



Ecole Mohammadia d'Ingénieurs
2 Année Génie Informatique et Digitalisation
2023-2024

NoSQL

Application web de gestion d'une bibliothèque

Réalisé par :

- Aymane Lahnin
- Mohamed Aabouche
- Marouane Debbagh
- Adnane El hassani

Encadré par:

Pr A.El kassiri

Sommaire

Introduction.....	3
Etude de l'existant.....	4
Problématique.....	4
Analyse des besoins et exigences fonctionnelles.....	4
Exigences non fonctionnelles.....	6
Conception de la BD et étapes de réalisation:.....	6
Étude de l'environnement de développement et Justifications des choix et technologies	11
Mode d'emploi de l'application.....	12
Conclusion.....	18

1. Introduction

Dans le cadre de ce projet, nous avons développé une application web de gestion de bibliothèque. Cette application vise à offrir une solution pour la gestion des livres, des emprunts, des retours et des auteurs, tout en facilitant l'accès à l'information pour les utilisateurs.

Pour atteindre ces objectifs, nous avons mis en place une architecture technique robuste et évolutive, en utilisant des technologies de pointe. Le backend de l'application a été développé avec **Spring Boot**, un framework puissant et flexible pour la création de services web et d'applications d'entreprise. Cette solution nous permet de bénéficier d'une grande modularité et d'une gestion simplifiée des dépendances et des configurations.

Le frontend de l'application a été réalisé avec **React.js**, une bibliothèque JavaScript populaire pour la construction d'interfaces utilisateur interactives et dynamiques. Grâce à React.js, nous avons pu créer une expérience utilisateur fluide et réactive, en garantissant une navigation rapide et une manipulation efficace des données.

Pour la gestion des données, nous avons opté pour des bases de données NoSQL, en combinant **MongoDB** et **Neo4j**. **MongoDB**, une base de données orientée document, nous offre une grande flexibilité dans la gestion des données non structurées et semi-structurées, tandis que **Neo4j**, une base de données orientée graphe, nous permet de modéliser et de requêter efficacement les relations complexes entre les différentes entités de la bibliothèque.

Cette combinaison de technologies nous permet de proposer une application complète, performante et adaptée aux besoins spécifiques de la gestion d'une bibliothèque. Au fil de ce rapport, nous détaillerons les choix techniques effectués, les étapes de développement, ainsi que les fonctionnalités et avantages de notre application de gestion de bibliothèque.

2. Etude de l'existant

Evergreen est une application de gestion de bibliothèque open source qui repose sur une base de données SQL, comme PostgreSQL ou MySQL/MariaDB. Cette application offre une gamme complète de fonctionnalités pour la gestion des collections de bibliothèques, des utilisateurs et des auteurs.



Cependant, l'utilisation d'une base de données relationnelle présente certaines limites :

1. **Scalabilité verticale** : Pour gérer une croissance importante des données ou une augmentation soudaine du nombre d'utilisateurs, il est souvent nécessaire d'augmenter les ressources matérielles (CPU, RAM) d'une seule machine (scalabilité verticale). Cela peut devenir coûteux et limite la capacité à gérer de très grandes bibliothèques.
2. **Schéma rigide** : Les bases de données relationnelles exigent généralement un schéma de données fixe et bien défini. Cela signifie que le modèle de données doit être structuré à l'avance avec des tables, des colonnes et des relations. L'adaptation rapide aux changements dans les besoins de la bibliothèque ou l'ajout de nouveaux types de données peut nécessiter des modifications de schéma substantielles, ce qui peut être complexe et coûteux.
3. **Performance dans les requêtes complexes** : Bien que les bases de données relationnelles soient robustes pour des requêtes complexes, des requêtes très complexes ou des volumes de données massifs peuvent parfois entraîner des performances dégradées, nécessitant des optimisations minutieuses et parfois des agrégations de données.

3. Problématique

Face aux limites inhérentes des bases de données relationnelles identifiées dans l'étude précédente, comment concevoir et mettre en œuvre une nouvelle application de gestion de bibliothèque utilisant une base de données NoSQL pour optimiser la scalabilité, la flexibilité du schéma et les performances des requêtes complexes, tout en gardant les mêmes fonctionnalités de gestions?

4. Analyse des besoins et exigence fonctionnelles

Notre application a 4 types d'utilisateurs classés en 4 niveau lesquels sont:

- **Niveau 1:Admin**

Sa tâche principale est la gestion des utilisateurs : effectuer les opérations CRUD sur les comptes. Il peut également réaliser les mêmes opérations que l'agent.

- **Niveau 2: Agent**

ses taches sont :

1.Gestion des Livres (par les agents et administrateurs)

- Ajouter un livre : Permettre l'ajout de nouveaux livres dans les deux bases de données mongodb et neo4j pour avoir une synchronisation.
- Mettre à jour un livre : Offrir la possibilité de modifier les détails des livres existants.
- Supprimer un livre : Autoriser la suppression des livres de la base de données.
- Consulter des livres.

2. Gestion des Adhérents (par les agents et administrateurs)

- Ajouter un adhérent : Permettre l'enregistrement de nouveaux adhérents avec leurs informations personnelles dans mongodb.
- Mettre à jour un adhérent : Offrir la possibilité de modifier les informations des adhérents existants.
- Supprimer un adhérent : Autoriser la suppression des adhérents de la base de données.

Gestion des Prêts (par les agents et administrateurs)

- Ajouter un prêt : Enregistrer un nouvel emprunt en précisant le livre emprunté, l'adhérent, et les dates de prêt et de retour prévues.
- Mettre à jour l'état d'un prêt : Permettre la mise à jour de l'état d'un prêt (par exemple, retour du livre).
- Supprimer un prêt : Autoriser la suppression d'un enregistrement de prêt de la base de données.
- Consulter des prêts : Permettre aux adhérents de consulter l'historique de leurs prêts et les détails des emprunts en cours.

