جـامـعـة ابـن طـفـيـل
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵉⴱⵏ ⵟⴼⴰⵢⵍ
Ibn Tofaïl University

FACULTE DES SCIENCES

UNIVERSITE IBN TOFAIL

FACULTE DES SCIENCES, KENITRA

MEMOIRE DE PROJET DE FIN D'ETUDES

MASTER INTELLIGENCE ARTIFICIELLE ET REALITE VIRTUELLE

I.A générative pour génération de présentations immersives

ÉTABLISSEMENT D'ACCUEIL : VERVE TECHNOLOGIES, CASABLANCA

ELABORE PAR : BOUKTITIYA HAMZA

ENCADRE PAR : MR ANASS NOURI (FSK UNIVERSITE IBN TOFAÏL)

MOHAMED BOUMENZEH (VERVE TECHNOLOGIES)

SOUTENU LE 24/09/2024, DEVANT LE JURY COMPOSE DE :

- MME TOUAHNI RAJA (FSK UNIVERSITE IBN TOFAÏL)
- MME EDDAROUICH SOUAD (CRMFE RABAT)
- MR MESSOUSSI ROCHDI (FSK UNIVERSITE IBN TOFAÏL)
- MR NOURI ANASS (FSK UNIVERSITE IBN TOFAÏL)
- MR IDRISS MOUMEN (FSK UNIVERSITE IBN TOFAÏL)

ANNEE UNIVERSITAIRE 2023/2024

Remerciement

C'est avec une sincère reconnaissance que je consacre cette section pour exprimer ma profonde gratitude envers toutes les personnes qui ont été d'un soutien indispensable pour la réussite de ce projet. En cette occasion spéciale, je tiens à exprimer ma gratitude envers la coordonnatrice du Master IARV, Madame RAJAA TOUAHNI, pour la qualité de la formation qu'elle m'a apportée, ainsi que pour son soutien et son accompagnement tout au long de mon parcours.

Mes sincères remerciements vont également à mon encadrant interne, M. Anass Nouri, pour ses précieuses orientations tout au long de ce travail. Je tiens également à remercier les membres du jury pour avoir accepté d'examiner et d'évaluer mon travail.

J'exprime également ma gratitude envers M. Mohamed Boumenzeh, mon encadrant externe, pour son accueil chaleureux, son encadrement efficace et son accompagnement tout au long de cette expérience professionnelle enrichissante.

Avec beaucoup de respect, j'exprime ma grande gratitude au corps professoral du Master intelligence artificielle et réalité virtuelle pour avoir porté un vif intérêt à ma formation, et pour m'avoir consacré le plus clair de leur temps, de leur attention et de leur énergie dans un cadre agréable de complicité et de respect.

Je tiens à exprimer ma plus profonde gratitude envers ma famille. À ma mère et mon père, pour leur amour inconditionnel, leur soutien constant et leurs sacrifices qui m'ont permis d'arriver où je suis aujourd'hui. À mon frère et ma sœur, pour leur encouragement, leur complicité et leur présence à mes côtés tout au long de ce parcours. Votre soutien a été une source inépuisable de motivation et de réconfort.

Je tiens également à exprimer ma reconnaissance envers toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce projet. Leur implication et leur soutien ont été essentiels, et je les remercie chaleureusement pour leur précieuse collaboration.

Résumé

Ce rapport synthétise le travail réalisé au cours de mon stage de fin d'études au sein de l'entreprise VERVE TECHNOLOGIES à Casablanca, dans le cadre de mon Master en Intelligence Artificielle et Réalité Virtuelle (IARV) à la Faculté des Sciences de l'Université Ibn Tofail (UIT) de Kénitra. Le projet sur lequel j'ai travaillé, intitulé « Génération de Présentations Immersives à l'aide de l'Intelligence Artificielle », vise à développer une application web innovante capable de créer des présentations interactives et immersives. Ce projet met en œuvre des technologies avancées d'intelligence artificielle pour automatiser la génération de texte et d'images, ainsi que pour transformer des diapositives classiques en expériences immersives sur plusieurs écrans.

L'application que j'ai développée intègre des fonctionnalités de génération de texte alimentées par des modèles de langage de grande taille (LLM). Ces modèles permettent de résumer, étendre des textes, et ajouter des annotations aux diapositives, avec une génération de contenu basée sur des sources telles que Wikipedia, le web scraping, ou des documents PDF. Un système de génération augmentée par récupération (RAG) a été mis en place pour contrôler efficacement la quantité de texte traité par le LLM et assurer des sorties fiables pour les utilisateurs, tout en adoptant la méthodologie agile Scrum, ce qui a facilité une exécution efficace et adaptable du projet.

En premier lieu, j'ai développé un backend robuste avec FastAPI, permettant la création de différents endpoints pour que les utilisateurs puissent interagir avec les modèles. Cette architecture inclut un système d'authentification pour garantir la sécurité et le contrôle d'accès des utilisateurs. Une fois le backend en place, j'ai concentré mes efforts sur la génération de texte et d'images. J'ai mis en place des APIs pour diverses tâches de génération de texte, notamment la synthèse, l'extension et l'ajout de notes aux diapositives. Pour la génération d'images, j'ai développé plusieurs endpoints permettant de créer des visuels basés sur le contenu des diapositives, incluant des options pour générer des vues naturelles ou des objets principaux sans arrière-plan sous forme d'images PNG.

Techniquement, le projet s'appuie sur des LLMs exécutés localement sur CPU, ainsi que sur l'intégration d'API tierces comme Groq pour accélérer la génération en temps réel. Les principaux outils utilisés incluent LangChain pour faciliter les interactions avec les LLMs et FastAPI pour créer des endpoints spécifiques.

Ce stage a été une expérience enrichissante, me permettant d'approfondir mes compétences en intelligence artificielle tout en développant des solutions pratiques pour améliorer l'expérience utilisateur. Je suis impatient de finaliser ce projet et de voir le produit aboutir.

Mots clés : VERVE TECHNOLOGIES, Génération de Présentations, LLM, RAG, Génération d'Images, LangChain, FastAPI.

Abstract

This report summarizes the work I completed during my end-of-studies internship at VERVE TECHNOLOGIES in Casablanca, as part of my Master's degree in Artificial Intelligence and Virtual Reality (IARV) at the Faculty of Science of Ibn Tofail University (UIT) in Kénitra. The project, titled "Generation of Immersive Presentations using Artificial Intelligence," focused on developing an innovative web application designed to create interactive and immersive presentations. This application leverages advanced artificial intelligence technologies to automate text and image generation, transforming traditional slides into multi-screen immersive experiences.

The application incorporates text generation features powered by large language models (LLMs), capable of summarizing, extending, and annotating slide content. Text generation is based on various sources, including Wikipedia, web scraping, and PDF documents. A Retrieval Augmented Generation (RAG) system was implemented to efficiently manage the amount of text processed by the LLM, ensuring accurate and reliable outputs. The project followed the agile Scrum methodology, which enabled a flexible and efficient execution process.

In addition to text generation, I developed several endpoints for image generation, enabling the creation of visuals based on slide content. These include options for generating natural scenes or focusing on main objects without backgrounds, saved as PNG images.

Technically, the project relies on LLMs running locally on CPUs, along with the integration of third-party APIs like Groq to enhance real-time generation speed. The primary tools used in the project include LangChain for facilitating LLM interactions and FastAPI for building specific endpoints.

This internship has been a valuable experience, deepening my expertise in artificial intelligence while allowing me to develop practical solutions that enhance user experience. I am eager to finalize this project and see the product come to life.

Keywords: *VERVE TECHNOLOGIES, Presentation Generation, LLM, RAG, Image Generation, LangChain, FastAPI.*

Table des matières

Liste des abréviations

Abréviation	Signification
NL	Natural language
LLM	Large language model
IA	Intelligence Artificielle (Artificial Intelligence)
ML	Machine Learning (Apprentissage automatique)
DL	Deep learning
SQL	Structured Query Language (Langage de requête structuré)
IDE	Integrated Development Environment (Environnement de développement intégré)
API	Application Programming Interface (Interface de programmation d'applications)
ETL	Extract, Transform, Load (Extraire, Transformer, Charger)
RAG	Retrieval Augmented Generation
Schema	Les instructions pour création d'une base de données

Liste des figures

Chapitre 1. Introduction générale

○ Introduction

Dans ce chapitre introductif, je présenterai tout d'abord l'organisme d'accueil VERVE TECHNOLOGIES, en citant ses clients, ses valeurs fondamentales et ses solutions innovantes. Ensuite, je procéderai à une présentation générale du projet et j'aborderai l'organisation de l'équipe de travail, les tâches qui m'ont été attribuées dans le contexte métier, ainsi que les technologies utilisées.

1.1. Présentation de l'organisme d'accueil

Figure 1.1: logo VERVE TECHNOLOGIES

VERVE TECHNOLOGIES est une entreprise de pointe spécialisée dans le secteur des technologies de l'information. Forte d'une équipe de 40 experts web cumulant plus de 12 ans d'expérience, VERVE TECHNOLOGIES accompagne ses clients dans le développement et la mise en œuvre de solutions innovantes en IoT, IA, web et mobile.

Sa mission est de permettre à ces clients d'exploiter pleinement le potentiel de l'internet en leur fournissant des solutions avant-gardistes fondées sur les nouvelles technologies. L'entreprise s'engage à propulser ces clients vers le succès en ligne grâce à son expertise en création de sites web et en marketing numérique.

Sa vision est de devenir un leader mondial en offrant des services inégalés et de qualité supérieure dans les domaines de l'IA, de l'IoT, ainsi que du développement web et mobile. VERVE TECHNOLOGIES se distinguons par sa capacité à fournir des solutions uniques et sur mesure, adaptées aux besoins spécifiques de chaque client.

1.1.1. Organigramme du VERVE TECHNOLOGIES

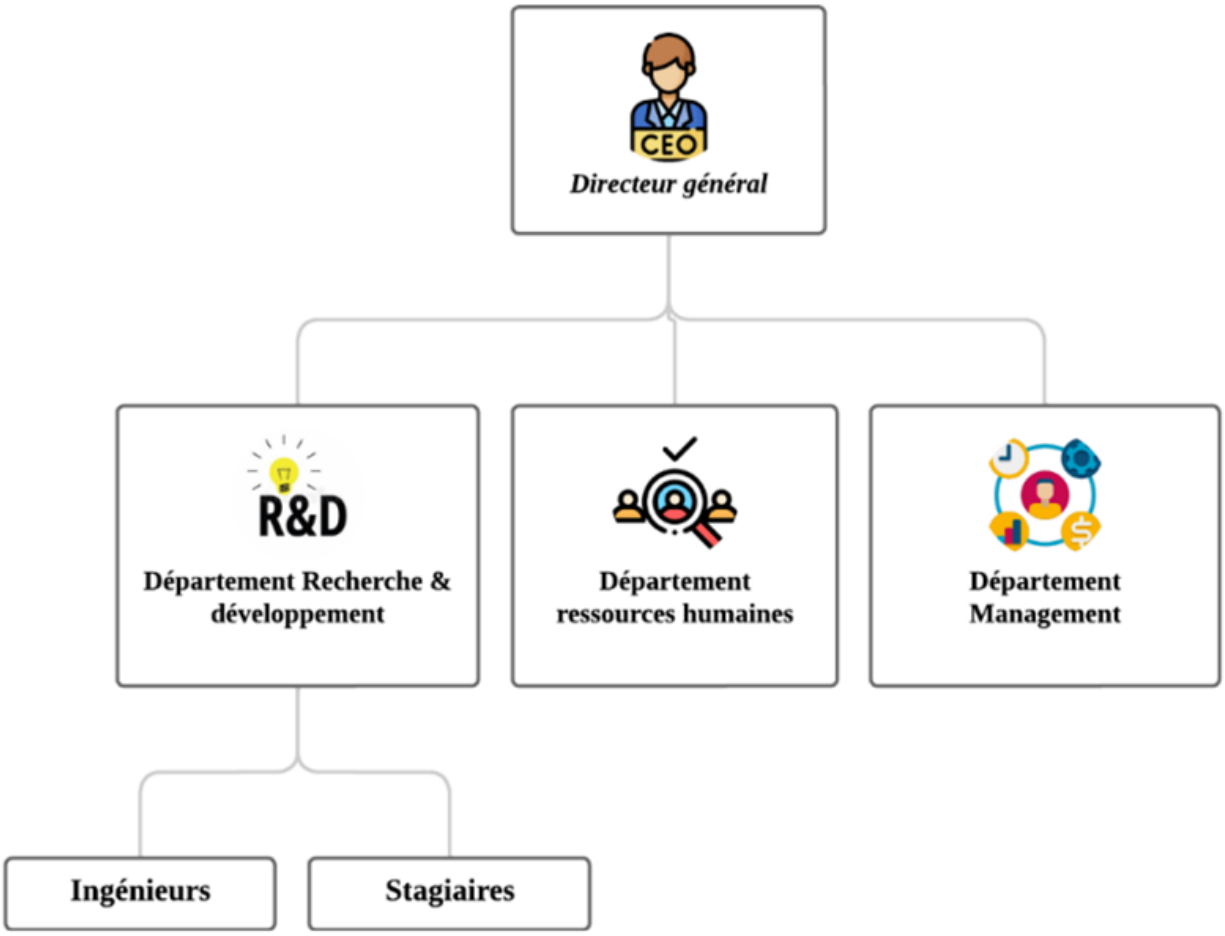

Figure 1.2 : l'organigramme de l'entreprise VERVE TECHNOLOGIES

L'organigramme est une représentation schématique des liens fonctionnels, organisationnels et hiérarchiques au sein d'une entreprise. Il offre une vue d'ensemble de la répartition des postes et des fonctions au sein de la structure. Cette cartographie simplifiée permet de visualiser les différentes relations de commandement et les rapports de subordination, fournissant ainsi une vision claire et concise de la structure organisationnelle.

