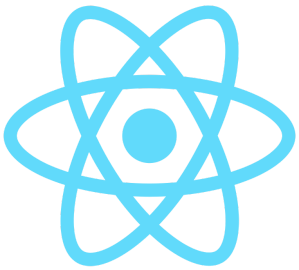
Université polytechnique Hauts-De-France
INSA Hauts-De-France

Projet Java-Spring



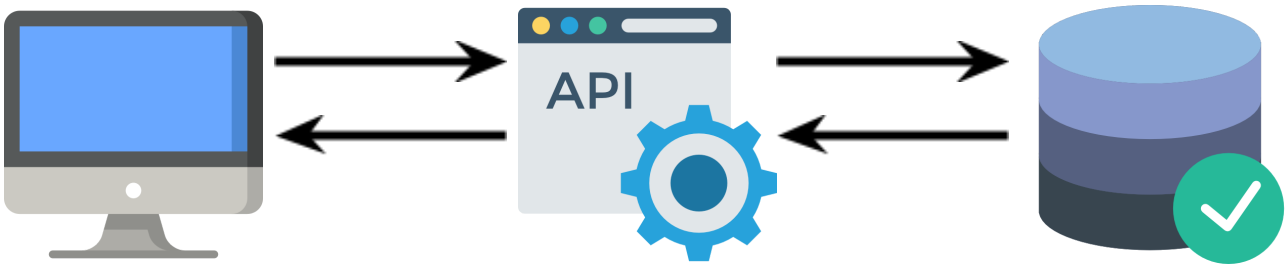
Martin THIBAUT

Basile THIRY

Monsieur Jérôme DEGROOTE

Tables de matières

Fonctionnement de l'application



Client (React) | Api (Java Spring Boot) | BDD (HiberNate)

Notre application fonctionne de la manière suivante :

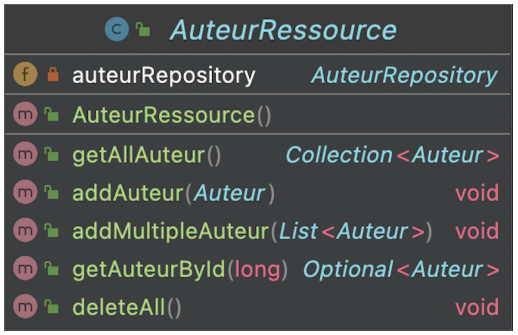
Le client Web interagit avec notre API qui elle interagit avec la base de données. Le client Web est développé en React, nous avons choisi React car c'est un techno que nous ne connaissions pas et nous voulions du challenge. Il fait des requêtes vers notre API, cela peut être des requêtes GET, POST ou DELETE.
L'API fait le traitement de l'information, demande ce qu'il faut à la base de données et les retourne au client pour un traitement plus visuel.

Lorsque l'on lance l'application, nous arrivons sur la liste des bibliothèques. On peut ensuite dans l'application (Voir l'image ci-dessous) :

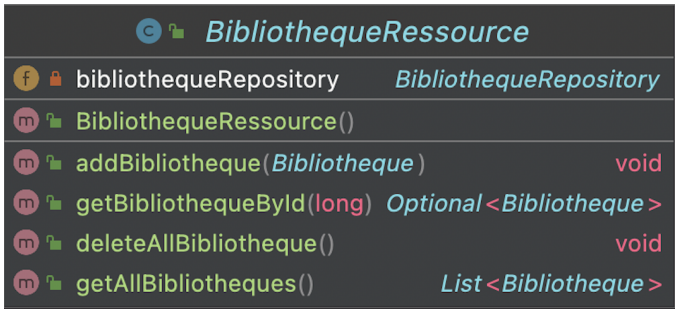
- Voir la liste de toutes les bibliothèques existantes.
- Voir la liste de tous les livres existants.
- Voir la liste de tous les auteurs existants.
- Voir la liste des livres contenus dans une bibliothèque spécifique.
- Dans chacune des pages précédentes, on peut rechercher dans la barre de recherche sur le haut de la page, en fonction du nom de la bibliothèque, du livre ou de l'auteur.
- En tant qu'administrateur :
    - Ajouter un livre dans une bibliothèque
    - Ajouter un auteur
- On peut également, en cliquant sur notre tête dans la barre latérale, accéder à notre profil Linkedin et en cliquant sur notre nom, nous envoyer un mail sur notre boîte mais universitaire.

