

Rapport de Projet

Scholarly+ : Système de Recherche et d'Assistance Académique

Réalisée par :

Alae ABID
Firdaous FATMI
Oumaima HSSAINE

Encadrée par:

Mr. Hakim
HAFIDI

Mr. Hamza
GAMOUH

Table des matières

1	Introduction	2
1.1	Contexte et Motivation	2
1.2	Objectifs du Projet	2
2	Préparation et Architecture du Système	3
2.1	Frontend (Page Web)	4
2.1.1	Recherche d'articles	4
2.1.2	Chatbot intelligent	5
2.2	Backend	5
2.3	Base de Données	6
2.3.1	Pinecone	7
2.3.2	mongoDB	8
3	Méthodologie	10
3.1	Architecture du projet	10
3.2	Collection et Préparation des Données	10
3.3	Développement de l'Interface Utilisateur	11
3.4	Intégration du Chatbot pour la Recherche Contextuelle	12
4	Résultats du Projet	12
4.1	Présentation des Résultats finaux	12
4.1.1	barre de recherche	12
4.1.2	chatbot	13
5	Conclusion	14
6	Références	14

1 Introduction

Le développement d'outils intelligents pour la recherche d'informations scientifiques est devenu crucial à une époque où la quantité de publications dans des domaines comme l'apprentissage automatique (Machine Learning) croît de manière exponentielle. Ce projet vise à concevoir une application Web innovante, Scholarly+, qui implémente les fonctionnalités de base d'un système de recherche sémantique et d'assistance interactive pour les publications scientifiques en Machine Learning. L'application s'appuie sur des technologies modernes telles que Python, MongoDB, Pinecone, Gemini AI, SentenceTransformer et Streamlit, pour offrir une plateforme robuste et conviviale. Les utilisateurs peuvent rechercher des articles de recherche en entrant des mots-clés et accéder à des informations clés comme le titre, l'abstract, les auteurs, l'année de publication et un lien vers le PDF source. Une fonctionnalité avancée de chatbot, alimentée par Gemini AI, permet également aux utilisateurs d'interagir avec le système pour poser des questions et obtenir des réponses contextualisées basées sur les articles trouvés. L'objectif principal de ce projet est de fournir un accès rapide, pertinent et personnalisé à la recherche scientifique en exploitant des techniques avancées de recherche sémantique et d'intelligence artificielle. Le système met en œuvre des optimisations comme le cache des résultats via MongoDB et l'utilisation d'une indexation vectorielle efficace avec Pinecone pour garantir des performances élevées. Ce rapport détaillera les différentes étapes de conception et de développement de l'application, les approches techniques utilisées et les résultats obtenus, illustrant ainsi comment Scholarly+ simplifie l'exploration de la littérature scientifique en apprentissage automatique.

1.1 Contexte et Motivation

Dans le contexte actuel de l'ère numérique, l'accès à une quantité croissante d'informations scientifiques et techniques est un défi majeur pour les chercheurs et les professionnels du domaine. La recherche d'articles académiques, notamment dans le domaine du machine learning, nécessite souvent de naviguer dans un grand volume de données non structurées. Les moteurs de recherche traditionnels, basés principalement sur des mots-clés, peuvent ne pas être suffisants pour fournir des résultats pertinents et détaillés. Ainsi, la motivation derrière ce projet est de développer une plateforme web intégrant une barre de recherche avancée permettant de trouver des articles scientifiques pertinents sur des sujets liés au machine learning, tout en fournissant des informations clés telles que les résumés, les auteurs, et les années de publication. Un autre aspect clé du projet est l'intégration d'un chatbot utilisant des modèles d'IA pour répondre de manière contextuelle aux questions des utilisateurs, améliorant ainsi l'expérience de recherche.

1.2 Objectifs du Projet

Le projet Scholarly+ a pour objectif de développer une application Web intelligente pour faciliter la recherche d'informations scientifiques en apprentissage automatique (Machine Learning). S'appuyant sur des technologies avancées telles que Python, MongoDB, Pinecone,

Gemini AI, SentenceTransformer et Streamlit, cette application permet aux utilisateurs de rechercher des articles pertinents en entrant des mots-clés. Les résultats incluent des informations clés comme le titre, l'abstract, les auteurs, l'année de publication et un lien vers le PDF source. De plus, une fonctionnalité de chatbot interactif, alimentée par Gemini AI, offre des réponses contextualisées basées sur les articles trouvés. Grâce à une gestion efficace du cache via MongoDB et une recherche sémantique rapide avec Pinecone, Scholarly+ offre une expérience utilisateur fluide, optimisant l'accès aux connaissances pour les chercheurs, étudiants et passionnés du domaine.

2 Préparation et Architecture du Système

L'architecture du système de Scholarly+ est conçue pour offrir une solution performante et scalable permettant la gestion et la recherche d'articles scientifiques via une interface interactive.