L'organigramme facilite la compréhension des flux de communication, des responsabilités et des niveaux de décision, contribuant à une gestion plus efficace et à une meilleure coordination des activités au sein de l'entreprise.

1.1.2. Centres de Services

VERVE TECHNOLOGIES, une entreprise dynamique et jeune, propose à ses clients une gamme variée de services en technologies de l'information, allant de la recherche et développement à l'internet des objets, en passant par le conseil et l'intelligence artificielle.

Développement IT :

VERVE TECHNOLOGIES excelle dans le développement d'applications mobiles et web pour ses partenaires, en utilisant des technologies de pointe telles que Java EE, Angular, AngularJS, .Net, C#, PL/SQL, SQL, ABAP/4, PHP, Python et Flutter. L'entreprise se concentre sur la création de solutions sur mesure qui répondent aux besoins spécifiques de chaque client, garantissant des performances optimales et une expérience utilisateur exceptionnelle.

Recherche et Développement :

VERVE TECHNOLOGIES mobilise ses ressources internes pour créer des solutions innovantes visant à répondre aux défis clés de la transformation économique et digitale actuelle. L'équipe de R&D est dédiée à l'exploration de nouvelles technologies et à l'amélioration continue des produits et services, garantissant que les clients restent à la pointe de l'innovation.

Qualité Logicielle et Testing :

L'entreprise met un accent particulier sur la qualité logicielle et les tests rigoureux pour assurer la fiabilité et la performance de ses solutions. Des processus de test approfondis, incluant des tests automatisés et manuels, sont mis en place pour identifier et corriger les éventuelles anomalies avant le déploiement, assurant ainsi des produits de haute qualité.

Intelligence Artificielle :

VERVE TECHNOLOGIES intègre des solutions d'intelligence artificielle pour améliorer l'efficacité opérationnelle et offrir des perspectives inédites aux entreprises. En utilisant des techniques avancées de machine learning, de traitement du langage naturel et d'analyse prédictive, l'entreprise aide ses clients à exploiter leurs données pour prendre des décisions éclairées et automatiser des processus complexes.

Internet des Objets (IoT) :

Dans le domaine de l'internet des objets, VERVE TECHNOLOGIES développe et met en œuvre des solutions connectées qui permettent aux entreprises de recueillir, analyser et utiliser des données en temps réel. Ces solutions IoT facilitent la gestion intelligente des actifs, l'optimisation des opérations et la création de nouvelles opportunités commerciales grâce à l'intégration transparente des technologies connectées.

1.1.3. produit Dariwatt

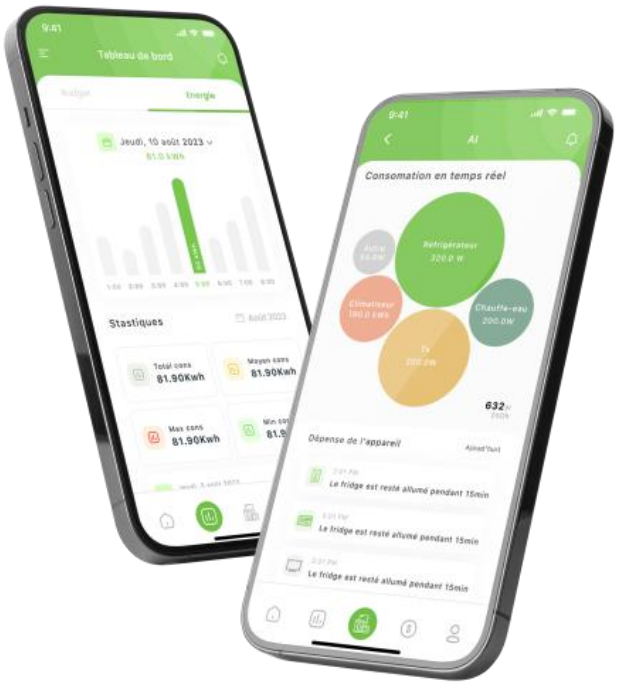

*Figure 1.3 :*DariWatt

DariWatt est une solution brevetée consistant en un dispositif qui s'intègre facilement sur votre tableau électrique et vous permet, via une application mobile, de consulter en détail l'énergie consommée dans votre maison, vous aidant ainsi à maitriser vos factures d'électricité.

1.2. Présentation du Projet

Introduction :

Ce projet, développé durant mon stage de fin d'études chez Verve Technology, a pour objectif de révolutionner la création de présentations en automatisant la génération de contenu textuel et visuel de haute qualité grâce à des technologies avancées d'intelligence artificielle. L'idée centrale est de permettre un gain de temps considérable et de simplifier le processus de création de contenu pour des utilisateurs tels que les professeurs, en leur permettant de se concentrer sur la conception et la présentation plutôt que sur la recherche et la génération de contenu.

Fonctionnalités clés du projet :

1. Gain de temps avec l'IA générative :
    - L'objectif principal est d'aider les utilisateurs à économiser un temps précieux en automatisant la création de contenu pour les diapositives à l'aide de modèles de langage de grande taille (LLM).
    - Que ce soit pour résumer du contenu complexe, prolonger un texte existant ou ajouter des annotations détaillées, le système génère du texte de qualité rapidement et efficacement.
    - Cela élimine la nécessité de rechercher et de rédiger manuellement le contenu de chaque diapositive, permettant aux utilisateurs de se concentrer sur la création et la structure de leur présentation.
2. Génération fluide de contenu à partir de différentes sources :
    - Le projet prend en charge plusieurs méthodes pour générer du contenu, incluant :

- Web scraping : depuis Wikipedia ou d'autres sources en ligne pour garantir l'actualité des informations.
- Traitement de documents PDF **:** pour extraire les points clés et générer un contenu pertinent à partir de publications académiques ou de rapports.

- Grâce au système Retrieval Augmented Generation (RAG), le texte généré est précis, pertinent et adapté au sujet de la présentation.

3. Génération d'images pour enrichir les visuels :
   - En plus du texte, le système génère des images correspondant au contenu de chaque diapositive.
   - Des options permettent de générer des scènes naturelles ou des visuels axés sur des objets principaux, ce qui enrichit l'aspect visuel de la présentation.
   - Ces images sont conçues pour être immersives, augmentant ainsi l'engagement de l'audience et rendant la présentation plus attrayante visuellement.
4. Environnement de présentation immersive :
   - Le système est conçu pour des environnements immersifs, transformant les diapositives classiques en une expérience de présentation multi-écran.
   - En automatisant la génération de contenu, nous garantissons que les visuels et le texte sont à la fois de haute qualité et contextuellement corrects, ce qui permet aux professeurs de se concentrer sur la conception de la mise en page et la structure de la présentation, sans se soucier de la création de contenu.

Comment le projet bénéficie aux professeurs et éducateurs :

- Accès rapide à l'IA générative :
  - Cette plateforme offre aux professeurs un accès simple et rapide aux outils de génération de contenu basés sur l'intelligence artificielle. Elle réduit

considérablement le temps nécessaire à la préparation des diapositives, leur permettant ainsi de se concentrer sur d'autres aspects de leur travail, tels que la planification des cours et l'interaction avec les étudiants.

- Génération de contenu basée sur les sujets du projet :
    - Le système génère du texte en fonction des sujets spécifiques de la présentation, garantissant que le matériel est pertinent et adapté à la thématique abordée.
    - Que ce soit à partir d'un PDF ou d'une ressource web, le contenu généré est précis, réduisant ainsi le besoin de recherches supplémentaires et de création manuelle.
- Intégration visuelle sans effort :
    - La fonctionnalité de génération d'images simplifie le processus d'ajout de visuels à la présentation. Plutôt que de rechercher des images manuellement, le système les génère automatiquement, garantissant leur cohérence avec le contenu de la diapositive.
    - Cette fonctionnalité est particulièrement précieuse pour les présentations immersives, où les visuels jouent un rôle clé dans l'engagement du public.

Chapitre 2. Etat de l'art sur les LLMs

○ Introduction

Dans un monde où l'efficacité et l'automatisation sont devenues des priorités pour les entreprises, l'intelligence artificielle, et en particulier les modèles de langage de grande taille (LLMs), joue un rôle de plus en plus central. Ces modèles sont capables de comprendre, générer et traiter du texte en langage naturel, ouvrant la voie à des innovations considérables dans divers secteurs. Dans le cadre de mon projet de présentation immersive, les LLMs ont été utilisés pour automatiser la génération de contenus textuels et visuels, facilitant ainsi la création de présentations intelligentes et interactives.

Les LLMs, basés sur l'architecture Transformer, ont la capacité de traiter d'énormes quantités de données textuelles et de capturer des relations sémantiques complexes entre les mots. Ce projet a exploité ces modèles pour différentes tâches comme la génération de résumés, l'extension de textes, la création de notes pour les diapositives, ainsi que la génération d‘un promts d'images à partir de descriptions textuelles. Ces technologies permettent d'automatiser des processus auparavant manuels et chronophages, augmentant ainsi la productivité et la qualité des contenus générés.

Un des aspects les plus intéressants de l'utilisation des LLMs dans ce projet est leur capacité à créer des présentations immersives en transformant un texte simple en plusieurs diapositives bien structurées. Cette approche permet de créer des expériences immersives dans lesquelles les utilisateurs peuvent visualiser des contenus sur plusieurs écrans, tout en bénéficiant d'une présentation fluide et cohérente. Les modèles de génération d'images, comme Stable Diffusion, ont également été intégrés pour produire des visuels uniques basés sur le contenu des diapositives, rendant chaque présentation plus engageante et esthétique.

L'utilisation de modèles open-source et la possibilité de les déployer localement ont été essentielles pour assurer la sécurité des données et la confidentialité, tout en maintenant un haut niveau de performance. Cela a permis de mettre en œuvre des solutions évolutives sans compromettre la sensibilité des informations manipulées.

Ce chapitre explorera en détail les technologies derrière ces modèles de langage, notamment comment les LLMs sont utilisés pour la génération de contenus textuels et visuels dans le cadre de mon projet. j’examinerons les méthodes utilisées pour structurer les présentations de manière intelligente, les algorithmes de génération d'images, ainsi que les avantages et défis associés à l'implémentation de ces technologies dans des environnements de présentation immersive

2.1. Les large language models : LLMs

2.1.1. Définition des LLMs

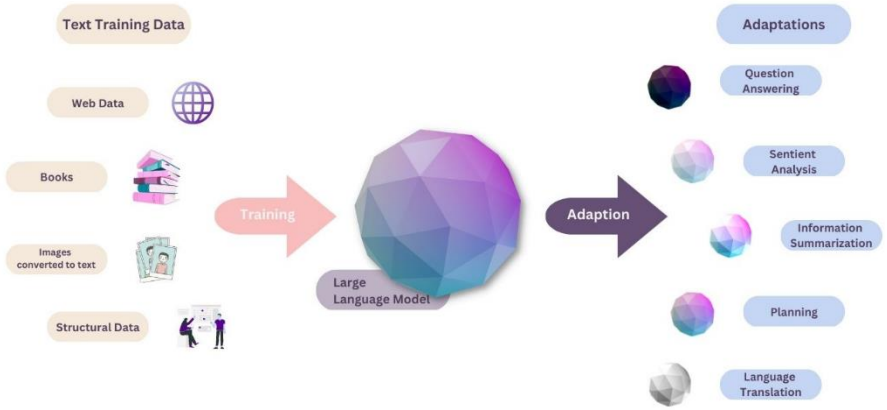

Figure 2.1: schéma qui démontre comment fonctionne les LLMs

Un large language model est un type avancé de modèle de langage qui est entraîné en utilisant des techniques d'apprentissage profond sur des quantités massives de données textuelles. Ces modèles sont capables de générer du texte semblable à celui produit par les humains et d'effectuer diverses tâches de traitement de langage naturelle.

En revanche, la définition d'un modèle de langage fait référence au concept d'attribuer des probabilités à des séquences de mots, basées sur l'analyse de corpus de textes.
Un modèle de langage peut avoir une complexité variable, allant de simples modèles n-gram à des modèles de neural networks plus sophistiqués.

Cependant, le terme "large language model" fait généralement référence à des modèles qui utilisent des techniques d'apprentissage profond et ont un grand nombre de paramètres, pouvant aller de millions à des milliards. Ces modèles d'IA peuvent capturer des modèles complexes dans le langage et produire du texte qui est souvent indiscernable de celui écrit par des humains.

