RAPPORT MIF11 :

Conception et développement d'une base de données requêtable de sujets de stages de biologie au sein de l'Alliance Université Européenne Arqus

Miryam ATAMNA, Marc FAUSSURIER, Mariam EL OUARRAD

FÉVRIER 2025

Résumé: Ce projet a pour but de créer et développer une base de données requêtable pour le site de dépôts/recherche de stage de biologie au sein de l'Alliance Université Européenne Arqus. Cette base de données sera utilisé afin de faciliter la recherche de stage selon des critères de recherche (domaine de recherche, établissement d'accueil, durée du stage, etc.). L'objectif étant de simplifier et optimiser la recherche de stages pour les étudiants ainsi que pour les enseignants et d'offrir un filtrage efficace.

Mots-clés: deep learning, base de données, requêtes, back-end.

1 INTRODUCTION

Le projet s'inscrit dans le cadre du développement du back-end de la plateforme en ligne de Arqus, une alliance universitaire européenne regroupant plusieurs universités de différents pays. Leur but est de favoriser la coopération académique, scientifique et culturelle. Cette alliance cherche à renforcer la mobilité des étudiants et des chercheurs, à développer des programmes communs et à encourager l'innovation dans l'enseignement supérieur. Au cours de ce projet nous avons été supervisés et guidés par Romain THINON (manager du programme d'alliance université européenne Arqus auquel prend part Lyon 1) , Laurent GUEGUEN (maître de conférences en biologie) et encadrés par Nadia KABACHI (professeure en informatique à l'Université Lyon 1).

L'objectif de ce projet est de développer une base de données requêtable de stages en permettant une recherche intelligente, qui pourra ensuite être utilisée ensuite sur la plateforme de stages d'Arqus.

2 PRÉSENTATION DU CONTEXTE

2.1 L'importance des bases de données

De nos jours, les bases de données jouent un rôle central dans la gestion de l'information. Elles permettent le stockage, l'organisation et l'extraction efficace de données structurées. Elles sont largement utilisées dans divers domaines, allant des systèmes d'information d'entreprise à la recherche scientifique et à la gestion des connaissances.

L'Alliance Universitaire Européenne Arqus dispose actuellement d'une plateforme de stages proposant une interface d'affichage des offres de stages. Cependant, cette interface est très limitée, ne permettant qu'un défilement manuel des offres sans possibilité de recherche avancée. Cette absence de filtrage et de fonctionnalités de recherche engendre des difficultés majeures pour les étudiants, qui doivent parcourir des listes longues et peu organisées pour trouver une offre correspondant à leurs critères.

Dans ce contexte, l'intégration d'une base de données relationnelle optimisée permettrait une meilleure structuration des offres, rendant les recherches plus intuitives et efficaces.

2.2 Optimiser la recherche d'information

Avec l'augmentation constante du volume d'informations en ligne, il devient essentiel d'optimiser la manière dont ces données sont recherchées et affichées. Sur la plateforme Arqus, les étudiants rencontrent actuellement des difficultés dans la consultation des offres de stage, celles-ci étant présentées sous forme de liste sans possibilité de recherche avancée ou de quelconque filtrage. Cette interface limitée contraint les utilisateurs à parcourir manuellement de nombreuses annonces, rendant le processus long et inefficace, étant donné le nombre de disciplines différentes proposées.

Pour remédier à ces limitations, plusieurs approches modernes issues du domaine de la gestion de l'information peuvent être mises en place. L'indexation et la recherche textuelle constituent une première solution en structurant les données sous forme de mots-clés, qui permettrait une récupération rapide des offres correspondant aux mots-clés donnés par les étudiants. Une telle optimisation offrirait un accès plus direct aux stages s'avérant pertinents, et éviterait aux utilisateurs de défiler inutilement parmi des offres qui ne correspondent pas à leurs critères.