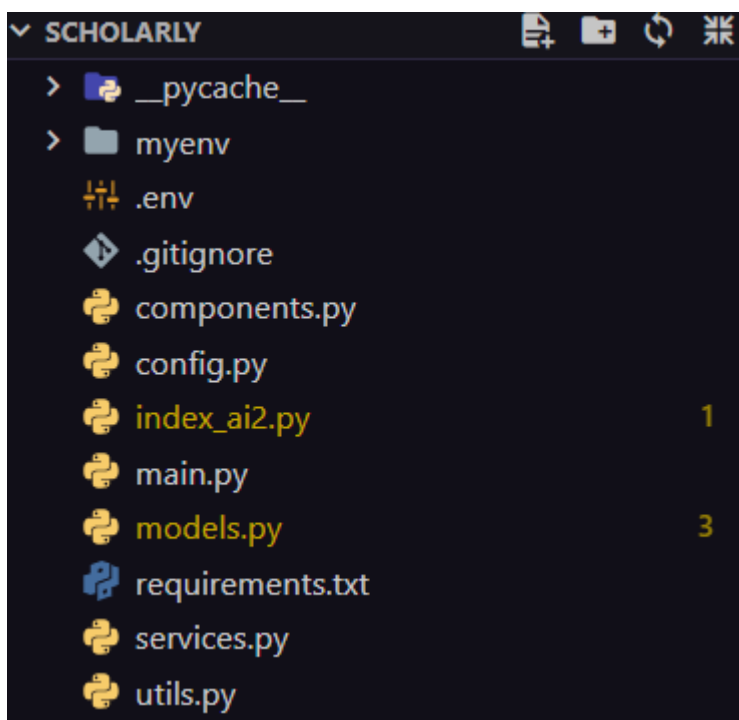


FIGURE 1 – structure de projet

2.1 Frontend (Page Web)

Le frontend est développé à l'aide de Streamlit, un framework Python idéal pour créer des page web interactives et réactives. Les fonctionnalités clés du frontend incluent :



FIGURE 2 – logo streamlit

2.1.1 Recherche d'articles

Les utilisateurs peuvent effectuer des requêtes pour trouver des articles pertinents en fonction de mots-clés ou de thèmes spécifiques. Les articles trouvés sont présentés sous forme de liste avec des détails tels que le titre, l'auteur, l'année de publication et un lien vers le PDF.

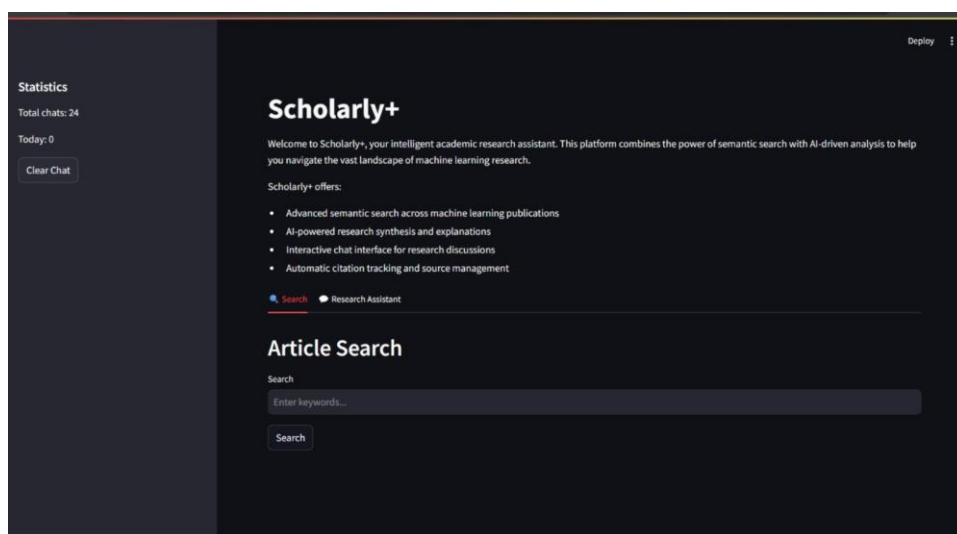


FIGURE 3 – barre de recherche

2.1.2 Chatbot intelligent

Le frontend intègre un chatbot permettant aux utilisateurs de poser des questions contextuelles pour des analyses plus approfondies. Les utilisateurs peuvent interagir avec le chatbot pour poser des questions sur les résultats de recherche, comme des explications sur le contexte des articles trouvés ou des éclaircissements sur les thèmes abordés dans les articles. Cela ajoute une dimension dynamique et personnalisée à l'expérience utilisateur.