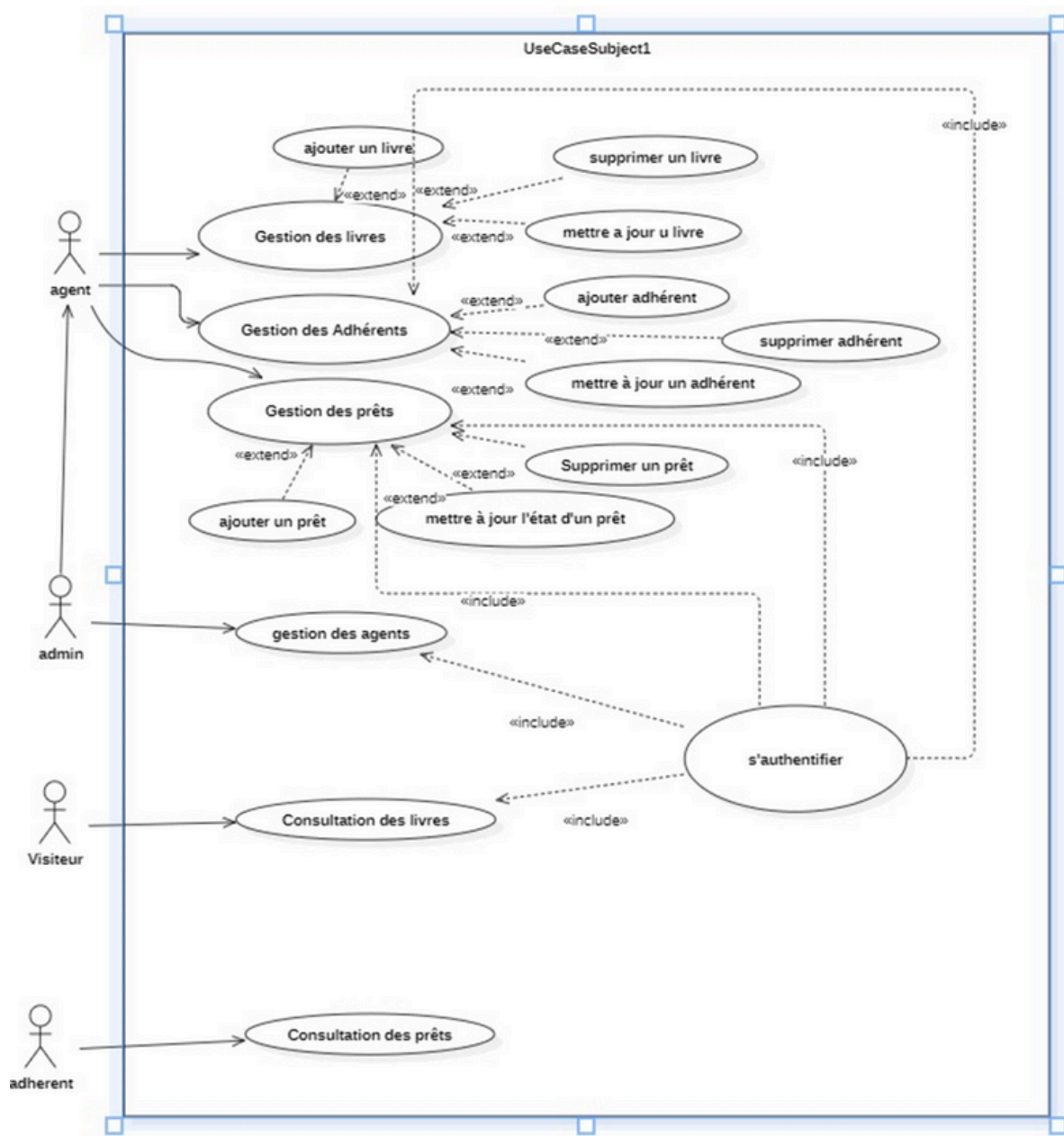
• Niveau 3: Adhérent

Après avoir s'authentifier l'adhérent peut consulter les opérations des prêts enregistrées pour son compte, comme il peut aussi consulter les livres de la bibliothèque

• Niveau 4: Visiteur

les utilisateurs de ce niveau ne peuvent que consulter les livres de la bibliothèque

Le diagramme des cas d'utilisation ci-dessous résume les fonctionnalités de l'application discutées ci-dessus:



5. Exigences non fonctionnelles

Performance

- Temps de Réponse : L'application doit fournir des réponses rapides aux requêtes des utilisateurs, idéalement en moins de 3 secondes pour la plupart des opérations courantes telles que la recherche de livres et la consultation des prêts.
- Scalabilité : L'architecture doit permettre l'ajout de nouveaux serveurs et bases de données pour gérer l'augmentation du nombre d'utilisateurs et du volume de données sans dégradation des performances.

Sécurité

- Authentification et Autorisation : Implémenter un système d'authentification sécurisé et des contrôles d'accès basés sur les rôles pour protéger les données sensibles.
- Chiffrement : Les données sensibles, telles que les mots de passe des utilisateurs, doivent être chiffrées avant qu'elles soient stockées dans la base de données.

Synchronisation des Données entre MongoDB et Neo4j

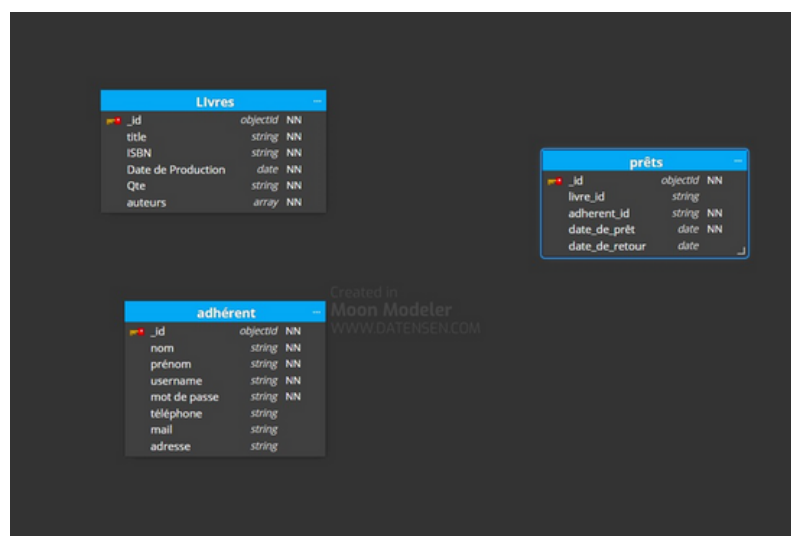
- Consistance des Données : Assurer que les données restent synchronisées entre MongoDB et Neo4j, particulièrement pour les entités et les relations gérées par ces deux bases de données.
- Mécanismes de Synchronisation : Création d'un processus d'arrière-plan pour synchroniser les données vers Neo4j.

6. Conception de la BD et étapes de réalisation:

Dans cette étape du projet, nous avons entrepris de mettre en œuvre une conception qui répond aux exigences fonctionnelles et non fonctionnelles.

6.1 Conception MongoDB

La conception de la base de données MongoDB a évolué à travers plusieurs versions pour s'assurer qu'elle répondait de manière optimale aux besoins fonctionnels et non fonctionnels du projet:



Version 1

La conception initiale de la base de données MongoDB se compose de trois collections principales : Livres, Adhérent, et Prêts. Voici une analyse de cette conception :

Collection Livres

- `_id` : Identifiant unique de chaque livre (type `objectId`).
- `title` : Titre du livre (type `string`).
- `ISBN` : Numéro international standardisé du livre (type `string`).
- `Date de Production` : Date de production ou de publication du livre (type `date`).
- `Qte` : Quantité de copies disponibles (type `number`). Elle permet de suivre le nombre de copies disponibles, ce qui est crucial pour la gestion des emprunts.
- `auteurs` : Liste des auteurs du livre (type `array`, éléments de type `string`) ceci permet de gérer facilement les livres ayant plusieurs auteurs.

