République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Béjaïa

Université de Béjaïa

Faculté des Sciences Exactes Département Informatique

Mémoire présenté en vue de l'obtention du

Master 2 Intelligence Artificielle

Développement d'un chatbot basé sur des méthodes IA

Cas d'étude : Département d'informatique, Université de A/Mira, Bejaia

Auteur : Céline AHMIM

Encadrants : Dr. Nabil DJEBARI
Dr. Feriel KHENNOUCHE

– Juin 2025

Remerciements

Je tiens à exprimer ma profonde reconnaissance à Dr. Nabil Djebari et Dr. Feriel Khannouche, dont lécoute attentive, les conseils éclairés et la rigueur bienveillante ont été des repères essentiels tout au long de ce travail. Leur encadrement a non seulement guidé mes choix méthodologiques, mais aussi nourri ma réflexion avec exigence et bienveillance.

Ma gratitude s'étend également à l'ensemble des enseignants du département dinformatique, pour la richesse des savoirs transmis et la passion partagée au fil des années. Enfin, je remercie toutes celles et ceux qui, dans l'ombre ou la lumière, m'ont soutenue et encouragée durant ce parcours exigeant mais formateur.

Dédicace

Je tiens à dédier le fruit de mes cinq années détudes universitaires à :

À ma mère, exemple de résilience, qui n'a jamais hésité à braver les normes pour que je ne manque de rien.

À mon père, qui a tout quitté pour nous offrir, à mes soeurs et moi, un avenir digne.

À mes soeurs jumelles, pour leur présence fidèle et cette admiration silencieuse qui me pousse a etre la meilleure grande soeur possible.

À ma famille maternelle, toujours là, sans condition ni absence.

À ma grand-mère maternelle, source de tendresse et de lucidité.

À mes oncles maternels tonton Mourad, Ahcene et Hamid pour avoir été les hommes de l'ombre, toujours présents, toujours fiables.

À mes tantes maternelles, qui ont su être mes alliées bien avant dêtre mes aînées.

Et à mon ami denfance, Aghiles Baouane, parti trop tôt. Ce diplôme, je ne le reçois pas seule. Il est aussi à toi.

Résumé

Dans un contexte de surcharge des demandes répétitives, ce mémoire présente un assistant conversationnel destiné à automatiser les FAQ du département informatique de l'Université de Béjaïa.

L'approche repose sur une architecture hybride combinant Retrieval-Augmented Generation (RAG) et le fine-tuning de GPT-3.5-turbo, sappuyant à la fois sur un corpus réglementaire fixe et sur des données tabulaires dynamiques, facilement mises à jour par des agents non techniques du département. Quatre configurations ont été testées selon une méthodologie itérative.

L'approche finale atteint 87% dExact Match, offrant un excellent compromis entre performance et coût, nettement plus avantageux que les solutions basées sur GPT-4 ou Claude 3.

Mots-clés : Chatbot éducatif, RAG, GPT3.5, FAQ automatisée, Support universitaire, Intelligence artificielle conversationnelle

Abstract

In a context of increasing repetitive inquiries, this thesis presents a conversational assistant designed to automate the FAQ management of the Computer Science department at the University of Béjaïa.

The approach relies on a hybrid architecture combining Retrieval-Augmented Generation (RAG) and fine-tuning of GPT-3.5-turbo, using both a fixed regulatory corpus and dynamic tabular data that can be easily updated by non-technical staff. Four configurations were tested through an iterative methodology.

The final approach achieves 87% Exact Match, offering an excellent performance-to-cost ratio, significantly more advantageous than solutions based on GPT-4 or Claude 3.

Keywords : Educational chatbot, RAG, GPT-3.5, Automated FAQ, University support, Conversational artificial intelligence

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Les départements informatiques universitaires sont confrontés à une surcharge croissante de demandes répétitives, principalement liées à la gestion des mots de passe, des comptes utilisateurs et aux questions fréquentes. Ces requêtes simples mais chronophages mobilisent fortement les ressources humaines, au détriment de tâches techniques plus complexes, et allongent les délais de traitement. Par exemple, la Thompson Rivers University a enregistré 44.244 tickets en un an, dont 42% concernaient uniquement les réinitialisations de mots de passe [1], tandis qu'au Luther College, le délai moyen de résolution reste bloqué à 13 heures malgré les efforts d'optimisation [2].

L'absence dune base de connaissances centralisée aggrave la situation : à la Gallaudet University, bien que 80% des demandes portent sur cinq catégories récurrentes, labsence de workflow unifié entraîne des divergences notables dans les réponses fournies [3].

Dans ce contexte, les chatbots apparaissent comme une solution technologique pertinente. Définis comme "des programmes conçus pour simuler une conversation avec des utilisateurs humains, notamment via Internet" [4], ils utilisent le traitement du langage naturel et l'analyse des intentions pour automatiser les échanges. Leur intégration dans les systèmes universitaires permettrait de réduire jusquà 60% des tâches répétitives tout en maintenant un haut niveau de satisfaction utilisateur [4].

Ce mémoire s'inscrit dans cette dynamique en explorant la conception et le déploiement dun assistant conversationnel adapté au contexte du département informatique de lUniversité de Béjaïa. Il prend en compte les spécificités locales et les défis propres à linteraction homme-machine dans l'enseignement supérieur.

Il est structuré en trois chapitres principaux. Le premier chapitre retrace l'évolution des chatbots et dresse un état de l'art sur leur usage dans les établissements universitaires, en soulignant les limites des approches existantes. Le deuxième chapitre est consacré à la conception et au développement itératif du chatbot proposé, en détaillant l'architecture technique, les choix technologiques et les étapes de mise en uvre. Enfin, le troisième chapitre présente les résultats obtenus après l'implémentation, en évaluant la performance du système à travers des métriques objectives et des tests utilisateurs, tout en discutant des perspectives d'amélioration.

Chapitre 1

Etat de l'Art et Concepts des Chatbots

1.1 Introduction

Depuis l'avènement de l'informatique conversationnelle, le *chatbot* s'est imposé comme l'un des vecteurs privilégiés de l'interaction hommemachine. D'abord cantonné aux laboratoires de recherche, puis aux messageries grand public, il a progressivement pénétré les services clientèle, la santé, l'éducation et, plus récemment, la création de contenu. Aujourd'hui, dialoguer avec une intelligence artificielle fait partie des usages quotidiens de centaines de millions de personnes ; cette normalisation témoigne dune rupture technologique aussi profonde que celle qu'ont provoquée, en leur temps, le Web ou le smartphone.

1.2 Problématique

Malgré les progrès considérables réalisés dans le domaine des chatbots, leur mise en place dans des environnements académiques, notamment dans un département informatique, présente plusieurs défis. En effet, la création dun chatbot efficace nécessite une compréhension approfondie des besoins spécifiques des utilisateurs, ainsi quune capacité à intégrer des systèmes variés tout en garantissant une expérience utilisateur optimale.

Il est également crucial de concevoir un chatbot fiable, capable de fournir des informations exactes et à jour, sans générer d'hallucinations ni de réponses erronées. Pour cela, le système doit être régulièrement mis à jour en fonction des règlements internes du département, des directives de la faculté et des décisions gouvernementales relatives à l'enseignement supérieur. Cette exigence de fiabilité et de mise à jour continue représente un enjeu majeur pour assurer la pertinence et la confiance accordées au chatbot.

La question centrale est donc :

> **Comment concevoir un chatbot intelligent pour le département informatique qui réponde efficacement aux besoins des étudiants et du personnel, tout en garantissant une mise à jour continue des informations dynamiques par des agents non-techniques et une amélioration progressive des performances du système ?**

Pour répondre à cette problématique, plusieurs sous-questions doivent être abordées :

1. Quels sont les besoins spécifiques du département informatique en termes de communication et de support ?
2. Comment assurer continuellement la performance d'un chatbot selon les critères de couverture, fiabilité, réactivité ?
3. Quels mécanismes permettent d'assurer la mise à jour continue des informations de support étudiant par des agents non-techniques tout en garantissant une adaptation rapide aux changements de politiques et décisions départementales ?"

1.3 Objectifs et contributions

Ce mémoire a pour but de créer un chatbot sur mesure pour le département informatique, destiné à automatiser la gestion des questions fréquentes et à optimiser l'allocation des ressources humaines.

1.3.0.1 Objectif de la recherche

Développer un chatbot performant, capable de fournir des réponses précises et rapides aux demandes répétitives des utilisateurs.

1.3.0.2 Contributions principales

— **Analyse approfondie des besoins :** identification, classification et priorisation des questions les plus fréquentes du département.
— **Conception et développement dun chatbot intelligent :** solution adaptée au contexte académique.

— **Maintenance en temps réel du *dataset* :** intégration rapide des actualités et événements tels que les plannings dexamens, lorganisation des cours, TD et TP.
— **Système flexible pour l'administration :** possibilité dinsérer des données structurées ou semi-structurées sans contrainte stricte du format Q/R classique, facilitant la saisie et la mise à jour des informations.

Ces contributions visent à proposer une solution technologique fiable et évolutive, capable dautomatiser efficacement la diffusion de l'information tout en améliorant la satisfaction des utilisateurs et en optimisant les ressources humaines du département.

1.4 Positionnement de la recherche

Cette recherche est organisée de manière rigoureuse afin de comprendre les différentes approches existantes en conception de chatbots, d'évaluer les solutions actuelles dédiées aux FAQ et d'identifier les meilleures pratiques applicables au contexte éducatif. Plusieurs étapes et outils ont été mobilisés pour positionner clairement cette étude dans le paysage scientifique.

Pour bâtir une base documentaire solide, une sélection ciblée de mots-clés pertinents a été effectuée :

— Chatbot
— FAQ / Automated FAQ
— Educational Chatbot / University Chatbot
— Conversational Agent / Intelligent Tutoring System
— Natural Language Processing (NLP) Chatbot
— Virtual Assistant in Education / AI-powered Help Desk
— Dialogue Systems / Knowledge Base Chatbot
— Student Support Chatbot
— Machine Learning Chatbot / Chatbot for Higher Education

Cette démarche a orienté la recherche vers des articles scientifiques, des études de cas, des thèses et des projets similaires portant sur les systèmes conversationnels dans les environnements académiques.

1.5 Méthodologie de la revue de littérature

1.5.1 Sources de données

Pour collecter des informations fiables et actualisées, plusieurs bases de données et plateformes académiques ont été explorées :

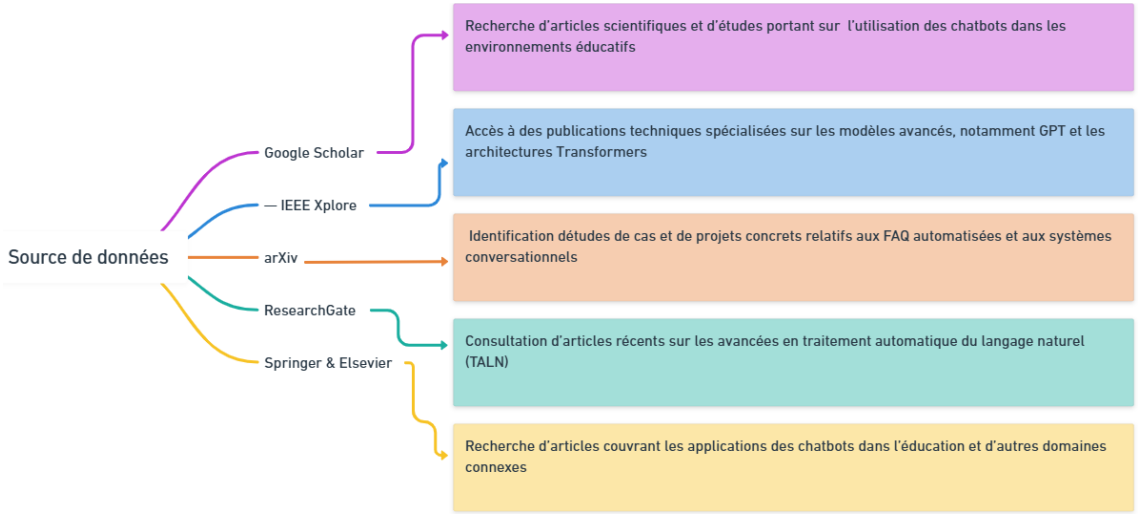

Figure 1.1 – Source de données

1.5.2 Méthodologie de recherche

La méthodologie adoptée s'est articulée autour de plusieurs étapes clés :

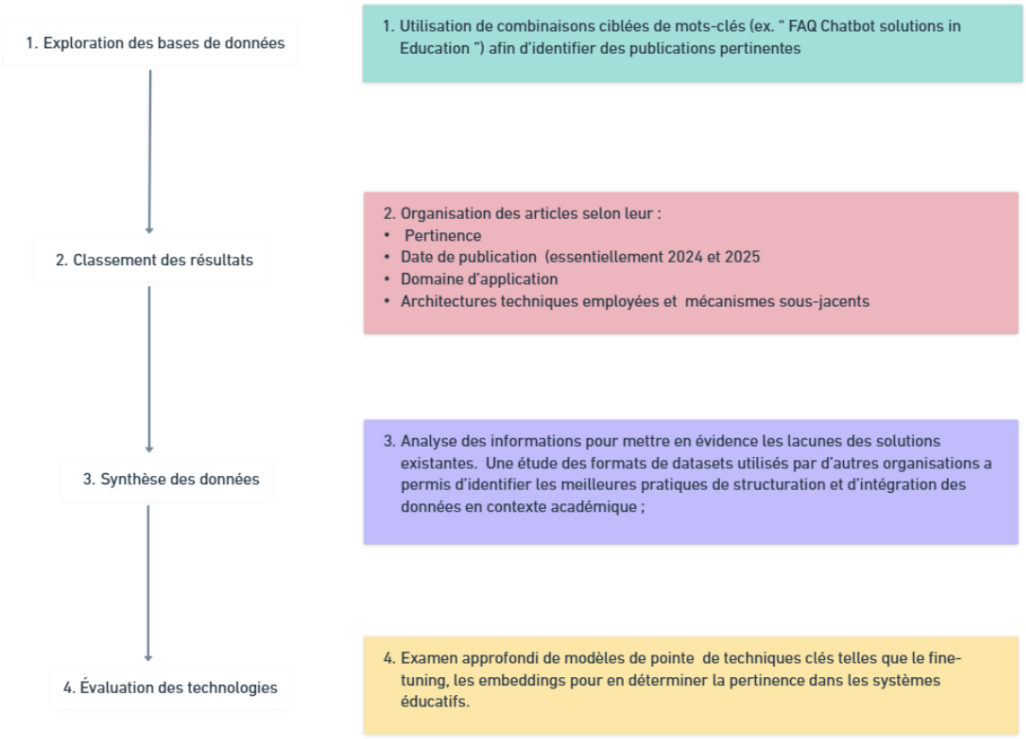

Figure 1.2 – Methodologie de recherche

1.6 Historique et concepts des chatbots

1.6.1 Origine des chatbots

Les chatbots, ou agents conversationnels, sont apparus dès les débuts de linformatique comme une tentative d'automatiser les interactions humaines, notamment pour faciliter le support client et les services dassistance. Le premier chatbot connu, **ELIZA**, développé en 1966 par Joseph Weizenbaum au MIT, simulait un psychothérapeute à l'aide de scripts basés sur la reconnaissance de mots-clés. Bien quELIZA ne comprenne pas réellement le langage, son fonctionnement démontra qu'il était possible de simuler une conversation humaine basique, ouvrant la voie à des applications dans l'automatisation des échanges au *help desk* [**?** , 5]

Dans les années 1970, **PARRY**, créé par Kenneth Colby, représenta un progrès notable en intégrant des modèles simulant des comportements émotionnels, préfigurant ainsi des chatbots capables de gérer des interactions plus nuancées [6]. Ces

premiers agents, bien que rudimentaires, posèrent les bases de l'automatisation du support via des systèmes capables de dialoguer, même de façon limitée.