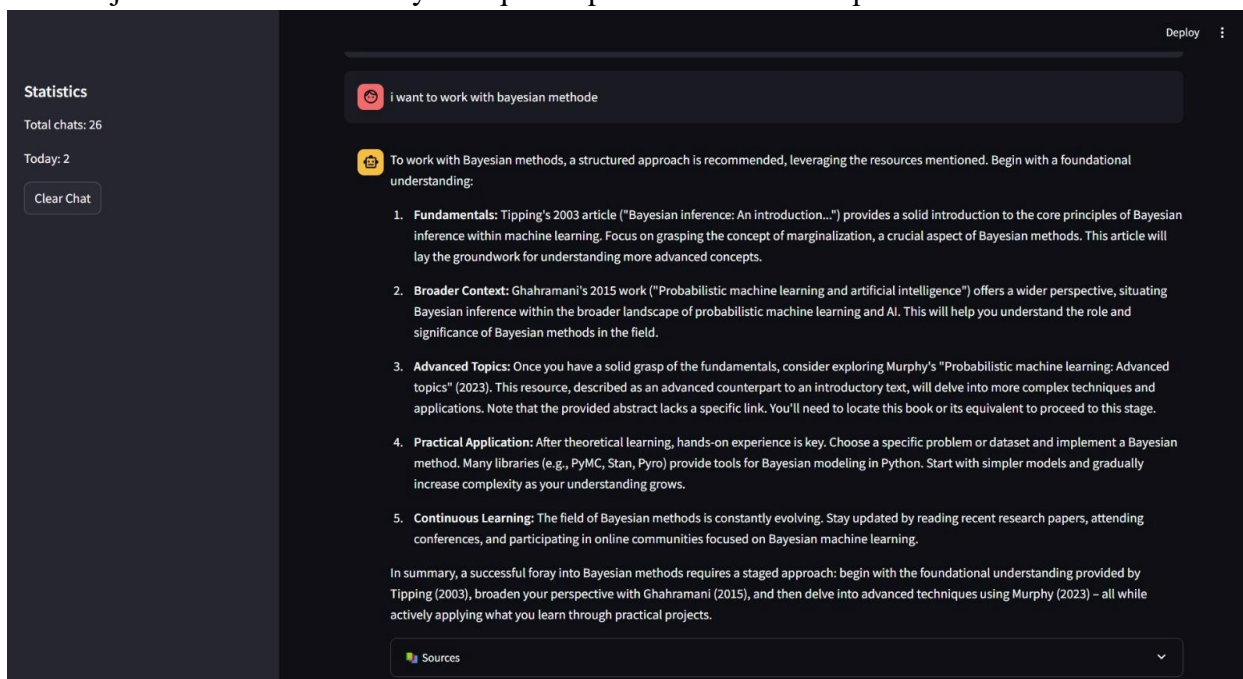


FIGURE 4 – chatbot interface

2.2 Backend

L'architecture backend de Scholarly+ est conçue pour offrir une plateforme intelligente et fluide d'assistance à la recherche académique. Elle intègre diverses technologies de pointe pour gérer le stockage des données, la recherche sémantique et l'analyse alimentée par l'IA.

Perspectives alimentées par l'IA avec Gemini 1.5 Flash : Le modèle génératif Gemini 1.5 Flash de Google est utilisé pour fournir des réponses détaillées et contextualisées aux requêtes des utilisateurs. Configuré avec un paramétrage de génération sur mesure, il génère des réponses à la fois précises et en adéquation avec le contexte de recherche fourni.

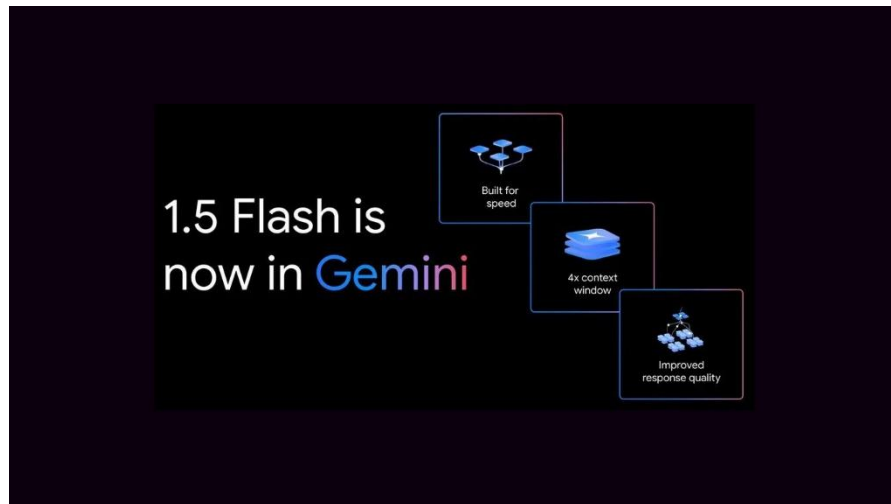


FIGURE 5 – logo gemini

Gestion de l'état de session : Le backend gère l'état de session spécifique à chaque utilisateur afin de conserver les messages de chat et les résultats de recherche entre les interactions. Cela permet une expérience personnalisée et interactive où les utilisateurs peuvent revoir les requêtes et résultats précédents.