Collection Adhérent

- `_id` : Identifiant unique de chaque adhérent (type `objectId`).
- `nom` : Nom de famille de l'adhérent (type `string`).
- `prenom` : Prénom de l'adhérent (type `string`).
- `username` : Nom d'utilisateur pour l'authentification (type `string`).
- `mot de passe` : Mot de passe pour l'authentification (type `string`).
- `téléphone` : Numéro de téléphone de l'adhérent (type `string`).
- `mail` : Adresse e-mail de l'adhérent (type `string`).
- `adresse` : Adresse postale de l'adhérent (type `string`).

Collection Prêts

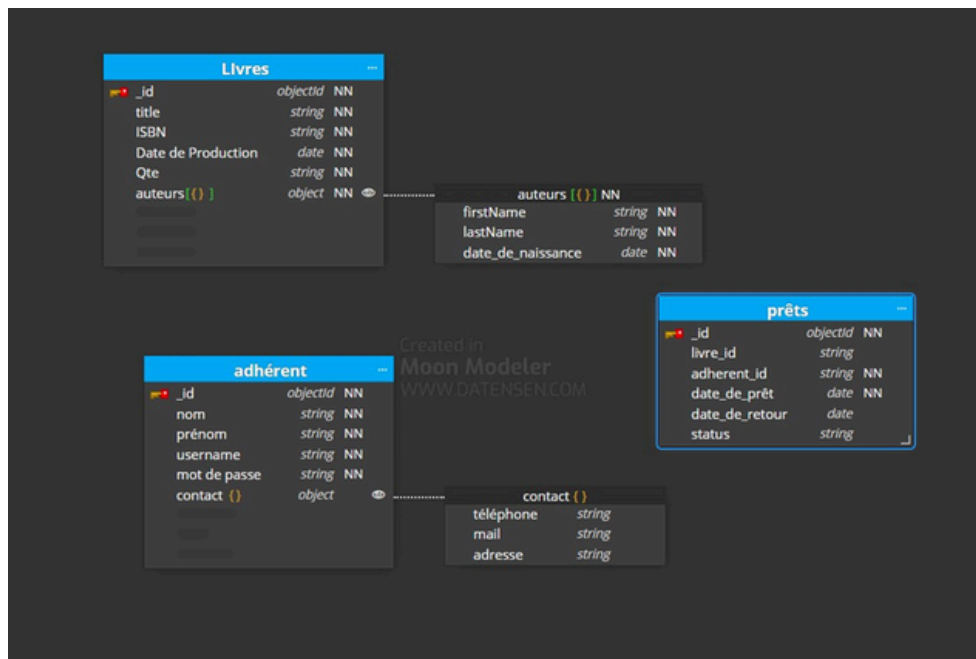
- `_id` : Identifiant unique de chaque prêt (type `objectId`).
- `livre_id` : Référence à l'identifiant du livre emprunté (type `string`).
- `adhérent_id` : Référence à l'identifiant de l'adhérent qui a emprunté le livre (type `string`).
- `date_de_pret` : Date à laquelle le livre a été emprunté (type `date`).
- `date_de_retour` : Date prévue pour le retour du livre (type `date`).

Commentaires:

- La collection Prêts relie efficacement les collections Livres et Adhérent via les champs `livre_id` et `adhérent_id`, permettant de suivre les emprunts.
- Dans l'analyse ci-dessus nous avons expliqué les types des données utilisés dans chaque champ et justifié même le choix de ces champs pour chaque collection.

Limites de cette version:

- Cette version présente des limites importantes en termes de performances des requêtes, notamment pour l'accès aux informations. Par exemple, si nous voulons accéder à l'auteur d'un livre, nous devons lancer une autre requête pour trouver cet auteur car la conception courante empêche un accès direct à partir du livre. Le même problème se pose pour accéder aux informations pour contacter un adhérent, pour récupérer toutes ses informations il faut passer par tous ses attributs.
- La collection des prêts ne possèdent aucune informations qui nous permette de suivre l'état d'un prêt quelconque



Version 2

La version 2 de la conception MongoDB montre des améliorations par rapport à la version précédente, avec une meilleure structuration des documents et des références. Voici une analyse détaillée de cette conception :

Collection Livres

- On a gardé les mêmes attributs sauf la liste des auteurs; pour ce champ on a mis une collection d'objet auteur et ceci grâce à la flexibilité des technologies utilisées qui permet de lier un objet de type document avec un objet de type nœud

Collection Adhérent

- On a gardé les mêmes attributs sauf le champ du contact; pour ce champ on a mis une sous-collection contact.

Collection Prêts

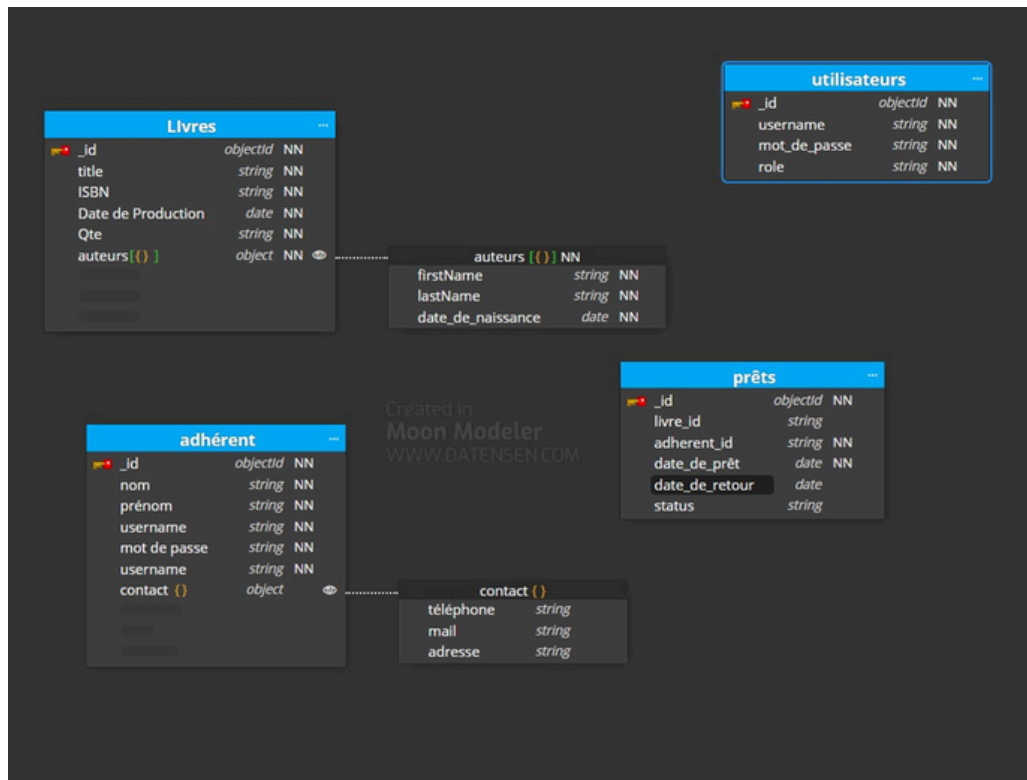
- On a ajouté un champ status pour suivre l'état du prêt est utile pour la gestion des prêts en cours et des retours.