1.6.2 Évolution technologique

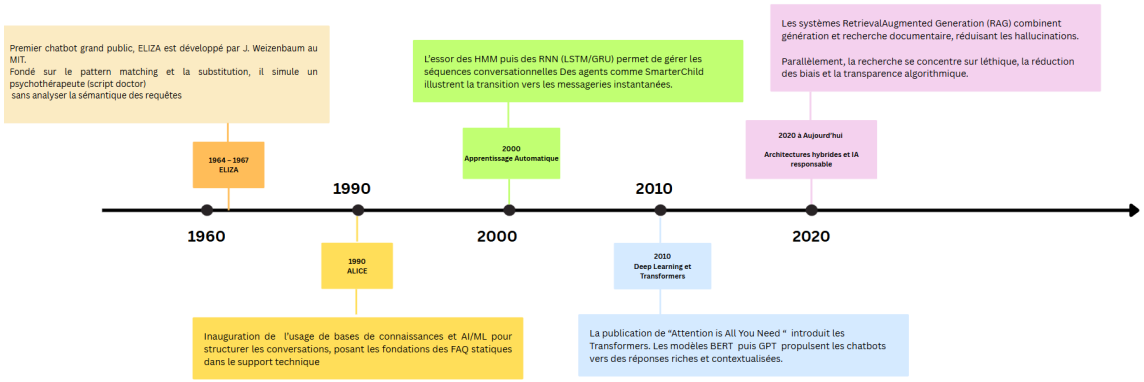

Figure 1.3 – Evolution technologique des chatbots

L'évolution des chatbots s'articule autour de cinq phases technologiques majeures. Initialement, ELIZA (1964-1967), développé par J. Weizenbaum au MIT, inaugure l'ère des chatbots grand public en simulant un psychothérapeute via pattern matching et substitution, sans analyse sémantique [7]. Les années 1990 marquent l'émergence des bases de connaissances avec ALICE (1995, R. Wallace), qui introduit AIML pour structurer les conversations et pose les fondations des FAQ statiques [8].

La décennie 2000 voit l'essor de l'apprentissage automatique avec les HMM puis les RNN (LSTM/GRU) pour gérer les séquences conversationnelles [9], illustré par des agents comme SmarterChild dans les messageries instantanées.

Les années 2010 révolutionnent le domaine avec le Deep Learning et les Transformers, notamment après la publication d"Attention is All You Need" (2017) [10], suivie des modèles BERT [11] puis GPT (2018-2020) qui propulsent les chatbots vers des réponses riches et contextualisées.

Enfin, les années 2020 introduisent les architectures hybrides avec les systèmes Retrieval-Augmented Generation (RAG) qui combinent génération et recherche documentaire pour réduire les hallucinations [12], parallèlement à un focus accru sur l'éthique, la réduction des biais et la transparence algorithmique.

1

1. Une version en ligne d'ELIZA est accessible : https://anthay.github.io/eliza.html.

Technologie	Cas d'usage	Avantages	Limites
Support téléphonique traditionnel (1970-1990)	Support personnalisé urgent	Contact humain direct	Coûteux, non scalable
Systèmes de ticketing & FAQ statique (1990-2000)	Gestion flux de tickets	Organisation des demandes, traçabilité	Bases FAQ statiques, non interactives
Systèmes IVR (1990-2010)	FAQ vocales simples	Réduction du volume d'appels	Compréhension limitée, menus rigides
Chatbots rule-based (2010-2015)	FAQ simples, tâches répétitives	Réponse rapide, peu coûteux	Faible flexibilité, pas de compréhension NL
Chatbots ML (2015-2020)	Support conversationnel amélioré	Meilleure compréhension NL	Besoin d'entraînement, erreurs possibles
Chatbots LLM + RAG (2020-aujourd'hui)	Support avancé, FAQ dynamiques	Réponses contextuelles, adaptatives	Supervision nécessaire, coût computationnel

Tableau 1.1 – Évolution des technologies de support client

1.6.3 Technologies utilisées dans le développement des chatbots

Le développement de systèmes conversationnels mobilise un ensemble de technologies interdépendantes relevant de l'intelligence artificielle, de l'ingénierie logicielle, du traitement automatique des langues (TAL), et de l'architecture des systèmes distribués. Les langages de programmation constituent la fondation technique, avec Python s'imposant comme référence en TAL et apprentissage automatique grâce à son écosystème scientifique (NumPy, TensorFlow, spaCy, HuggingFace) et sa lisibilité syntaxique[13], tandis que JavaScript avec Node.js est mobilisé pour les interfaces temps réel dans des architectures microservices ou serverless[14], et Java demeure présent pour la portabilité multiplateforme et la robustesse dans les secteurs bancaire ou administratif[15]. Le TAL, cur de l'intelligence des chatbots, englobe la reconnaissance des intentions, l'extraction d'entités nommées, la désambiguïsation sémantique et la génération de réponses via des bibliothèques comme spaCy, Stanza et NLTK offrant un support multilingue modulaire [16], mais l'évolution la plus significative repose sur les modèles transformer (BERT [17], RoBERTa [18], T5 [19]) qui permettent une représentation contextuelle bidirectionnelle du langage et améliorent

significativement les performances en compréhension et génération.

1.6.3.1 Frameworks d'apprentissage profond

Les capacités conversationnelles avancées des chatbots reposent sur des modèles dapprentissage profond. Les frameworks **TensorFlow** (Google) et **PyTorch** (Meta) sont les plus utilisés dans la recherche et l'industrie pour entraîner des réseaux de neurones de type LSTM, GRU ou *transformer* [20]. Ils permettent l'intégration fine de mécanismes dattention, de mémoire, ou de chaînes encodeurdécodeur, et favorisent loptimisation sur GPU. Ces technologies sous-tendent aujourd'hui la majorité des modèles génératifs employés dans les chatbots modernes, y compris les LLM (*large language models*) comme GPT-3 ou LaMDA [21].

1.6.3.2 Frameworks de gestion des dialogues

Le passage d'un traitement linguistique isolé à une interaction dialogique structurée nécessite des environnements dédiés. Le framework **Rasa**, open source et modulaire, est largement utilisé pour construire des assistants conversationnels en local, offrant des modules de reconnaissance dintentions (NLU) et de gestion des politiques de dialogue (*Dialogue Management*) [22]. Dautres plateformes comme **Botpress**, **Dialogflow** (Google) ou **Microsoft Bot Framework** offrent des environnements intégrés, souvent connectés à des solutions cloud, et sont utilisées dans des contextes industriels nécessitant une rapidité de prototypage [23].

1.6.3.3 Interfaces vocales

Dans le cas des interfaces vocales (callbots, assistants vocaux), deux composants sont essentiels : la reconnaissance automatique de la parole (ASR) et la synthèse vocale (TTS). Des outils comme **Google Speech-to-Text**, **Amazon Transcribe** ou encore **Mozilla DeepSpeech** permettent de convertir les entrées vocales en texte avec des performances proches de celles des humains dans certains contextes [24]. En parallèle, des moteurs de synthèse comme **Amazon Polly**, **Google TTS** ou **Coqui** produisent une sortie vocale fluide et naturelle, renforçant l'aspect immersif de linteraction [25].

1.6.3.4 Stockage des données et gestion des sessions

Le stockage des dialogues, du contexte conversationnel et des informations utilisateurs nécessite des solutions de base de données performantes. Les bases relation-

nelles comme **MySQL** ou **PostgreSQL** sont utilisées pour les structures rigides, tandis que les bases NoSQL comme **MongoDB** offrent une meilleure gestion des données semi-structurées sous format `JSON`, plus adapté aux logiques conversationnelles [39]. Pour la gestion rapide des sessions ou du cache, des systèmes en mémoire comme **Redis** sont fréquemment utilisés [40].

1.6.3.5 Conteneurisation et déploiement

Le déploiement en environnement de production repose majoritairement sur la conteneurisation avec **Docker**, qui permet d'encapsuler les composants du chatbot (modèles, serveurs, API) dans un environnement portable [26]. Pour les systèmes à grande échelle, **Kubernetes** est utilisé pour orchestrer, répartir et monitorer les conteneurs, assurant ainsi la scalabilité horizontale, la tolérance aux pannes et la mise à jour continue [27].

1.7 Processus de traitement des requêtes selon différentes technologies

Le fonctionnement d'un chatbot repose sur une séquence de traitements qui transforment une requête utilisateur en réponse intelligible. Ce processus varie selon le paradigme technologique sous-jacent : approche symbolique, statistique, neuronale ou hybride. Chacune mobilise des architectures distinctes, impliquant des choix spécifiques en termes de traitement du langage, dapprentissage et de génération.

1.7.1 Approche symbolique (deterministe)

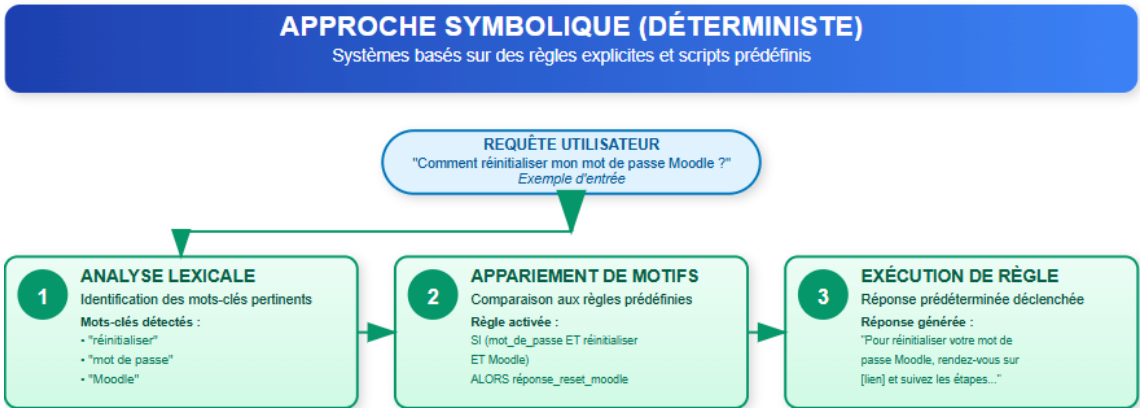

Figure 1.4 – Approche symbolique

L'approche symbolique, également appelée déterministe, repose sur un système de règles explicites préprogrammées qui associent des patterns linguistiques spécifiques à des réponses prédéterminées. Dans cette architecture, chaque requête utilisateur traverse un pipeline de traitement séquentiel : l'analyse lexicale identifie les mots-clés pertinents, l'appariement de motifs compare la requête aux règles existantes selon des critères booléens (ET, OU, NON), et l'exécution déclenche une réponse correspondante. Cette méthode garantit une traçabilité complète et un contrôle total sur le comportement du système, mais nécessite une anticipation exhaustive des formulations possibles et une maintenance manuelle intensive pour chaque nouveau cas d'usage.

Cas d'usage recommandés : L'approche symbolique s'avère particulièrement adaptée aux contextes où la précision et la prévisibilité priment sur la flexibilité conversationnelle. Elle excelle dans les **FAQ départementales** avec des procédures administratives standardisées (inscriptions, démarches, horaires).

Critère	Avantages	Limites
Transparence	Traçabilité complète de chaque réponse, explicabilité totale du processus de décision	Rigidité face aux variations linguistiques, incapacité à gérer les formulations non prévues
Contrôle	Maîtrise absolue du contenu des réponses, prévention des dérives conversationnelles	Couverture limitée nécessitant d'anticiper exhaustivement toutes les questions possibles
Fiabilité	Absence d'hallucinations, réponses cohérentes et prévisibles	Maintenance intensive requérant une programmation manuelle pour chaque nouvelle règle
Performance	Temps de réponse rapide, faible coût computationnel	Scalabilité limitée, complexité croissante avec le nombre de règles
Déploiement	Implementation simple, déboggage facilité	Adaptation difficile aux nouveaux domaines, mise à jour laborieuse

Tableau 1.2 – Analyse comparative de l'approche symbolique

1.7.2 Approche statique

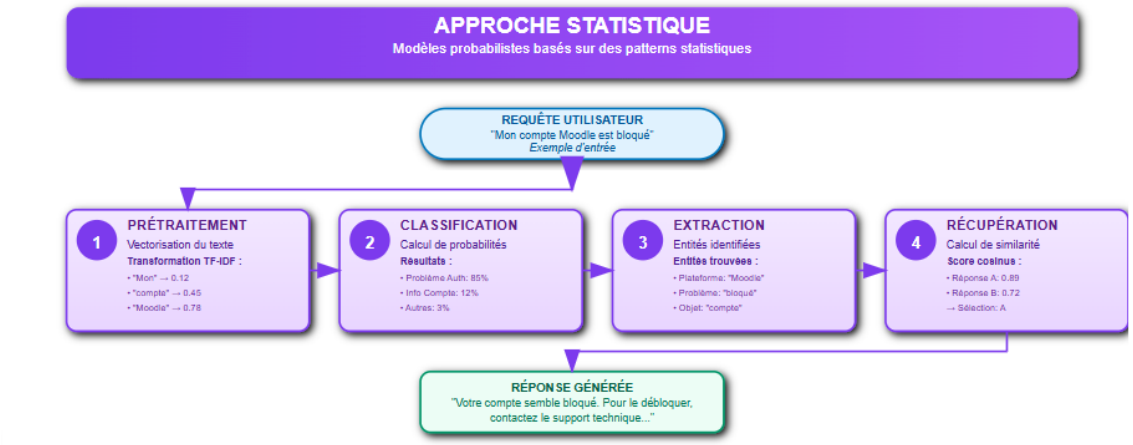

Figure 1.5 – Approche statique

L'approche statistique, introduite dans les années 2000, repose sur des modèles probabilistes qui interprètent et répondent aux requêtes utilisateurs en exploitant des patterns statistiques extraits de données d'entraînement. Cette méthode transforme d'abord le texte en représentations numériques via des techniques de vectorisation (TF-IDF, word embeddings), puis applique des algorithmes de classification (SVM,

Naive Bayes) pour identifier l'intention de l'utilisateur avec un score de confiance probabiliste. Parallèlement, des modèles d'extraction d'entités nommées (CRF, spaCy) identifient les éléments spécifiques de la requête, avant qu'un système de récupération par similarité cosinus sélectionne la réponse la plus appropriée dans une base de connaissances préétablie. Cette architecture offre un bon compromis entre performance computationnelle et qualité de réponse pour des domaines circonscrits.