Intégration et évolutivité : Le backend est conçu pour évoluer de manière fluide en fonction de la demande des utilisateurs. En combinant des API et des bibliothèques avec une configuration robuste, il garantit une haute disponibilité et des performances optimales.

2.3 Base de Données

L'intégration entre MongoDB et Pinecone dans le système permet de gérer efficacement les requêtes de recherche et les résultats. Lorsqu'un utilisateur effectue une recherche, le système vérifie d'abord le cache MongoDB pour trouver les résultats précédemment stockés liés à la requête. Si les résultats sont trouvés dans le cache, ils sont immédiatement récupérés et affichés, ce qui permet de gagner du temps et des ressources. Si les résultats ne sont pas dans le cache, Pinecone est utilisé pour effectuer une recherche sémantique en générant un vecteur d'embedding de la requête et en interrogeant l'index Pinecone. Les résultats de Pinecone sont ensuite stockés dans MongoDB pour une utilisation future, garantissant que le système peut fournir des réponses plus rapides pour les requêtes suivantes. Cette approche améliore les performances en combinant les avantages des deux bases de données : MongoDB pour un stockage et une récupération rapides, et Pinecone pour une recherche sémantique évolutive et haute performance.

2.3.1 Pinecone

Cette base de données vectorielle est utilisée pour stocker les embeddings des articles scientifiques. Les embeddings sont des représentations numériques générées à partir des titres, des auteurs et des résumés à l'aide du modèle SentenceTransformer. Chaque embedding est un vecteur dans un espace multidimensionnel, permettant de mesurer la similarité sémantique entre les articles et les requêtes.






FIGURE 6 – logo pinecone

Indexation : Lors de l'ingestion des données, chaque article scientifique est transformé en embedding et indexé dans Pinecone.



FIGURE 7 – index

Cette indexation permet des recherches ultrarapides basées sur la distance entre vecteurs (par exemple, la distance cosinus ou euclidienne). Traitement des requêtes utilisateur : Lorsqu'un utilisateur effectue une recherche, la requête est encodée en vecteur à l'aide du modèle SentenceTransformer.

1	ID	VALUES	
	412	0.00541541912, -0.0497728325, 0.0971337259, 0.038749177, 0.0635424629, 0.0395645872, 0.0...	  
SCORE	METADATA		
0.9982	<p>abstract: "This paper presents a new approach to feature selection for machine learning that uses a correlation based heuristic to evaluate the merit..."</p> <p>authors: ["MA Hall", "LA Smith"]</p> <p>title: "Practical feature subset selection for machine learning"</p> <p>url: "https://researchcommons.waikato.ac.nz/bitstream/handle/10289/1512/Practicalfeaturesubset?sequence=1"</p> <p>Show 1 more</p>		




2	ID	VALUES	
	617	-0.0108257569, -0.090066649, 0.0584755614, 0.00995188393, 0.0541051291, 0.0491120331, -0.0...	  
SCORE	METADATA		
0.8140	<p>abstract: "A central problem in machine learning is identifying a of feature selection for machine learning through a correlation based Three machine..."</p> <p>authors: ["MA Hall"]</p> <p>title: "Correlation-based feature selection for machine learning"</p> <p>url: "https://researchcommons.waikato.ac.nz/bitstream/handle/10289/15043/thesis.pdf?sequence=1&isAllowed=y"</p> <p>Show 1 more</p>		

FIGURE 8 – db pinecone

Recherche sémantique : Ce vecteur est utilisé pour interroger l'index Pinecone et récupérer les articles les plus pertinents en fonction de la similarité vectorielle.

float

2.3.2 mongoDB

Cette base de données NoSQL est utilisée pour stocker plusieurs types d'informations :



FIGURE 9 – logo mongoDB

Métadonnées des articles : Inclut des champs tels que le titre, l'auteur, l'année de publication, et le lien vers le PDF. Ces informations permettent d'enrichir les résultats de recherche présentés à l'utilisateur.

