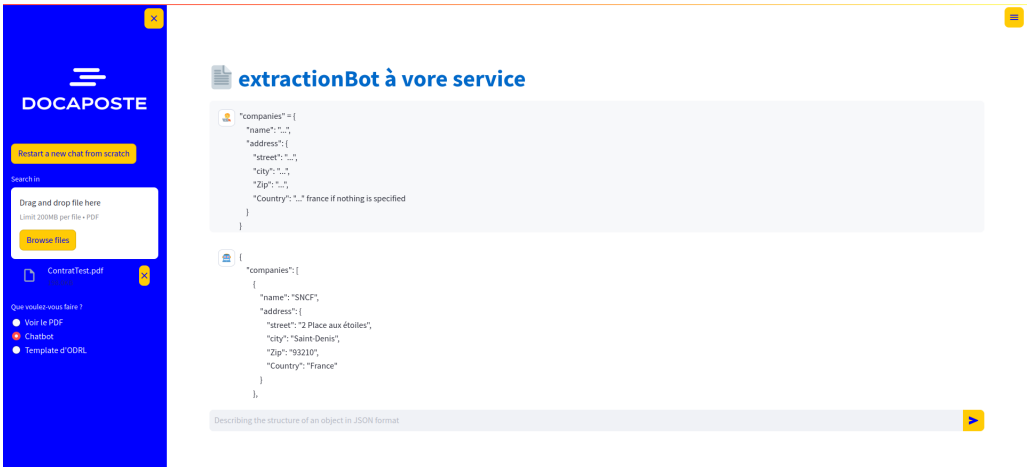

FIGURE 6 – Illustration de l'application web permettant de structurer les fichiers onglet 2

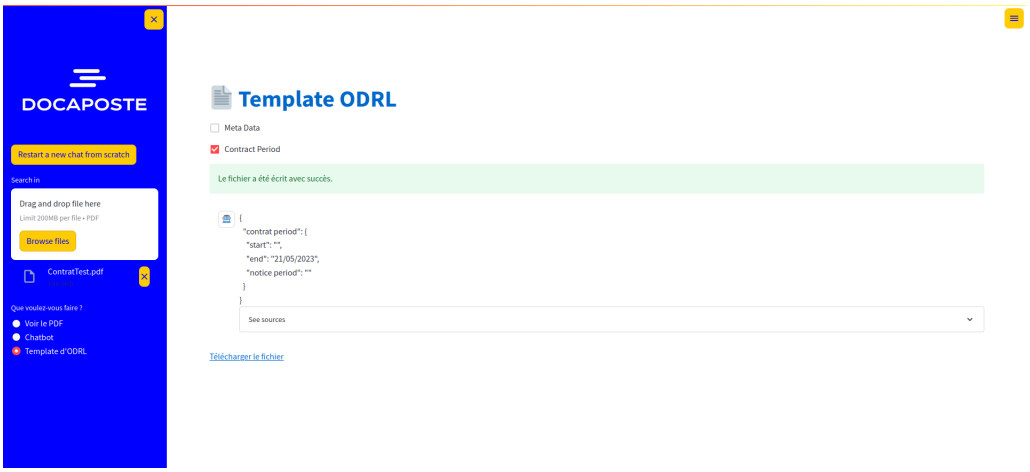

FIGURE 7 – Illustration de l'application web permettant de structurer les fichiers onglet 3

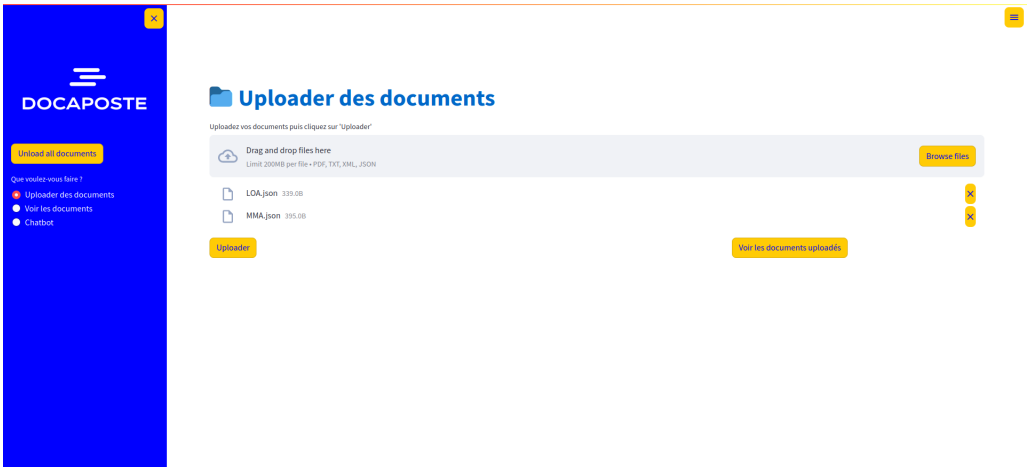

FIGURE 8 – Illustration de l'application web permettant de questionner les fichiers structurés onglet 1