Par ailleurs, l'intégration de techniques de Traitement du Langage Naturel (NLP) améliorerait encore davantage la recherche en analysant non seulement les mots-clés, mais aussi le contexte des requêtes . Contrairement aux systèmes classiques qui reposent sur des correspondances exactes, cette approche permettrait d'élargir la recherche en prenant en compte la proximité sémantique des termes. Pour illustrer, un étudiant cherchant un stage en biologie marine pourrait ainsi se voir proposer des offres en écologie marine, même si l'intitulé exact diffère, grâce à une analyse approfondie des descriptions de stages.

Enfin, les systèmes de recommandation peuvent eux aussi améliorer l'expérience utilisateur. En exploitant l'historique des recherches et les préférences des étudiants, ils permettent de suggérer automatiquement des offres similaires à celles précédemment consultées. Cette fonctionnalité pourrait réduire considérablement le temps de recherche en proposant des stages adaptés aux besoins de chaque utilisateur.

L'association de ces trois approches pourrait transformer Arqus en une plateforme plus performante et intuitive. En facilitant l'accès aux opportunités et en améliorant la pertinence des résultats, ces améliorations garantiraient une meilleure expérience pour les étudiants en quête d'un stage.

3 CONCEPTION DE LA BDD

3.1 Analyse des besoins

L'un des défis majeur a été de traduire ces besoins en un schéma relationnel cohérent et optimisé. Cet exercice s'inscrit pleinement dans les objectifs pédagogiques de l'UE de bases de données suivie en L2 ainsi qu'en L3, en mettant en pratique les principes théoriques dans un contexte concret.
La structuration de la base de données a été pensée pour répondre effacement aux besoins spécifiques du projet. En effet, chaque table a été le fruit d'une réflexion menée en collaboration avec les responsables de ce projet, afin que le modèle soit fidèle a leur attentes. L'un des défis majeur a été de traduire ces besoins en un schéma relationnel cohérent et optimisé.
Ainsi, on a pu identifier deux types d'utilisateurs : les étudiants, qui recherchent des stages selon des critères spécifiques, et les administrateur, qui peuvent vouloir consulter les offres pour orienter leurs étudiants, mais ont aussi la possibilité de supprimer les stages qu'ils ont proposés.
D'autre part les critères de recherche principaux :

- Le domaine du stage (ex. biologie, informatique, chimie).
- L'université proposant l'offre.
- La durée du stage.
- Les mots-clés associés à l'offre.

Et des critères plus précis tel que le rôle de la personne qui pose un stage. Il a ainsi été précisé que l'auteur de l'offre n'est pas nécessairement le maître de stage, et que ce dernier n'est pas forcément un chercheur. Cette approche permet entre autre la clarté du code, les limitations des erreurs de saisie etc.

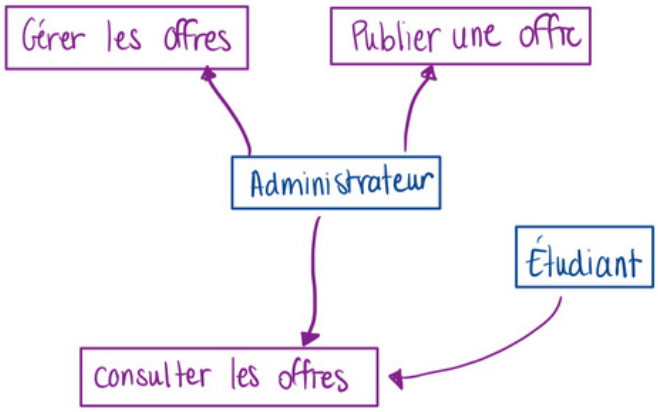

Figure 1 - Diagramme d'interactions

3.2 Modèle relationnel et structure de la base de données

La base de données repose sur plusieurs entités connectés par des associations :

- Stages : représente chaque offre de stage, avec des informations comme le titre, la description, la durée et la date de publication.
- Universités : liste les universités partenaires d'Arqus.
- Superviseurs : référence les encadrants académiques pouvant superviser des stages.
- Disciplines : permet de classer les stages par domaine d'étude.
- Mots-clés (Keywords) : contient les mots-clés associés à une offre de stage.