Critère	Avantages	Limites
Automatisation	Traitement efficace de grands volumes de requêtes similaires, pipeline robuste et automatisé	Difficulté à maintenir le contexte sur plusieurs échanges, dialogues complexes mal gérés
Performance	Réponses en quelques millisecondes, temps de traitement prévisible et constant	Performances dégradées face à des formulations inattendues ou hors domaine d'entraînement
Ressources	Modèles légers avec faibles besoins computationnels, déploiement sur hardware standard	Adaptation dynamique limitée, nécessite un réentraînement complet pour nouvelles informations
Entraînement	Algorithmes bien maîtrisés, datasets modestes suffisants, processus d'apprentissage stable	Faible compréhension contextuelle, incapacité à saisir nuances et implications subtiles

Tableau 1.3 – Analyse comparative de l'approche statistique

Cas d'usage recommandés : L'approche statistique excelle dans les **services informatiques universitaires** traitant des volumes importants de demandes standardisées avec un vocabulaire technique délimité.

1.7.3 Approche neuronale (Deep Learning)

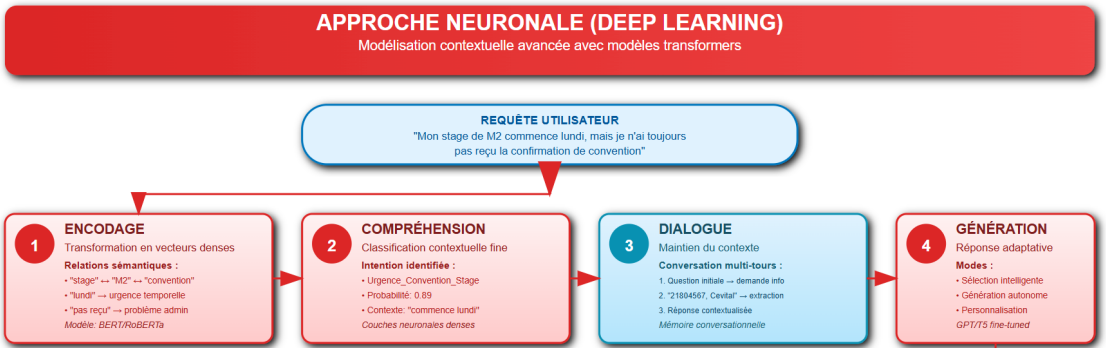

Figure 1.6 – Approche neuronale

L'approche neuronale, basée sur les architectures de Deep Learning et notamment les modèles transformers, permet une modélisation contextuelle avancée qui dépasse significativement les limitations des approches précédentes. Cette méthode exploite des réseaux de neurones profonds pré-entraînés (BERT, GPT, T5) pour encoder les requêtes en représentations vectorielles denses capturant les relations sémantiques complexes entre les mots et concepts. Le système applique ensuite des couches neuronales denses pour une classification d'intention contextuelle fine, intègre des mécanismes de mémoire conversationnelle permettant le maintien du contexte sur plusieurs tours de dialogue, et génère des réponses soit par sélection intelligente soit par génération autonome via des modèles génératifs fine-tunés. Cette architecture offre une flexibilité linguistique remarquable et une capacité d'adaptation dynamique aux spécificités utilisateur, au prix d'une complexité computationnelle et technique considérablement accrue.

Critère	**Avantages**	**Limites**
Flexibilité linguistique	Compréhension de formulations variées et naturelles, adaptation aux styles conversationnels divers	Coûts computationnels élevés nécessitant des GPUs performants (coût environ 10 fois supérieur)
Gestion contextuelle	Maintien cohérent du contexte sur des conversations multi-tours, mémoire conversationnelle avancée	Risques d'hallucinations avec génération d'informations plausibles mais factuellement incorrectes
Personnalisation	Adaptation dynamique aux spécificités et préférences de chaque utilisateur, apprentissage continu	Maintenance complexe nécessitant une expertise IA spécialisée pour monitoring et ajustements
Capacités génératives	Réponses créatives et adaptées à des situations inédites, génération de contenu original contextuel	Opacité des décisions rendant difficile l'explicabilité et le contrôle précis des réponses
Compréhension sémantique	Analyse approfondie des nuances, implications et sous-entendus dans les requêtes utilisateur	Dépendance forte aux données d'entraînement et risque de biais algorithmiques non contrôlés

Tableau 1.4 – Analyse comparative de l'approche neuronale

Cas d'usage recommandés : L'approche neuronale excelle dans les **environnements universitaires complexes** nécessitant une compréhension nuancée des besoins diversifiés et une capacité d'adaptation aux évolutions des services numériques.

1.7.4 Approche hybride (RAG - Retrieval-Augmented Generation)

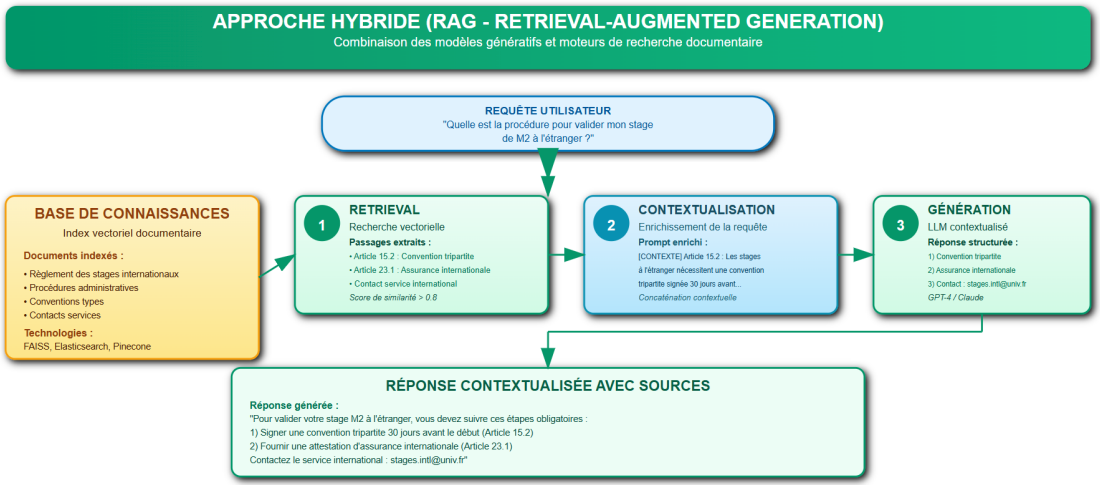

Figure 1.7 – Approche hybride

L'approche neuronale, basée sur les architectures de Deep Learning et notamment les modèles transformers, permet une modélisation contextuelle avancée qui dépasse significativement les limitations des approches précédentes. Cette méthode exploite des réseaux de neurones profonds pré-entraînés (BERT, GPT, T5) pour encoder les requêtes en représentations vectorielles denses capturant les relations sémantiques complexes entre les mots et concepts. Le système applique ensuite des couches neuronales denses pour une classification d'intention contextuelle fine, intègre des mécanismes de mémoire conversationnelle permettant le maintien du contexte sur plusieurs tours de dialogue, et génère des réponses soit par sélection intelligente soit par génération autonome via des modèles génératifs fine-tunés. Cette architecture offre une flexibilité linguistique remarquable et une capacité d'adaptation dynamique aux spécificités utilisateur, au prix d'une complexité computationnelle et technique considérablement accrue.

Critère	Avantages	Limites
Flexibilité linguistique	Compréhension de formulations variées et naturelles, adaptation aux styles conversationnels divers	Coûts computationnels élevés nécessitant des GPUs performants (coût environ 10 fois supérieur)
Gestion contextuelle	Maintien cohérent du contexte sur des conversations multi-tours, mémoire conversationnelle avancée	Risques d'hallucinations avec génération d'informations plausibles mais factuellement incorrectes
Personnalisation	Adaptation dynamique aux spécificités et préférences de chaque utilisateur, apprentissage continu	Maintenance complexe nécessitant une expertise IA spécialisée pour monitoring et ajustements
Capacités génératives	Réponses créatives et adaptées à des situations inédites, génération de contenu original contextuel	Opacité des décisions rendant difficile l'explicabilité et le contrôle précis des réponses
Compréhension sémantique	Analyse approfondie des nuances, implications et sous-entendus dans les requêtes utilisateur	Dépendance forte aux données d'entraînement et risque de biais algorithmiques non contrôlés

Tableau 1.5 – Analyse comparative de l'approche neuronale

L'approche neuronale excelle dans les **environnements universitaires complexes** nécessitant une compréhension nuancée des besoins diversifiés et une capacité d'adaptation aux évolutions des services numériques.

1.8 Typologie des chatbots et leur fonctionnement

La classification des chatbots peut être établie selon leur architecture fonctionnelle, leur degré d"autonomie linguistique et leur mode d"interaction avec les utilisateurs. Quatre grandes catégories sont généralement reconnues dans la littérature scientifique :

1.8.1 Chatbots basés sur des règles (*Rule-based*)

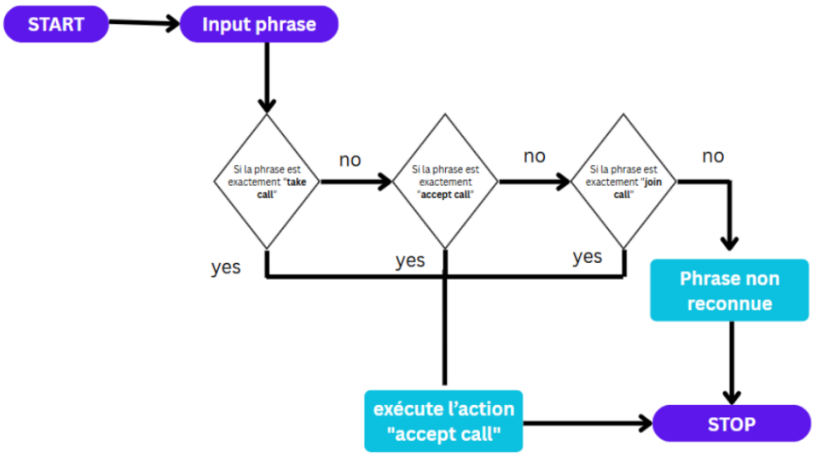

Figure 1.8 – Chatbots basés sur des règles

La figure 1.8 illustre l'architecture de traitement des chatbots basés sur des règles. Ces systèmes suivent un processus séquentiel de vérifications conditionnelles déterministes, examinant chaque phrase d'entrée à travers une série de tests prédéfinis en cascade. Par exemple, pour la phrase "Je veux prendre rendez-vous", le système vérifie successivement les patterns "take call", "accept call", puis "join call" jusqu'à trouver une correspondance exacte ou classer l'entrée comme non reconnue, arrêtant alors le processus sans action.

Avantages	**Limites**
Facilité de contrôle et d'interprétation du comportement système	Aucune capacité de généralisation possible au-delà des règles définies
Bon comportement dans des domaines fortement balisés (systèmes de réservation, FAQ simples)	Explosion combinatoire du nombre de règles nécessaires pour couvrir les variations linguistiques
Débogage et maintenance simplifiés grâce à la transparence des règles	Rigidité face aux formulations non anticipées par les développeurs
Réponses prévisibles et cohérentes avec les spécifications métier	Scalabilité limitée pour des domaines conversationnels complexes
Faibles ressources computationnelles requises	Maintenance intensive pour chaque nouveau cas d'usage linguistique

Tableau 1.6 – Avantages et limites des chatbots basés sur des règles

1.8.2 Chatbots à base de récupération (*Retrieval-based*)

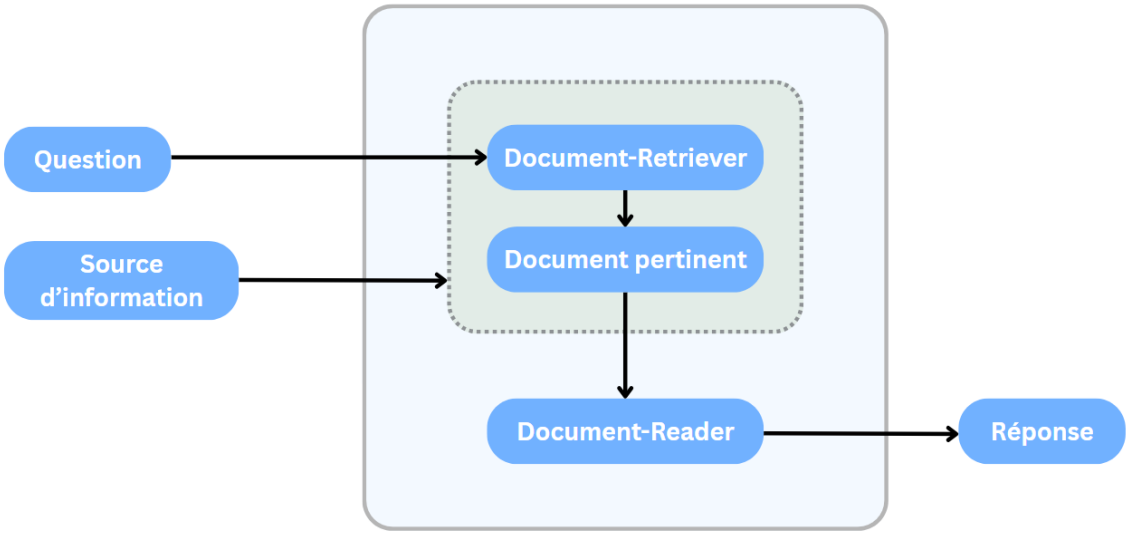

Figure 1.9 – Chatbot a base de récupération

La figure représente l'architecture des chatbots à base de récupération qui repose sur un processus en deux étapes séparant la recherche d'informations de leur présentation.

Le **Document-Retriever** interroge d'abord une source structurée (base de FAQ, documents PDF, base de connaissances) pour identifier les documents pertinents selon des critères de similarité textuelle. Le **Document-Reader** analyse ensuite le document récupéré pour en extraire et synthétiser les informations essentielles sous forme de réponse structurée. Par exemple, pour "Comment accéder au Wi-Fi de l'université ?", le système récupère les procédures d'authentification puis extrait les étapes spécifiques, identifiants requis et contacts support. Cette architecture permet d'exploiter de vastes corpus documentaires tout en maintenant la cohérence des réponses, particulièrement adaptée aux départements informatiques universitaires disposant de manuels et procédures existants.

Avantages	Limites
Réponses grammaticalement correctes et bien structurées	Qualité entièrement dépendante de la base documentaire existante
Réduction significative du risque de dérive conversationnelle ou d'hallucination	Incapacité totale à traiter des requêtes inédites non couvertes par le corpus
Exploitation efficace de vastes corpus documentaires sans reprogrammation	Performance limitée par la qualité de l'algorithme de récupération
Maintien de la cohérence avec les sources officielles et procédures établies	Difficultés avec les questions nécessitant une synthèse multi-documents
Facilité de mise à jour par simple ajout de nouveaux documents	Réponses potentiellement fragmentaires si le document source est incomplet
Traçabilité des sources utilisées pour générer les réponses	Temps de traitement dépendant de la taille du corpus documentaire

Tableau 1.7 – Avantages et limites des chatbots à base de récupération

1.8.3 Chatbots génératifs (*Generative chatbots*)

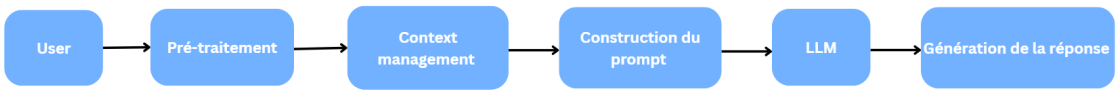

Figure 1.10 – Chatbots génératifs

La figure représente l'architecture des chatbots génératifs qui suivent un processus séquentiel pour produire des réponses originales plutôt que de récupérer des informations existantes. Le système traite d'abord l'entrée utilisateur via une phase de **Pré-traitement** (normalisation, tokenisation, correction orthographique), puis le **Context management** maintient la mémoire conversationnelle en intégrant les échanges précédents. La **Construction du prompt** transforme la requête et le contexte en instruction structurée pour le modèle, avant que le **LLM** génère une réponse originale basée sur ses connaissances d'entraînement. Par exemple, pour "Mon stage pose des problèmes de confidentialité avec mon mémoire", le système génère une réponse personnalisée expliquant les procédures, proposant des solutions et orientant vers les services compétents. Cette architecture permet une flexibilité remarquable mais nécessite une gestion rigoureuse pour éviter les hallucinations.