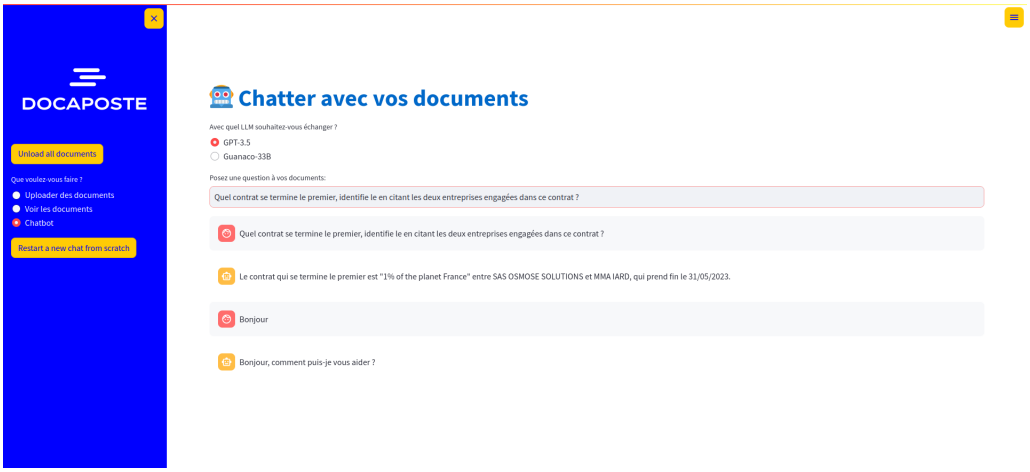

FIGURE 9 – Illustration de l'application web permettant de questionner les fichiers structurés onglet 2

TABLE 4 – Planning

Semaine	Contenu
1	Introduction au stage et prise en main des technologies
2	Anlyse des problématiques pour la formation au prompt engineering, commencement du projet
3	Formation au prompt engineering et analyse fonctionnelle des cas d'usages
4	Codage du cas d'usage et présence à vivatech
5	Présentation formation prompt engineering et Codage du cas d'usage
6	Codage cas d'usage et rendez-vous client
7	Analyse et comparaison des différentes IA et rendez-vous client
8	Déploiement Llama2 et codage cas d'usage
9	Analyse et comparaison solution Oblige vs solution OpenValue
10	Rendu et documentation

Références

[1] Easy With AI. *LLama 2*. URL : https://easywithai.com/resources/llama-2/.

[2] ANTHROPIC. *Claude 2*. URL : https://www.anthropic.com/index/claude-2.

[3] DECRYPT. *Falcon LLM : Le modèle LLM open source le plus puissant à ce jour*. URL : https://pandia.pro/actualite/falcon-llm-le-modele-llm-open-source-le-plus-puissant-a-ce-jour/.

[4] Hugging FACE. *Open LLM Leaderbird*. URL : https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.

[5] HAILEYSCHOELKOPL. *Eleuther AI Language Model Evaluation Harness*. URL : https://github.com/EleutherAI/lm-evaluation-harness.

[6] LIGHTON. *Le grand modèle de langage de LightOn de 40 milliards de paramètres : MINI*. URL : https://www.lighton.ai/fr/blog/blog-4/le-grand-modele-de-langage-de-lighton-de-40-milliards-de-parametres-mini-19.

[7] META. *Introducing Llama 2*. URL : https://ai.meta.com/llama/.

[8] META. *Introducing LLaMA : A foundational, 65-billion-parameter large language model*. URL : https://ai.meta.com/blog/large-language-model-llama-meta-ai/.

[9] OPENVALUE. *Notre Vocation*. URL : https://openvalue.co/.

[10] Ertugrul PORTAKAL. *Llama 2 vs ChatGPT d'OpenAI : Lequel est le meilleur ?* URL : https://textcortex.com/fr/post/llama-2-vs-chatgpt.

[11] WORLDOFAI. *How To Install Llama 2 Locally and On Cloud - 7B, 13B, 70B Models !* URL : https://www.youtube.com/watch?v=SbuhznykQBg&t=238s.