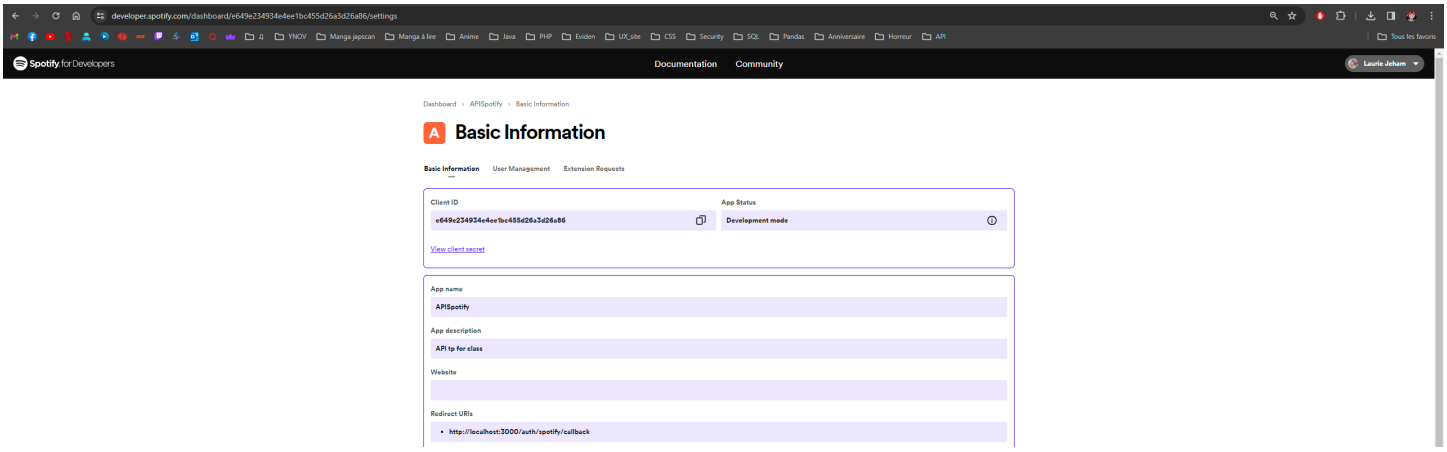
RAPPORT
API SPOTIFY

LAURIE JEHAM DAVY MARTHELLY

Introduction

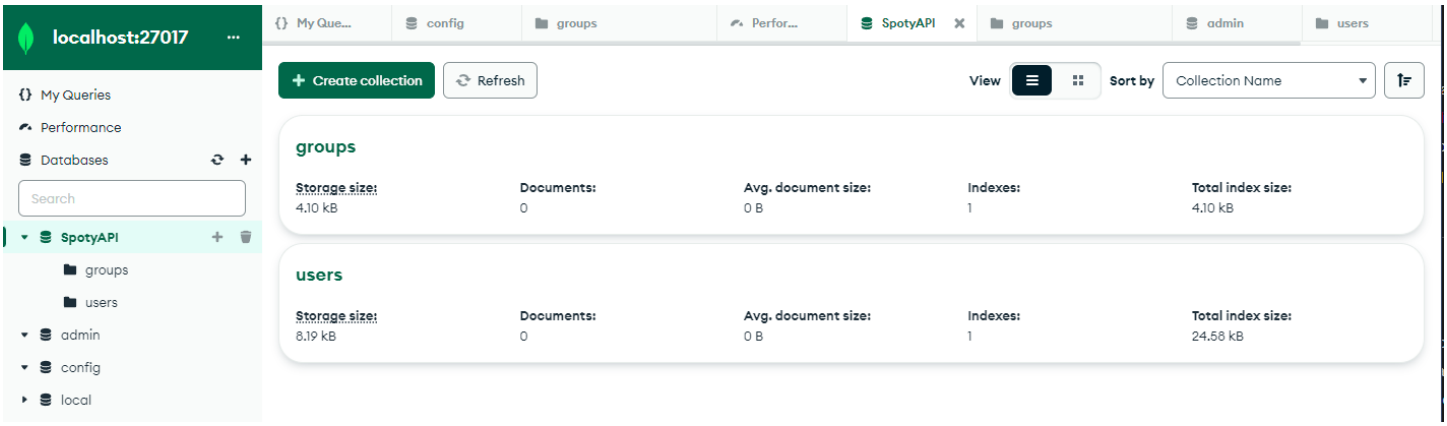
Ce rapport documente le développement d'une extension de l'API Spotify, conçue pour enrichir l'expérience utilisateur en fournissant des fonctionnalités de gestion de groupes et de personnalisation de playlists. L'objectif était de créer une application intuitive et performante qui s'intègre harmonieusement à l'API Spotify existante, tout en offrant une documentation claire et accessible via SwaggerUI.