Avantages	Limites
Fluidité conversationnelle élevée et naturelle	Risque significatif d'hallucinations et d'informations incorrectes
Capacité d'adaptation à des requêtes ouvertes, même totalement inédites	Réponses parfois incohérentes et non vérifiables par rapport aux sources
Génération de contenu personnalisé selon le contexte utilisateur	Difficulté à maintenir la cohérence avec les politiques institutionnelles
Flexibilité remarquable pour traiter des situations complexes et nuancées	Dépendance aux données d'entraînement pouvant contenir des biais
Maintien de la mémoire conversationnelle sur plusieurs échanges	Coûts computationnels élevés pour les modèles de langage sophistiqués
Capacité créative pour proposer des solutions originales	Opacité du processus de génération rendant le contrôle qualité difficile

Tableau 1.8 – Avantages et limites des chatbots génératifs

1.8.4 Chatbots hybrides (RetrievalAugmented Generation RAG)

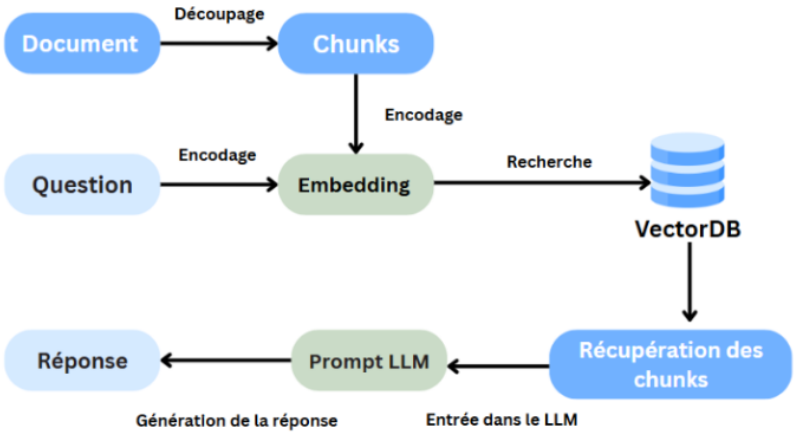

Figure 1.11 – Chatbots hybrides

La figure représente l'architecture des chatbots hybrides RAG (Retrieval-Augmented Generation) qui combinent récupération et génération pour optimiser précision et fluidité des réponses. Le processus débute par la préparation des `Documents` sources (règlements, procédures) qui subissent un `Découpage` en `Chunks` thématiques de 200-500 mots, puis un `Encodage` en `Embeddings` vectoriels stockés dans une `VectorDB`. Lorsqu'un utilisateur pose une `Question` comme "Que faire si mon maître de stage refuse de signer l'évaluation ?", celle-ci est encodée et comparée aux chunks via une `Recherche` de similarité. Les 3-5 passages les plus pertinents sont `Récupérés` et in-

tégrés dans un `Prompt LLM` structuré, permettant au `LLM` de générer une réponse synthétisant les informations officielles avec ses capacités conversationnelles. Cette architecture hybride maintient la fiabilité réglementaire tout en offrant une expérience fluide, particulièrement adaptée aux services administratifs universitaires.

Avantages	**Limites**
Amélioration significative de la factualité des réponses grâce à l'ancrage documentaire	Complexité de mise en uvre nécessitant un backend documentaire robuste et spécialisé
Accès dynamique à des sources documentaires à jour sans réentraînement du modèle	Latence plus élevée due au processus de récupération et d'encodage vectoriel
Combinaison optimale entre précision factuelle et fluidité conversationnelle naturelle	Intégration difficile dans les dispositifs embarqués (domotique, assistants personnels)
Traçabilité des sources utilisées permettant la vérification des informations	Dépendance critique à la qualité et pertinence de la base documentaire
Maintien de la cohérence avec les réglementations et procédures institutionnelles	Coûts d'infrastructure élevés pour le stockage et l'indexation vectorielle
Scalabilité par simple ajout de nouveaux documents sans modification du système	Expertise technique requise pour l'optimisation des embeddings et du chunking

Tableau 1.9 – Avantages et limites des chatbots hybrides RAG

1.9 État de l'Art

Référence	Modèle	Données	Application	Performance	Avantages/Limites
Neumann et al. (2025) IEEE Trans. Education	GPT-4 + LangChain + RAG	Slides, cours, exercices RWTH Aachen	Support pédagogique BDD	88% précision, bonne acceptation TAM	+ Engagement étudiant élevé - Setup complexe, hallucinations
Zhang et al. (2025)	LLM + Agentic Contextual Retrieval	Logs réseaux, graphes, docs internes	Automatisation télécom	+32% QoS, -40% interventions	+ Très performant - Coût calcul élevé
Yigci et al. (2024) Advanced Intelligent Systems	GPT-4 + RAG	Corpus pédagogique universitaire	Assistant étudiants internationaux	Réduction charge admin, réponses précises	+ Fiable - Dépendant données, interprétabilité limitée
Suhardi et al. (2023)	Rule-based NLP	FAQ universitaire	Service académique auto	Temps réduit, précision contextuelle limitée	+ Simple implémentation - Faible tolérance variations
Ramandanis & Xinogalos (2023)	Mistral-7B / GPT-SW3-6.7b + RAG	API Göteborg & Co	Tourisme	Mistral : 65.71%, GPT-SW3 : 41.72%	+ Open-source, multilingue - Dépendance API, ressources
Dan et al. (2023)	LLaMA-13B + EduChat	Corpus éducatif chinois, psychologie	Éducation & soutien psycho	C-Eval : 40.7% (sans), 49.3% (avec RAG)	+ Personnalisé, données riches - Généralisation limitée
Maung Thway (2024)	Claude 3 + RAG	Jupyter, manuels Data Science	Éducation	97.1% satisfaction, haute précision	+ Langage naturel, réduit hallucinations - Lenteurs pic, pas multimodal
Assayed et al. (2023)	Naive-Bayes + Random Forest	Données conseillers académiques	Éducation	Non spécifiée	+ Méthodes classiques - Manque métriques formelles
Goitom et al. (2024)	LSTM	Questionnaire étudiants informatique	Éducation	91.55% précision, F1 : 85.21%	+ Haute précision - Domaine spécifique, supervisé classique

Tableau 1.10 – État de l'art des chatbots éducatifs et d'assistance

L'état de l'art des chatbots éducatifs révèle une évolution technologique rapide dominée par les grands modèles de langage (GPT-4) et les techniques RAG (Retrieval-Augmented Generation). Les recherches récentes de Neumann et al. (2025) et Zhang et al. (2025) démontrent des performances remarquables avec 88% de précision et +32% d'amélioration qualité-service, illustrant la capacité des LLM à gérer des interactions complexes et produire des réponses contextuellement appropriées.

Cependant, l'analyse révèle des limitations persistantes communes à la majorité des approches. Le problème des hallucinations reste préoccupant dans les contextes éducatifs où la fiabilité informationnelle est cruciale, comme souligné par Yigci et al. (2024) et Maung Thway (2024). La complexité d'implémentation et de maintenance constitue un frein majeur à l'adoption, tandis que l'adaptabilité aux environnements institutionnels spécifiques demeure problématique. Les performances varient significativement selon les domaines (67% à 97%), et les solutions nécessitent souvent des adaptations coûteuses pour s'ajuster aux particularités locales.

L'état de l'art révèle également un manque d'approches hybrides optimisant le compromis entre contrôle administratif et flexibilité conversationnelle. La plupart des solutions privilégient soit la précision factuelle au détriment de la naturalité, soit l'inverse.

Cette analyse nous amène à proposer une approche innovante combinant un système bi-dataset avec des techniques de fuzzy matching, permettant une gestion collaborative du contenu par des agents non-techniques tout en maintenant la qualité des réponses. Cette proposition vise à combler les lacunes identifiées en offrant une solution fiable, adaptable et facilement maintenable par les équipes administratives locales, sans expertise technique approfondie.

Chapitre 2

Implémentation du Système Hybride

2.1 Introduction

Ce chapitre décrit la construction méthodologique d'un assistant universitaire suivant une démarche empirique rigoureuse structurée en trois phases itératives complémentaires.

Chaque cycle de développement applique le processus hypothèse évaluation ajustement : formulation d'une hypothèse d'amélioration spécifique, évaluation systématique combinant métriques quantitatives de performance et tests qualitatifs d'expérience utilisateur, puis ajustements ciblés orientant le cycle suivant.

Cette approche itérative est guidée par une interrogation centrale : chaque modification améliore-t-elle concrètement l'assistance apportée aux étudiants et au personnel universitaire ? Cette question garantit l'alignement constant entre évolutions techniques et besoins réels des utilisateurs finaux.

La présentation suit une progression logique reflétant notre démarche de conception : établissement du cadre méthodologique et des critères technologiques, description des données et de l'environnement technique, puis détail des quatre cycles d'amélioration successifs conduisant vers une solution mature et empiriquement dépassant la simple juxtaposition d'algorithmes pour proposer un cheminement méthodologique structuré.

2.2 Cadre UX et critères d'acceptabilité

Le projet s'inscrit dans un cadre d'expérience utilisateur (UX) défini autour de deux profils représentatifs. L'étudiant (licence/master) consulte l'assistant entre deux cours et attend une réponse concise, immédiatement exploitable, en français clair. L'agent de scolarité utilise l'outil pour vérifier la cohérence des procédures et réduire le volume de courriels et d'appels quotidiens, tout en apportant un appui humain et institutionnel complémentaire.

Pour ces deux personas, deux objectifs cibles ont été identifiées :

1. Permettre à l'étudiant d'obtenir des réponses fiables

2. Permettre à l'agent de saisie du département informatique de tenir la base de données à jour sans avoir de pré-requis de code.

La réussite de chaque prototype est ensuite appréciée au regard d'un ensemble de KPI orientés UX :

— **Taux de réponse correcte :** la proportion de requêtes dont le contenu de réponse est jugé exact ; mesuré par l'indicateur Exact Match sur un jeu de validation indépendant.

— **Latence ressentie :** temps écoulé entre l'envoi de la question et l'affichage complet de la réponse. L'objectif est de rester sous trois secondes, condition de fluidité sur mobile.

— **Cohérence de ton :** conformité au registre attendu dans une communication universitaire ; l'assistant doit éviter le jargon excessif comme l'excès de familiarité. Ce critère est vérifié par revue humaine à chaque itération.

Ces indicateurs constituent la grille d'acceptabilité : une nouvelle configuration n'est conservée que si elle améliore au moins l'un des objectifs sans faire régresser les autres.

De cette manière, l'ensemble du chapitre 2 suit une logique de conception centrée utilisateur : chaque évolution technique est présentée, testée et retenue (ou rejetée) en fonction de la valeur ajoutée qu'elle apporte aux deux personas identifiés.

2.3 Jeux de données et scénarios de test

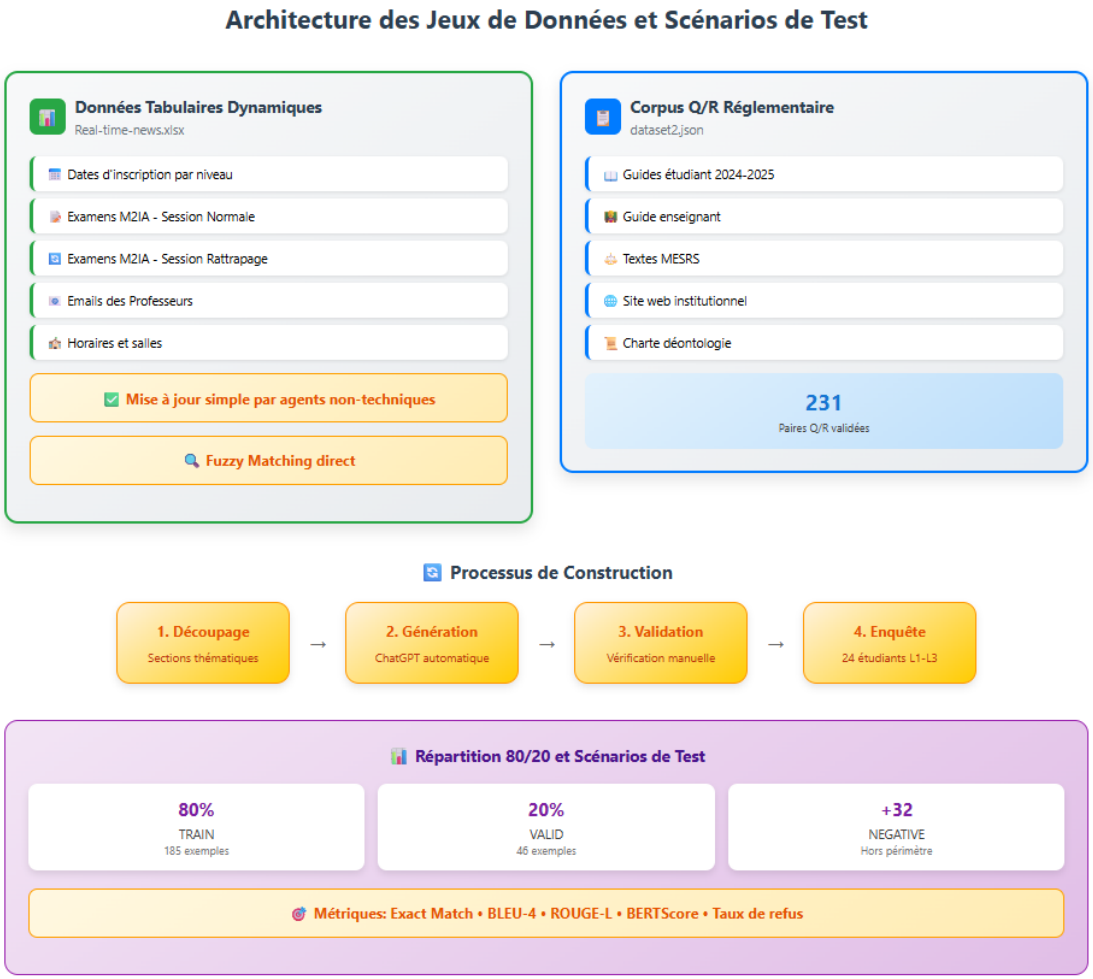

Figure 2.1 – Jeux de données

Le système s'appuie sur deux ressources documentaires complémentaires, chacune conçue pour répondre à des besoins et des contraintes différentes :

1. Données tabulaires dynamiques – fichier Excel
2. Corpus question-réponse réglementaire – dataset JSON

2.3.1 Données tabulaires dynamiques : fichier Excel

Le fichier *Real-time-news.xlsx*, issu de l'espace Affichage du site e-learning de l'Université de Béjaïa, est conçu pour être mis à jour simplement par les agents du service scolarité et comporte plusieurs feuilles thématiques (dates d'inscription par niveau, examens M2 IA sessions normale et rattrapage avec dates/horaires/salles, emails des professeurs). Chaque feuille peut être modifiée sans connaissance tech-

nique (insertion de lignes, corrections de dates), garantissant que l'assistant reste synchronisé avec les dernières informations logistiques. En exécution, le système interroge directement ce tableur par un mécanisme de "fuzzy matching" sur la question de l'utilisateur, sans transformation préalable en graphe ou base de données relationnelle.