2.1.3.1. GroqAPI

GroqAPI: Cost and Speed Advantage :

GroqAPI offers a significant advantage in both cost and speed compared to traditional APIs like OpenAI's GPT-4, making it an attractive option for businesses looking to use large language models (LLMs) at scale.

Cost Comparison

- GroqAPI (Llama 3.1 8B Instant 128k):
  - Input cost: 20 million tokens per $1
  - Output cost: 12.5 million tokens per $1
- OpenAI GPT-4o 128k:
  - Input cost: $5.00 per 1 million input tokens
  - Output cost: $15.00 per 1 million output tokens

In this example, GroqAPI is drastically cheaper than OpenAI's GPT-4 for similar token usage. You save over 99% in total cost when using GroqAPI, which makes it ideal for heavy usage scenarios.

Speed Comparison :

GroqAPI is designed for high performance, using Groq hardware, which accelerates LLM computations with low-latency and high-speed throughput. This allows GroqAPI to handle models like LLaMA 3.1 faster than traditional cloud services that rely on more general-purpose hardware.

For enterprises, this means faster model inference, quicker response times, and the ability to process larger amounts of data more efficiently. The speed gains, combined with lower costs, provide a dual benefit that is hard to match with other LLM providers.

2.1.4. Contrôle de la Génération de Texte avec RAG

2.1.4.1. Introduction au RAG

La génération augmentée par récupération (RAG) est une technique de traitement du langage naturel (NLP) qui combine les points forts des modèles d'intelligence artificielle (IA) basés sur la récupération et ceux basés sur la génération.

Exemple d'utilisation du RAG : RAG permet à un modèle de langage de grande taille (LLM) de connaître le prix actuel du Bitcoin lorsqu'on lui demande combien de BTC on peut acheter pour 1 000 $.

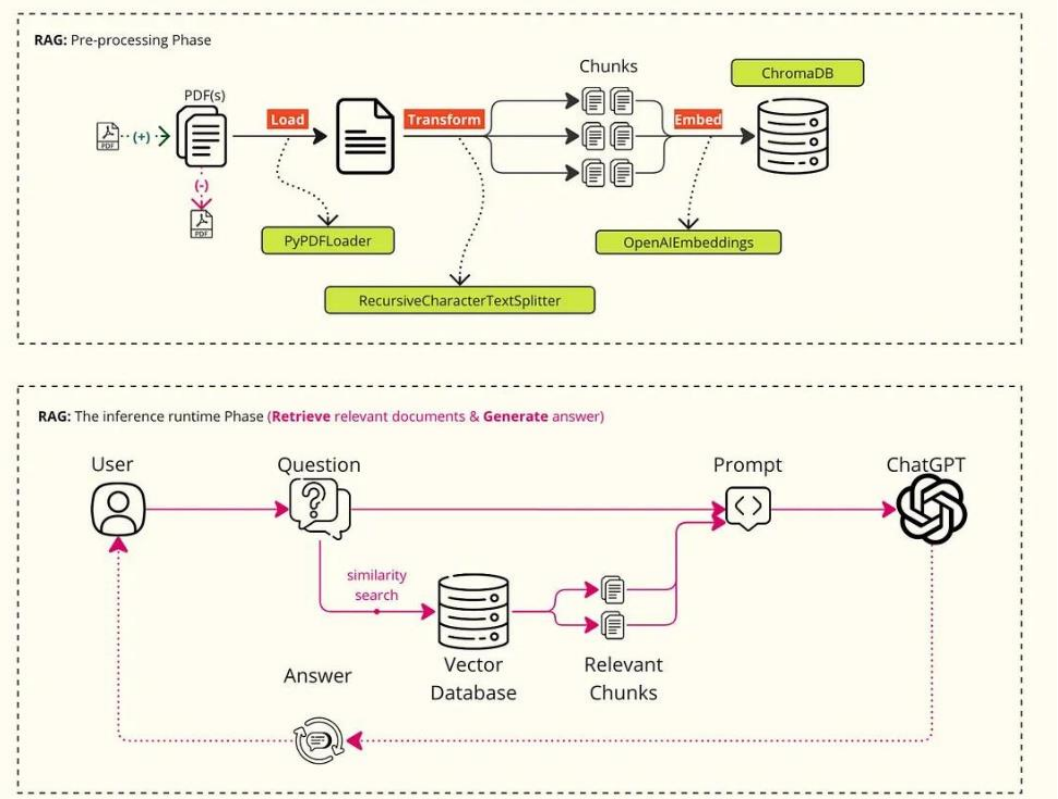

Figure 2.2: génération augmentée par récupération (RAG)

Qu'est-ce que le Traitement du Langage Naturel (NLP) ? :

Le traitement du langage naturel (NLP) est une branche de l'informatique, plus précisément de l'intelligence artificielle (IA), qui vise à permettre aux ordinateurs de comprendre le texte et les mots parlés de manière similaire à ce que font les humains.
Le NLP combine la modélisation basée sur des règles du langage humain avec des modèles statistiques, d'apprentissage automatique (machine learning) et d'apprentissage profond (deep learning).
Ces technologies permettent aux ordinateurs de traiter le langage humain sous forme de texte ou de données vocales et de "comprendre" son sens complet, y compris l'intention et le sentiment de l'auteur.
Les modèles d'IA génératifs populaires tels que Bard, ChatGPT et Grok sont des exemples d'applications du NLP.

Modèles Basés sur la Récupération dans le NLP :

Les modèles basés sur la récupération dans le NLP sont conçus pour sélectionner une réponse appropriée parmi un ensemble prédéfini de réponses en fonction de la requête d'entrée.

Ces modèles comparent le texte d'entrée (une question ou une requête) avec une base de données de réponses prédéfinies. Le système identifie la réponse la plus appropriée en mesurant la similarité entre l'entrée et les réponses stockées en utilisant des techniques telles que la similarité cosinus ou d'autres méthodes de correspondance sémantique. Les modèles basés sur la récupération sont efficaces pour des tâches telles que la réponse à des questions, où les réponses sont souvent factuelles et disponibles sous une forme structurée.

Modèles Basés sur la Génération dans le NLP :

Les modèles basés sur la génération créent des réponses à partir de zéro. Ces modèles utilisent des algorithmes complexes, souvent basés sur des réseaux neuronaux, pour générer du texte ou des réponses semblables à celles des humains.

Contrairement aux modèles basés sur la récupération, les modèles basés sur la génération ne reposent pas sur des réponses prédéfinies. Au lieu de cela, ils apprennent à générer des réponses en prédisant le mot suivant ou la séquence de mots en fonction du contexte fourni par l'entrée. Cette capacité à générer des réponses nouvelles et contextuellement appropriées rend les modèles de génération très polyvalents et adaptés à des tâches telles que l'écriture créative, la traduction et le dialogue, où les réponses doivent être diverses et riches en contexte.

Pourquoi le RAG est-il Utile ?

Les modèles de base (comme ChatGPT d'OpenAI ou LLaMA de Meta) sont généralement entraînés hors ligne, ce qui rend le modèle indépendant de toute donnée créée après l'entraînement du modèle. De plus, les modèles de base sont formés sur des corpus de domaine très général, ce qui les rend moins efficaces pour les tâches spécifiques à un domaine, bien qu'ils soient plus efficaces pour les tâches générales.

La Génération Augmentée par Récupération (RAG) peut être utilisée pour récupérer des données provenant de sources extérieures à un modèle de base et enrichir les invites en ajoutant les données récupérées pertinentes en contexte.

Comment cela fonctionne-t-il ?

1. Une invite donnée est saisie par un utilisateur.
2. Cette invite est comparée à une base de données / une source d'information que le modèle de base ne connaît pas.

3. Un contexte spécifique qui aidera le modèle de génération (modèle de base) à répondre de manière optimale est retourné par le modèle basé sur la récupération et ajouté à l'invite.

4. L'invite (maintenant enrichie du contexte) est donnée au modèle de génération, et le modèle répond mieux qu'il ne l'aurait fait sans le contexte supplémentaire.

En résumé, RAG est la manière dont Bard connaît la température extérieure, combien de temps il faut pour se rendre au parc, et combien de temps cela fait vraiment depuis que vous avez touché l'herbe.

○ Conclusion

Les modèles de langage (LLM) comme ceux fournis par GroqAPI ne sont pas simplement des outils, mais de véritables catalyseurs pour repousser les frontières de l'efficacité et de la créativité. En comparaison avec d'autres solutions, notamment OpenAI, GroqAPI se démarque par son rapport coût/performance exceptionnel et ses temps de réponse rapides, rendant l'accès aux LLM plus abordable et plus rapide.

Dans ce projet, l'implémentation de la solution GroqAPI, couplée à un système RAG (Retrieval-Augmented Generation), constitue une approche idéale. Le RAG permet de contextualiser et d'enrichir les réponses de l'IA avec des données externes spécifiques, assurant des informations plus précises et pertinentes dans les présentations générées. Cela permet d'améliorer la qualité des applications tout en minimisant les coûts par rapport à des solutions comme OpenAI, où les frais d'utilisation peuvent devenir prohibitifs à grande échelle.

En définitive, l'intégration de GroqAPI avec un système RAG offre non seulement un excellent compromis entre rapidité, efficacité, et prix, mais positionne également cette combinaison comme une solution de pointe dans le domaine des applications basées sur l'IA. Ce modèle propose une alternative robuste et économique face à OpenAI, tout en garantissant un accès fluide et une scalabilité adaptée aux besoins croissants de l'innovation tech

Chapitre 3. La génération des images en utilisant l'I.A

○ Introduction à la génération d'images

3.1.3. Texte-à-Image et génération d'Images

Avez-vous déjà visualisé une image à partir d'une histoire captivante ? Ou imaginé un paysage qui n'existe que dans vos rêves ? L'intersection de la synthèse texte-à-image et de la génération d'images pilotée par l'intelligence artificielle est l'endroit où ces fantasmes prennent vie. Explorons ces technologies révolutionnaires, leurs différences et leur impact incroyable sur notre monde.

Texte-à-Image et génération d'images :

- Texte-à-Image : C'est une technologie fascinante qui transforme les mots en images. Que ce soit une scène descriptive tirée d'un roman ou une simple requête comme « Dessine-moi un coucher de soleil sur l'océan », la synthèse texte-à-image utilise l'intelligence artificielle pour traduire le texte en art visuel. Il s'agit de combler le fossé entre le langage et l'imagerie.
- Génération d'images par l'IA **:** Au-delà de simples traductions, la génération d'images par IA consiste à créer des visuels entièrement nouveaux. Alimentée par des algorithmes complexes et l'apprentissage automatique, cette innovation produit des images uniques sans nécessiter de descriptions textuelles spécifiques. C'est comme exploiter la créativité illimitée d'un artiste numérique.

Différences Entre Texte-à-Image et génération d'images :

- Texte-à-Image traduit des descriptions spécifiques en images, transformant des mots en réalité visuelle.
- Génération d'Images explore des territoires inexplorés, créant de nouveaux visuels sans guide textuel prédéfini.

Ensemble, ces technologies façonnent l'avenir des médias visuels, du divertissement, du design et bien plus encore.

3.1.4. Aperçu historique

Synthèse d'Images à ses Débuts (Avant 2000) :

- Fractales et Techniques Procédurales :
  Les fractales et les techniques procédurales étaient des approches mathématiques permettant de créer des images complexes à partir d'équations simples. Elles ont permis de générer des formes naturelles comme les montagnes, les arbres et les nuages sans avoir à modéliser ces objets manuellement.
- Lancer de Rayons (Ray Tracing) :
  Développé dans les années 1980, le lancer de rayons est une technique de rendu pour simuler la manière dont la lumière interagit avec les objets dans un environnement 3D. Cela a jeté les bases des images de synthèse modernes en permettant la génération d'images avec des reflets et des ombres réalistes.
- Mappage de Textures :
  Dans les années 1990, le mappage de textures est devenu une technique courante pour ajouter du détail visuel aux modèles 3D en appliquant des images 2D aux surfaces des objets. Cela a permis une amélioration significative de la qualité visuelle des jeux vidéo, des films et des simulations.

Introduction des Réseaux de Neurones et de l'IA (Début des Années 2000) :

- Réseaux Neuronaux Feedforward :
  Les réseaux neuronaux de type feedforward ont marqué le début de l'utilisation de l'IA pour l'image. Cependant, leur capacité à modéliser des relations complexes dans les

images était limitée, car ils ne pouvaient pas capter efficacement les structures spatiales.

- Réseaux Convolutionnels (CNN) :

  Introduits par Yann LeCun, les CNN ont révolutionné la reconnaissance d'images grâce à leur capacité à extraire des caractéristiques hiérarchiques des images, telles que des contours, des textures et des formes. Les CNN ont permis une amélioration considérable des algorithmes de reconnaissance d'image, ouvrant la voie aux développements ultérieurs dans la génération d'images.