(Developer.spotify obligatoire afin d'utiliser l'API)



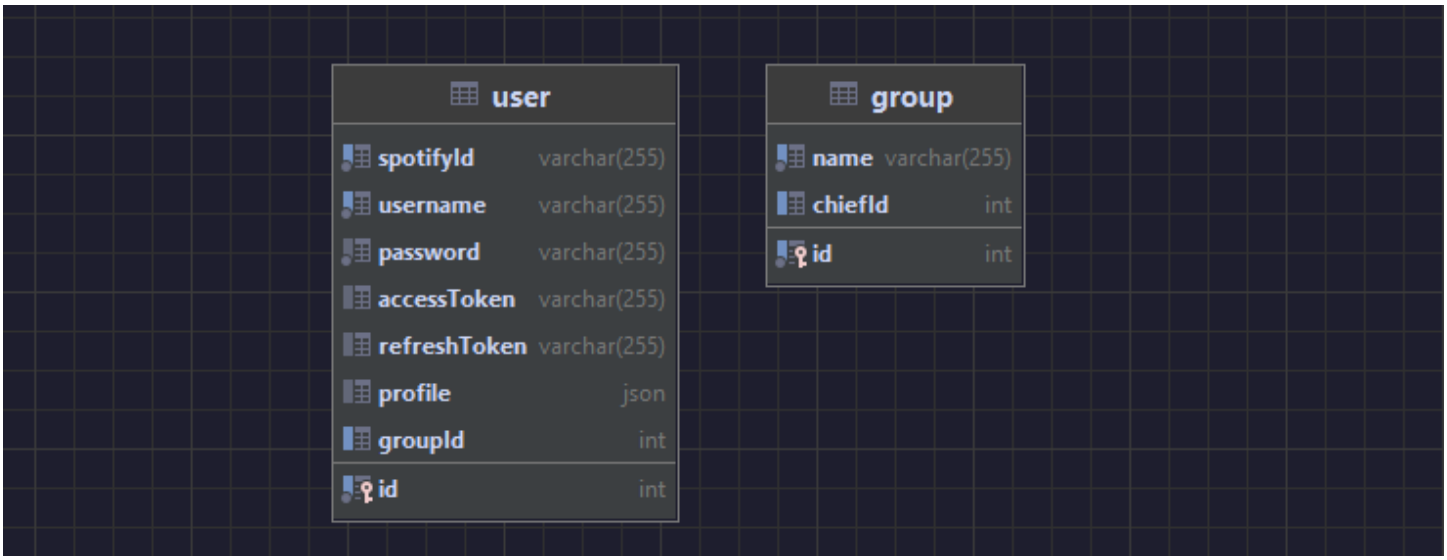
Choix Techniques et Implémentation

Utilisation d'Express.js et Mongoose



Le code montre l'utilisation d'Express.js comme framework pour la construction de l'infrastructure du serveur. Express.js est particulièrement apprécié pour sa simplicité et sa flexibilité dans la définition de routes et la gestion des requêtes HTTP. Parallèlement, Mongoose est employé pour simplifier les interactions avec la base de données MongoDB, en fournissant un mapping objet-document (ODM) intuitif.

Dans le fichier app.js, les routes sont définies de manière claire. Par exemple, la route /register est utilisée pour l'inscription des utilisateurs, où les données sont validées et sauvegardées dans la base de données MongoDB.



Authentification avec Passport.js

Passport.js est un outil puissant pour l'authentification dans les applications Node.js. Dans ce projet, deux stratégies d'authentification sont mises en œuvre : la stratégie locale pour l'authentification basée sur le nom d'utilisateur et le mot de passe, et la stratégie Spotify pour l'authentification via OAuth avec Spotify.

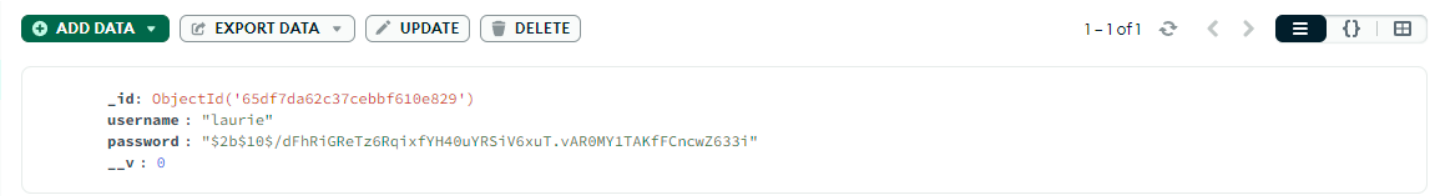
Le fichier passport.js illustre la configuration de Passport.js avec ces deux stratégies. La stratégie locale vérifie les informations d'identification fournies par l'utilisateur, tandis que la stratégie Spotify

utilise les clés d'API de Spotify pour valider les utilisateurs.