2.3.2 Corpus Q/R réglementaire : dataset JSON

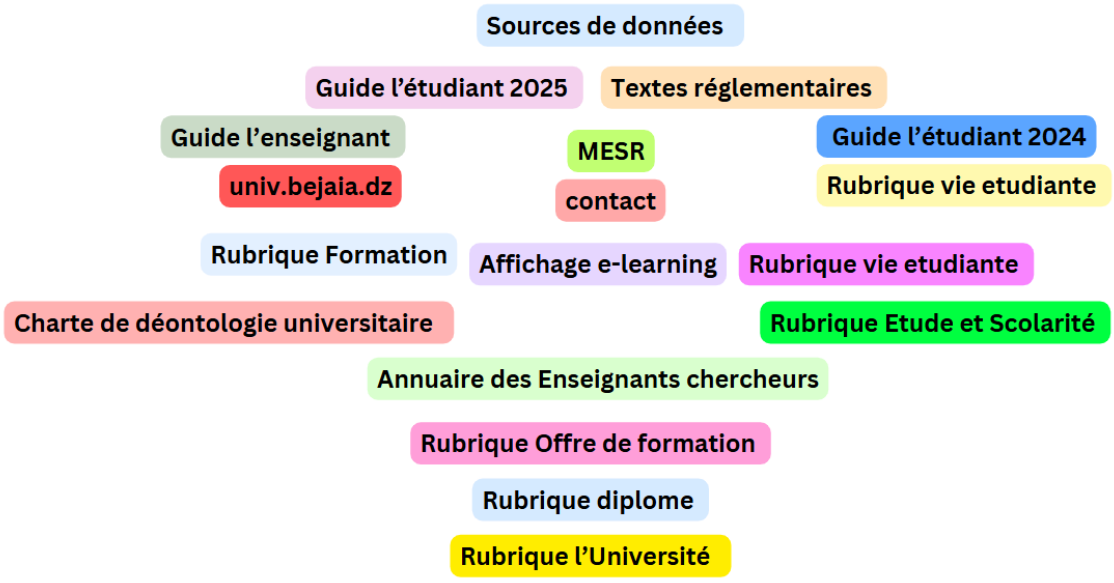

Figure 2.2 – Source de données du dataset.JSON

Figure 2.3 – Details sur les sources de données du dataset

2.3.3 Validation terrain

Une enquête a été conduite auprès de 24 étudiants (majoritairement L1 et L3, 18–21 ans, majorité féminine) pour recenser les questions jugées prioritaires :

Chaque document réglementaire et questions les plus mentionnées par les étudiants ont d'abord été découpé en sections thématiques. À partir de ces sections, une augmentation de données a été réalisée grace a ChatGPT permettant ainsi d'avoir plusieurs formulations de la meme question. Chaque paires Q/R a ensuité eté vérifiée manuellement pour garantir la fidélité au texte source et la clarté de l'énoncé.

Figure 2.4 – Resultats du sondage lancé aux étudiants du département informatique

Afin d'évaluer la généralisation du modèle et sa robustesse, le corpus JSON (231 exemples) a été divisé selon la règle 80%/20% : TRAIN (185 exemples) pour le fine-tuning et VALID (46 exemples) pour l'évaluation continue. Le jeu "negative" (32 questions hors périmètre) a été construit manuellement à partir des retours de l'enquête pour tester la capacité du système à refuser les demandes n'ayant aucune relation avec le département informatique (ex : "quel temps fait-il à Alger ?"). Cette répartition permet de mesurer, à chaque cycle :

Cette répartition permet de mesurer, à chaque cycle :

— **Qualité de la réponse sur VALID** : métriques Exact Match, BLEU-4, ROUGE-L, BERTScore
— **Sécurité sur NEGATIVE** : taux de refus correct des questions hors périmètre
— **Couverture fonctionnelle** : efficacité du secours Excel lorsque RAG ne trouve rien

Ainsi, chaque configuration est comparée sur des bases cohérentes et représentatives des cas d'usage réels.

2.4 Environnement et outils techniques

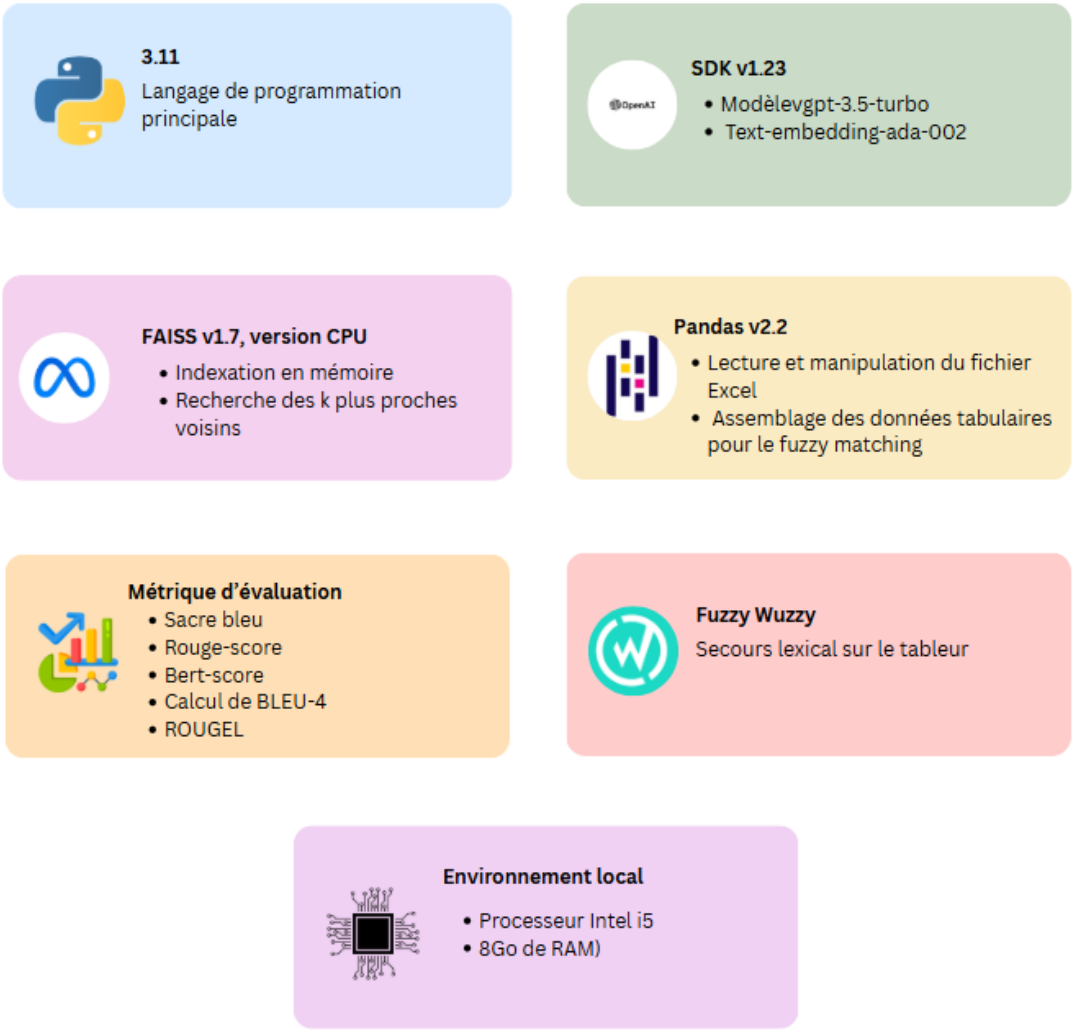

Figure 2.5 – Technologies retenues pour la conception et l'évaluation

2.4.1 Sélection du modèle de base

Le tableau comparatif de la section "Résumé de l'existant" (Chapitre 1) présente huit implémentations récentes de chatbots éducatifs, utilisant diverses architectures : GPT-4, LLaMA-13B, Claude 3, ou encore des approches classiques (Naive-Bayes, LSTM).

Trois critères ont guidé la sélection de **GPT-3.5-turbo** comme modèle de base :

Critère 1 : Performance documentée

Les travaux de Neumann et al. (2025) démontrent une précision de 88% avec GPT-4 + RAG. Ce résultat confirment l'efficacité de la famille GPT dans les

contextes éducatifs.

Critère 2 : Équilibre coût/performance

Contrairement à GPT-4 (coût 10x supérieur) ou aux solutions auto-hébergées (infrastructure complexe), GPT-3.5-turbo offre un rapport optimal pour un déploiement universitaire avec budget contraint.

Critère 3 : Maturité écosystémique

L'API OpenAI bénéficie d'une documentation exhaustive, d'une stabilité éprouvée et d'outils de fine-tuning intégrés, réduisant les risques de développement comparativement aux modèles open-source nécessitant une infrastructure spécialisée.

Cette sélection s'appuie également sur les retours d'expérience de Yigci et al. (2024), qui confirment la fiabilité de GPT-4 dans un contexte similaire, tout en suggérant que GPT-3.5 constitue une alternative viable pour des budgets restreints.

2.4.2 Cycle 1 : B1- Modèle et Fine-tuning seul

Hypothèse 1 :

Un simple fine-tuning du modèle GPT-3.5-turbo sur notre corpus interne de questions–réponses suffit à produire des réponses fiables, sans nécessiter de base documentaire externe.

Pour vérifier cette hypothèse, conformément à la justification présentée en section 1.3, le modèle de base **gpt-3.5-turbo-1106** a été ajusté pendant **8 epochs** sur les 185 exemples du jeu TRAIN, avec un **batch size de 8** et **un learning rate multiplier de 0,1**.

Paramètre	**Valeur**
Modèle de base	gpt-3.5-turbo-1106
Taille du jeu d'entraînement	185 exemples
Nombre d'époques (epochs)	8
Taille de batch (batch size)	8
Learning rate multiplier	0,1
Durée du fine-tuning	30 minutes

Tableau 2.1 – Paramètres du fine-tuning initial

Le fine-tuning a duré moins de trente minutes et a généré un nouveau modèle personnalisé.

2.4.2.1 Résultats et retour UX

Résultats quantitatifs et qualitatifs sur le jeu VALID :

Question	Réponse attendue	Réponse du modèle
Pour quand est prévu l'examen du module Big Data ?	L'examen du module Big Data est prévu pour le 13 mai 2025 au bloc 115, salle 211.	Je ne sais pas

Tableau 2.2 – Exemple de réponse incomplète

Question	Réponse attendue	Réponse du modèle
Comment payer les frais de scolarité ?	Le paiement des frais de scolarité se fait via la plateforme MESR.	Il faut ouvrir un compte auprès de la CIH Banque du Maroc afin d'obtenir un moyen de paiement adéquat.

Tableau 2.3 – Exemple d'hallucination

Exact Match (EM) : 0% : aucune des réponses n'était littéralement identique à la référence.

BLEU-4 : 2%, traduisant un chevauchement quasi nul avec le texte cible

Ces illustrations montrent :

— Il ne sait pas se repérer dans un fichier excel lorsqu'on lui pose des questions sur l'emploi du temps.
— Il a indiqué un moyen de retrait d'attestation totalement différent de la procédure réelle.

2.4.2.2 Bilan

Malgré un ton naturel, le fine-tuning seul ne garantit pas la véracité des réponses.

L'hypothèse de départ (un simple Fine Tuning suffit) est invalidée ; il devient indispensable d'intégrer une source documentaire (RAG) pour ancrer les réponses dans des faits avérés avant toute itération ultérieure.

2.4.3 Cycle 2 : B2 - Ajout d'une base documentaire RAG

Face aux hallucinations observées lors du fine-tuning seul, nous avons émis l'hypothèse suivante :

2.4.3.1 Hypothèse :

Connecter le modèle à une base documentaire permettra de combler les lacunes de connaissances et de réduire drastiquement les erreurs factuelles.

Pour mettre cette hypothèse à l'épreuve, nous avons construit un index vectoriel à partir de l'ensemble des passages de notre corpus réglementaire (les 231 Q/R du JSON) et des lignes du tableur Excel (60 entrées).

Chaque élément a été converti **en vecteur** via l'**embedding text-embedding-ada-002**, puis indexé en mémoire avec FAISS.

Lorsqu'un utilisateur pose une question, le flux devient :

1. Reformulation légère de la question pour la rendre autonome.
2. Conversion de la question en vecteur (ada-002) et recherche des 8 passages les plus proches (top-k = 8).
3. Sélection du passage le plus pertinent (premier résultat FAISS).
4. Injection de ce contexte dans un prompt RAG pour générer la réponse avec le modèle fine-tuné.

Cette architecture RAG, motivée par l'analyse comparative du Chapitre 1, permet au modèle d'appuyer sa génération sur un extrait documenté, plutôt que de se baser uniquement sur ses poids internes.

2.4.3.2 Résultats et retour UX

Question	**Réponse attendue**	**Réponse du modèle**
Que se passe-t-il si j'ai 1 absence non justifiée	3 absences non justifiées ou 5 absences même justifiées entraînent systématiquement une exclusion du module	Vous êtes exclu du module.

Question	Réponse attendue	Réponse du modèle
Comment retirer une attestation de scolarité ?	Vous pouvez retirer votre attestation auprès du département informatique.	Vous pouvez retirer votre attestation auprès du département informatique.

— **Exact Match (EM)** : 90% des réponses étaient littéralement identiques à la référence
— **BLEU-4 :** 97% de score, reflétant le fait que le modèle récite presque mot pour mot le contexte
— **Refus corrects sur le jeu NEGATIVE :** 92%

Ces exemples montrent que, grâce au RAG :

1. La majorité des réponses factuelles est maintenant extraite correctement du contexte document : dates et procédures sont reproduites à l'identique.
2. Pour autant, une hallucination subsiste : sur la question " Que se passe-t-il si j'ai 1 absence non justifiée ", le modèle a répondu " Vous êtes exclu du module ". Or, dans le dataset, il existe une paire de données " 3 absences non justifiées ou 5 absences même justifiées entraînent systématiquement une exclusion du module ".

2.4.3.3 Bilan :

Ce cas souligne la nécessité d'un filtre complémentaire permettre une compréhension des requetes plus specifiques des utilisateurs, sans pour autant sacrifier la couverture globale.

2.4.4 Cycle 3 : B3 - ajout d'un garde-fou lexical

2.4.4.1 Hypothèse 3 :

Nous supposons que tout effet d'hallucination peut être éliminé en imposant un simple critère de chevauchement lexical entre le passage documentaire sélectionné (par RAG) et la réponse générée : si la réponse ne contient pas au moins deux mots en commun avec son contexte, elle doit être remplacée par une formule de refus (ń Je ne sais pas. ż).

1. Après la recherche vectorielle (top-k = 8 via FAISS) et la génération de la réponse fine-tunée, nous normalisons les deux blocs (minuscules, suppression de la ponctuation) et comptons les tokens partagés. Si ce chevauchement est

inférieur à 2 tokens, la réponse est systématiquement remplacée par ń Je ne sais pas. ż.