Développement de Modèles Génératifs (Milieu des Années 2010) :

- Generative Adversarial Networks (GANs) – 2014 :

  Proposés par Ian Goodfellow, les GANs ont introduit un nouveau paradigme dans la génération d'images. Deux réseaux de neurones s'affrontent : un générateur qui crée des images à partir de bruit aléatoire, et un discriminateur qui évalue si les images sont réelles ou générées. Les GANs ont ouvert la voie à une génération d'images réalistes, mais ils restaient difficiles à entraîner et souvent instables.
- Variational Autoencoders (VAEs) – 2013 :

  Les VAEs, introduits par Kingma et Welling, sont des modèles probabilistes capables de générer des images réalistes à partir d'une distribution latente. Contrairement aux GANs, les VAEs sont plus stables et permettent de capturer une variété plus large de données génératives, bien qu'ils ne produisent pas toujours des images aussi précises.
- Deep Convolutional GANs (DCGANs) – 2015 :

  Les DCGANs ont amélioré les GANs en introduisant des réseaux convolutionnels profonds, ce qui a rendu la génération d'images beaucoup plus réaliste et détaillée. Cette avancée a permis de générer des images de qualité à partir de descriptions simples

.

Révolution Texte-à-Image (Fin des Années 2010)

- StackGAN (2017) :

  StackGAN a marqué une percée dans la synthèse texte-à-image en introduisant une approche en deux étapes. La première étape génère une image basique à partir du texte, et la deuxième étape affine l'image en ajoutant plus de détails, ce qui a considérablement amélioré la qualité des images générées à partir de descriptions textuelles.

- AttnGAN (2018) :

  L'introduction d'AttnGAN a intégré un mécanisme d'attention dans le processus de génération d'images, permettant au modèle de se concentrer sur différentes parties du texte lors de la génération d'images. Cela a permis de générer des images beaucoup plus précises et cohérentes avec les descriptions textuelles.

Techniques Contemporaines (2021-présent) :

BigGAN (2018) :

BigGAN a augmenté la capacité des GANs en utilisant de très grands modèles et ensembles de données, ce qui a permis de générer des images de très haute qualité, souvent presque indiscernables des images réelles.

StyleGAN (2019-2020) :

StyleGAN, développé par NVIDIA, a révolutionné la génération d'images en introduisant des contrôles sur les caractéristiques génératives à différents niveaux de l'image. Cela a permis aux utilisateurs de modifier des aspects spécifiques des images, tels que les

expressions faciales, les cheveux ou le fond, ouvrant la voie à des applications plus interactives.

DALL-E (2021) :

DALL-E d'OpenAI a marqué un tournant dans la génération d'images à partir de texte en proposant un modèle capable de générer des images très complexes et précises à partir de descriptions textuelles simples. Il a été l'un des premiers modèles à montrer un potentiel véritablement créatif en termes de génération d'images à la demande.

CLIP (2021) :

CLIP (Contrastive Language-Image Pre-training), également d'OpenAI, a introduit une nouvelle approche permettant de relier efficacement des images et des textes, facilitant la recherche, l'édition et la génération d'images en fonction de commandes textuelles.

Imagen (2022) :

Imagen, développé par Google AI, utilise des modèles de diffusion pour générer des images de très haute qualité à partir de descriptions textuelles. Il a montré des performances supérieures à DALL-E dans de nombreux cas, notamment en termes de réalisme et de cohérence.

Stable Diffusion (2022) :

Stable Diffusion de Stability AI a introduit une méthode de diffusion qui a permis une génération d'images haute qualité à partir de descriptions textuelles, tout en réduisant les besoins en calcul. Ce modèle est particulièrement apprécié pour sa capacité à générer des images avec des descriptions succinctes, et il est devenu un acteur majeur dans les outils de génération d'images IA.

MidJourney (2022-2023) :

MidJourney s’est distingué en tant qu'outil de génération d'images pour les créateurs et les artistes. Avec un style distinctif, il permet de créer des œuvres visuelles en fonction des descriptions, souvent en concurrence avec des modèles comme DALL-E et Stable Diffusion.

3.2. Mise en œuvre dans les projets

Dans cette section, je vais décrire comment Stable Diffusion a été intégré dans mon projet pour générer des images, y compris la suppression des arrière-plans des images. La solution a été développée en utilisant FastAPI, en exploitant la puissance des LLMs (Large Language Models) pour générer des prompts significatifs et en intégrant les modèles de Hugging Face pour produire des visuels de haute qualité. Ci-dessous, je vais expliquer comment j'ai utilisé différents espaces de Hugging Face et géré les points de terminaison de génération d'image dans le projet.

Espace Hugging Face et Intégration de Stable Diffusion :

Pour générer des images dans le projet, j'ai utilisé le modèle Stable Diffusion 3 de Hugging Face. Stable Diffusion est un modèle puissant capable de générer des images à partir de descriptions textuelles, offrant une large gamme de personnalisations en termes de style, de résolution et de spécificité des prompts.

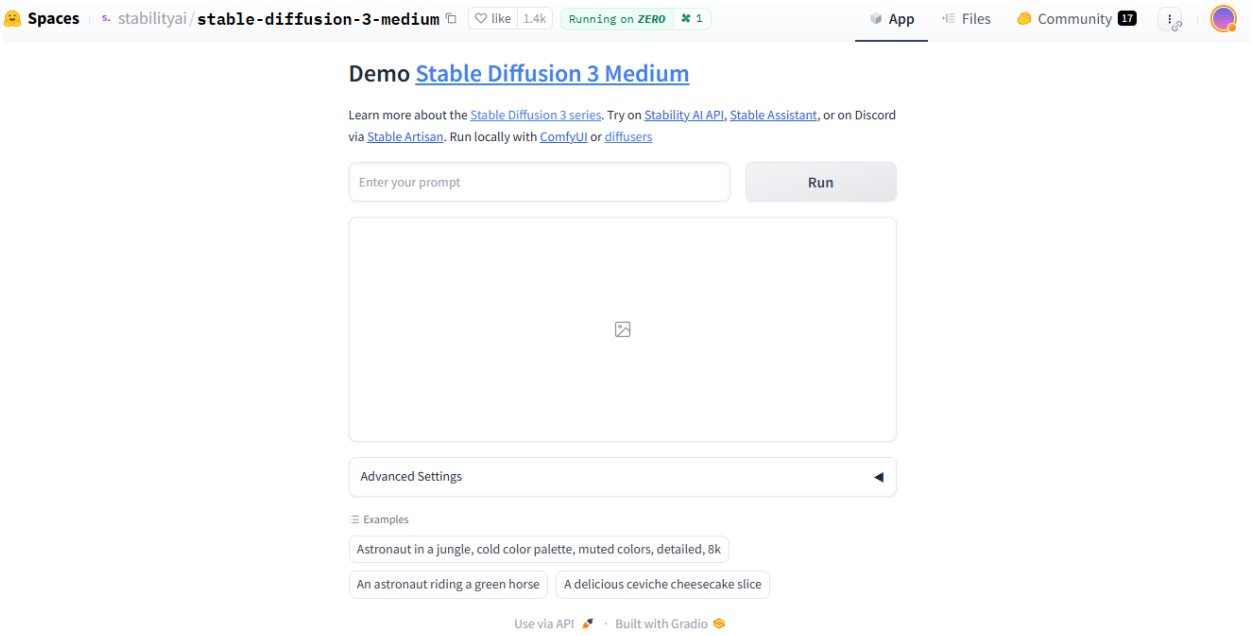

*Figure 3.1:*Stable Diffusion 3 de Hugging Face

J'ai employé l'espace Hugging Face.

`"stabilityai/stable-diffusion-3-medium"` pour générer des images basées sur les prompts fournis par l'utilisateur ou générés dynamiquement via les LLMs. L'intégration commence par l'appel à l'API pour prédire l'image en fonction d'une description textuelle (`prompt`) en utilisant les paramètres suivants :

- Prompt : La description textuelle principale fournie par l'utilisateur ou générée par le LLM.
- Negative Prompt : Pour filtrer les éléments non désirés.
- Seed : Pour la reproductibilité.
- Largeur et Hauteur : Pour contrôler les dimensions de l'image.
- Guidance Scale et Inference Steps : Pour ajuster la qualité et la créativité de l'image générée.

Génération d'Images via les Endpoints FastAPI :

J'ai conçu une série de points de terminaison FastAPI pour gérer la génération d'images en fonction des entrées utilisateur ou des prompts automatisés générés par les LLMs. Chaque endpoint suit un processus structuré pour récupérer la requête, générer l'image, puis la retourner au format requis (PNG).

○ Conclusion

L'intégration de la génération d'images dans mon projet a permis d'explorer les capacités des modèles d'intelligence artificielle, notamment Stable Diffusion, pour créer des visuels adaptés à des présentations immersives. Grâce à l'utilisation d'espaces comme Hugging Face, combinée avec des LLMs pour générer des prompts optimisés, il est possible de produire des images de haute qualité de manière automatisée.

De plus, la mise en place de points de terminaison spécifiques pour la génération d'images sans arrière-plan a ajouté une nouvelle dimension de flexibilité, en facilitant la personnalisation et la manipulation d'éléments visuels isolés. L'intégration de services comme remove.bg ou des techniques similaires a permis d'affiner encore plus les résultats en rendant les images générées plus propres et prêtes à l'emploi pour diverses présentations.

Ce projet a ainsi démontré l'importance de combiner plusieurs technologies, notamment la génération d'images par IA et le traitement automatique des langues, pour enrichir l'expérience visuelle et interactive des utilisateurs, ouvrant la voie à de nouvelles opportunités dans le domaine de la création visuelle automatisée et des présentations assistées par l'IA.

Chapitre 4. Outils et Technologies utilisés

4.1. Language, Frameworks et bibliothèques

○ Introduction

Dans ce chapitre, je vais passer en revue les principales technologies et outils utilisés dans mon projet de Text-to-SQL basé sur les LLMs. Pour chaque élément, je fournirai une brève définition, mettant en lumière son rôle spécifique dans mon développement. Cette approche permettra de donner un aperçu clair et concis de ma stack technologique, tout en soulignant l'importance de chaque composant dans la réalisation de ma solution.

4.1.1. Language de programmations

4.1.1.1. Python

Figure 4.1 : logo python

Python est un langage de programmation de haut niveau, interprété et polyvalent, créé par Guido van Rossum et lancé en 1991. Il se distingue par sa syntaxe claire et lisible, qui met l'accent sur la simplicité et la facilité d'utilisation.

Utilisation en IA :

Python est devenu le langage de prédilection pour le développement en intelligence artificielle et en apprentissage automatique pour plusieurs raisons :

- Simplicité et lisibilité : Sa syntaxe claire permet aux chercheurs et développeurs de se concentrer sur la résolution de problèmes complexes plutôt que sur les détails de la programmation.
- Riche écosystème de bibliothèques : Python dispose d'une vaste collection de bibliothèques spécialisées pour l'IA et l'apprentissage automatique, telles que TensorFlow, PyTorch, scikit-learn, et NumPy, qui facilitent le développement de modèles sophistiqués.
- Communauté active : Une large communauté de développeurs contribue constamment à l'amélioration des outils et bibliothèques, offrant un support précieux et des ressources abondantes.
- Flexibilité : Python s'intègre facilement avec d'autres langages et technologies, ce qui est crucial pour les projets d'IA complexes.
- Prototypage rapide : La nature interprétée de Python permet un développement et un test rapides des idées, ce qui est essentiel dans la recherche en IA.

Ces caractéristiques font de Python un choix privilégié pour le développement en IA, de la recherche fondamentale aux applications industrielles.

Google Colab

4.1.2. Plateformes

4.1.2.1. Hugginface

Figure 4.2: huggingface logo

Hugging Face est une plateforme open-source et une communauté dédiée à l'apprentissage automatique et au traitement du langage naturel (NLP). Depuis sa création en 2016, elle est devenue un centre de référence pour le développement, le partage et l'utilisation de modèles de langage pré-entraînés, ainsi que d'outils liés à l'intelligence artificielle.

Rôle dans notre projet de génération d'images :

- Exploration des modèles : Hugging Face m'a permis de rechercher et d'explorer des modèles pré-entraînés spécifiques à la génération d'images, tels que Stable Diffusion, pour répondre à nos besoins.
- Téléchargement des modèles :J'ai pu facilement télécharger ces modèles pour les utiliser dans mon projet et les adapter à la génération d'images avec et sans arrière-plan.
- Utilisation d'API :En utilisant les API disponibles sur Hugging Face, j'ai pu intégrer des modèles directement dans mes endpoints, facilitant ainsi la génération d'images à partir de prompts créés avec des LLMs.
- Partage des résultats :Une fois les modèles intégrés et les images générées avec succès, Hugging Face m'a offert une plateforme pour partager mes résultats avec la communauté.

- Bibliothèque Transformers : J'ai utilisé la bibliothèque Transformers pour optimiser les performances des modèles génératifs et améliorer l'interaction avec les API de génération d'images.