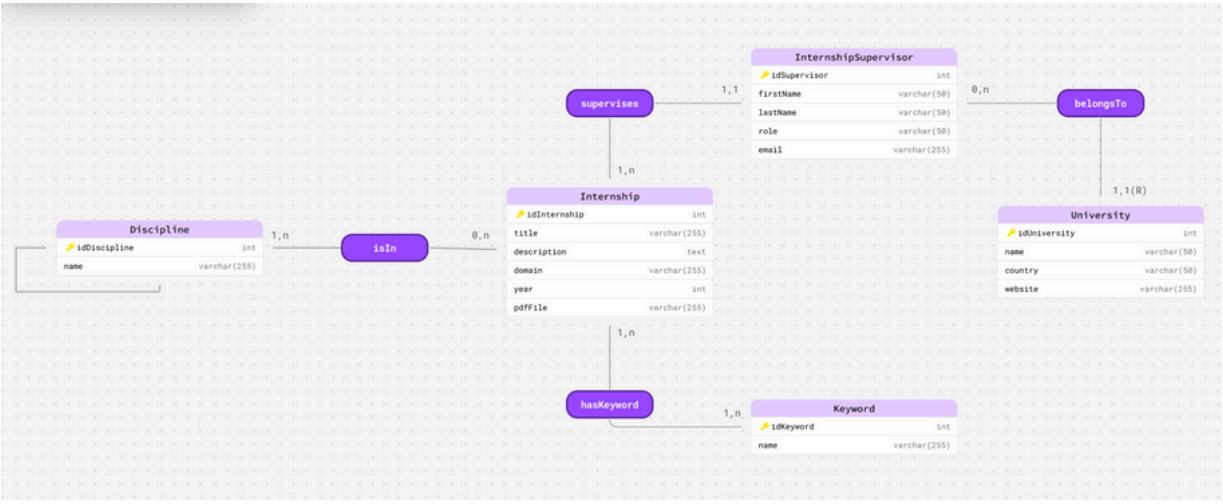

Figure 2 - Schéma E\A

La complexité du modèle réside dans la gestion des relations entre ces entités, qui s'appuient sur différents types d'associations :

- Relation many-to-many entre Stages et Disciplines : Un même stage peut appartenir à plusieurs domaines scientifiques. Par exemple, un stage sur la biodiversité des récifs coralliens peut être classé à la fois en biologie marine et en écologie.
- Relation many-to-one entre Stages et Superviseurs : Un superviseur peut être responsable de plusieurs stages, mais chaque stage n'est associé qu'à un seul encadrant principal.
- Relation many-to-one entre Superviseurs et Universités : Chaque superviseur est affilié à une université, mais une université peut compter plusieurs superviseurs parmi ses membres.
- Relation many-to-many entre Stages et Mots-clés : Cela permet d'associer plusieurs mots-clés à une même offre de stage, facilitant ainsi la recherche avancée par thématique. Les mots-clés eux peuvent être associés à plusieurs stages.

3.3 Choix des technologies

Pour faire fonctionner notre plateforme, nous avons besoin d'un système capable de stocker et organiser les offres, de gérer les interactions entre les utilisateurs et ces offres, et de permettre une recherche rapide et efficace. Pour cela, nous avons choisi trois outils principaux : PostgreSQL, l'ORM SQLAlchemy et FastAPI. Ces outils assurent ensemble une gestion fluide et performante des données.

PostgreSQL est la base de données où sont stockées toutes les informations sur les stages, les universités, les superviseurs et les disciplines. Il permet de retrouver rapidement les offres correspondant aux critères des étudiants, en organisant ces données de manière efficace.

SQLAlchemy est un outil qui facilite la communication entre la base de données et le reste de l'application. Il permet d'accéder aux informations de manière plus simple et d'éviter des erreurs lorsqu'on effectue des recherches ou des modifications dans la base.

FastAPI est utilisé en backend pour assurer la communication entre l'application et la base de données. Il permet d'envoyer et de recevoir des informations rapidement. Par exemple lorsqu'un étudiant effectue une recherche ou qu'un administrateur ajoute une nouvelle offre. Même si l'utilisateur final ne le voit pas directement, il joue un rôle essentiel dans la fluidité et la rapidité du système.