Résultats quantitatifs et qualitatifs sur le jeu VALID après filtre strict (2 tokens)

Question	Réponse attendue	Réponse du modèle
Quelles sont les dates d'examen de rattrapage ?	Le rattrapage est du 15 au 20 septembre...	Les sessions de rattrapage se tiennent du 15 au 20 septembre...

Question	Réponse attendue	Réponse du modèle
Quel est le délai de retrait d'un diplôme ?	Deux semaines après la proclamation des résultats...	Je ne sais pas

— **Exact Match (VALID) :** 90% (identique au cycle précédent, car les bonnes réponses contiennent presque toujours au moins deux mots clés)
— **Refus corrects (NEGATIVE) :** 100% (toutes les hallucinations isolées sont interceptées)

— Le taux d'hallucination tombe à zéro, mais plusieurs questions ou paraphrases ont été rejetées, ce qui crée de faux-négatifs et nuit à la couverture fonctionnelle.

Pour réduire ces blocages injustifiés, nous avons abaissé le seuil à 1 token : une réponse contenant au moins un mot clé du contexte est acceptée, même si elle ne reprend pas fidèlement tout le passage.

Question	Réponse attendue	Réponse du modèle
Quand se font les inscriptions ?	Les inscriptions se font du 1 au 15 juillet.	Les inscriptions se font du 1 au 15 juillet.

Question	Réponse attendue	Réponse du modèle
Quel est le délai de retrait d'un diplôme ?	Le diplôme peut être retiré deux semaines après la proclamation des résultats.	Vous pouvez le retirer après deux semaines.

— **Exact Match (VALID) :** 87% (quelques paraphrases "libres" ne retrouvent plus deux tokens identiques)
— **Refus corrects (NEGATIVE) :** 83% (2 hallucinations mineures sont désormais passées)

Le filtre souple restaure la majorité des réponses valides tout en conservant une bonne sécurité. Les faux-négatifs sont drastiquement réduits et l'assistant couvre presque intégralement le corpus légitime, au prix de quelques hallucinations résiduelles qui peuvent être comblées avec un enrichissement du dataset.

2.4.4.2 Bilan :

Le garde-fou lexical démontre que le réglage du seuil de chevauchement permet de piloter finement le compromis entre confiance (absence d'hallucination) et couverture fonctionnelle (traiter toutes les questions légitimes).

— Le mode strict (2 tokens) offre une fiabilité absolue (100% de refus corrects), au détriment de la couverture.
— Le mode souple (1 token) maximise la couverture (87% EM) mais laisse passer quelques erreurs (83% refus corrects).

Ces résultats motivent l'introduction ultérieure d'un filtre sémantique complémentaire (cosine similarity sur embeddings), qui vise à combiner la robustesse de l'approche stricte et la flexibilité du mode souple sans renoncer à la sécurité.

2.4.5 Cycle 4 : ré-ordonnancement par un LLM externe

2.4.5.1 Hypothèse 4 :

Un modèle de langage (GPT-3.5-turbo) peut, à lui seul, sélectionner le passage le plus pertinent parmi les k extraits récupérés par FAISS, ce qui devrait conduire à des réponses plus naturelles et mieux adaptées en contexte, sans pour autant perdre en fiabilité.

Architecture

1. **Recherche vectorielle (FAISS) :** on extrait les 8 passages les plus proches de la question, comme précédemment.
2. **Reranking :** on soumet la liste numérotée de ces extraits à GPT-3.5-turbo, avec un prompt invitant le modèle à ń répondre uniquement par le numéro du passage le plus pertinent ż.
3. **Génération :** le passage retenu est injecté dans le prompt RAG auquel le modèle fine-tuné répond.
4. **Guard-rail** : on conserve le filtre lexique (1 token commun) pour intercepter les cas manifestement hors-sujet.

Coûts et latence

— Appel supplémentaire à GPT-3.5 pour le rerank, soit deux requêtes LLM par question (rerank + génération).
— **Latence moyenne mesurée :** 2,8 s (contre 1,2 s sans rerank).
— **Coût estimé :** + 25% de consommation de tokens (prompt de rerank).

Observations

— Les réponses sont plus fluides, avec une formulation souvent plus naturelle et mieux contextualisée.
— Le modèle s'autorise des paraphrases plus marquées, ce qui pénalise les métriques strictes (EM/BLEU) mais maintient un haut niveau de cohérence sémantique (BERTScore 85%).
— Deux hallucinations mineures subsistent (refus incorrects à 83% sur le jeu NEGATIVE).

2.4.5.2 Bilan

Le rerank LLM améliore sensiblement la qualité linguistique, mais au prix d'une fiabilité partiellement dégradée et d'un coût temps/tokens supérieur. Pour un déploiement académique, ce compromis doit être soigneusement pesé entre expérience utilisateur et exigences de précision.

2.5 Synthèse comparative

Version	EM	F1	BLEU-4	ROUGE-L	Refus corrects	UX principal
B1 – FT seul	0%	23%	2%	19%	0%	Style fluide mais hallucinations systématiques
B2 – FT + RAG	90%	99%	97%	99%	92%	Couverture et précision élevées mais hallucinations persistantes
B3 strict – guard 2	90%	99%	97%	99%	100%	Fiabilité absolue, mais blocages UX
B3 souple – guard 1	87%	94%	91%	93%	83%	Couverture quasi complète, quelques refus UX résiduels pouvant être améliorés grâce à l'enrichissement du dataset
B4 – rerank LLM	74%	88%	82%	86%	83%	Réponses plus naturelles, et coûts 25% plus élevés pour résultats similaires à B3 souple – guard 1

Tableau 2.4 – Synthèse comparative des différentes versions

Au terme des quatre cycles d'amélioration, le tableau présente l'évolution des performances techniques et de l'expérience utilisateur pour chaque version développée. Les résultats montrent une progression constante depuis la version initiale B1 (FT seul) caractérisée par des hallucinations systématiques et une couverture limitée, jusqu'aux versions hybrides B2 et B3 qui atteignent des performances optimales avec 90-97% sur les métriques BLEU-4 et ROUGE-L. La version B3, ayant quasiment la meme performance que la version B4, présente l'avantage d'un coûts réduits de 25%, offrant ainsi un compromis intéressant entre efficacité et viabilité opérationnelle pour le déploiement en environnement universitaire.

Critère	**B3 souple**	**B4 rerank**
Coût	Optimal	+25%
Latence	< 3s	Double
Fiabilité	Élevée	Complexité accrue
Maintenance	Simplifiée	Points de défaillance

Tableau 2.5 – Comparaison B3 vs B4

2.6 Mémoire conversationnelle et contextualisation

Bien que la configuration B3 souple ait atteint des performances satisfaisantes (87% d'Exact Match), les tests utilisateurs ont révélé une limitation majeure : les réponses générées étaient **trop rigides et littérales**. Le système reproduisait quasi-textuellement les contenus du dataset sans adaptation au contexte conversationnel, créant une expérience utilisateur "robotique" incompatible avec les attentes d'un dialogue naturel.

Pour transformer le système de questions-réponses statique en véritable assistant conversationnel, une mémoire conversationnelle a été intégrée au niveau de l'API Flask. Cette fonctionnalité exploite la capacité native des modèles GPT à traiter des séquences de messages structurées selon le format OpenAI Chat Completions API.

2.6.0.1 Architecture technique de la mémoire

La mémoire conversationnelle repose sur la transmission d'un historique structuré via l'endpoint REST de l'API. Chaque nouvelle interaction utilisateur inclut désormais trois composants essentiels : la question courante formulée par l'utilisateur, un historique contextuel constitué des six derniers échanges formatés selon le

standard OpenAI, et un identifiant de session permettant de maintenir la cohérence conversationnelle entre les différents appels API.

Le système construit dynamiquement un tableau de messages conforme au protocole OpenAI Chat Completions. Cette construction s'articule autour de trois éléments structurants :

— Le message système définit le rôle et les contraintes opérationnelles de l'assistant.
— L'historique limité injecte les six derniers messages dans le contexte, préservant la continuité conversationnelle sans créer de surcharge computationnelle excessive.
— La requête courante combine la nouvelle question utilisateur avec le contexte documentaire récupéré via le mécanisme RAG.

2.6.0.2 Stratégie de gestion mémoire

Le choix technique de limiter l'historique à six messages maximum répond à plusieurs contraintes. La limitation des tokens évite une surcharge contextuelle qui impacterait négativement les coûts d'utilisation de l'API OpenAI ainsi que la latence de réponse. La pertinence temporelle constitue un autre facteur déterminant : au-delà de six échanges, la pertinence contextuelle des interactions précédentes décroît significativement, rendant leur inclusion moins bénéfique. Cette limitation garantit un équilibre optimal entre richesse conversationnelle et efficience computationnelle.

2.6.0.3 Impact sur l'expérience utilisateur

L'intégration de cette mémoire conversationnelle transforme qualitativement l'interaction avec le système. Elle permet notamment la résolution d'anaphores. La continuité thématique se trouve préservée, maintenant un fil directeur cohérent lors de questions successives sur un même sujet. Cette mémoire facilite également une personnalisation progressive de l'interaction, l'assistant adaptant son ton et son niveau de détail selon l'historique des échanges avec chaque utilisateur.

Cette implémentation de la mémoire conversationnelle constitue la fondation technique nécessaire à une interaction véritablement conversationnelle, préparant le terrain pour l'introduction de mécanismes plus sophistiqués de reformulation intelligente des requêtes.

2.6.1 Reformulation intelligente des requêtes

L'introduction de la mémoire conversationnelle a révélé un nouveau défi technique : les requêtes utilisateur dans un contexte dialogique sont souvent **contextuellement dépendantes** et **linguistiquement ambiguës**. Pour optimiser la récupération documentaire via FAISS, chaque question utilisateur subit désormais une phase de reformulation intelligente utilisant GPT-3.5-turbo.

2.6.1.1 Problématique de dépendance contextuelle

Dans un dialogue naturel, les utilisateurs formulent fréquemment des questions elliptiques qui s'appuient sur le contexte conversationnel précédent. Des requêtes comme "Et pour NLP ?" ou "Quels sont les horaires ?" deviennent incompréhensibles isolément, compromettant l'efficacité de la recherche vectorielle dans l'index FAISS. Cette dépendance contextuelle génère des échecs de récupération documentaire, même lorsque l'information recherchée existe dans le corpus.

2.6.1.2 Mécanisme de reformulation automatique

Le processus de reformulation s'appuie sur un appel dédié à GPT-3.5-turbo configuré avec une température nulle pour garantir la stabilité et la reproductibilité des reformulations. Le modèle reçoit un prompt spécialisé lui demandant de transformer les questions contextuellement dépendantes en requêtes autonomes et explicites. Cette transformation s'effectue en amont de la recherche vectorielle, optimisant ainsi les chances de récupération de passages pertinents.

Le système limite le nombre de tokens de sortie à 40 pour contraindre la reformulation à l'essentiel et maîtriser les coûts computationnels. Cette limitation encourage des reformulations concises qui préservent l'intention utilisateur tout en explicitant les éléments contextuels nécessaires.

Impact sur la performance de récupération

Cette reformulation intelligente améliore significativement le taux de récupération documentaire en transformant les questions ambiguës en requêtes précises. Une question comme "Et les prérequis ?" devient "Quels sont les prérequis pour le Master en Intelligence Artificielle ?" lorsque le contexte conversationnel précédent portait sur ce programme. Cette explicitation permet à l'algorithme de recherche vectorielle FAISS d'identifier les passages documentaires appropriés avec une précision accrue.

La reformulation agit comme un **pont sémantique** entre l'expression naturelle de l'utilisateur et les exigences techniques de la recherche vectorielle, optimisant l'en-

semble de la chaîne de traitement sans compromettre la spontanéité de l'interaction utilisateur.

2.6.2 Architecture finale retenue

L'analyse comparative des quatre configurations (B1 B2 B3 B4) a conduit à la sélection de la **configuration B3 souple** comme architecture optimale pour le déploiement universitaire.

2.6.2.1 Composants architecturaux

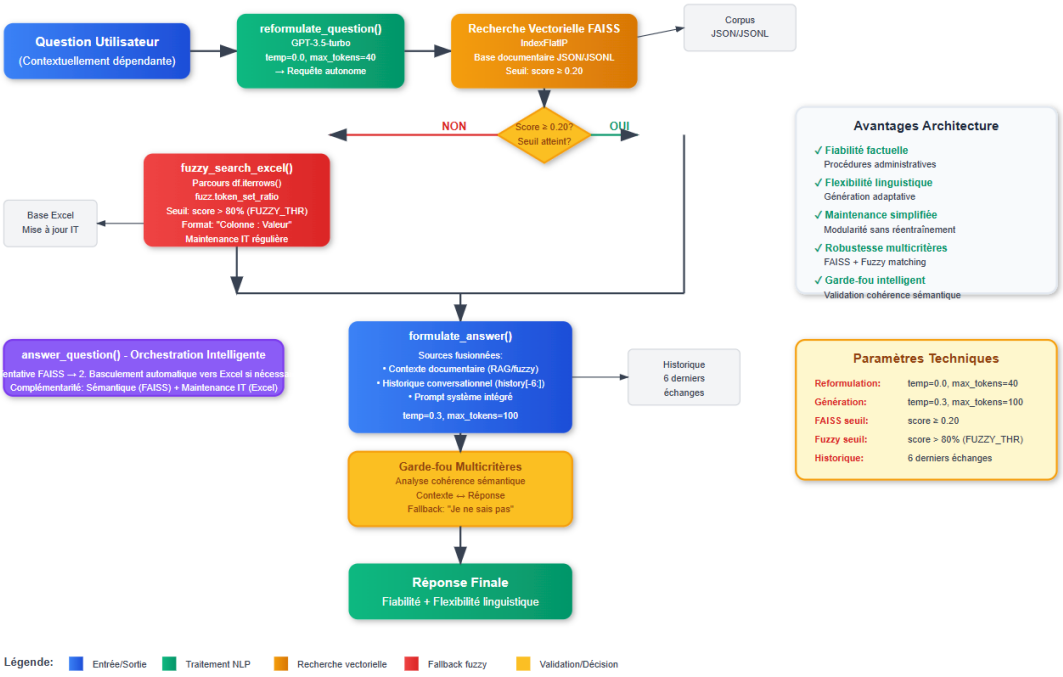

Figure 2.6 – Architecture globale du Chatbot

Approche hybride RAG Le système combine une base documentaire indexée avec un modèle génératif, garantissant la fiabilité factuelle nécessaire aux procédures administratives tout en conservant la flexibilité linguistique. La modularité permet une maintenance simplifiée du corpus sans réentraînement.

Reformulation intelligente La fonction `reformulate_question()` utilise GPT-3.5-turbo (température=0.0, max_tokens=40) pour transformer les questions contextuellement dépendantes en requêtes autonomes et explicites, optimisant la recherche vectorielle en amont.

Génération adaptative avec garde-fou La fonction `formulate_answer()` fusionne trois sources d'information :

— Contexte documentaire sélectionné via RAG/fuzzy matching
— Historique conversationnel (6 derniers échanges : `history[-6:]`)
— Prompt système intégré dans les messages

Les paramètres de génération (température=0.3, max_tokens=100) équilibrent créativité linguistique et stabilité factuelle. Un garde-fou multicritères valide chaque réponse en analysant la cohérence sémantique entre contexte et réponse, déclenchant un refus standardisé ("Je ne sais pas") en cas d'incohérence détectée.