Commentaire :

La nouvelle conception a permis de résoudre le problème du suivi de l'état d'un prêt. De plus, elle a amélioré l'accès rapide aux informations des auteurs. Cependant, les relations entre les entités "livre" et "auteur" ou les adhérents et leurs adresses sont fortement liées et rarement modifiées. L'utilisation d'objets intégrés permet de gérer ces relations de manière optimale.

Limites

Bien que ce modèle améliore la clarté des données et la vitesse d'accès, il ne présente aucune solution pour gérer les utilisateurs de l'application. Avec ce modèle, il n'est pas possible de créer des comptes utilisateurs ou de leur affecter des privilèges selon leur rôle (niveau d'interaction).

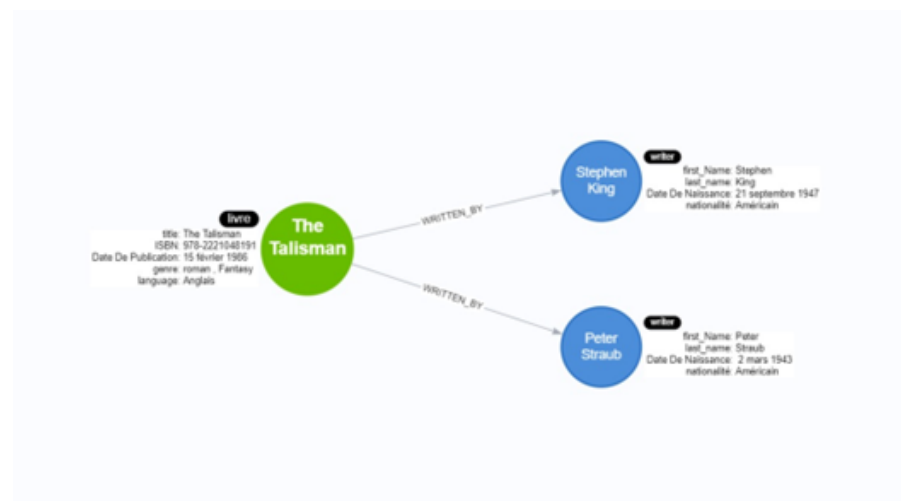


Version 3

Pour cette version, nous avons résolu le problème en ajoutant une nouvelle collection pour les utilisateurs. Les privilèges de chaque utilisateur sont affectés en fonction de la valeur de l'attribut "rôle".

6.1 Conception Neo4j

Tout comme pour la conception de MongoDB, la conception de Neo4j a également évolué à travers plusieurs versions



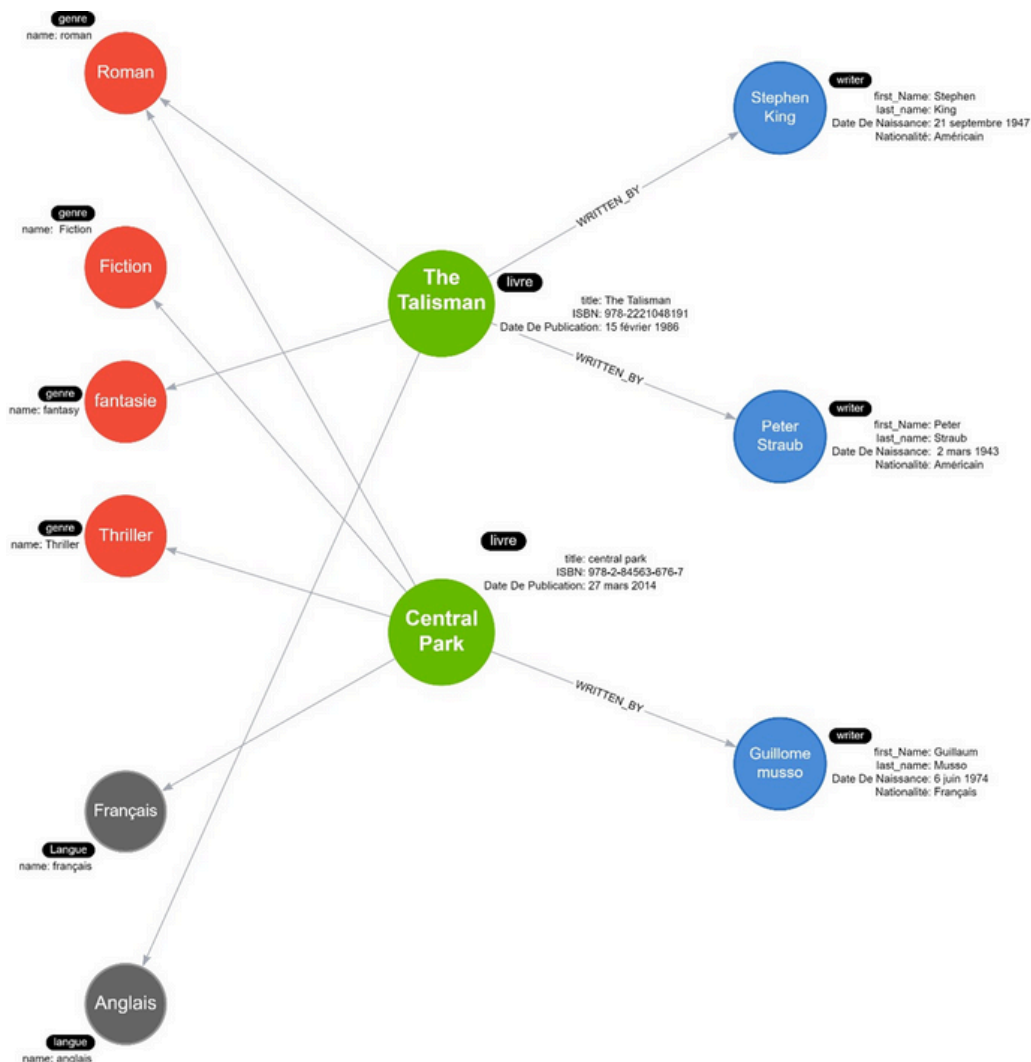
version 1

Description

La version 1 de la conception présente les nœuds principaux incluant le livre, avec des attributs détaillés tels que le titre, l'ISBN, la date de publication, le genre et la langue, ainsi que ses auteurs, chacun ayant des informations personnelles comme le prénom, le nom de famille, la date de naissance et la nationalité. Les relations "WRITTEN_BY" relient le livre à ses auteurs, illustrant clairement les connexions entre ces entités.