L'utilisation de Hugging Face dans notre projet a permis de simplifier l'accès à des modèles performants, d'accélérer le processus de développement, et d'assurer une flexibilité dans l'intégration de nouvelles fonctionnalités d'IA. Cela a également favorisé un partage de connaissances avec la communauté tout en respectant les meilleures pratiques en matière d'IA.

4.1.2.2. Hugging Face Spaces

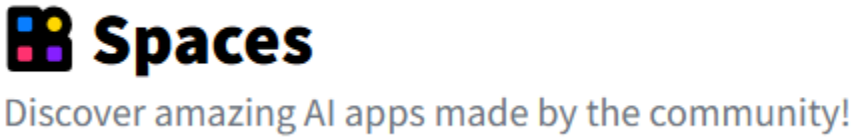

Figure 4.3: huggingface spaces logo

Hugging Face Spaces : Déployer vos Modèles en Quelques Clics

En plus des modèles et des API, Hugging Face propose également une fonctionnalité puissante et pratique appelée Spaces. Il s'agit d'un environnement de déploiement qui permet de créer, tester et partager facilement des applications basées sur des modèles d'IA.

Qu'est-ce que Hugging Face Spaces ?

- Spaces est une plateforme d'hébergement d'applications qui permet aux développeurs de déployer des modèles d'IA sous forme d'applications interactives.
- Ces applications peuvent être construites en utilisant des frameworks populaires comme Gradio ou Streamlit, facilitant ainsi l'intégration et l'interaction avec les modèles d'IA.

Comment utiliser Hugging Face Spaces ? :

1. Création d'un Space : Vous pouvez créer un Space gratuitement sur Hugging Face en accédant à l'onglet "Spaces" sur la plateforme. Vous pouvez choisir entre plusieurs frameworks, comme Gradio ou Streamlit, en fonction de vos préférences.
2. Déploiement d'un modèle : Une fois le Space créé, il est possible de déployer un modèle directement depuis le dépôt de modèles de Hugging Face. Il suffit de sélectionner un modèle (comme Stable Diffusion pour la génération d'images), de l'intégrer dans l'application, et de configurer les paramètres.
3. Exécution de l'application : L'application déployée sera automatiquement hébergée sur Hugging Face et accessible via un lien public. Cela permet de tester l'interaction avec le modèle en temps réel et de partager les résultats avec la communauté.
4. Partage et collaboration : Hugging Face Spaces facilite la collaboration avec d'autres développeurs et chercheurs en permettant le partage facile des Spaces avec un lien public, rendant vos projets accessibles à d'autres utilisateurs de la plateforme.

Types de GPU fournis par Hugging Face Spaces :

Pour les projets nécessitant une grande puissance de calcul, comme la génération d'images ou l'entraînement de modèles, Hugging Face offre un accès à des GPU directement dans les Spaces

- T4 GPUs : Fournis par Nvidia, ces GPU sont utilisés pour des tâches nécessitant une accélération modérée, comme le fine-tuning de modèles de NLP ou la génération d'images.
- A10G GPUs : Ce GPU de Nvidia est optimisé pour les charges de travail de deep learning plus intensives, comme la génération d'images haute résolution ou l'inférence rapide sur de grands modèles.
- A100 GPUs : Également de Nvidia, l'A100 est l'un des GPU les plus puissants disponibles pour les projets IA. Il est idéal pour des tâches extrêmement lourdes comme l'entraînement de modèles de très grande taille ou l'inférence en temps réel sur des ensembles de données volumineux.

Les Spaces de Hugging Face permettent ainsi de déployer des modèles rapidement et de les utiliser directement avec des ressources adaptées à vos besoins, qu'il s'agisse de simples tests ou de projets nécessitant une puissance de calcul importante.

4.1.2.3. Google Colab

Figure 4.4: Google colab logo

Google Colab (abréviation de *Google Collaboratory*) est un outil en ligne gratuit fourni par Google qui permet d'exécuter du code Python directement dans un navigateur. Il est très

populaire auprès des chercheurs, développeurs et étudiants pour la réalisation de projets d'intelligence artificielle, d'apprentissage automatique et d'analyse de données, notamment grâce à l'accès à des ressources matérielles puissantes, comme les GPU et les TPU.

Rôle dans notre projet :

- Environnement de développement pratique :Google Colab m'a fourni un environnement de développement accessible depuis n'importe quel appareil avec une connexion Internet. Cela a facilité la collaboration et le partage de notebooks entre les membres de l'équipe, permettant ainsi d'expérimenter et de tester rapidement mes modèles d'IA.
- Accès aux GPU/TPU : L'une des plus grandes forces de Google Colab est la possibilité d'accéder gratuitement à des GPU et TPU pour des tâches intensives, ce qui a considérablement accéléré les phases d'entraînement et d'inférence de nos modèles, en particulier ceux de génération d'images et Texts.
- Intégration avec des bibliothèques de deep learning : Colab prend en charge l'installation de bibliothèques telles que TensorFlow, PyTorch, et Transformers de Hugging Face. Cela m'a permis d'expérimenter facilement avec des modèles d'IA et de bénéficier d'une exécution rapide et fiable.
- Sauvegarde sur Google Drive : Grâce à l'intégration avec Google Drive, j'ai pu sauvegarder et charger mes modèles et jeux de données de manière fluide, garantissant ainsi la continuité du projet sans avoir à configurer de serveurs supplémentaires.
- Partage et collaboration : Le format notebook de Google Colab a facilité la documentation du code et des expériences. Chaque membre de l'équipe pouvait accéder aux notebooks, ajouter des commentaires ou des modifications en temps réel, favorisant ainsi un travail collaboratif efficace.

En résumé, Google Colab a été un outil essentiel pour le développement de notre projet, offrant à la fois la puissance de calcul nécessaire pour entraîner nos modèles et une interface intuitive pour tester, collaborer et itérer rapidement sur nos idées.

4.1.2.4. Papers with code

Figure 4.5: papers with code logo

Papers with Code est une plateforme open-source qui regroupe des articles de recherche en apprentissage automatique avec leurs implémentations de code correspondantes. Elle a joué un rôle essentiel dans notre projet de génération de présentations immersives, notamment en facilitant l'accès aux recherches récentes et à leurs implémentations.

Rôle dans notre projet de génération de présentations immersives :

- Exploration de l'état de l'art :J'ai exploré les avancées les plus récentes dans les domaines de la génération d'images et de la génération de texte grâce à Papers with Code, en accédant aux articles de recherche pertinents et à leur code.
- Benchmarks et évaluations :La plateforme m'a permis de consulter des classements et évaluations de modèles pour des tâches spécifiques liées à mon projet, telles que la génération d'images à partir de texte et la génération de contenus immersifs.

- Inspiration pour l'implémentation : Je me suis inspiré des implémentations disponibles pour structurer mon propre pipeline de génération de contenu, en adaptant certaines techniques à mes besoins spécifiques.
- Reproductibilité : Grâce aux codes sources publiés sur la plateforme, j'ai pu reproduire certains résultats de modèles d'IA pertinents, ce qui m'a permis de mieux comprendre leur fonctionnement et de les intégrer à mon projet.
- Veille technologique :La plateforme a facilité mon suivi des innovations dans l'IA, notamment en matière de modèles de diffusion pour la génération d'images et de modèles textuels, ce qui a influencé mes choix technologiques et méthodologiques.
- Comparaison des performances : J'ai comparé les performances de mes propres modèles avec celles présentes sur la plateforme, ce qui m'a permis de situer mon travail par rapport aux standards actuels et d'identifier des pistes d'amélioration.

L'utilisation de Papers with Code a enrichi mon projet en me donnant accès à une vaste base de connaissances et de ressources pratiques, m'aidant ainsi à concevoir des solutions performantes pour la création de présentations immersives.

4.1.3. Bibliothèques

Bitsandbytes

Définition : Une bibliothèque légère pour des calculs efficaces avec de grands modèles.

Utilisation : Utilisée pour quantifier et optimiser de grands modèles de langage pour des performances efficaces sur du matériel grand public.

Datasets

Définition : Une bibliothèque Hugging Face pour la gestion des ensembles de données.

Utilisation : Permet d'accéder à un large éventail d'ensembles de données pour l'entraînement et l'évaluation de modèles d'apprentissage automatique, y compris des méthodes de prétraitement et de manipulation des données.

Accelerate

Définition : Une bibliothèque permettant d'exécuter les modèles PyTorch sur plusieurs appareils (CPU/GPU) avec un minimum de changements dans le code. Utilisation : Simplifie l'apprentissage des modèles dans les environnements distribués, en supportant les configurations de précision mixte et multi-GPU.

Transformers

Définition : Une bibliothèque Hugging Face fournissant des modèles d'apprentissage automatique de pointe, en particulier des transformateurs. Utilisation : Couramment utilisée pour implémenter et affiner les modèles de transformateurs pour des tâches telles que la classification et la génération de textes.

Torch

Définition : PyTorch, une bibliothèque d'apprentissage automatique open-source. Utilisation : Fournit des outils pour l'apprentissage profond et le calcul tensoriel, largement utilisés pour l'entraînement des réseaux neuronaux.

Tensorboard

Définition : Outil de visualisation des expériences d'apprentissage automatique.
Utilisation : Enregistre et visualise les mesures pendant l'apprentissage du modèle, ce qui facilite le débogage et l'optimisation.

Evaluate
Définition : Une bibliothèque "Hugging Face" pour l'évaluation des modèles d'apprentissage automatique. Utilisation : Fournit des méthodes standardisées pour évaluer la performance des modèles sur différentes tâches et benchmarks.

Flash-attn
Définition : Flash Attention, un mécanisme d'attention optimisé pour les transformateurs.
Utilisation : Améliore l'efficacité et la vitesse des calculs d'attention dans les modèles de transformateurs, en particulier sur les GPU modernes.

Diffusers
Définition : Une bibliothèque pour les modèles de diffusion. Utilisation : Implémente des modèles génératifs basés sur la diffusion pour des tâches telles que la synthèse d'images.

huggingface_hub
Définition : Une bibliothèque pour interagir avec le Hugging Face Hub. Utilisation : Fournit des outils pour accéder, télécharger et gérer des modèles et des ensembles de données sur le Hugging Face Hub.

Langchain

Définition : Une bibliothèque pour la création d'applications utilisant de grands modèles de langage. Utilisation : Permet le développement d'applications complexes utilisant des modèles de langage, y compris l'intégration et l'orchestration de divers composants.

SQLAlchemy

Définition : Une bibliothèque ORM pour Python qui facilite les interactions avec les bases de données. Utilisation : Gère les opérations de base de données, définit les modèles de table et gère les sessions avec MySQL.

Passlib

Définition : Une bibliothèque pour la gestion des mots de passe en Python. Utilisation : Fournit des outils pour le hachage et la vérification sécurisés des mots de passe.

GPT4All

Définition : Une bibliothèque pour l'utilisation de modèles de langage, en particulier pour les tâches de génération de texte. Utilisation : Implémente des modèles pour la génération de langage naturel et les applications basées sur le texte.

Faiss-cpu

Définition : Facebook AI Similarity Search, une bibliothèque pour la recherche de similarités vectorielles. Utilisation : Recherche efficacement les vecteurs similaires, utile pour le traitement de grands ensembles de données textuelles.

python-dotenv

Définition : Une bibliothèque pour gérer les variables d'environnement à partir de fichiers .env.
Utilisation : Charge les variables d'environnement dans la configuration de l'application.

uvicorn
Définition : Serveur ASGI permettant de déployer des applications FastAPI. Utilisation : Servir les applications FastAPI en gérant les requêtes asynchrones.

rembg
Définition : Outil permettant de supprimer l'arrière-plan des images. Utilisation : Utilisé pour supprimer les arrière-plans des images, améliorant ainsi le contenu visuel.

Code Overview
mon code comprend divers composants permettant de travailler avec de grands modèles linguistiques, de générer et de traiter du texte et de gérer l'authentification. Il intègre plusieurs bibliothèques pour construire un système complet de génération et de gestion des présentations, notamment :

Modèles d'invite : Définissent la manière dont les entrées sont formatées et traitées pour différentes tâches (par exemple, génération de titres, résumé, génération d'images). Chaînes LLM : Mise en œuvre de la logique d'exécution des tâches du modèle de langage à l'aide des invites définies. Modèles pydantiques : Définir les structures de données pour le traitement des demandes et la gestion des utilisateurs. Authentification : Gérer l'authentification et l'autorisation des utilisateurs avec JWT et le hachage de mot de passe. Intégration de base de données : Utiliser SQLAlchemy pour les opérations de base de données et la gestion des sessions.

4.2. Outils d'industrialisation

4.2.1. Git

Figure 4.6: logo Git

Git est un système de contrôle de version distribué, libre et open source créée par Linus Torvalds en 2005. Il est devenu l'un des outils de gestion de code source les plus populaires et largement utilisé dans le développement de logiciels et de projets informatiques. Voici quelques-unes de ses fonctionnalités :

- Collaboration : Git est conçu pour faciliter la collaboration entre les développeurs. Il offre des outils pour partager des versions de code, gérer les conflits et gérer les autorisations d'accès.
- Distribution : Git est un système de contrôle de version distribué, ce qui signifie que toutes les copies de l'historique des versions sont stockées localement.
- Branchement et fusionnement : Git permet aux développeurs de travailler sur des fonctionnalités ou des corrections de bugs en créant des branches distinctes. Les modifications peuvent ensuite être fusionnées avec la branche principale en utilisant un processus de fusion.
- Révisions : les commits sont stockés sous forme de révisions, chacune contenant un ensemble de modifications de code. Git offre la possibilité d'explorer l'historique des versions et restaurer les révisions précédentes.