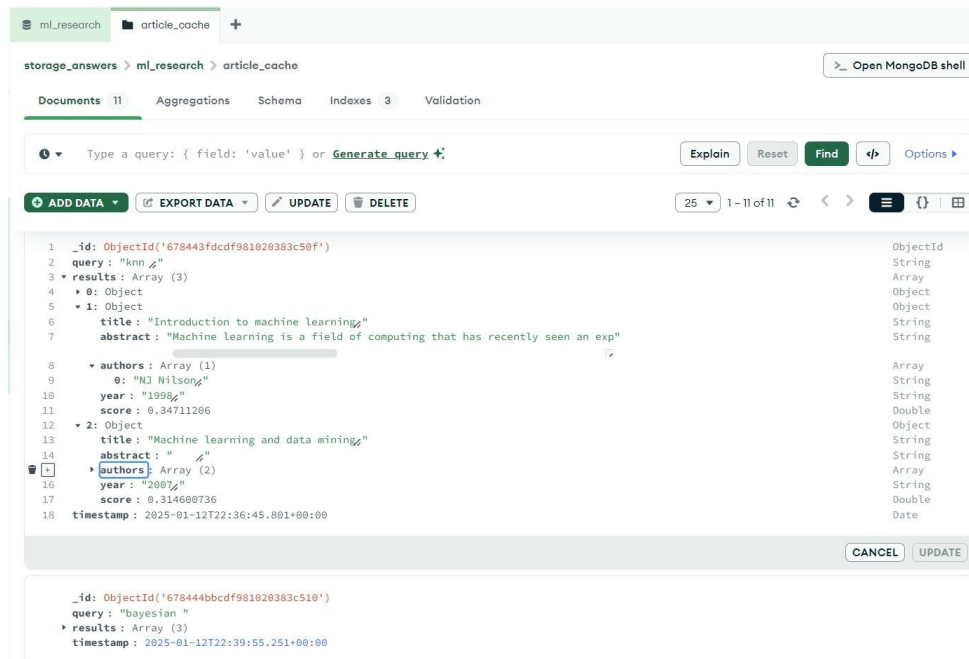


FIGURE 10 – stored data

Historique des recherches : Enregistre les requêtes effectuées par les utilisateurs, facilitant ainsi le suivi et l’amélioration de l’expérience utilisateur.

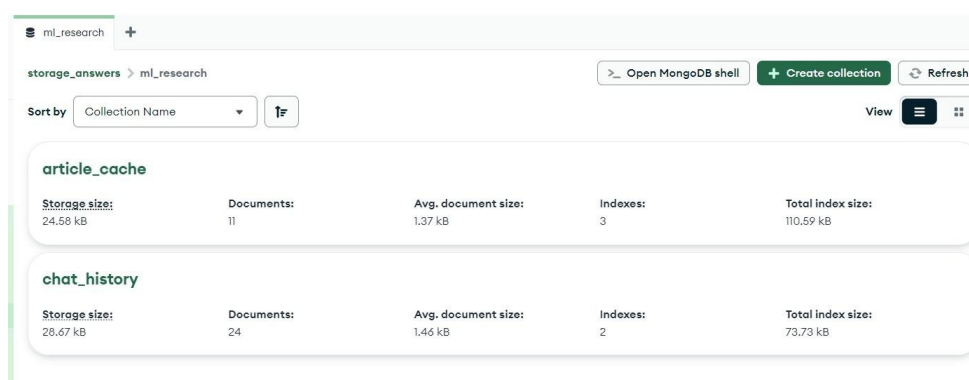
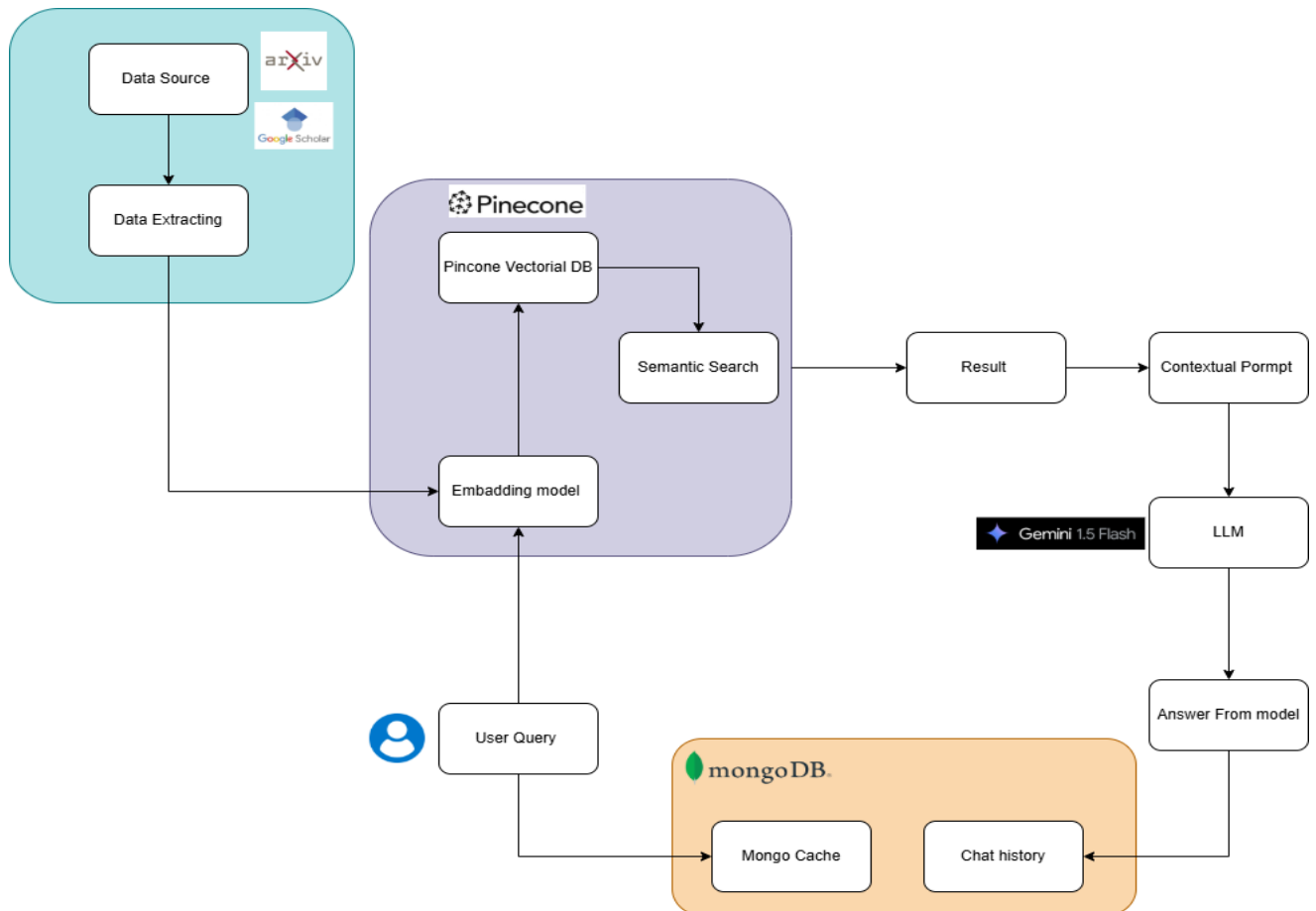