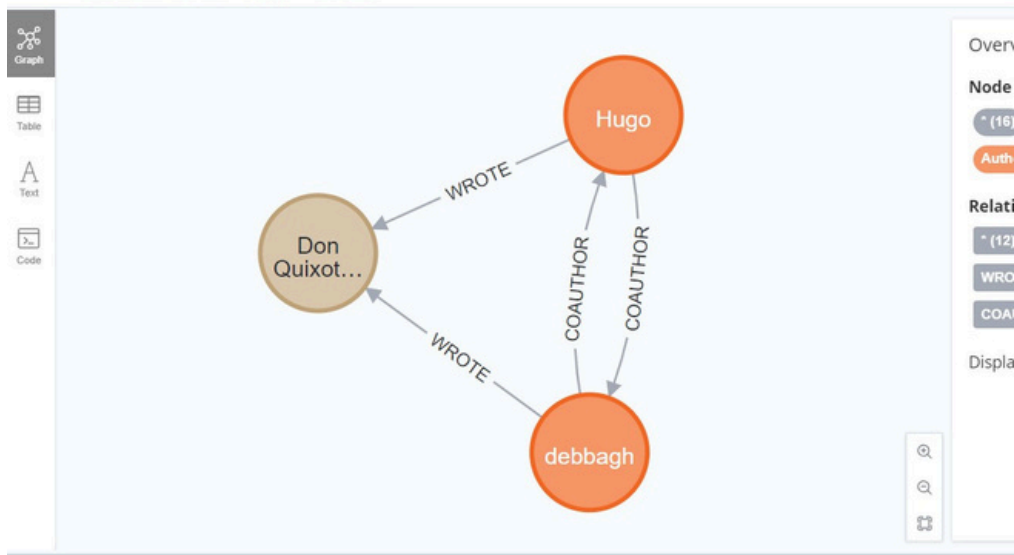
Limites:

Un problème dans cette conception est l'absence de gestion des exemplaires individuels du livre, qui est crucial pour une bibliothèque. Sans des nœuds distincts pour chaque exemplaire avec des attributs tels que le statut (emprunté, réservé), il devient difficile de suivre la disponibilité et la circulation des livres, limitant ainsi l'efficacité de la gestion des emprunts et des réservations.



Version 2

Dans la version 2, nous avons résolu le problème, mais le modèle courant ne présente pas la relation entre les auteurs du même livre.



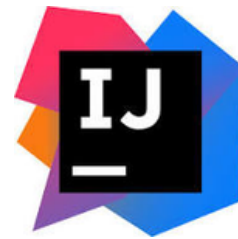
Version 3

Dans cette version on a ajouté une relation entre les auteurs du même livre avec cette conception il sera facile d'effectuer des modification et d'effectuer des recherches à-propos des auteurs

7. Étude de l'environnement de développement et Justifications des choix et technologies

7.1 Étude de l'environnement de développement

- Éditeur Intelligent : Doté d'un éditeur intelligent, il propose une assistance contextuelle, facilitant ainsi la rédaction de code efficace et la résolution instantanée des problèmes.
- Gestion Avancée de la Base de Données : IntelliJ IDEA Ultimate inclut une gestion avancée de la base de données, simplifiant les opérations liées à la base de données directement depuis l'environnement de développement.
- Intégration Continue : Offrant une intégration continue, il assure un flux de travail harmonieux, facilitant la collaboration et la livraison de logiciels de haute qualité.
- Support des Frameworks Populaires : En offrant un support complet des frameworks populaires et des technologies modernes, il facilite le développement dans des contextes variés
- Débogage Puissant : Son environnement de débogage puissant, associé à des outils de profilage et une analyse statique du code, assure une résolution efficace des problèmes et une amélioration continue du code.



7.2 Justifications des choix et technologies

a Frontend

Le choix du framework React.js pour le développement du frontend

se justifie par plusieurs facteurs. Tout d'abord, son architecture reposant sur un DOM virtuel garantit des performances élevées et une réactivité optimale lors de la mise à jour de l'interface utilisateur. De plus,

sa conception modulaire axée sur des composants réutilisables favorise une structure de code maintenable et évolutive. Sans oublier ses capacités à créer des interfaces utilisateur interactives et dynamiques.



b Backend

Le framework Spring Boot a été sélectionné pour le développement du backend pour plusieurs raisons.

Premièrement,

il permet de simplifier et accélérer le processus de développement grâce à des configurations prédéfinies et une approche "convention plutôt que configuration".



En outre, Spring Boot offre une grande flexibilité grâce à de nombreuses fonctionnalités intégrées telles que la gestion des dépendances, la sécurité et la prise en charge des services RESTful. Sa compatibilité avec les bases de données NoSQL comme MongoDB et Neo4j constitue un avantage supplémentaire.

c Base de données

La sélection de bases de données NoSQL pour l'application de gestion de bibliothèque se justifie par leur aptitude à traiter efficacement de grands volumes de données non structurées ainsi que par leur flexibilité d'extensibilité horizontale.

MongoDB, une base de données orientée document, permet une gestion optimisée des enregistrements hétérogènes et une exécution rapide des requêtes complexes