Ces trois technologies travaillent ensemble pour offrir une plateforme simple d'utilisation, où les étudiants peuvent facilement trouver un stage correspondant à leurs besoins.

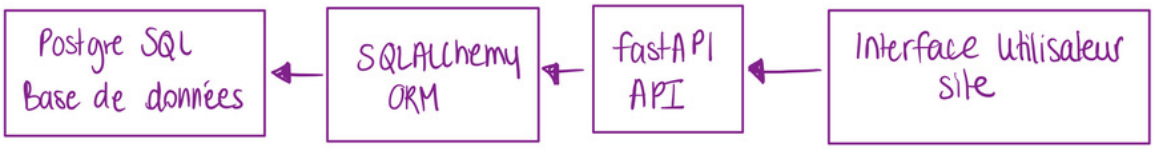

Figure 3 - Technologies utilisés dans l'architecture du projet

4 DÉVELOPPEMENT ET IMPLÉMENTATION

4.1 Mise en place de l'API REST

L'API joue un rôle central, elle permet la communication entre l'application et la base de données, via un ensemble de requêtes qui facilitent l'ajout, la modification et la recherche des stages. Elle a été développée avec FastAPI comme précisé précédemment. Elle agit comme un intermédiaire entre la base de données et l'interface utilisateur, recevant des requêtes et retournant des réponses structurées en JSON.

Le fonctionnement général est le suivant :

1. Un utilisateur effectue une requête (ex. rechercher un stage par mots-clés).
2. L'API interroge la base de données via SQLAlchemy, qui génère et exécute une requête SQL optimisée.
3. Les résultats sont retournés en format JSON et utilisé pour être affiché sur l'interface utilisateur (frontend).

4.2 Endpoints principaux

L'API joue un rôle essentiel dans l'application en permettant de récupérer et d'afficher les données stockées dans la base sur le site. Chaque endpoint correspond à une action bien précise et facilite l'échange d'informations entre la base de données et l'interface utilisateur.

Par exemple, lorsqu'un étudiant remplit un formulaire de recherche de stage, l'application doit lui afficher la liste des universités. Pour cela, elle envoie une requête à l'endpoint **/universities**, qui récupère les établissements enregistrés dans la base de données.

```python
@router.get("/universities")
async def get_supervisors(db: Session = Depends(get_db)):
    return db.query(University).all()
```

Figure 4 - Exemple de endpoint /universities

L'API comprend d'autres endpoints, qui permettent d'effectuer différentes actions essentielles :

- /disciplines : Récupère toutes les disciplines disponibles pour aider les étudiants à filtrer les stages.
- /search : Permet d'effectuer une recherche avancée en filtrant par mots-clés, disciplines, durée, etc.
- /stages : Récupère la liste complète des offres de stage disponibles sur la plateforme.
- /stages/{id} : Affiche les détails d'un stage spécifique lorsqu'un étudiant clique sur une offre.
- /download/{pdf_filename} : Permet de télécharger le fichier PDF associé à une offre de stage.
- /upload : Réservé aux enseignants, cet endpoint permet d'ajouter une nouvelle offre avec ses informations et son fichier PDF.

Chaque endpoint automatise la communication entre l'interface et la base de données, garantissant une mise à jour dynamique et une recherche fluide des stages.

5 RECHERCHE INTELLIGENTE ET OPTIMISATION

5.1 Optimisation des requêtes SQL et filtrage avancé

La première étape pour améliorer la recherche des stages a été d'optimiser les requêtes SQL afin d'accélérer l'accès aux données tout en permettant des filtres avancés. L'objectif était d'assurer une recherche fluide et performante, évitant ainsi aux étudiants de devoir parcourir une liste longue et non triée d'offres.

Pour ce faire, plusieurs améliorations ont été mises en place. Tout d'abord, l'indexation des colonnes clés a été appliquée aux champs fréquemment utilisés dans les requêtes, notamment les titres, les disciplines et les mots-clés associés aux stages. Cette indexation permet de réduire considérablement le temps de réponse lorsqu'un utilisateur effectue une recherche. Ensuite, la structure relationnelle de la base de données a été pensée pour faciliter les jointures entre les différentes entités : stages, disciplines, superviseurs et universités. L'utilisation de requêtes SQL bien optimisées permet ainsi d'obtenir des résultats précis en tenant compte des différents filtres sélectionnés par l'étudiant.