En ce qui concerne le flux de Git, je présente le schéma ci-dessous qui décrit l'ensemble des phases utilisées par Git pour envoyer les changements vers et les recevoir depuis le repository distant :

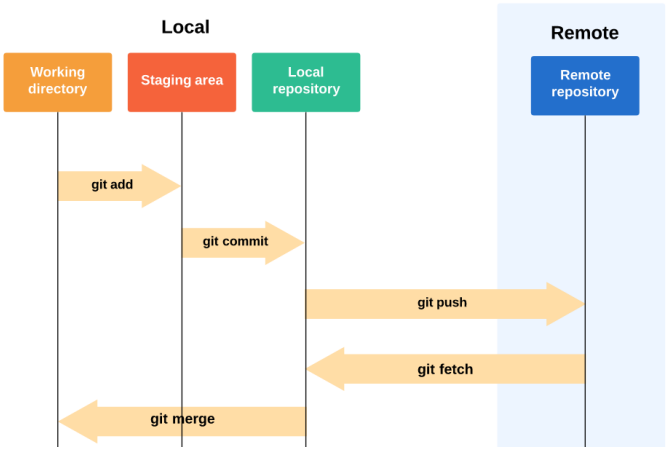

Figure 4.7: flux Git

Tout au long du flux Git, je fais des modifications dans le répertoire de travail, les déplace vers la zone de staging à l'aide de « git add » pour les préparer à la validation, les valide dans le référentiel local avec « git commit », puis pousse ces validations vers le repository distant à l'aide de « git push » pour les mettre à disposition des collaborateurs.

Pour extraire les modifications du repository distant vers le répertoire de travail, j'utilise d'abord « git fetch », qui apporte les modifications du référentiel distant vers le référentiel local. Ensuite, j'utilise « git merge » pour fusionner les modifications du référentiel local dans le répertoire de travail. Cela garantit que les dernières modifications du référentiel distant sont incorporées dans le répertoire de travail, ce qui permet la synchronisation.

4.2.2. GitHub

Figure 4.8: logo GitHub

GitHub est une plateforme de développement collaboratif qui permet aux développeurs de travailler ensemble sur des projets logiciels. Elle offre des fonctionnalités pour la gestion de code, le suivi des bugs, la planification des projets, et bien plus encore. Il se caractérise pase :

- Gestion de Code Centralisée : GitHub permet aux développeurs de suivre les modifications de code, de collaborer sur des projets, et de gérer des versions de manière efficace. Les utilisateurs peuvent cloner des dépôts, soumettre des modifications, et fusionner des demandes de tirage.

- Collaboration Facilitée : La plateforme offre des outils pour la collaboration en temps réel, y compris des discussions intégrées, des revues de code, et des notifications pour garder tout le monde informé des dernières mises à jour.
- Intégration Continue et Déploiement Continu : GitHub s'intègre avec des outils d'intégration continue et de déploiement continu, permettant aux équipes de tester et de déployer leur code automatiquement.
- Communauté et Apprendre : GitHub héberge une communauté active de développeurs qui partagent des connaissances, prodiguent des conseils, et contribuent à des projets open source.
- Sécurité et Confidentialité : GitHub offre des options de sécurité robustes, y compris l'authentification à deux facteurs, des permissions granulaires pour le code, et la possibilité de rendre des dépôts privés.

4.2.3. Visual Studio Code (VS Code)

Figure 4.9: logo de VScode IDEA

Visual Studio Code, communément appelé VS Code, est un éditeur de code source gratuit et open-source développé par Microsoft. Il est largement apprécié par les développeurs pour sa légèreté, sa flexibilité et son extensibilité, ce qui en fait un outil incontournable pour le développement logiciel moderne.

Caractéristiques Clés de VS Code

- Édition de Code Efficace : VS Code offre une expérience d'édition de code puissante avec des fonctionnalités telles que la coloration syntaxique, l'autocomplétion intelligente, le reformatage de code et la refactorisation. Ces fonctionnalités facilitent la lecture et l'écriture de code propre et maintenable.
- Extensibilité via les Extensions : L'une des forces majeures de VS Code réside dans son vaste écosystème d'extensions. Les développeurs peuvent personnaliser et améliorer les fonctionnalités de l'éditeur en installant des extensions pour différents langages de programmation, frameworks et outils. Cela permet d'adapter l'environnement de développement aux besoins spécifiques de chaque projet.

- Intégration du Contrôle de Version : VS Code s'intègre parfaitement avec des systèmes de contrôle de version tels que Git et SVN. Les développeurs peuvent effectuer des opérations de contrôle de version directement depuis l'éditeur, y compris le commit, le push, le pull et la gestion des branches, ce qui simplifie le suivi et la gestion des modifications de code.
- Debugging Intégré : L'éditeur fournit des outils de débogage intégrés qui permettent d'exécuter et de tester le code en temps réel. Les développeurs peuvent définir des points d'arrêt, inspecter les variables et parcourir le code pas à pas pour identifier et résoudre efficacement les bugs.
- Terminal Intégré : VS Code inclut un terminal intégré qui permet d'exécuter des commandes shell sans quitter l'environnement de développement. Cela facilite l'exécution de scripts, la gestion des packages et l'interaction avec divers outils de ligne de commande.

Utilisation de VS Code dans le Projet "GenerativeAI pour une Présentation Immersive"

Dans le cadre du projet *GenerativeAI pour une présentation immersive*, VS Code a été utilisé comme environnement de développement principal pour les raisons suivantes :

Développement Python Optimisé : Grâce aux extensions dédiées comme *Python* et *Pylance*, VS Code a offert un support complet pour le développement en Python, incluant l'autocomplétion, la détection d'erreurs en temps réel et le support pour les environnements virtuels, ce qui a grandement amélioré l'efficacité du développement des fonctionnalités d'IA du projet.

Gestion Facilitée des Projets : Les fonctionnalités de gestion de projets et de dossiers de VS Code ont permis d'organiser et de naviguer facilement entre les différents fichiers et modules du projet, assurant une structure de code claire et cohérente.

Collaboration et Contrôle de Version : L'intégration native avec Git a simplifié la collaboration entre les membres de l'équipe, permettant un suivi transparent des modifications de code, la gestion des branches de développement et la résolution efficace des conflits.

Debugging et Tests Efficaces : Les outils de débogage intégrés ont aidé à identifier et corriger rapidement les problèmes dans le code, assurant ainsi la stabilité et la fiabilité de l'application tout au long du cycle de développement.

4.2.4. Intégration de GitHub Copilot

Figure 4.10: Logo de GitHub Copilot

GitHub Copilot est un outil d'intelligence artificielle développé conjointement par GitHub et OpenAI, conçu pour assister les développeurs en suggérant des lignes ou des blocs de code contextuellement pertinents pendant l'écriture. Il utilise des modèles de langage avancés pour comprendre le contexte du code et proposer des complétions intelligentes.

Avantages de l'utilisation de GitHub Copilot dans le projet :

- Augmentation de la Productivité : Copilot a permis de réduire le temps d'écriture de code en suggérant rapidement des implémentations possibles basées sur les commentaires et les noms de fonctions, ce qui a accéléré le développement des fonctionnalités complexes du projet.
- Amélioration de la Qualité du Code : Les suggestions fournies par Copilot ont aidé à adopter de meilleures pratiques de codage et à éviter les erreurs courantes, contribuant ainsi à la production d'un code plus propre et plus maintenable.
- Facilitation de l'Apprentissage et de l'Exploration : Pour les aspects moins familiers du développement, Copilot a servi de guide en proposant des exemples de code pertinents, facilitant l'apprentissage de nouvelles bibliothèques ou techniques nécessaires à la réalisation du projet.
- Support Multilingue : Étant donné que le projet peut impliquer divers langages et technologies, Copilot a offert une assistance précieuse en fournissant des suggestions de code pour différents langages de programmation, assurant une cohérence et une intégration fluides entre les différents composants de l'application.

Mise en œuvre concrète dans le projet :

Lors du développement de l'application, Copilot a été utilisé pour :

- Générer des fonctions de traitement de données et de manipulation de texte nécessaires à la préparation du contenu des présentations immersives.
- Proposer des structures de code pour l'intégration avec des API externes, telles que celles utilisées pour la génération d'images et de textes via des modèles d'IA.
- Suggérer des tests unitaires pour vérifier la fonctionnalité et la robustesse du code développé.
- Offrir des exemples de configuration pour des outils et des frameworks utilisés dans le projet, comme FastAPI pour le développement backend et Streamlit pour la création de l'interface utilisateur.

○ Conclusion

Dans ce chapitre, j'ai examiné les principaux outils et technologies qui ont soutenu le développement de mon projet. Chaque outil a été choisi avec soin pour répondre aux besoins spécifiques du projet, qu'il s'agisse de la génération d'images ou de l'intégration de modèles de langage.

- Hugging Face m'a offert une plateforme ouverte pour explorer, intégrer et partager des modèles, en particulier pour la génération d'images. Cela m'a permis d'accéder à des modèles pré-entraînés et de les adapter à mes besoins, tout en partageant mes résultats avec la communauté.
- Google Colab s'est avéré être une ressource précieuse en offrant un environnement de développement collaboratif avec des capacités de calcul avancées. Cela m'a permis d'entraîner et d'expérimenter avec les modèles de manière efficace, sans avoir à déployer une infrastructure coûteuse.

L'utilisation conjointe de ces outils a non seulement facilité le développement, mais a également permis d'accélérer la mise en œuvre des fonctionnalités critiques, comme la génération d'images et le retrait des arrière-plans. Ces technologies se sont révélées essentielles pour assurer la qualité et la performance du projet final.

Chapitre 5. Réalisation et concrétisation

○ Introduction

Dans ce chapitre, je vais explorer la phase de réalisation concrète du projet intitulé "Generative AI pour une présentation immersive". Après avoir étudié les fondements théoriques et techniques de l'intelligence artificielle générative et les outils utilisés, il est temps de détailler la mise en œuvre pratique de ce projet.

L'objectif principal de ce projet est de simplifier la tâche des enseignants et des étudiants en automatisant la création de présentations visuelles, en leur faisant gagner du temps lors de la recherche d'informations et d'images. Grâce à l'IA générative, il devient possible de générer rapidement du contenu pertinent et des visuels adaptés aux sujets abordés. L'intégration de ces technologies permet de réduire la charge de travail liée à la préparation des présentations, en optimisant à la fois le contenu textuel et visuel.

Ce chapitre détaillera les étapes du développement du projet, en mettant l'accent sur les modules de génération d'images et de textes, l'automatisation des tâches répétitives, et les solutions trouvées pour aider les utilisateurs à obtenir des résultats de haute qualité sans effort. Je discuterai également des défis rencontrés et des résultats obtenus en facilitant l'utilisation de l'IA pour la création de présentations.

5.1. Création du Backend et Authentification avec FastAPI

Dans le cadre du projet **"Generative AI pour une Présentation Immersive"**, la création d'un backend robuste et sécurisé était essentielle pour la gestion des utilisateurs et des interactions en temps réel avec les modèles d'IA. J'ai choisi d'utiliser FastAPI pour développer le backend en raison de ses capacités exceptionnelles en matière de performance et de simplicité dans la création d'API modernes.

5.1.3. Création du Backend avec FastAPI

FastAPI est un framework moderne, rapide et facile à utiliser pour construire des API en Python. Il offre des performances élevées grâce à une architecture asynchrone, ce qui le rend particulièrement adapté pour des applications nécessitant une gestion efficace des requêtes concurrentes. Voici les étapes clés de la création du backend :

- Configuration du Projet : J'ai d'abord configuré un projet FastAPI en créant un environnement virtuel et en installant les dépendances nécessaires via `pip`. Les principales bibliothèques utilisées incluent `fastapi`, `uvicorn` pour le serveur ASGI, et `sqlalchemy` pour la gestion de la base de données.
- Définition des Modèles et des Schémas : J'ai défini des modèles SQLAlchemy pour représenter les données des utilisateurs, telles que les informations d'identification et les métadonnées du compte. Les schémas Pydantic ont été utilisés pour valider les données d'entrée et de sortie des API.
- Implémentation des Routes et des Endpoints : Les routes de l'API ont été mises en place pour gérer les opérations CRUD (Create, Read, Update, Delete) sur les données des utilisateurs, y compris la création de nouveaux utilisateurs, la gestion des informations de profil et la suppression des comptes.