FIGURE 11 – cache

Interactions avec le chatbot : Stocke les échanges entre l’utilisateur et le chatbot, ce qui permet une analyse approfondie des besoins des utilisateurs et une amélioration continue des réponses fournies.

3 Méthodologie

3.1 Architecture du projet



3.2 Collection et Préparation des Données

La première étape consiste à identifier des sources fiables d'articles scientifiques. Les métadonnées des articles, telles que le titre, l'auteur, l'année de publication, le résumé et le lien vers le PDF, sont extraites et organisées. Une fois collectées, les données sont nettoyées pour éliminer les doublons et standardiser les formats. Ensuite, des embeddings sont générés pour chaque article à l'aide du modèle SentenceTransformer. Ces embeddings sont indexés dans Pinecone, permettant ainsi une recherche rapide et précise basée sur la similarité des vecteurs.

```
papers_data.json > 83 > abc author
1 [
2   {
3     "title": "Lecture Notes: Optimization for Machine Learning",
4     "abstract": "Lecture notes on optimization for machine learning, derived from a course at Princeton University and tutored by Elad Hazan.",
5     "author": "Elad Hazan",
6     "year": "2019",
7     "url": "https://arxiv.org/pdf/1909.03550v1.pdf"
8   },
9   {
10    "title": "An Optimal Control View of Adversarial Machine Learning",
11    "abstract": "I describe an optimal control view of adversarial machine learning, where the dynamical system is the machine learning process.",
12    "author": "Xiaojin Zhu",
13    "year": "2018",
14    "url": "https://arxiv.org/pdf/1811.04422v1.pdf"
15  },
16  {
17    "title": "Minimax deviation strategies for machine learning and recognition with short learning samples",
18    "abstract": "The article is devoted to the problem of small learning samples in machine learning. The flaws of maximum deviation strategies are analyzed.",
19    "author": "Michail Schlesinger, Evgeniy Vodolazkiy",
20    "year": "2017",
21    "url": "https://arxiv.org/pdf/1707.04849v1.pdf"
22  },
23  {
24    "title": "Machine Learning for Clinical Predictive Analytics",
25    "abstract": "In this chapter, we provide a brief overview of applying machine learning techniques for clinical predictive analytics.",
26    "author": "Wei-Hung Weng",
27    "year": "2019",
28    "url": "https://arxiv.org/pdf/1909.09246v1.pdf"
29  },
30  {
31    "title": "Towards Modular Machine Learning Solution Development: Benefits and Trade-offs",
32    "abstract": "Machine learning technologies have demonstrated immense capabilities in various domains. They play a key role in many applications.",
33    "author": "Samiyuru Menik, Lakshmi Ramaswamy",
34    "year": "2023",
35    "url": "https://arxiv.org/pdf/2301.09753v1.pdf"
36  },
37  {
38    "title": "Introduction to Machine Learning: Class Notes 67577",
39    "abstract": "Introduction to Machine learning covering Statistical Inference (Bayes, EM, NML/MaxEnt duality), algebraic geometry, and optimization.",
40    "author": "Amnon Shashua",
41    "year": "2009",
42    "url": "https://arxiv.org/pdf/0904.3664v1.pdf"
43  }
44 ]
```