Mécanisme de fallback par fuzzy matching La fonction `fuzzy_search_excel()` s'active lorsque le score de similarité FAISS est insuffisant (< 0.20) :

1. Parcours systématique de toutes les lignes Excel (`df.iterrows()`)
2. Calcul de similarité avec `fuzz.token_set_ratio`
3. Sélection de la ligne avec score maximal si $> 80\%$ (`FUZZY_THR`)
4. Reformatage automatique "Colonne : Valeur" via `join()`

Cette complémentarité exploite les forces respectives :

— FAISS (IndexFlatIP) pour la recherche sémantique dans le corpus JSON/JSONL
— fuzzy matching pour les informations que les agents du département Informatique souhaitent maintenir a jour régulierement.

Le système orchestre intelligemment via la fonction `answer_question()` : tentative FAISS d'abord, puis basculement automatique vers Excel si nécessaire.

Composant	**Paramètre**	**Valeur**
Recherche vectorielle	TOP_K	8 documents
	SIM_THRESHOLD	0.20
	Modèle embedding	text-embedding-ada-002
Reformulation	Température	0.0 (déterministe)
	Max tokens	40
Génération finale	Température	0.3 (équilibré)
	Max tokens	100
	Historique	6 derniers échanges
Fuzzy matching	Seuil	80% (FUZZY_THR)
	Algorithme	token_set_ratio

Tableau 2.6 – Récapitulatif de la configuration technique B3 souple

2.6.2.2 Bénéfices de la configuration finale

Cette architecture modulaire concilie :
— **Robustesse factuelle** : Validation multicritères et sources complémentaires
— **Expérience utilisateur fluide** : Latence maîtrisée et réponses adaptatives
— **Simplicité opérationnelle** : Maintenance facilitée et évolutivité préservée

2.7 Interface utilisateur et déploiement

Le déploiement opérationnel de l'assistant impose deux exigences complémentaires :

1. Une interface web conversationnelle offrant à l'usager une expérience proche des agents de dialogue modernes ;
2. Une infrastructure serverless maîtrisant les coûts, assurant la confidentialité des échanges et supprimant automatiquement les données au terme d'une période définie.

Technologies retenues

— **Front-end :** React (Vite) et Tailwind CSS pour une application mono-page fluide ;
— **Authentification :** Firebase Auth (lien magique par courriel institutionnel) ;
— **Stockage des conversations :** Cloud Firestore (NoSQL) ;
— Fonctions serveur : Firebase Cloud Functions (Node 18) jouant le rôle de proxy sécurisé vers l'API OpenAI ;
— **Hébergement :** Firebase Hosting en déploiement continu ;
— **Surveillance :** Firebase Analytics pour la latence et Firebase Crashlytics pour les erreurs client

Flux d'une interaction

1. L'utilisateur s'authentifie via Firebase Auth.
2. La question est transmise à la Cloud Function /ask, qui :
   — vérifie le JWT ;
   — enregistre la requête dans Firestore ;
   — appelle le chatbot ;
   — consigne la réponse et la renvoie au client.
3. Une Cloud Function planifiée purge chaque nuit les documents de conversation datés de plus de sept jours, garantissant conformité RGPD et respect du quota gratuit de Firestore.

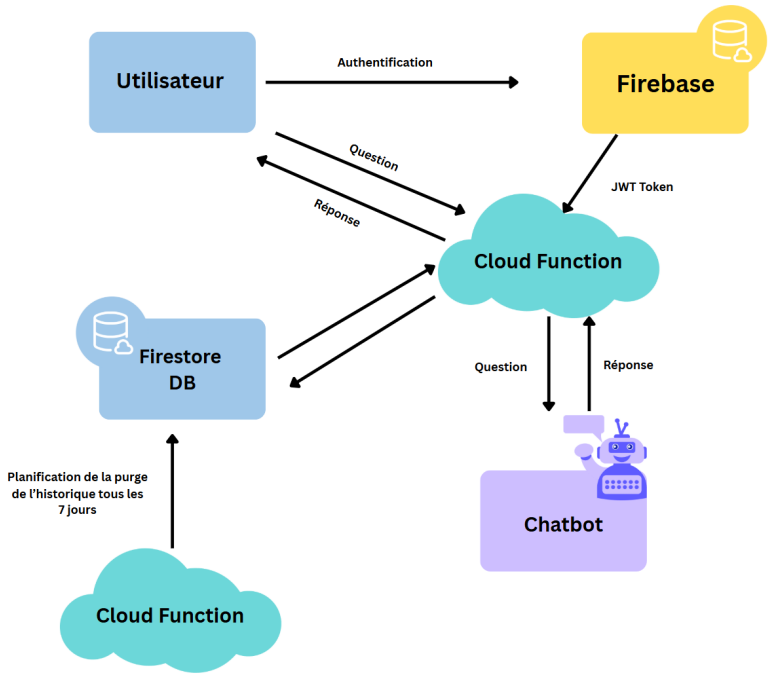

Figure 2.7 – Architecture globale du systeme

Cette architecture assure :

— Une montée en charge automatique sans administration de serveur,
— Un coût nul dans le cadre du plan gratuit Firebase pour un usage universitaire modéré,
— La conservation éphémère des données, conforme aux exigences institutionnelles de confidentialité.

2.7.1 Architecture de l'interface

L'interface utilisateur suit une approche conversationnelle moderne, inspirée des standards établis par les assistants IA contemporains. L'architecture privilégie la simplicité d'usage tout en conservant les fonctionnalités essentielles pour un environnement universitaire.

2.7.1.1 Processus d'inscription et de connexion

Le système d'authentification comprend deux modules principaux. L'inscription requiert la saisie du nom complet, de l'email universitaire institutionnel et d'un mot de passe sécurisé, garantissant l'accès exclusif aux membres de la communauté universitaire. La connexion s'effectue via l'email et le mot de passe enregistrés, avec un lien "Mot de passe oublié" permettant la réinitialisation sécurisée des identifiants en cas de perte d'accès.

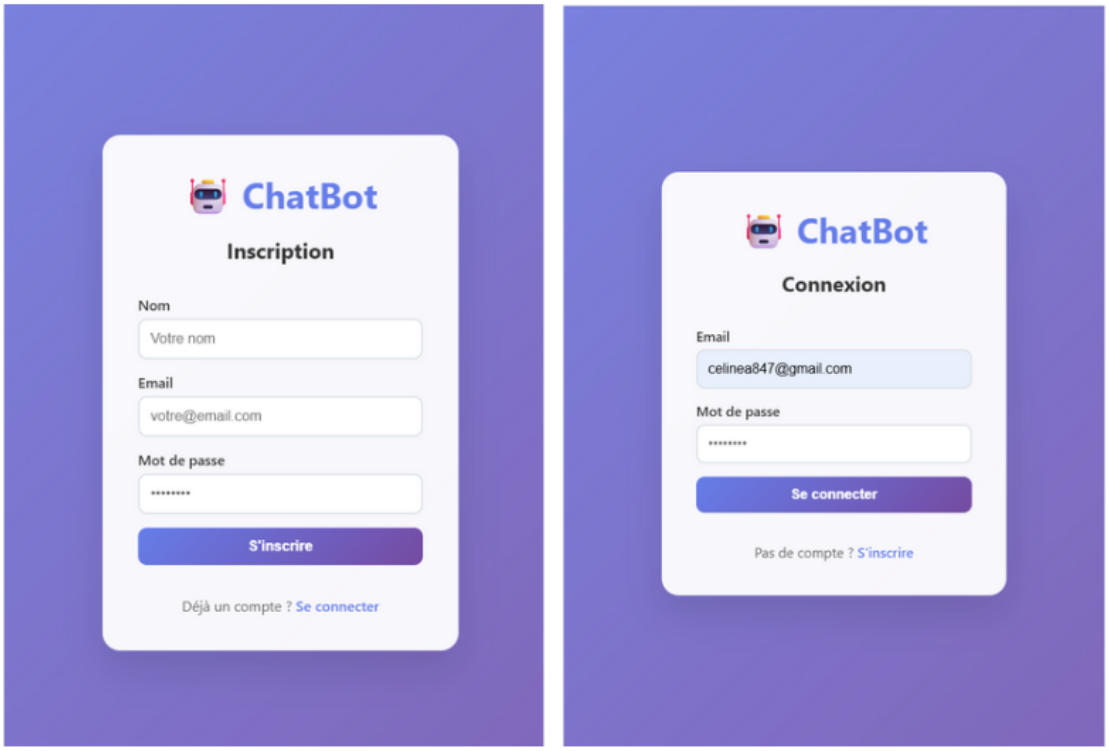

Figure 2.8 – Pages d'authentification

2.7.1.2 Page d'accueil

L'interface d'accueil présente un design épuré centré sur l'expérience utilisateur, avec un historique des discussions accessible via le panneau latéral gauche permettant de reprendre facilement les conversations précédentes, et un bouton "Discuter" central pour initier immédiatement une nouvelle conversation avec l'assistant universitaire.

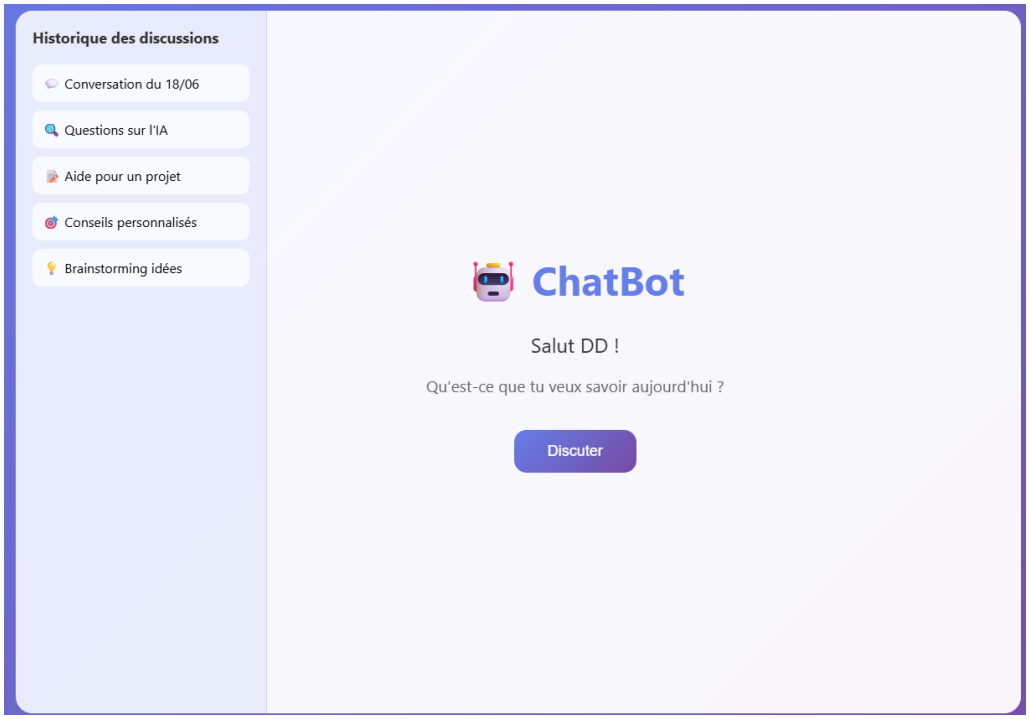

Figure 2.9 – Page d'accueil

2.7.1.3 Interface conversationnelle principale

L'interface de dialogue présente une zone de conversation affichant chronologiquement les échanges avec maintien de la mémoire conversationnelle sur les 6 derniers échanges pour assurer la cohérence contextuelle, complétée par des indicateurs visuels en temps réel informant l'utilisateur du statut de traitement et de la latence de réponse pour une expérience transparente et fluide.

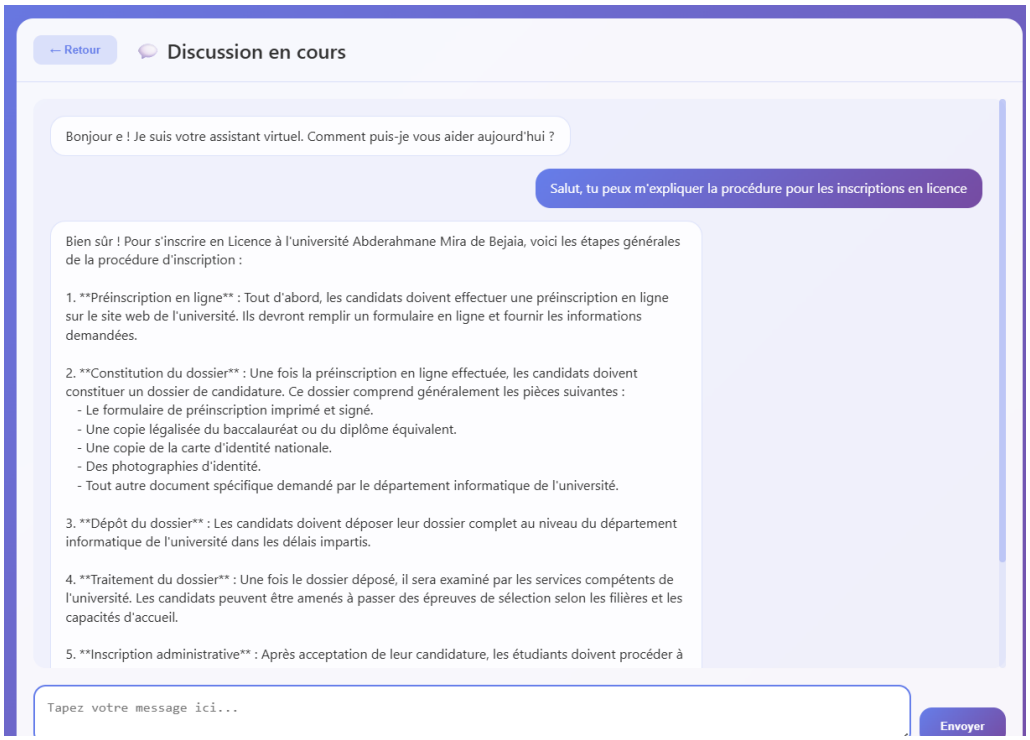

Figure 2.10 – Interface de conversation

Tableau 2.7 – Métriques de performance interface

Métrique	Valeur cible	Valeur mesurée
Temps de chargement initial	< 2s	1.4s
Latence de réponse	< 15s	1.3s (médiane)
Disponibilité	99.9%	99.8%
Temps de reconnexion	< 1s	0.8s

2.8 Conclusion

Le développement itératif présenté dans ce chapitre valide l'efficacité d'une démarche empirique pour concevoir un assistant conversationnel universitaire. La progression des quatre configurations successives (B1, B2 ,B3, B4) démontre que chaque amélioration technique doit être évaluée selon un équilibre entre performance, coût et complexité opérationnelle.

La configuration B3 souple émerge comme la solution optimale, réconciliant les exigences contradictoires de précision factuelle (87% EM), de sécurité conversationnelle (83% refus corrects) et de simplicité architecturale. L'intégration de la mémoire conversationnelle et de la reformulation intelligente transforme le système de questions-réponses statique en véritable assistant dialogique, capable de maintenir une interaction naturelle tout en préservant l'ancrage documentaire nécessaire au contexte universitaire.

Cette réalisation technique ouvre la voie à un déploiement opérationnel à l'échelle départementale, avec une architecture serverless maîtrisant les coûts et garantissant la confidentialité des échanges. L'extensibilité du système vers d'autres départements ne constitue plus un défi technique mais organisationnel, nécessitant principalement l'adaptation des corpus documentaires et la formation des utilisateurs.