Les différentes ressources

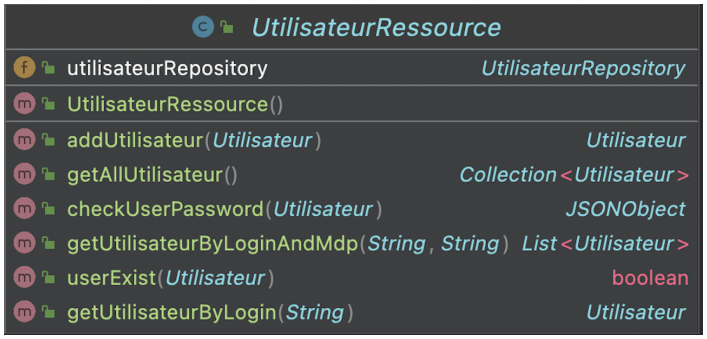
Nous avons tout d'abord la ressource **Livre**, elle caractérise un livre. Un livre à une auteur mais aussi une bibliothèque.
Chaque livre est à une bibliothèque afin de pouvoir les lister par la suite.



Ensuite, nous avons la ressource **Auteur**, elle caractérise un auteur. Un auteur à une liste de livre ainsi que d'autres attribut tels que son nom, son genre ou encore sa nationalité.



De plus, nous avons la ressource **Bibliothèque,** elle caractérise une bibliothèque. Une bibliothèque à une liste de livres mais aussi un nom et une description.



Pour finir, une ressource **Utilisateur** qui caractérise les utilisateurs admin ou non de l'application. Un utilisateur est caractérisé par un login et un mot de passe. Il lui permet de s'authentifier lorsqu'il veut ajouter un livre ou un auteur.

La Base De Données

Afin de mener à bien ce projet, nous avons utilisé la base de données HiberNate.

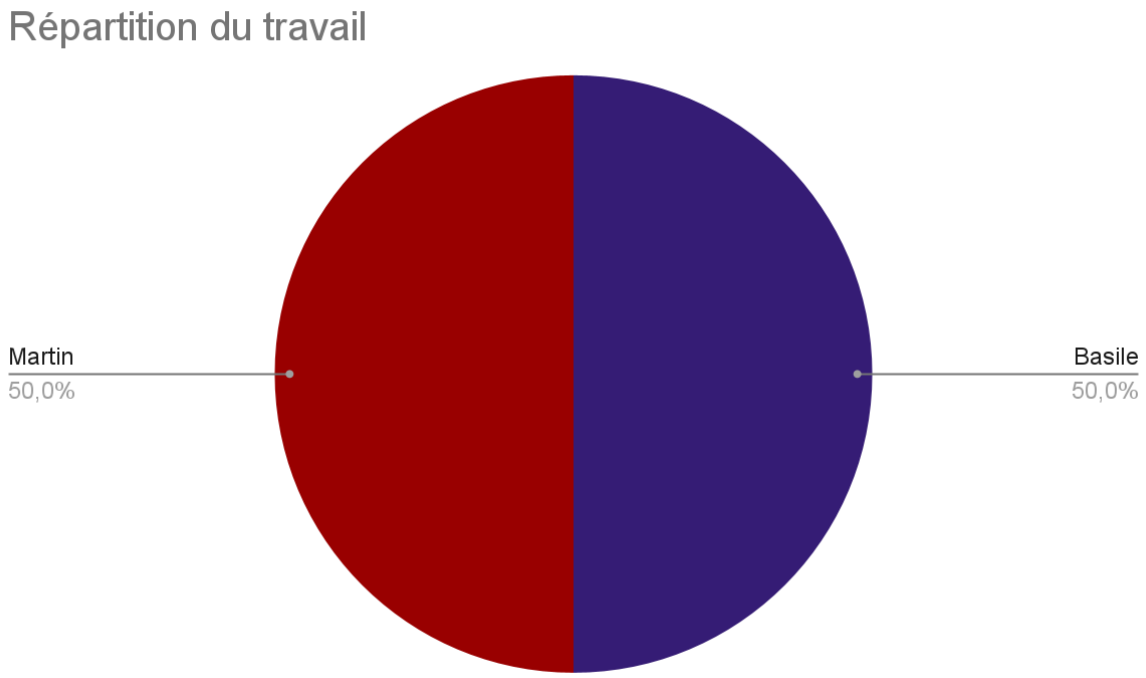
Dans la table Utilisateur, les mots de passe sont stockés de manières hashés. Lorsque l'on crée un utilisateur administrateur avec PostMan, on hash le mot de passe pour plus de sécurité.

Dans le sens inverse, quand la personne veut se login avant d'ajouter un livre ou un auteur, on regarde si le mot de passe entré dans le formulaire correspond au hash de la base de données.