5.1.4. Authentification et Sécurité

L'authentification des utilisateurs est cruciale pour sécuriser les interactions avec l'application. Pour cela, j'ai implémenté un système d'authentification complet utilisant OAuth2 et JWT (JSON Web Tokens). Voici comment j'ai abordé cette partie :

- Gestion des Utilisateurs : J'ai créé un endpoint pour l'inscription des utilisateurs, avec un système de hachage des mots de passe pour garantir la sécurité des informations sensibles. Les mots de passe sont hachés à l'aide de la bibliothèque `bcrypt`.

- Authentification avec JWT : Lors de la connexion, les utilisateurs reçoivent un token JWT, qui est ensuite utilisé pour authentifier les requêtes API. Le token inclut des informations sur l'utilisateur et a une durée de vie limitée, assurant ainsi une sécurité supplémentaire.
- Sécurisation des Endpoints : Les endpoints nécessitant une authentification sont protégés par des dépendances FastAPI qui vérifient la validité du token JWT avant d'accéder aux données protégées.

5.1.5. Méthodes Asynchrones pour une Meilleure Scalabilité

L'une des caractéristiques clés de FastAPI est son support pour les méthodes asynchrones, permettant de gérer efficacement un grand nombre d'utilisateurs simultanément. J'ai utilisé des méthodes asynchrones pour :

- Optimisation des Requêtes : Les requêtes à la base de données et aux services externes sont traitées de manière asynchrone, réduisant ainsi les temps d'attente et améliorant les performances globales de l'application.
- Gestion des Concurrences : Grâce aux fonctionnalités asynchrones de FastAPI, j'ai pu concevoir une API capable de gérer plusieurs connexions simultanées sans compromettre la réactivité du système.
- Réponses en Temps Réel : Les opérations telles que la génération de texte et d'images en temps réel sont traitées de manière asynchrone, garantissant une expérience utilisateur fluide et réactive même lorsque de nombreux utilisateurs sont connectés simultanément.

En résumé, la création du backend avec FastAPI et la mise en œuvre d'un système d'authentification sécurisé ont été des étapes cruciales pour assurer le bon fonctionnement et la sécurité de l'application. L'utilisation de méthodes asynchrones a permis de garantir une

performance optimale, même dans des conditions de charge élevée, contribuant ainsi au succès global du projet **"Generative AI pour une Présentation Immersive"**.

5.2. Implémentation de l'API Groq et Création des Différents Endpoints .

L'intégration de l'API Groq a joué un rôle clé dans le projet "Generative AI pour une Présentation Immersive", permettant de générer des textes, résumés, et images basés sur le contenu fourni par l'utilisateur. L'objectif était de créer plusieurs endpoints pour faciliter la génération automatique de contenus adaptés aux présentations immersives. FastAPI a permis de structurer l'API et d'utiliser des méthodes asynchrones pour offrir une réactivité optimale.

```python
from langchain_groq import ChatGroq

groq_api_key = os.getenv('GROQ_API_KEY')
# llm=ChatGroq(groq_api_key=groq_api_key,model_name="llama3-70b-8192")
llm = ChatGroq(groq_api_key=groq_api_key, model_name="mixtral-8x7b-32768")

```

Figure 5.1: code d'intégration de l'API Groq

5.2.3. Génération de Contenus Basée sur les Chaînes LangChain

Plusieurs endpoints ont été développés pour des tâches spécifiques telles que la génération de texte de diapositives, de titres, de résumés, et d'images. Voici un aperçu des fonctionnalités implémentées dans ces endpoints :

Génération de texte pour une diapositive : Cet endpoint permet de générer du contenu textuel pour une diapositive en fonction du contenu fourni par l'utilisateur. En fonction de la langue spécifiée (français ou anglais), des chaînes distinctes sont utilisées pour produire un texte adapté.

Génération de contenu Wikipedia : Cet endpoint permet d'extraire des informations de Wikipedia en fonction du prompt donné, puis génère un texte de diapositive ainsi qu'un titre. Il utilise une recherche sur Wikipedia (en français ou en anglais selon la langue) et combine les résultats avec des chaînes LangChain.

5.2.4. Génération de Résumés, Textes Courts et Longs

Trois endpoints ont été créés pour générer des résumés, raccourcir du texte ou encore allonger un texte donné. Ces fonctionnalités sont particulièrement utiles pour adapter le contenu d'une présentation à différents formats et longueurs :

Génération de Résumés : Cet endpoint permet de créer un résumé du texte fourni.

```python
# Generate summary endpoint
@app.post("/generate-summary")
async def generate_summary(request: RequestWithToken, db: Session = Depends(get_db)):
    try:
        # Validate the token
        current_user = await get_current_user(token=request.token, db=db)
        await get_current_active_user(current_user=current_user)

        # Run the summary chain
        summarize = summarize_chain.run(request.prompt)
        return {"summarize": summarize}
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

```

Figure 5.2: Code Génération de Résumés

Raccourcir et Allonger du Texte : Ces deux endpoints permettent respectivement de raccourcir un texte ou de l'allonger en fonction des besoins de l'utilisateur.

5.2.5. Génération de Prompts pour Images et Objets

Deux endpoints ont été créés pour générer des prompts pour des images et des objets en fonction du contenu des diapositives. Ces fonctionnalités permettent de créer des descriptions adaptées pour des générateurs d'images et d'objets basés sur les titres et le texte des présentations.

Génération de prompts pour des images : Utilisant les chaînes LangChain, cet endpoint génère un prompt pour créer une image en fonction du titre de la présentation et du texte de la diapositive. Génération de prompts pour des objets : Similaire à la génération de prompts pour les images, cet endpoint se concentre sur la création de descriptions adaptées pour générer des images d'objets spécifiques à partir du contenu des diapositives.

```python

@app.post("/generate-image-promt")
async def generate_image(request: imageRequest ,current_user: UserModel = Depends(get_current_active_user)):
    try:
        image = generate_image_chain.run(presentation_title=request.presentation_title, slide_text=request.slide_text)
        return {"image": image}
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

@app.post("/generate-object-promt")
async def generate_object(request: imageRequest ,current_user: UserModel = Depends(get_current_active_user)):
    try:
        image = generate_image_chain.run(presentation_title=request.presentation_title, slide_text=request.slide_text)
        return {"image": image}
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))
```

Figure 5.3: Code Génération de prompts pour des objets

5.2.6. Liste des Titres et Génération de Titres

Pour améliorer la structure et l'organisation des diapositives, deux endpoints ont été développés pour générer des titres multiples à partir du contenu fourni, ainsi qu'un titre unique pour chaque diapositive :

Génération d'une liste de titres : Cet endpoint crée une liste de titres séparés à partir d'un texte source. Il utilise la chaîne de génération de titres adaptée à la langue choisie.

Génération d'un titre unique : Utilisé pour générer un titre clair et concis pour une diapositive spécifique.

```python

# Generate summary endpoint
@app.post("/generate-summary")
async def generate_summary(request: RequestWithToken, db: Session = Depends(get_db)):
    try:
        # Validate the token
        current_user = await get_current_user(token=request.token, db=db)
        await get_current_active_user(current_user=current_user)

        # Run the summary chain
        summarize = summarize_chain.run(request.prompt)
        return {"summarize": summarize}
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

# Shorten text endpoint
@app.post("/shorten-text")
async def shorten_text(request: RequestWithToken, db: Session = Depends(get_db)):
    try:
        # Validate the token
        current_user = await get_current_user(token=request.Token, db=db)
        await get_current_active_user(current_user=current_user)

        # Run the shorten text chain
        shorten_text = shorten_text_chain.run(request.prompt)
        return {"shorten_text": shorten_text}
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

# Lengthen text endpoint
@app.post("/lengthen-text")
async def lengthen_text(request: RequestWithToken, db: Session = Depends(get_db)):
    try:
        # Validate the token
        current_user = await get_current_user(token=request.Token, db=db)
        await get_current_active_user(current_user=current_user)

        # Run the lengthen text chain
        lengthen_text = lengthen_text_chain.run(request.prompt)
        return {"lengthen_text": lengthen_text}
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

```

Figure 5.4: Code Génération d'un titre unique

Ces différentes implémentations d'endpoints ont permis de rendre le projet "Generative AI pour une Présentation Immersive" extrêmement polyvalent, capable de gérer la génération de textes, d'images et de contenus structurés en fonction des besoins des utilisateurs. La flexibilité offerte par l'API Groq et les chaînes LangChain a permis de créer une interface conviviale pour les utilisateurs finaux, tout en assurant une scalabilité pour de multiples utilisateurs.

5.3. Système RAG (Retrieval-Augmented Generation)

Le système RAG (Retrieval-Augmented Generation) a été intégré dans votre projet pour améliorer la génération de contenu en combinant la récupération d'informations pertinentes et la génération de texte basée sur ces informations. Voici un aperçu détaillé de chaque composant du système RAG que vous avez mis en place.

5.3.3. Endpoint RAG-HTTP

Cet endpoint est conçu pour traiter des requêtes HTTP en utilisant des documents chargés à partir d'un lien web. Il intègre un mécanisme de récupération et de génération de texte basé sur le contenu des documents.

5.3.4. Endpoint RAG-Wiki

Cet endpoint utilise du contenu Wikipedia pour générer un bloc de texte pour une diapositive. Le contenu Wikipedia est d'abord récupéré, puis traité pour créer des documents pertinents pour le modèle RAG

5.3.5. Endpoint Traitement de PDF avec RAG

Cet endpoint traite les fichiers PDF pour extraire le texte, le diviser en documents, et les utiliser pour générer des réponses en fonction du contenu extrait.

5.3.6. Explication du Système RAG

Le système RAG (Retrieval-Augmented Generation) combine des techniques de récupération d'informations avec des modèles de génération de texte pour fournir des réponses plus précises et informées à partir d'une large base de documents. Voici les étapes et technologies utilisées dans notre projet :

Chargement et Prétraitement des Données

Les documents sont chargés depuis différentes sources telles que des pages web, Wikipedia, des fichiers PDF, et d'autres bases de données. Ces documents sont ensuite divisés en fragments plus petits pour être facilement traités par les modèles de génération. Cela permet d'améliorer l'efficacité du système tout en assurant que les informations pertinentes sont récupérées.

Vectorisation des Données

Une fois les documents fragmentés, ils sont transformés en vecteurs à l'aide de l'index FAISS (Facebook AI Similarity Search). FAISS est un outil puissant pour effectuer des recherches vectorielles à grande échelle. Dans notre projet, il est utilisé pour indexer et rechercher rapidement des morceaux de texte pertinents en fonction des requêtes de l'utilisateur.

- Vectorisation utilisée : FAISS (utilisé pour convertir les documents en vecteurs afin de permettre une récupération efficace des informations).
- Modèle d'embeddings utilisé(all-minilm-l6-v2) :J'utilise des embeddings personnalisés via embeddings2 pour transformer les textes en représentations vectorielles.

```python
from langchain_community.embeddings import GPT4AllEmbeddings

# Initialize embeddings
model_name = "all-MiniLM-L6-v2.gguf2.f16.gguf"
gpt4all_kwargs = {'allow_download': 'True'}
embeddings2 = GPT4AllEmbeddings(
    model_name=model_name,
    gpt4all_kwargs=gpt4all_kwargs
)
```

Figure 5.5: Modèle d'embeddings utilisé

Chaîne de Documents (Document Chain)

Après la vectorisation, une chaîne de documents est créée en utilisant un modèle de génération de langage (LLM). Ce modèle LLM est chargé de formuler des réponses basées sur les documents récupérés à l'étape précédente.

Traitement des Requêtes

Lorsqu'une requête est soumise, elle est traitée via une chaîne de récupération. Cette chaîne utilise l'index FAISS pour récupérer les morceaux de texte les plus pertinents et les combiner avec le modèle LLM pour générer une réponse. Le modèle de génération est guidé par des prompts adaptés pour produire des réponses cohérentes et informatives.

Gestion des Langues

Le système est capable de gérer plusieurs langues, notamment le français et l'anglais. Selon la langue spécifiée par l'utilisateur, un modèle de prompt spécifique est utilisé pour générer des réponses pertinentes dans la langue demandée.

Nettoyage des Données :

Pour optimiser l'utilisation des ressources, un nettoyage automatique des données et des objets temporaires est effectué après chaque traitement. Cela permet de libérer les ressources et d'éviter les fuites de mémoire, garantissant ainsi des performances optimales.

Technologies Utilisées pour le Prétraitement :

Text Splitting (Découpage du Texte)

Pour le découpage du texte, j'utilise RecursiveCharacterTextSplitter. Cet outil divise les documents en morceaux gérables de manière à maintenir un certain chevauchement entre les fragments, assurant ainsi que les idées complètes ne sont pas interrompues.

- Outil de découpage : `RecursiveCharacterTextSplitter`
- Taille des morceaux : 1000 caractères avec un chevauchement de 200 caractères.

PDF Processing (Traitement des PDF)

Pour le traitement des documents PDF, j'utilise PyPDF2 pour extraire le texte des fichiers PDF. Ce texte est ensuite traité de manière similaire aux autres documents en le divisant en fragments plus petits.

- Lecteur PDF utilisé : `PyPDF2`
- Processus : Extraction du texte page par page et création de documents à partir de ce texte.