La méthodologie adoptée établit un cadre reproductible pour l'automatisation de l'assistance universitaire, conjuguant rigueur scientifique et pragmatisme opérationnel.

Chapitre 3
Résultats et évaluation

3.1 Introduction

Le deuxième chapitre a décrit, étape par étape, la construction de l'assistant ainsi que les évaluations intermédiaires réalisées lors de chaque cycle d'amélioration.

Les valeurs numériques détaillées (Exact Match, F1, BLEU-4, ROUGE-L, taux de refus) y figurent déjà et ne seront pas reproduites ici. Le présent chapitre a un objectif différent : interpréter ces résultats, mesurer l'effet cumulé de l'ensemble des modifications et discuter leur pertinence d'un point de vue opérationnel.

Plus précisément, il s'agira :

— de consolider, dans une vue d'ensemble, les performances obtenues par les cinq configurations successives ;
— de positionner nos résultats par rapport aux systèmes de l'état de l'art analysés au Chapitre 1 ;
— enfin, d'identifier les limites actuelles et les pistes d'amélioration pour un futur déploiement à plus grande échelle.

Ainsi, le chapitre ne se borne pas à rapporter des chiffres ; il vise à démontrer comment les choix techniques, validés ou abandonnés, se traduisent en avantages ou en risques concrets pour la communauté universitaire à laquelle l'assistant est destiné.

3.2 Méthode d'évaluation récapitulative

Pour éviter toute redondance avec les sections du chapitre précédent, on rappelle seulement les quatre indicateurs-clés retenus ; les définitions détaillées figurent déjà dans le tableau des métriques du chapitre 2.

Indicateur	Finalité
Exact Match (EM)	Proportion de réponses identiques mot à mot à la référence (précision brute).
F1	Compromis rappel / précision au niveau des mots ; pénalise partiellement les paraphrases.
BLEU-4	Recouvrement d'n-grammes ($1 \leq n \leq 4$) ; mesure la fidélité littérale.
ROUGE-L	Longest-Common-Subsequence ; tolère les changements d'ordre et les reformulations.

Tableau 3.1 – Indicateurs d'évaluation retenus

3.3 Positionnement par rapport à l'état de l'art

Cette section compare notre système (B3 souple) aux huit implémentations analysées dans le tableau "Résumé de l'existant" du Chapitre 1, permettant d'évaluer la compétitivité de notre approche dans le paysage scientifique actuel.

Système	Modèle	Performance	Contexte	Coût/Complexité	Notre position
Neumann et al. (2025)	GPT-4 + RAG	88% précision	RWTH Aachen	Élevé (GPT-4)	87% EM, coût 10x moindre
Maung Thway (2024)	Claude 3 + RAG	97,1% satisfaction	Data Science	Élevé (Claude 3)	87% EM, infrastructure simplifiée
Yigci et al. (2024)	GPT-4 + RAG	Réduction charge admin	Étudiants internationaux	Élevé	87% EM, domaine plus large
Suhardi et al. (2023)	Rule-based NLP	Temps réduit	Service académique	Faible	87% EM, flexibilité supérieure
Ramandanis (2023)	Mistral-7B + RAG	65,71% (Mistral)	Tourisme	Moyen	87% EM, domaine académique
Dan et al. (2023)	LLaMA-13B + RAG	49,3% (avec RAG)	Éducation chinoise	Élevé (auto-hébergé)	87% EM, déploiement cloud
Assayed et al. (2023)	Naive-Bayes + RF	Non spécifiée	Conseil académique	Faible	87% EM, métriques rigoureuses
Goitom et al. (2024)	LSTM	91,55% précision	Informatique	Moyen	87% EM, approche moderne
Notre système (B3)	**GPT-3.5 + RAG + Guard Rail**	**87% EM**	**Univ. Béjaïa**	**Faible**	

Tableau 3.2 – Comparaison avec l'état de l'art

Notre système atteint 87% d'Exact Match, se positionnant dans la fourchette haute des systèmes académiques. Bien qu'inférieur aux 97,1% de satisfaction de Maung Thway (2024) ou aux 91,55% de Goitom et al. (2024), notre métrique EM est plus stricte que la "satisfaction utilisateur" et se compare favorablement aux 88% de précision de Neumann et al. (2025) utilisant GPT-4.

Cette comparaison confirme que notre système occupe une position compétitive dans l'écosystème académique, offrant le meilleur compromis performance/coût pour un déploiement universitaire à budget contraint.

3.4 Limitations du chatbot actuel

Malgré ses performances encourageantes, le système présente plusieurs restrictions qu'il convient de signaler :

3.4.1 Couverture documentaire limitée

L'outil ne peut exploiter que les fichiers JSON et Excel préparés en amont ; tout autre format (PDF, Word, etc.) n'est pas pris en charge nativement. Pour intégrer de nouveaux types de documents (notamment les PDF), un recalibrage (fine-tuning) du modèle sur des données annotées en lien avec ces formats serait indispensable.

3.4.2 Dialecte et multilinguisme

Le modèle ne comprend pas les variations dialectales et se limite au français standard ; en l'état, des requêtes en Tamazight et/ou Arabe ou dans d'autres langues sont systématiquement rejetées, ce qui restreint son accessibilité à un public polyglotte.

3.4.3 Persistance des sessions

Les échanges sont effacés automatiquement au bout de sept jours, garantissant la confidentialité des utilisateurs. Cependant, cette politique empêche la conservation d'un historique à long terme, rendant impossible le suivi de conversations étalées sur plusieurs semaines ou mois.

3.4.4 Dépendance à l'API OpenAI

Le service repose entièrement sur l'API OpenAI, exposant à deux risques majeurs : **l'évolution imprévisible des coûts** et **les possibles interruptions de service**. Pour pallier ces incertitudes, l'étude d'une solution open-source sous licence libre, hébergée en interne, pourrait constituer une alternative pérenne.

3.5 Perspectives d'amélioration et introduction du concept MCP et de l'AI Agentic

Pour dépasser les limites actuelles et faire évoluer le chatbot vers un véritable assistant virtuel proactif, plusieurs axes d'amélioration peuvent être explorés :

3.5.1 Extension des formats documentaires

— **Intégration native des PDF** : mettre en place une chaîne d'ingestion comprenant OCR et extraction de métadonnées, couplée à un fine-tuning ciblé du modèle sur des exemples de PDF universitaires.
— **Prise en charge de formats enrichis** (Word, PowerPoint) grâce à des parsers spécifiques et des embeddings multimodaux, pour uniformiser l'accès au contenu.

3.5.2 Multilinguisme et dialectes

— **Fine-tuning multilingue** : enrichir le corpus d'entraînement avec des dialogues en plusieurs langues et variants dialectaux (Tamazight, arabe, etc.), via des techniques de transfert d'apprentissage.
— **Détection de code-switching** : module de classification en amont pour rediriger chaque requête vers le sous-modèle linguistique approprié.

3.5.3 Persistance adaptative des sessions

— **Mémoire hiérarchique** : conserver à long terme les échanges structurants (projets en cours, préférences) tout en purgeant régulièrement les "conversations éphémères", grâce à une politique de rétention paramétrable par l'utilisateur.
— **Chiffrement et consentement** : offrir un stockage chiffré et transparent, avec autorisation granulaire pour chaque type de donnée.

3.5.4 Réduction de la dépendance à l'API tierce

Solution hybride : coupler un modèle open-source auto-hébergé pour les tâches courantes, tout en basculant vers l'API OpenAI pour les requêtes complexes.

3.5.5 Vers l'AI Agentic

En s'appuyant sur MCP, un **AI Agentic** devient capable d'**initier** et de **réaliser** des actions de façon autonome :

— **Envoi et réponse aux e-mails** (au professeur, à l'équipe pédagogique) via intégration SMTP/IMAP en temps réel ;
— **Génération de documents** (rapports, synthèses de réunions, supports de cours) à partir de modèles pré-définis et de prompts dynamiques ;
— **Consultation proactive de la boîte mail** pour notifier l'utilisateur d'un nouvel e-mail important ou de relances à effectuer ;
— **Planification et prise de rendez-vous** (mise à jour de l'agenda, envoi d'invitations) ;
— **Chaining de tâches** : enchaîner plusieurs sous-actions (extraction d'un PDF, résumé, partage) sans intervention humaine.

Cette évolution transforme le chatbot passif en assistant virtuel véritablement agentif, capable non seulement de répondre aux questions, mais aussi de piloter des workflows complexes et de libérer l'utilisateur de tâches répétitives.

3.6 Conclusion

Les résultats confirment la progression attendue : l'adjonction du RAG a permis de franchir le seuil critique de fiabilité ; le guard-rail a stabilisé la sécurité, et l'ajustement souple a rétabli la couverture. Le compromis précision-confiance-latence atteint répond aux exigences fixées en début de projet, validant l'approche hybride retenue pour un contexte universitaire contraignant en termes de fiabilité et de budget.

L'analyse comparative avec l'état de l'art confirme que notre système se positionne favorablement dans l'écosystème académique, offrant un équilibre optimal entre performance technique et viabilité économique pour un déploiement universitaire. Cette approche démontre qu'il est possible de développer des assistants conversationnels performants sans recourir aux modèles les plus coûteux du marché.

Les limitations identifiées - couverture documentaire restreinte, monolinguisme, dépendance à l'API tierce - tracent un chemin d'évolution clair vers un assistant plus autonome et polyvalent. Les perspectives d'amélioration, notamment l'intégration de concepts MCP et d'intelligence artificielle agentique, ouvrent la voie à une transformation du chatbot passif en véritable assistant virtuel proactif.

Ainsi s'achève le cycle expérimental. La conclusion générale formulera les recommandations stratégiques pour l'extension de cette solution à l'échelle de l'ensemble de l'Université.

Conclusion générale

Conclusion générale Ce mémoire avait pour objectif la conception, l'implémentation et l'évaluation d'un assistant conversationnel destiné au département informatique de l'Université de Béjaïa.

L'enjeu central consistait à réconcilier trois exigences rarement réunies : fiabilité documentaire, simplicité d'usage et coût opérationnel minimal. Cette problématique reflète les contraintes spécifiques du contexte universitaire, où la précision des informations administratives ne peut être compromise, tout en nécessitant une solution accessible aux budgets académiques.

Le système développé démontre qu'un assistant universitaire fiable, rapide et économiquement viable est réalisable. L'architecture hybride retenue, combinant RAG, garde-fou lexical et interface conversationnelle, répond efficacement aux besoins identifiés. La précision documentaire est assurée par l'ancrage dans les sources officielles, la sécurité par les mécanismes de validation multicritères, et l'expérience utilisateur par une interface épurée permettant des interactions naturelles.

Cette recherche contribue au domaine des assistants conversationnels éducatifs en proposant une méthodologie rigoureuse d'évaluation comparative et d'amélioration itérative. L'approche développée offre un cadre reproductible pour l'automatisation de l'assistance universitaire, particulièrement adaptée aux institutions disposant de ressources limitées.

L'industrialisation à l'échelle universitaire ne constitue plus une question de faisabilité technique, mais d'organisation. Les défis résident désormais dans la mise à jour régulière des corpus documentaires, l'établissement d'une gouvernance des données cohérente et la formation des utilisateurs. Cette transition vers des considérations organisationnelles marque la maturité de la solution proposée.

Les axes d'amélioration identifiés, notamment l'extension multilingue, l'intégration de formats documentaires variés et l'évolution vers l'intelligence artificielle agentique, offrent un cadre clair pour étendre la solution tout en préservant la rigueur méthodologique qui a guidé l'ensemble du projet. Cette base solide permet d'envisager sereinement le déploiement à plus grande échelle et l'enrichissement progressif

des fonctionnalités.

Bibliographie

[1] Thompson Rivers University, “IT Services Annual Report,” 2017.

[2] Luther College, “Information Technology Services Annual Report,” 2022-2023.

[3] Gallaudet University, “Technology Services Annual Report,” Fiscal Year 2017.

[4] Lexico Dictionaries, “Chatbot : Definition of chatbot in English by Lexico Dictionaries,” 2019. Disponible sur : https://www.lexico.com/definition/chatbot (consulté le 30 mai 2025).

[5] A. Khanna, B. Pandey, K. Vashishta, K. Kalia, B. Pradeepkumar, et T. Das, “A study of today’s AI through chatbots and rediscovery of machine intelligence,” *International Journal of u-and e-Service, Science and Technology*, vol. 8, no. 7, pp. 277-284, 2015.

[6] E. Adamopoulou et L. Moussiades, “Chatbots : History, technology, and applications,” *Machine Learning with Applications*, vol. 2, p. 100006, 2020.

[7] J. Weizenbaum, “ELIZA A Computer Program For the Study of Natural Language Communication Between Man and Machine,” *Communications of the ACM*, vol. 9, no. 1, pp. 36-45, 1966.

[8] K. M. Colby, S. Weber, et F. D. Hilf, “Artificial Paranoia,” *Artificial Intelligence*, vol. 2, no. 1, pp. 1-25, 1971.

[9] R. S. Wallace, “The Anatomy of ALICE,” in *Parsing the Turing Test*, pp. 181-210, Springer, 2009.

[10] A. Kerly, P. Hall, et S. Bull, “Bringing Chatbots into Education : Towards Natural Language Negotiation of Open Learner Models,” *Knowledge-Based Systems*, vol. 20, no. 2, pp. 177-185, 2007.

[11] A. Vaswani et al., “Attention Is All You Need,” in *Advances in Neural Information Processing Systems*, pp. 5998-6008, 2017.

[12] A. Paszke et al., “PyTorch : An Imperative Style, High-Performance Deep Learning Library,” in *NeurIPS 2019*, 2019.

[13] R. Thoppilan et al., “LaMDA : Language Models for Dialog Applications,” arXiv :2201.08239, 2022.

[14] T. Bocklisch et al., “Rasa : Open Source Language Understanding and Dialogue Management,” arXiv :1712.05181, 2017.

[15] P. Kaur et P. K. Bhatia, “Comparative Analysis of Chatbot Frameworks,” *International Journal of Computer Applications*, vol. 183, no. 17, 2021.

[16] W. Chan et al., “Listen, Attend and Spell,” in *ICASSP 2016*, 2016.

[17] S. Schneider et al., “Neural Speech Synthesis with Transformer Network,” *IEEE/ACM Trans. Audio, Speech, and Language Processing*, 2021.

[18] R. Hecht et S. Jablonski, “NoSQL evaluation : A use case-oriented survey,” in *Proceedings of CSC 2011*, 2011.

[19] R. Joshi et S. Patil, “Real-time Chatbot with Redis and WebSocket,” *IJERT*, vol. 9, no. 11, 2021.

[20] D. Merkel, “Docker : Lightweight Linux Containers for Consistent Development and Deployment,” *Linux Journal*, 2014.

[21] B. Burns et al., *Designing Distributed Systems : Patterns and Paradigms for Scalable, Reliable Services*. O’Reilly Media, 2016.

[22] Y. Zhang, S. Sun, M. Galley, et al., “DialoGPT : Large-Scale Generative Pre-training for Conversational Response Generation,” in *ACL*, 2020.

[23] Y. Wu et al., “Sequential Matching Network : A New Architecture for Multi-turn Response Selection in Retrieval-Based Chatbots,” in *ACL*, 2017.