Pour ajouter un utilisateur administrateur, dans PostMan, il faut utiliser la requête *addUtilisateur* et dans le body, cela doit être de la forme :

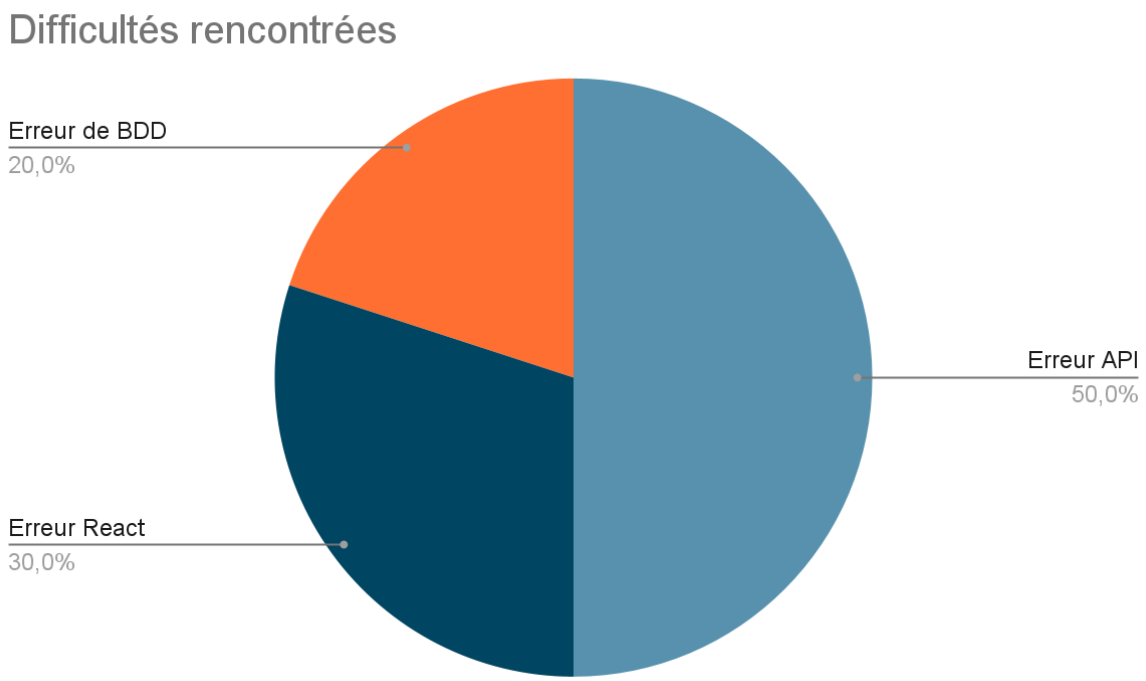
La répartition du travail

Front End	**Back End**
Martin, Basile	Basile, Martin



Nous avons tous les deux participé à chaque partie du projet. Basile et moi avons commencé par développer l'API.
Une fois l'API assez fonctionnelle, je me suis lancé sur le Front End avec l'aide de Basile car nous avons travaillés avec un plugin *CodeWithMe.*

Les difficultés rencontrées



Concernant les difficultés rencontrées, elles sont relativement minimes.

- Les erreurs API sont les principales difficultés que nous avons rencontrées. Cela pouvait être une erreur dans le type de retour, une erreur dans le fonctionnement de CRUD (un oublie de @Entity, @OneToMany, fetch = FetchType.EAGER) ou encore des erreurs de path.
- Les erreurs liées au React sont principalement des erreurs de syntaxes, le React été pour nous inconnu, c'était de la découverte alors nous faisons quelques erreurs de syntaxes.
- Concernant les erreurs de BDD, c'est principalement l'initialisation de la base de données avec l'url de la base et le driver.

Conclusion personnel

Martin :

Pour ma part, ce projet m'a apporté beaucoup de connaissances.
Tout d'abord, j'ai découvert Spring Boot et CRUD. Je sais désormais développer une API en Java Spring Boot.

Ensuite, j'ai découvert et appris à utiliser React. Une librairie que je ne savais pas utiliser et que je sais désormais utiliser.

Enfin, j'ai également découvert HiberNate, je ne connaissais pas cet outil. Un outil pas complexe d'utilisation, mais assez sympathique pour des petits projets comme celui-ci.

Basile :

Pour ma part, ce projet m'a beaucoup apporté dans l'apprentissage du web.
D'abord via l'API à créer, et via Spring boot, où j'ai commencé à bien comprendre comment pouvaient fonctionner certaines parties du web sans passer par que du JS / html / CSS.

Ensuite sur la partie affichage, avec le React, j'ai tout découvert de ce langage car je n'y connaissais rien, et j'ai apprécié découvrir un nouveau langage concernant le web. De plus avant je n'aimais pas développer pour du web, et avec ce projet j'ai appris à apprécier ça et à découvrir des aspects que je ne connaissais pas et qui sont agréables.