Cappellina Arthur
Deveaux Paul
Mentec Jérémi

Projet NoSQL

1. Introduction Notre projet vise à concevoir et déployer un système basé sur des bases de données NoSQL pour analyser et visualiser des relations complexes entre des séquences protéines. Le système repose sur des technologies NoSQL qui permettent de gérer des données volumineuses et non structurées.

2. Technologies utilisées

- **MongoDB** : Utilisée pour le stockage des données biologiques. MongoDB offre une grande flexibilité pour les documents JSON. Il est surtout utile pour stocker les données volumineuses.
- **Neo4j** : Base de données orientée graphe, il sert à modéliser et interroger les relations complexes entre protéines.
- **D3Blocks :** Bibliothèque utilisée pour créer des visualisations interactives et dynamiques basées sur D3.js.
- **Flask** : Framework web léger pour construire une interface utilisateur qui permet d'interagir avec les bases de données et de visualiser les résultats sous forme de graphes. Il permet également de faire différentes requêtes et d'afficher des statistiques.
- **Python** : Langage principal utilisé pour l'interrogation des bases, la création de ces dernières et toutes requêtes avec elles.
- **HTML/CSS** : Pour la conception des pages web interactives.

3. Fonctionnalités du système

- **Interrogation de MongoDB** : On utilise MongoDB pour récupérer les données nécessaires au site internet et au besoin de l'utilisateur.
- **Exploration des similitudes entre protéines** : Neo4j stocke les protéines et leurs relations sous forme de graphes, permettant des requêtes rapides pour explorer leurs similitudes.
- **Visualisation des similitudes** : Graphes et représentations interactives des données grâce à Neo4J et D3Block.
- **Statistiques** : Affichage de statistiques descriptives sur les données.

4. Architecture du projet

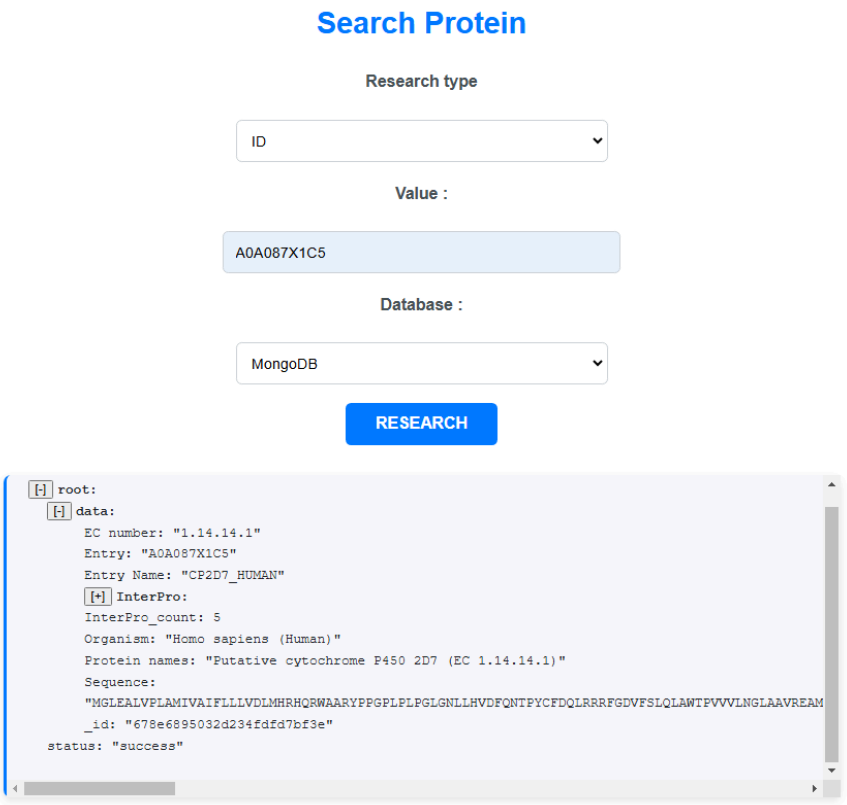
- **Données** : Les fichiers sources sont stockés dans le répertoire `data/` pour initialiser les bases de données.
- **Modules de connexion** : Les scripts Python dans `db_utils/` gèrent les connexions à MongoDB et Neo4j, ainsi que les requêtes complexes. Le script `database_creation.py` à la racine permet d'initialiser les bases de données et de les remplir.
- **Frontend** : Le répertoire `templates/` contient les fichiers HTML pour afficher les graphes, statistiques et formulaires de recherche.

- **Backend** : Le fichier `app.py` constitue le point d'entrée du projet, gérant le routage entre l'interface utilisateur et les bases de données.

5. Exemples

La page d'accueil permet d'effectuer certaines requêtes avec différentes conditions. Nous pouvons choisir quel champ est recherché et quelle base de données est utilisée. En sortie, nous obtenons les données sous forme de json.