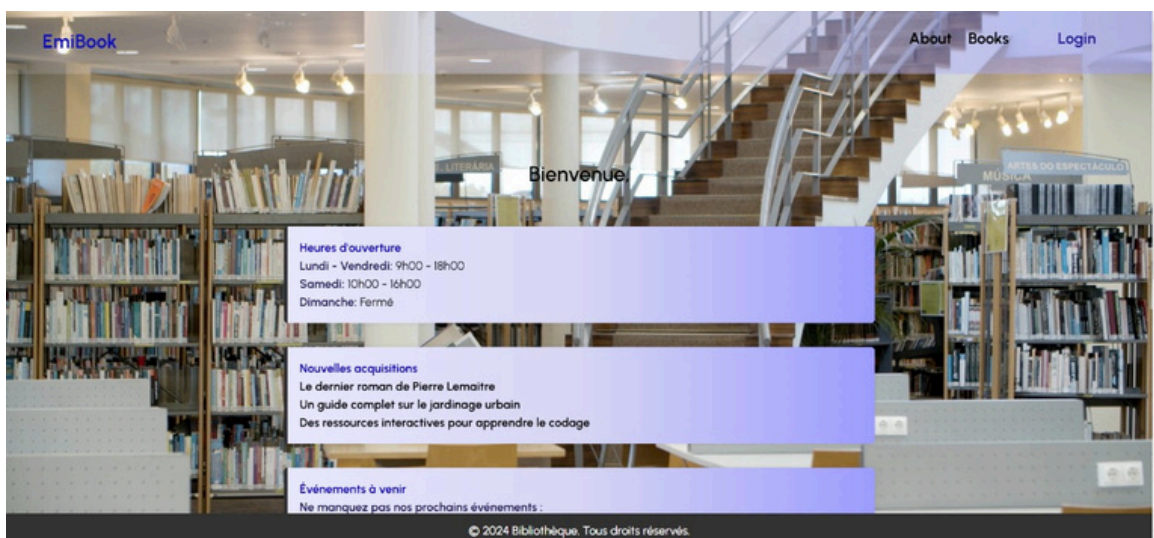


Neo4j, une base de données orientée graphe, s'avère particulièrement appropriée pour modéliser et interroger les relations complexes entre les différentes entités, telles que les liens entre auteurs, livres et emprunteurs, offrant ainsi des performances accrues pour les requêtes relationnelles.



8. Mode d'emploi de l'application

La première page de notre application



La consultation des livres ne nécessite pas d'authentification. Autrement dit, le visiteur n'a pas besoin d'être adhérent et d'utiliser son compte pour cette action.

Événements à venir
Ne manquez pas nos prochains événements :
- Atelier de lecture pour enfants - 12 juin à 14h00
- Conférence sur l'histoire locale - 20 juin à 18h00
- Club de lecture mensuel - 25 juin à 17h00

Search...

Titre	ISBN	Auteurs	Date de publication	copies disponibles
book 1	1	ayman	1/1/2002	99
book 3	12	debbagh	2024-06-26	122
book 4	1223	debbagh	2024-06-13	1222
les mesirables	123	hugo	2024-06-21	21
XXXX	2906	ccc		5
The Hunchback of Notre Dame	1234	Hugo	1979-07-12	200

© 2024 Bibliothèque. Tous droits réservés.

Pour obtenir des privilèges de niveau 1, il est nécessaire de se connecter en tant qu'admin

EmiBook

About Contact Login

Login

Username: admin

Password: [masked]

Login

© 2024 Bibliothèque. Tous droits réservés.

Après la connexion en tant qu'admin, il peut voir ses privilèges

EmiBook

About Contact logout

Home

Books

Adherents

prets

Bienvenue

Heures d'ouverture
Lundi - Vendredi: 9h00 - 18h00
Samedi: 10h00 - 16h00
Dimanche: Ferme

Nouvelles acquisitions
Le dernier roman de Pierre Lemaitre
Un guide complet sur le jardinage urbain
Des ressources interactives pour apprendre le codage

Événements à venir
Ne manquez pas nos prochains événements :

© 2024 Bibliothèque. Tous droits réservés.

Comme mentionné dans la section des fonctionnalités, la tâche principale de l'admin est la gestion des comptes des agents, qui eux-mêmes gèrent les comptes des adhérents. Cependant, l'admin dispose également des privilèges des agents. La figure suivante présente comment l'admin crée un agent

The screenshot shows the 'Add Adherent' form with the following data:

premier nom	marouane	nom	debbagh
username	marouane	email	marouane@gmail.com
téléphone	0667217946	adresse	Av. Ibn Sina, Rabat
password	2406/2007		

Réussite de la création du nouvel adhérent

premier nom	nom	username	email	téléphone	adresse	
adnane	adnane	adnane	adnane	0667217946	raba	delete
ayman	ayman	ayman	ayman	07788888	raba	delete
asmae	el kassiri	asmae	asmae	io		delete
mohamed	aabouch	mohamed	mohamedaabouch@gmail.com	0667217946		delete
marouane	debbagh	marouane	marouane@gmail.com	0667217946	Av. Ibn Sina, Rabat	delete

La procédure d'ajout d'un livre : pour vérifier sa réussite il suffit de cliquer sur le bouton All Books

The screenshot shows the 'Add Book' form with the following data:

Titre	The Hunchback of Notre Dame	ISBN	1234
Date de publication	07/12/1979	Copies disponibles	200
Auteurs			
Hugo	victor	02/18/1965	

On peut remarquer que le livre a été bien enregistré

Événements à venir
Ne manquez pas nos prochains événements :
- Atelier de lecture pour enfants - 12 juin à 14h00
- Conférence sur l'histoire locale - 20 juin à 18h00
- Club de lecture mensuel - 25 juin à 17h00

Search ..

Titre	ISBN	Auteurs	Date de publication	copies disponibles
book 1	1	ayman	1/1/2002	99
book 3	12	debbagh	2024-06-26	122
book 4	1223	debbagh	2024-06-13	1222
les misérables	123	hugo	2024-06-21	21
XXXX	2906	ccc		5
The Hunchback of Notre Dame	1234	Hugo	1979-07-12	200

© 2024 Bibliothèque. Tous droits réservés.

L'ajout d'un prêt

EmiBook About Contact Logout

Add pret All pret

Home Books Adherents prêts

adherent marouane livre The Hunchback of Notre Dame

date De Pret 11/06/2024 date De Retour date De Retour

Add Prêt

© 2024 Bibliothèque. Tous droits réservés.

En cliquant sur All prêt on peut voir l'enregistrement

EmiBook About Contact Logout

Add pret All pret

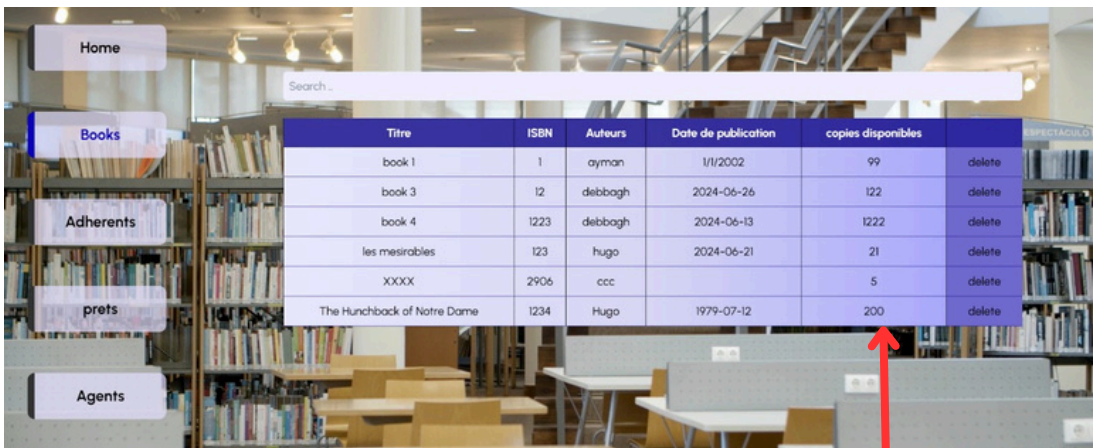
Home Books Adherents prêts

adherent	livre	date De Pret	date De Retour	status
marouane	The Hunchback of Notre Dame	11/06/2024		prête

© 2024 Bibliothèque. Tous droits réservés.