FIGURE 12 – data

3.3 Développement de l'Interface Utilisateur

La collecte des données est une étape fondamentale pour la réussite du projet. Elle commence par l'identification de sources fiables d'articles scientifiques, telles que des bases de données reconnues comme arXiv, PubMed, ou Google Scholar. Ces plateformes offrent un large éventail de publications scientifiques qui couvrent divers domaines. Une fois ces sources sélectionnées, les métadonnées des articles, à savoir le titre, l'auteur, l'année de publication, le résumé et le lien vers le PDF, sont extraites automatiquement à l'aide d'outils de scraping ou d'APIs fournies par ces bases de données. Ces informations sont ensuite organisées dans une base de données pour un accès et une gestion optimaux.

Afin d'assurer la qualité et la cohérence des données, un nettoyage est effectué pour supprimer les doublons et les informations inutiles, ainsi que pour standardiser les formats (par exemple, la date de publication, les noms d'auteurs, etc.). Cette étape est cruciale pour éviter toute confusion dans l'analyse et la recherche. Ensuite, les données textuelles des articles sont transformées en embeddings, des vecteurs numériques qui représentent de manière compacte le contenu des articles. Cette opération est réalisée à l'aide du modèle SentenceTransformer, qui est particulièrement adapté pour générer des embeddings de haute qualité à partir de textes. Une fois les embeddings générés, ils sont indexés dans Pinecone, un moteur de recherche vectoriel puissant, permettant de récupérer rapidement les articles les plus pertinents en fonction de la similarité des vecteurs.

3.4 Intégration du Chatbot pour la Recherche Contextuelle

L'ajout d'un chatbot basé sur un modèle de langage, tel que GPT, permet de rendre la recherche encore plus interactive et personnalisée. Le chatbot est capable d'interagir avec les utilisateurs en répondant à leurs questions sur les articles scientifiques récupérés. Lorsqu'un utilisateur consulte les résultats de recherche ou pose une question spécifique, le chatbot peut utiliser les données indexées dans Pinecone et stockées dans MongoDB pour fournir des réponses contextuelles, précises et détaillées.

Le chatbot peut également effectuer des actions supplémentaires, comme résumer des articles ou comparer plusieurs résultats entre eux, selon les demandes de l'utilisateur. Par exemple, si l'utilisateur souhaite un résumé rapide d'un article ou une comparaison entre plusieurs études sur un même sujet, le chatbot peut fournir ces informations directement. Cette fonctionnalité permet d'offrir une expérience de recherche enrichie et d'aider les utilisateurs à trouver rapidement des informations pertinentes sans avoir à consulter chaque article en détail. En outre, le chatbot peut être amélioré pour effectuer des recherches par questions complexes, en tenant compte du contexte de la recherche et des préférences personnelles de l'utilisateur, rendant ainsi l'interaction encore plus naturelle et intuitive.

4 Résultats du Projet

4.1 Présentation des Résultats finaux

4.1.1 barre de recherche

La barre de recherche permet aux utilisateurs de saisir des requêtes pour trouver des articles scientifiques pertinents en temps réel. Son interface est intuitive et réactive, offrant des résultats rapidement en fonction de la similarité des textes.

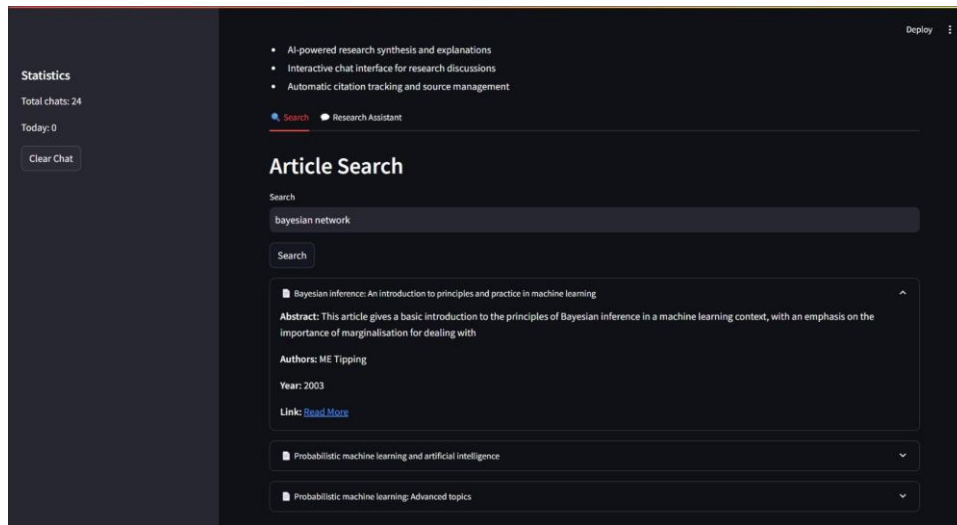


FIGURE 13 – resultat de recherche

4.1.2 chatbot

Le chatbot intégré offre une interaction fluide, permettant aux utilisateurs de poser des questions contextuelles liées aux résultats de recherche. Il fournit des réponses personnalisées en utilisant les données disponibles, améliorant ainsi l'expérience de recherche.