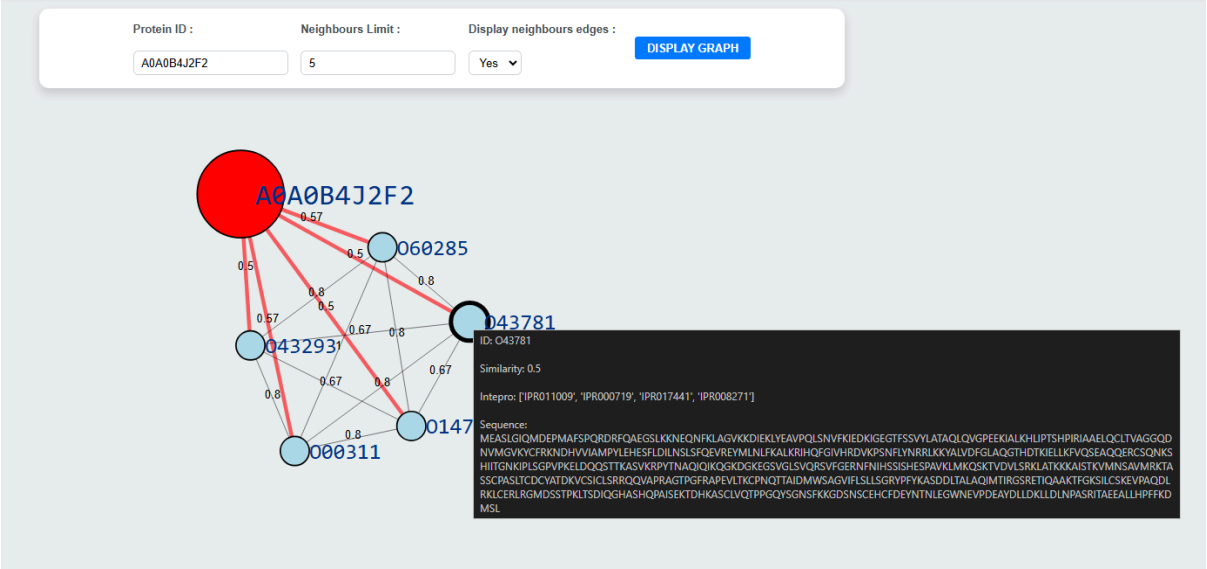
Avec mongodb, nous obtenons les données de la protéine stockée dans la base.



Avec neo4j, nous obtenons les données de la protéine, ses voisins de degré 1 et 2, ainsi que la similarité les liant.

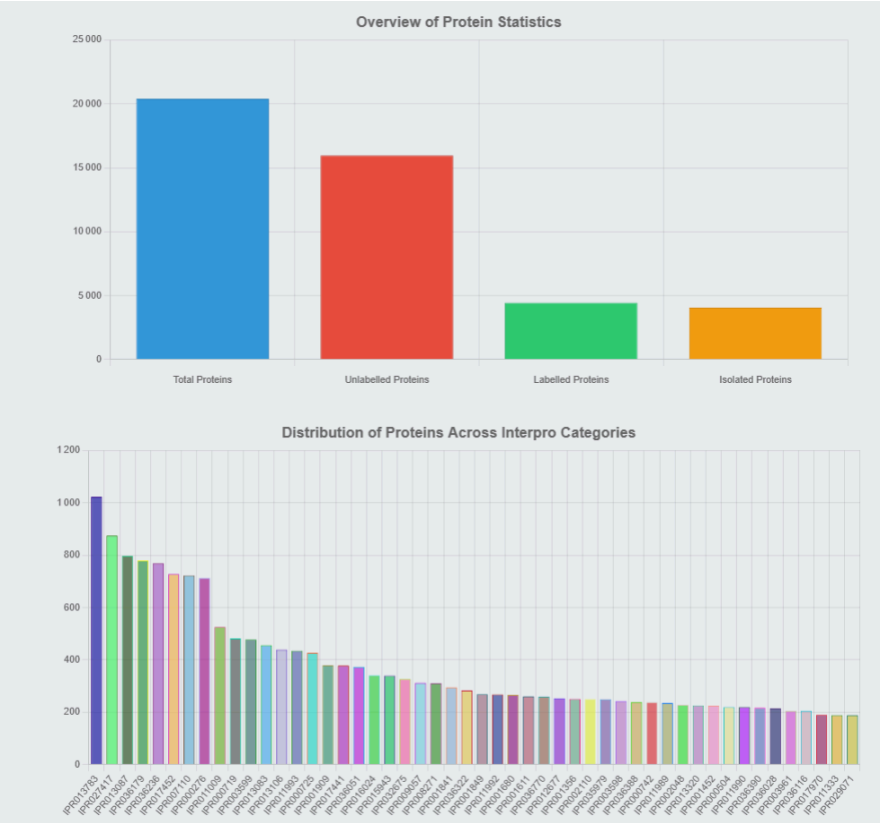
La page graph nous permet de visualiser les liens de similitude entre une protéine choisie et les autres protéines. Des options permettent de choisir le nombre de protéines maximum à afficher, ainsi que de choisir si l'on veut visualiser les liens entre les autres protéines ou non.

En survolant un noeud avec la souris, on peut obtenir des informations sur elle comme la similarité, ses InterPros et sa séquence.



La page statistique nous permet de voir les proportions des données labellisées ou non ainsi que les protéines isolées des autres grâce à un bar chart.

Elle permet également de voir la répartition des protéines par rapport à leurs interpros.



6. Conclusion

Le projet NoSQL met en avant les avantages des bases de données NoSQL pour traiter des données volumineuses et complexes. En combinant MongoDB et Neo4j, nous avons créé un système robuste et polyvalent qui répond aux besoins des utilisateurs en matière de gestion et de visualisation de données.