Pour tester la fonctionnalité des prêts, on compare le nombre de copies de The Hunchback of Notre Dame avant et après l'emprunt.

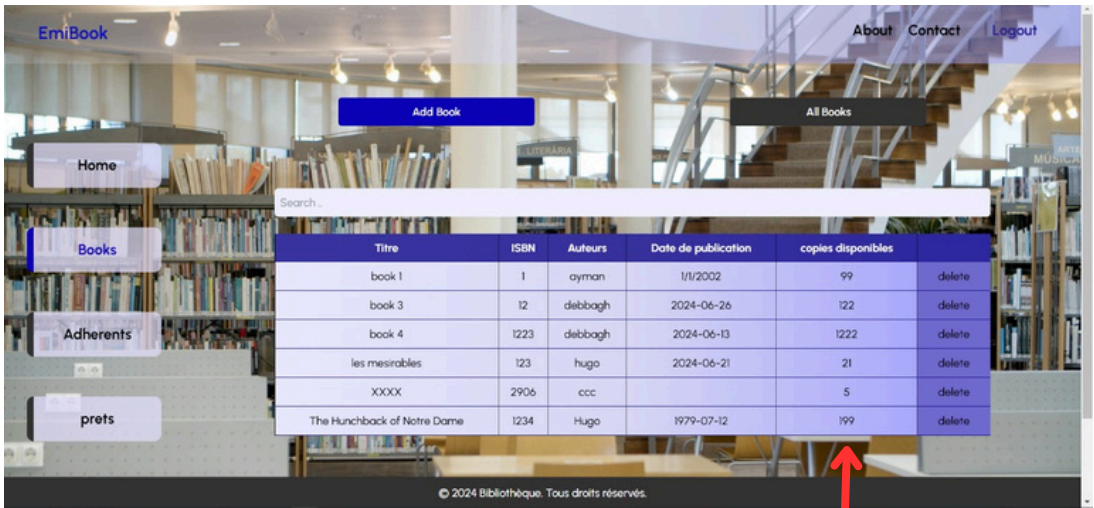
Avant prêt il y avait 200 copies



The screenshot shows the 'Books' section of the EmiBook application. A sidebar on the left contains navigation links: Home, Books, Adherents, prêts, and Agents. The main area displays a table of books. A red arrow points to the 'copies disponibles' column for 'The Hunchback of Notre Dame', which shows 200.

Titre	ISBN	Auteurs	Date de publication	copies disponibles	
book 1	1	ayman	1/1/2002	99	delete
book 3	12	debbagh	2024-06-26	122	delete
book 4	1223	debbagh	2024-06-13	1222	delete
les mesirables	123	hugo	2024-06-21	21	delete
XXXX	2906	ccc		5	delete
The Hunchback of Notre Dame	1234	Hugo	1979-07-12	200	delete

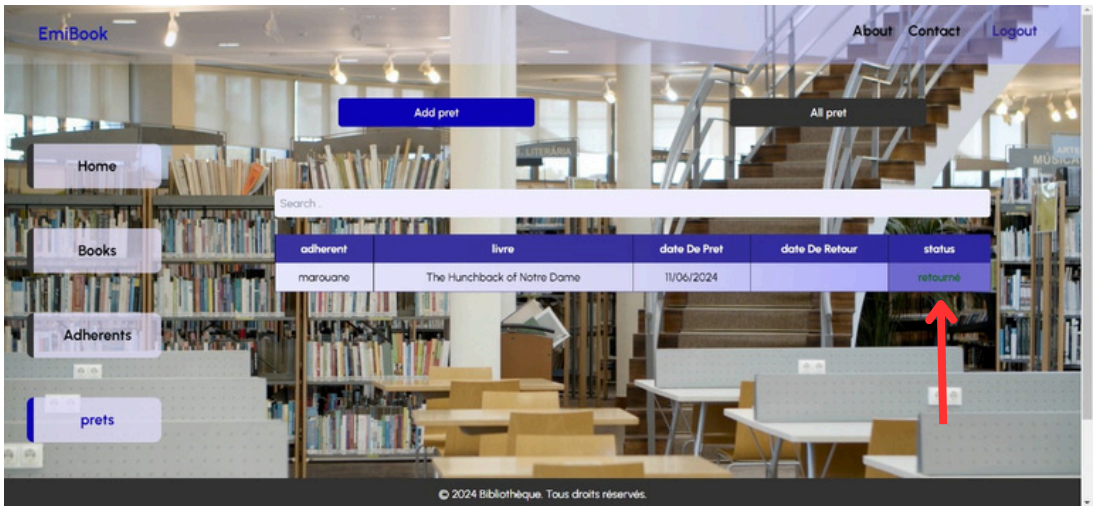
Après le prêt il y a 199 copies



The screenshot shows the 'Books' section after a loan. The 'copies disponibles' for 'The Hunchback of Notre Dame' has decreased to 199. A red arrow points to this value. The footer indicates '© 2024 Bibliothèque. Tous droits réservés.'

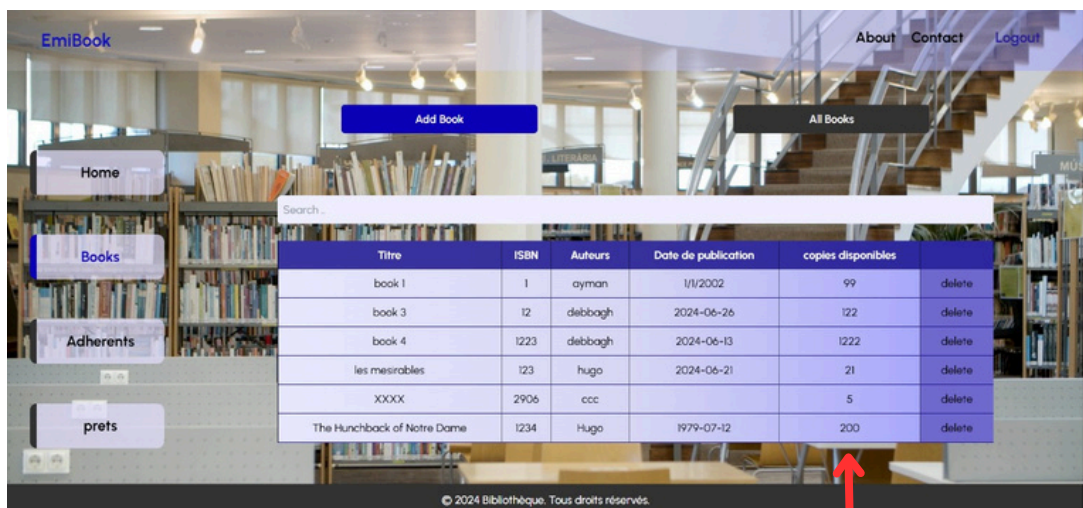
Titre	ISBN	Auteurs	Date de publication	copies disponibles	
book 1	1	ayman	1/1/2002	99	delete
book 3	12	debbagh	2024-06-26	122	delete
book 4	1223	debbagh	2024-06-13	1222	delete
les mesirables	123	hugo	2024-06-21	21	delete
XXXX	2906	ccc		5	delete
The Hunchback of Notre Dame	1234	Hugo	1979-07-12	199	delete