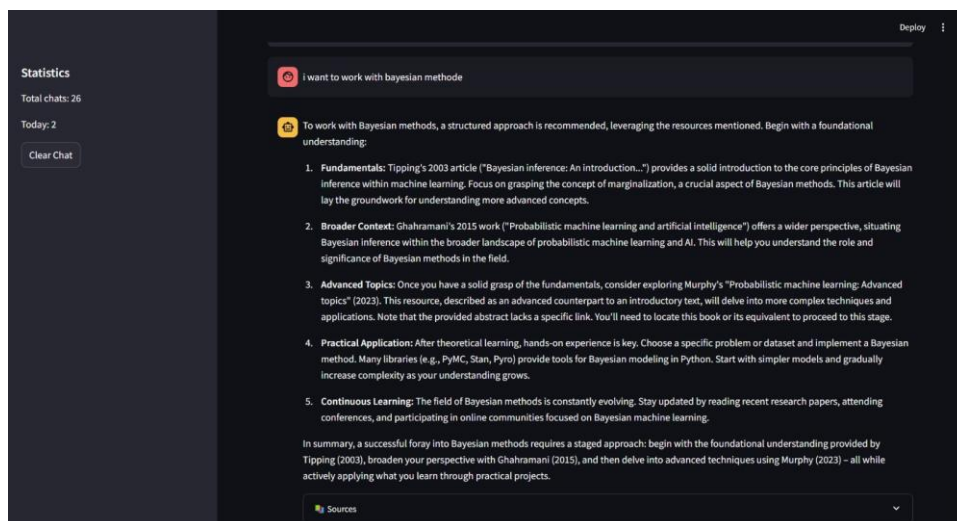


FIGURE 14 – interaction result

5 Conclusion

Le projet a permis de développer une page web complète et fonctionnelle dédiée à la recherche d'articles scientifiques, basée sur leur contenu, et intégrant une méthode avancée de génération de réponses intelligentes. L'utilisation d'une architecture moderne combinant Streamlit pour l'interface utilisateur, Pinecone pour la recherche sémantique, Google Gemini 1.5 Flash pour la génération de contenu, et MongoDB pour la gestion des données a permis de créer une solution efficace et évolutive, répondant aux objectifs fixés.

Grâce à l'intégration de modèles d'apprentissage automatique pour l'encodage des requêtes et des techniques de recherche avancée, la plateforme permet une expérience utilisateur fluide et pertinente. La gestion des résultats de recherche et des historiques de conversation a été optimisée par l'utilisation d'un système de cache et d'une base de données robuste, assurant une navigation rapide et efficace.

Le projet a mis en lumière certains défis, tels que la gestion de grandes quantités de données académiques, l'amélioration de la pertinence des réponses générées par l'IA, et l'intégration fluide des différents composants techniques. Ces défis ont nécessité des optimisations dans la gestion des données et des algorithmes de recherche pour garantir une performance constante.

Les résultats obtenus confirment la pertinence des choix technologiques et offrent de nombreuses perspectives d'amélioration, telles que l'élargissement de la base de données pour inclure davantage de publications, l'ajout de fonctionnalités de recommandations personnalisées, ou encore l'optimisation continue des modèles d'IA pour générer des réponses encore plus contextuelles. Ce projet constitue ainsi une base solide pour faciliter l'accès à la recherche académique en ligne et offre un potentiel d'évolution vers des solutions encore plus avancées.

6 Références

Pinecone. Getting Started with Pinecone. Retrieved from <https://docs.pinecone.io/guides/get-started/overview>

Hugging Face. Sentence Transformers: all-MiniLM-L6-v2. Retrieved from <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>

Streamlit. Streamlit Documentation. Retrieved from <https://docs.streamlit.io/>

Google Scholar. Retrieved from <https://scholar.google.com/>

PyPI. Scholarly: Python Package for Academic Publications and Metadata. Retrieved from <https://pypi.org/project/scholarly/>

Python Software Foundation. xml.etree.ElementTree — The ElementTree XML API. Retrieved from <https://docs.python.org/3/library/xml.etree.elementtree.html>

Kaggle. Retrieved from <https://www.kaggle.com/>