Dans notre projet, les utilisateurs peuvent affiner leur recherche grâce à plusieurs critères tels que la discipline, l'université d'accueil, la durée du stage ou encore les mots-clés associés. Par exemple, une requête cherchant tous les stages en informatique proposés par l'Université de Lyon 1 et débutant après janvier 2025 prendra en compte tous ces paramètres et retournera uniquement les offres pertinentes. Cette approche offre une expérience utilisateur bien plus intuitive, en permettant aux étudiants de trouver rapidement les offres qui correspondent à leurs attentes.

5.2 Intégration de l'intelligence artificielle pour une recherche plus pertinente

L'un des défis majeurs de la recherche d'offres de stage était d'améliorer la pertinence des résultats tout en offrant une plus grande flexibilité aux utilisateurs dans la formulation de leurs requêtes. En effet, une approche basée uniquement sur des mots-clés exacts restait trop rigide et pouvait conduire à des résultats peu satisfaisants. Pour pallier cette limite, nous avons intégré une approche issue du Traitement du Langage Naturel (NLP) qui permet d'analyser le contenu des offres et de proposer des résultats pertinents même lorsque la formulation d'une requête diffère de celle utilisée dans la description des stages.

Le NLP (Natural Language Processing) est un domaine de l'intelligence artificielle qui permet aux machines de comprendre et d'analyser le langage humain. Dans le cadre de notre projet, il est utilisé pour comparer le texte d'une requête utilisateur avec les descriptions des stages disponibles, en tenant compte non seulement des mots employés mais aussi du contexte sémantique global. Cette approche permet d'éviter les recherches trop strictes basées uniquement sur des mots-clés et d'afficher des résultats plus riches et adaptés aux intentions réelles des utilisateurs. Par exemple, un étudiant cherchant un stage en biologie marine pourra aussi voir des offres contenant les termes écologie marine ou étude des écosystèmes aquatiques, car ces concepts sont liés.

L'implémentation du NLP a été réalisée à l'aide de SciBERT, une version spécialisée du modèle BERT, entraînée sur des corpus scientifiques et académiques et BioBERT, spécifquement entrainée en biologie. Ce modèle est capable d'extraire des représentations vectorielles du texte, ce qui permet d'évaluer la similarité entre deux phrases même si elles ne contiennent pas les mêmes mots exacts. Dans notre projet, chaque offre de stage est ainsi convertie en une représentation numérique, qui capture son sens global. De la même manière, lorsqu'un étudiant effectue une recherche, sa requête est également transformée en vecteur, ce qui permet de comparer efficacement sa signification avec celle des stages enregistrés.

Cependant, le principal défi réside dans la capacité à effectuer cette comparaison rapidement, en particulier lorsqu'il y a un grand nombre d'offres enregistrées. Pour résoudre ce problème, nous avons trouvé une solution : FAISS (Facebook AI Similarity Search), une bibliothèque d'indexation vectorielle qui permet d'accélérer la recherche en facilitant la comparaison entre les vecteurs générés par SciBERT. Concrètement, FAISS permet de retrouver en quelques millisecondes les offres les plus proches d'une requête utilisateur, en cherchant uniquement dans l'espace vectoriel plutôt que d'effectuer une recherche textuelle classique dans la base de données.
L'intégration de SciBERT et FAISS ensemble a transformé la manière dont les stages sont recherchés sur la plateforme. Avant cette amélioration, un étudiant devait souvent essayer plusieurs formulations différentes pour espérer voir toutes les offres pertinentes. Désormais, grâce à l'analyse sémantique et à l'indexation vectorielle, la recherche est bien plus intuitive : un étudiant n'a plus besoin de taper des termes exacts pour accéder aux résultats les plus appropriés. De plus, l'algorithme classe directement les stages en fonction de leur pertinence réelle par rapport à la requête, évitant ainsi aux utilisateurs d'avoir à parcourir manuellement une liste non triée d'annonces.
Les premiers tests effectués après l'intégration de cette approche ont montré une amélioration significative en termes de pertinence des résultats et de rapidité d'exécution. Grâce à SciBERT et BioBERT, le moteur de recherche est capable d'établir des relations entre les termes employés dans une requête et ceux présents dans les descriptions de stages, permettant ainsi une meilleure mise en correspondance des offres avec les attentes des étudiants. D'un autre côté, l'utilisation de FAISS a permis d'optimiser considérablement le temps de réponse des recherches, rendant la navigation sur la plateforme beaucoup plus fluide et efficace. En combinant ces deux technologies, nous avons réussi à dépasser les limitations des recherches traditionnelles basées sur des mots-clés, offrant ainsi une expérience utilisateur bien plus satisfaisante et adaptée aux besoins des étudiants.