Après avoir retourné le prêt le statuts du prêt est modifié et le nombre des copies a augmenté encore



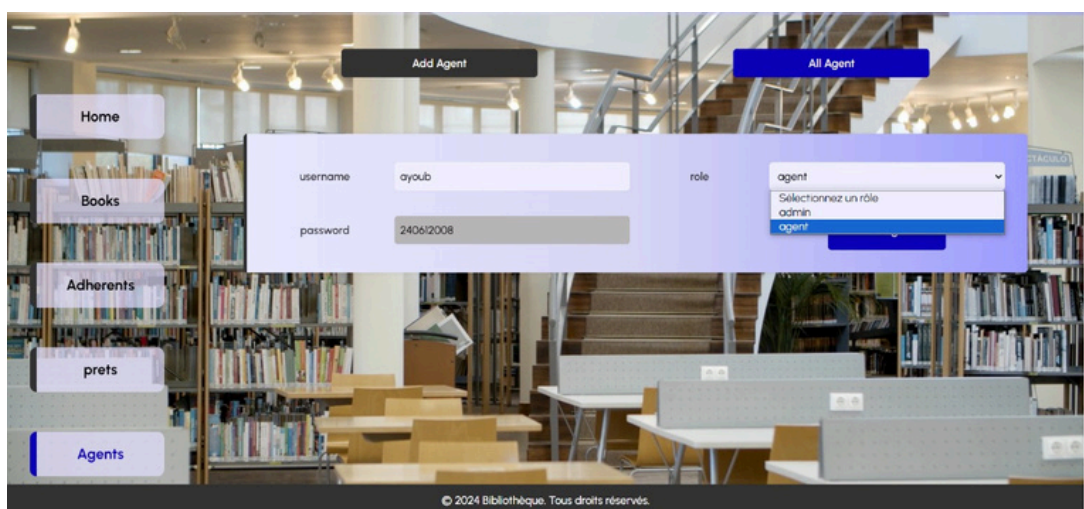
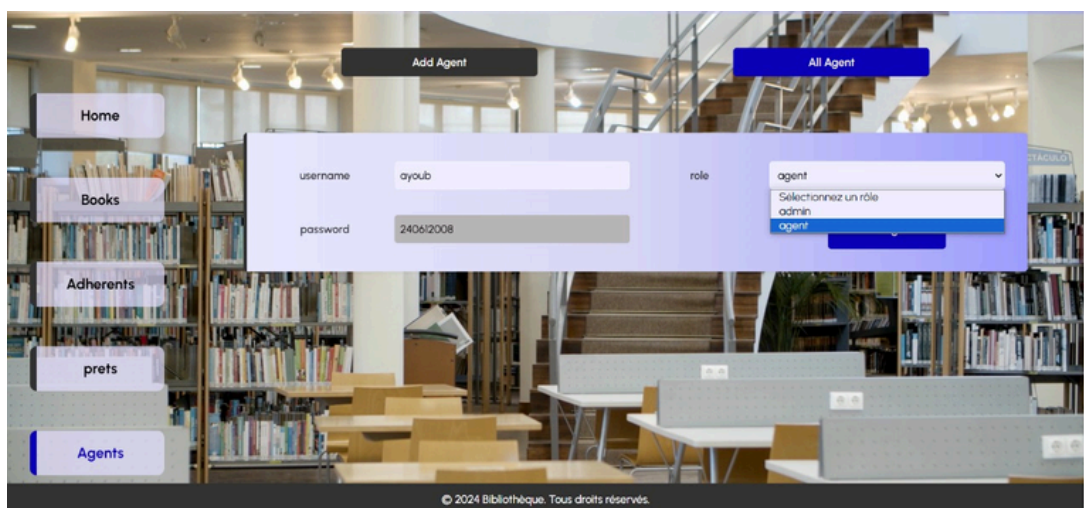
The screenshot shows the 'prêts' (loans) section of the EmiBook application. A sidebar on the left contains navigation links: Home, Books, Adherents, prêts, and Agents. The main area displays a table of loans. A red arrow points to the 'status' column for a loan of 'The Hunchback of Notre Dame', which shows 'retourné'.

adherent	livre	date De Pret	date De Retour	status
marouane	The Hunchback of Notre Dame	11/06/2024		retourné



Création d'agent

En cliquant sur add agent on sauvegarde cet agent et pour le voir il suffit de cliquer sur All Agent



Conclusion

Le principal objectif de ce projet était de tester nos connaissances en développement de manière approfondie, en mettant un accent particulier sur les bases de données NoSQL. Nous avons exploré divers concepts de modélisation de données, d'optimisation des requêtes et de gestion des utilisateurs dans un contexte réel, à savoir la gestion d'une bibliothèque.

Avec MongoDB, nous avons amélioré la gestion des données structurées de notre application de gestion de bibliothèque en utilisant des collections spécifiques pour les livres, les utilisateurs, les adhérents et les prêts. Cette approche a considérablement boosté les performances des requêtes tout en garantissant une gestion sécurisée des utilisateurs grâce à des rôles et des permissions bien définis.

L'intégration de Neo4j a enrichi notre approche en permettant de modéliser les relations complexes sous forme de graphes. Cette technologie s'est révélée particulièrement efficace pour représenter les liens entre les livres et leurs auteurs, ainsi que pour gérer les interactions entre ces derniers. L'utilisation des graphes a simplifié la navigation à travers les relations, améliorant la visualisation et l'analyse des données interconnectées.

Enfin, synchroniser les données entre MongoDB et Neo4j a constitué un défi significatif mais essentiel pour maintenir la cohérence et l'intégrité des informations dans les deux bases de données. Cette synchronisation nous a permis de tirer parti des forces respectives de chaque technologie tout en conservant la flexibilité et les performances nécessaires pour répondre aux exigences évolutives de notre application.

En conclusion, ce projet a consolidé nos compétences en développement logiciel en mettant en lumière les avantages et les défis spécifiques aux bases de données NoSQL. Il nous a également offert une expérience pratique précieuse dans l'implémentation de solutions technologiques modernes pour répondre aux besoins complexes des applications de gestion de données en temps réel.