HTTP Scraping (Récupération de Contenu Web)

Pour récupérer du contenu depuis le web, notamment des articles Wikipedia, j'utilise des appels HTTP avec des API adaptées pour obtenir le texte brut. Ce texte est ensuite traité pour générer des réponses à partir des informations récupérées.
Scraping de contenu web : Utilisation d'API pour accéder à des contenus comme Wikipedia et d'autres sources.

Résumé du Processus :

- Chargement des Données : Les données sont récupérées depuis des fichiers PDF ou des pages web.
- Découpage et Vectorisation : Le contenu est divisé et vectorisé à l'aide de FAISS pour permettre une recherche efficace.

- Chaîne de Documents : Le modèle LLM génère des réponses en fonction du texte récupéré.
- Génération et Nettoyage : Après génération de la réponse, le système nettoie les données utilisées pour libérer les ressources.

Cette architecture me permet d'offrir un système robuste et efficace pour le RAG, capable de gérer des requêtes complexes tout en intégrant une gestion multilingue et des ressources variées comme les PDF et les contenus en ligne.

5.4. image generation

5.4.3. EndPoints

`generate_image`: Génère une image unique basée sur l'invite de saisie.

`generate_image_slide`: Génère une image à partir d'invites textuelles pour les diapositives.

`generate_object_slide`: Génère une image avec l'objet principal, supprime l'arrière-plan et la renvoie au format PNG.

Processus de base de la génération d'images :

`generate_image_f`: Utilise un client pour interagir avec un modèle (probablement la diffusion stable) afin de générer une image basée sur l'invite d'entrée.

L'image est d'abord générée sous la forme d'un `.webp` puis converti en `.png` en utilisant le format `PIL`.

Résumé du flux de travail pour la génération d'images :

Image Prompting: Pour les trois points d'accès, la demande prend une forme de `GenerateImageRequest` contenant l'invite et le jeton. Cette invite est transmise à un client, qui interagit probablement avec un modèle tel que Stable Diffusion, afin de générer une image.

Token Authentication: Chaque demande vérifie d'abord l'utilisateur actuel et son statut actif, en utilisant la fonction `get_current_user` et `get_current_active_user`, garantir un accès authentifié.

Image Processing: Après avoir généré l'image, l'objet(`BytesIO`) est traité ultérieurement, par exemple en supprimant l'arrière-plan ou en le convertissant dans différents formats.

5.4.3.1. Génération d'images de diapositives

Pour les `generate_image_slide` l'image est générée sur la base du contenu de la diapositive et traitée ultérieurement à l'aide d'une chaîne, ce qui peut impliquer la génération d'un texte descriptif ou l'affinement de l'invite pour un meilleur alignement de l'image de la diapositive.

```python

@app.post("/generate-image-slide")
async def generate_image_slide(request: GenerateImageRequest , db: Session = Depends(get_db)):
    try:
        current_user = await get_current_user(token=request.Token, db=db)
        await get_current_active_user(current_user=current_user)
        image_data = generate_image_f(request)
        image = generate_image_chain.run( slide_text=request.prompt)
        req=GenerateImageRequest(prompt=image)
        image_data = generate_image_f(req)
                # Convert the BytesIO image to base64
        # output_bytes = base64.b64encode(image_data.getvalue()).decode("utf-8")

        # # Return the base64-encoded image
        # return JSONResponse(content={"image": output_bytes})
        return StreamingResponse(image_data, media_type="image/png", headers={"Content-Disposition": "attachment; filename=output.png"})
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))


```

Figure 5.6: code generate_image_slide

5.4.3.2. *Traitement d'images basé sur les objets*

La fonction generate_object_slide va un peu plus loin en générant une image de l'objet principal, puis le traiter pour supprimer l'arrière-plan à l'aide de la fonction `remove()`, et renvoie le fichier PNG sans arrière-plan.

```python

@app.post("/generate-object-slide")
async def generate_object_slide(request: GenerateImageRequest ,db: Session = Depends(get_db)):
    try:
        current_user = await get_current_user(token=request.Token, db=db)
        await get_current_active_user(current_user=current_user)

        # Generate the object prompt

        # image = generate_object_chain.run(presentation_title=request.presentation_title, slide_text=request.slide_text)
        image = generate_object_chain.run(slide_text=request.prompt)

        # Create the image generation request
        req = GenerateImageRequest(prompt=image)

        # Generate the image
        image_data = generate_image_f(req)

        # Check if the image_data is a BytesIO object
        if not isinstance(image_data, BytesIO):
            raise ValueError("Generated image data is not a BytesIO object")

        # Convert BytesIO to PIL Image
        pil_image = PILImage.open(image_data)

        # Remove the background from the image
        output_image = remove(pil_image)

        # Save the output image to BytesIO
        output_bytes = BytesIO()
        output_image.save(output_bytes, format="PNG")
        output_bytes.seek(0)
        # Convert the BytesIO image to base64
        output_bytes = base64.b64encode(output_bytes.getvalue()).decode("utf-8")

        # Return the base64-encoded image
        return JSONResponse(content={"image": output_bytes})
        # Return the image with background removed
        # return StreamingResponse(output_bytes, media_type="image/png", headers={"Content-Disposition": "attachment; filename=output.png"})
    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

```

Figure 5.7: code generate_object_slide

5.4.4. Avantages de ce système

Génération d'images dynamiques :

- Le système génère dynamiquement des images en fonction du texte saisi, ce qui peut s'avérer très utile dans les applications où le contenu visuel doit être créé à la volée (par exemple, diapositives, présentations ou expériences immersives).

Personnalisation en fonction des données fournies :

- En tirant parti des chaînes (telles que `generate_image_chain` et `generate_object_chain`), vous pouvez personnaliser le processus de génération d'images. Par exemple, la génération d'invites textuelles pour les diapositives et les objets permet de s'assurer que les images sont adaptées au contexte spécifique.

Traitement avancé des images :

- Background Removal: Le système peut traiter les images pour supprimer les arrière-plans (via remove()) et les convertir au format PNG. Cette fonction est utile pour isoler les objets principaux et manipuler les images pour des présentations ou des applications web.
- Base64 Encoding: L'option permettant de renvoyer les images sous forme de chaînes encodées en base64 facilite l'intégration dans les frameworks frontaux et les applications web où les images peuvent être directement incorporées.

Évolutivité avec une diffusion stable :

- Modèle de diffusion stable : En interagissant avec les modèles de diffusion stable (e.g., `stabilityai/stable-diffusion-3-medium`),Le système s'appuie sur la technologie descapacités de génération d'images à la pointe de la technologie. Cela

permet de garantir la qualité des images générées.reste élevée, tout en offrant la possibilité d'ajuster des paramètres tels que la semence, la largeur, la hauteur et l'échelle d'orientation.

Optimisation des performances :

StreamingResponse: En utilisant `StreamingResponse` pour l'envoi des images, le système peut efficacement renvoyer les images générées au client sans avoir à les charger complètement en mémoire. Cette fonction est essentielle pour traiter des images de grande taille ou de nombreuses demandes simultanées.

Conclusion

Mon stage au sein de Verve Technology, centré sur le développement d'un système de génération de présentations immersives en utilisant des techniques avancées d'intelligence artificielle, en particulier des modèles génératifs, a été une expérience profondément formatrice et enrichissante. Ce projet m'a permis de plonger dans les technologies d'IA de pointe, telles que les modèles de langage de grande taille (LLM) et les modèles de génération d'images, et de les appliquer à des défis réels liés à l'automatisation et à la génération de contenus pour des présentations.

La collaboration avec une équipe pluridisciplinaire, composée de chercheurs en IA, de développeurs web et d'ingénieurs en données, a élargi ma compréhension des complexités et du potentiel des solutions basées sur l'IA dans un contexte professionnel. Cette collaboration m'a permis de surmonter les défis techniques et pratiques liés à l'intégration de ces modèles dans des applications web, notamment pour la génération de texte et d'images.

Parmi les principaux défis rencontrés, j'ai dû gérer et optimiser les différents modèles d'IA, notamment en conciliant les performances des modèles avec l'efficacité computationnelle. Le fine-tuning et l'inférence avec de grands modèles, particulièrement sur CPU, ont nécessité une gestion rigoureuse des ressources et des stratégies d'optimisation, ce qui m'a amené à expérimenter des techniques telles que la quantification des modèles, l'ingénierie des prompts et l'amélioration des chaînes de génération.

La génération de diapositives immersives a également présenté des défis uniques. Il s'agissait de trouver un équilibre entre la qualité esthétique des diapositives générées et la clarté du contenu. Cela a exigé des tests et ajustements répétés pour affiner les modèles. De plus, l'intégration des pipelines de génération de texte et d'images devait fonctionner de manière fluide pour produire des résultats de haute qualité, adaptés aux cas d'utilisation professionnels.

Malgré ces défis, ce projet m'a offert une opportunité précieuse d'élargir mon expertise technique, notamment dans des domaines tels que la génération augmentée par la récupération d'informations (RAG), le traitement de texte basé sur les LLM, et la génération d'images avec des modèles comme Stable Diffusion. Cette expérience a renforcé ma compréhension de l'importance de la flexibilité, de la scalabilité et de l'adaptabilité lors du déploiement de systèmes d'IA dans des environnements de production.

Références

Site Web VERVE TECHNOLOGIES	• https://verve.ma/
Paper with code text to code	• https://paperswithcode.com/task/text-to-code-generation
Stable Diffusion Clearly Explained!	• Stable Diffusion Clearly Explained! (readmedium.com)
Understanding the Evolution of Stable Diffusion Models	• Understanding the Evolution of Stable Diffusion Models (readmedium.com)
A Comprehensive History of Text-to-Image and Generative Imaging	• Creating Reality: A Comprehensive History of Text-to-Image and Generative Imaging (readmedium.com)
The Evolution of AI Image Generators	• The Evolution of AI Image Generators: From Inception to Modern-Day Applications (readmedium.com)
3 Tools to Generate PowerPoint with AI from Text	• 3 Tools to Generate PowerPoint with AI from Text | by Kevin Goedecke | Medium
How to Build: a Text-to-PowerPoint Application (LangChain, OpenAI, CopilotKit & Next.js)	How to Build: a Text-to-PowerPoint Application (LangChain, OpenAI, CopilotKit & Next.js) - DEV Community
Ressources visuelles :	Chatbots with RAG: LangChain Full Walkthrough How to scrape the web for LLM in 2024: Jina AI (Reader API), Mendable (firecrawl) and Scrapegraph-ai Gen AI Course | Gen AI Tutorial For Beginners LangChain GEN AI Tutorial – 6 End-to-End Projects using OpenAI, Google Gemini Pro, LLAMA2 Ollama - Local Models on your machine Build a Large Language Model AI Chatbot using Retrieval Augmented Generation LangChain & Hugging Face - Run Any Language Model Locally (Code Walkthrough)

Conclusion

This project focused on developing a comprehensive backend system using FastAPI for authentication and interaction. JWT-based authentication was implemented to ensure secure access for users, creating a robust login and token management system.
Additionally, we integrated LLM LLaMA 3.1 using Groq to enable fast and efficient interaction with users. This allowed us to enhance text generation capabilities and make user interaction smoother.
To further enrich model outputs, we implemented Retrieval-Augmented Generation (RAG), enabling users to generate text based on PDF documents or HTTP links with web scraping, thus providing more contextual and relevant results.
For image generation, we employed Stable Diffusion 1.4, integrated through Hugging Face Spaces, allowing users to create high-quality images efficiently. The combination of text generation, RAG, and image generation provided a powerful toolset for users to produce presentations quickly and with high relevance, benefiting educators, students, and professionals alike.
This project demonstrated how the synergy of advanced language models and image generation can streamline content creation and reduce the time spent on manual tasks.

Conclusion

Ce projet a consisté à développer un système backend complet en utilisant FastAPI pour l'authentification et l'interaction. L'authentification basée sur JWT a été mise en œuvre pour assurer un accès sécurisé aux utilisateurs, en créant un système de gestion des connexions et des jetons robuste.
De plus, nous avons intégré le modèle LLM LLaMA 3.1 en utilisant Groq pour permettre une interaction rapide et efficace avec les utilisateurs, ce qui a amélioré les capacités de génération de texte et rendu l'interaction utilisateur plus fluide.
Pour enrichir davantage les sorties des modèles, nous avons mis en place la génération augmentée par récupération (RAG), permettant aux utilisateurs de générer du texte basé sur des documents PDF ou des liens HTTP avec du web scraping, fournissant ainsi des résultats plus contextuels et pertinents.
Pour la génération d'images, nous avons utilisé Stable Diffusion 1.4, intégré via Hugging Face Spaces, permettant aux utilisateurs de créer des images de haute qualité de manière efficace.
La combinaison de la génération de texte, du RAG, et de la génération d'images a fourni un ensemble d'outils puissant permettant aux utilisateurs de produire des présentations rapidement et avec une grande pertinence, au bénéfice des enseignants, étudiants et professionnels.
Ce projet a démontré comment la synergie entre les modèles de langage avancés et la génération d'images peut rationaliser la création de contenu et réduire le temps consacré aux tâches manuelles.