5.3 Utilisation de l'intelligence artificielle pour le pré-remplissage du formulaire de création de stage

L'un de nos objectifs était d'extraire des informations spécifiques à partir d'une offre de stage au format PDF afin de fluidifier la création de stage. Plus précisément, il s'agissait de récupérer le titre, la description, les dates de début et de fin, les sous-disciplines scientifiques (choisies parmi une liste préétablie), ainsi que les mots-clés associés, afin de préremplir le formulaire de publication de stage pour faciliter la vie des chercheurs. Pour ce faire, nous avons utilisé le LLM Qwen Instruct en version 3B avec des poids compressés en 4 bits afin de garder un maximum de flexibilité dans nos prompts tout en essayant d'utiliser un minimum de ressources possible. Cette fonctionnalité s'exécute en une dizaine de secondes sur un petit GPU de 4 Go de VRAM, tel qu'un GTX 980 dans nos tests.
Cette fonctionnalité est optionnelle et peut être activée ou désactivée via le fichier de configuration .env.

5.4 Limites

Nous avons identifié deux limitations dans notre approche pour faire de la recherche avancée. Premièrement, nous souhaitions entraîner notre propre modèle pour qu'il soit spécifiquement adapté aux descriptions de stages présentes sur la plateforme. Cela aurait permis de mieux capter les particularités de ce type de contenu et d'améliorer encore la précision de la recherche sémantique. Cependant, le modèle entraîné par les documents fourni par nos responsables n'a pas produit des résultats suffisamment pertinents pour nos besoins. Les résultats avec les modèles BERT ont été beaucoup plus concluant que le modèle entrainé par les documents fournis.
Deuxièmement, une autre fonctionnalité envisagée mais non mise en place concerne l'intégration d'un système de recommandation basé sur l'historique des recherches des utilisateurs. Ce type de système aurait permis de suggérer des offres similaires aux recherches précédentes, rendant la navigation encore plus intuitive. Son absence se justifie par une priorité donnée à l'amélioration de la recherche sémantique.

6 CONCLUSION

Au cours de ce projet nous avons créer une base données requêtable, nous l'avons entrainée à reconnaitre des similarités de sujets avec un certain pourcentage et nous avons collaboré avec un autre groupe d'etudiant.e.s responsable du front-end afin de corroborer nos versions et fournir un rendu directement exploitable. Ainsi, nous avons été soumis à différentes problématiques. Notamment celui d'adapter notre approche aux contraintes réelles des données disponibles. La gestion et l'exploitation des offres de stage nous ont amenés à réfléchir à des solutions alternatives pour compenser l'absence d'un corpus suffisamment vaste pour entraîner un modèle sur mesure. Nous avons dû nous appuyer sur des outils préexistants tout en optimisant leur utilisation pour répondre aux besoins spécifiques de la plateforme.

Cette première expérience de la recherche nous a permis d'approfondir nos compétences en base de données, ce qui sera sans doute très utile dans la suite de notre parcours en M2 Data Science. Elle nous a également offert l'opportunité de mettre en pratique, dans un contexte réel, les enseignements reçus tout au long de notre formation, et en particulier ceux de l'UE d'apprentissage des données, suivie parallèlement à ce projet.