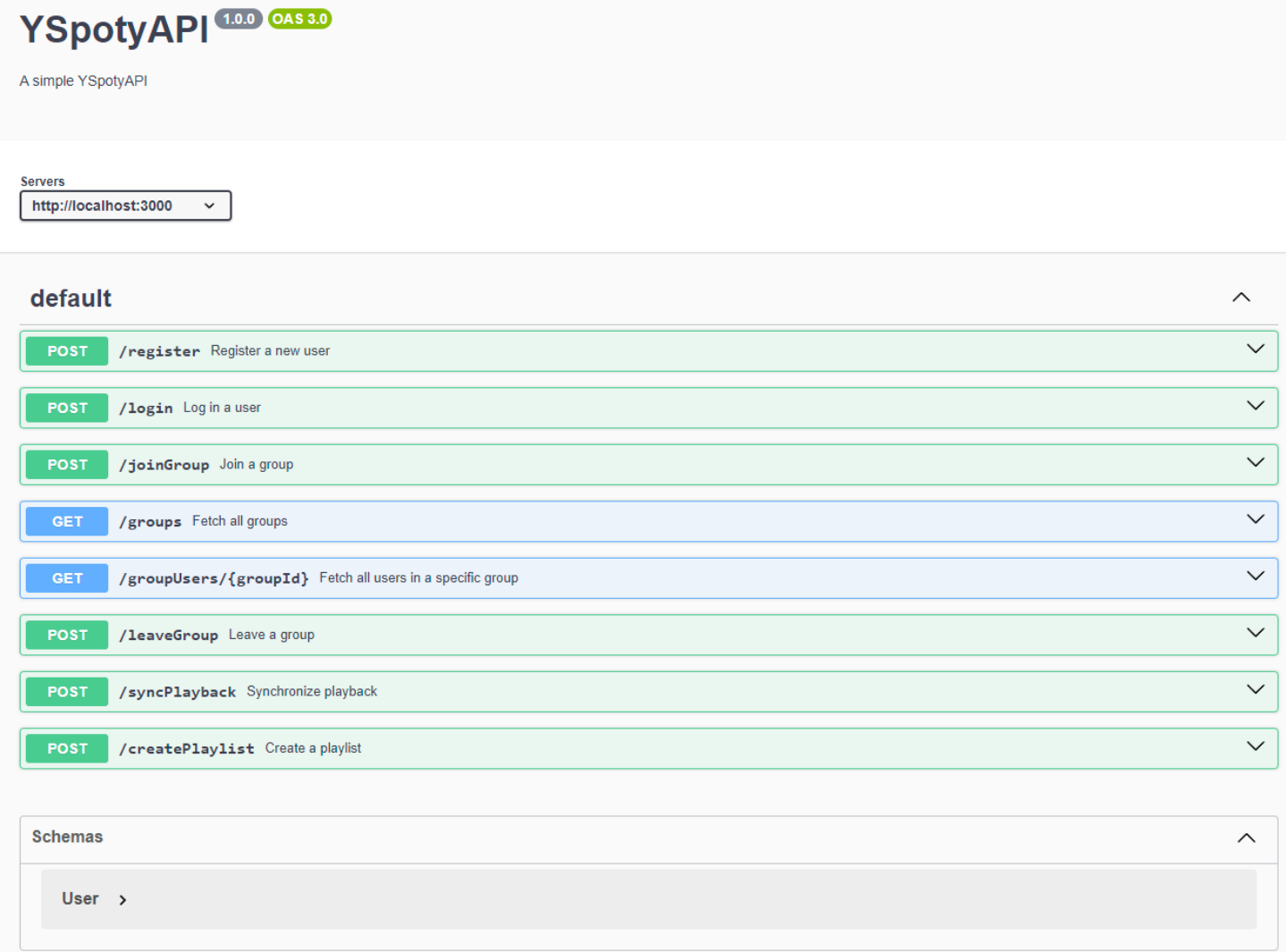
Stockage des Données avec MongoDB

MongoDB est choisi comme système de gestion de base de données en raison de sa flexibilité et de sa scalabilité. Les schémas de données sont définis à l'aide de Mongoose, permettant ainsi de structurer les données utilisateur, les groupes, et les sessions de manière organisée et cohérente.

Le fichier user.js contient la définition des schémas de données pour les utilisateurs et les groupes. Ces schémas spécifient les champs requis, les contraintes d'unicité, et les relations entre les entités.



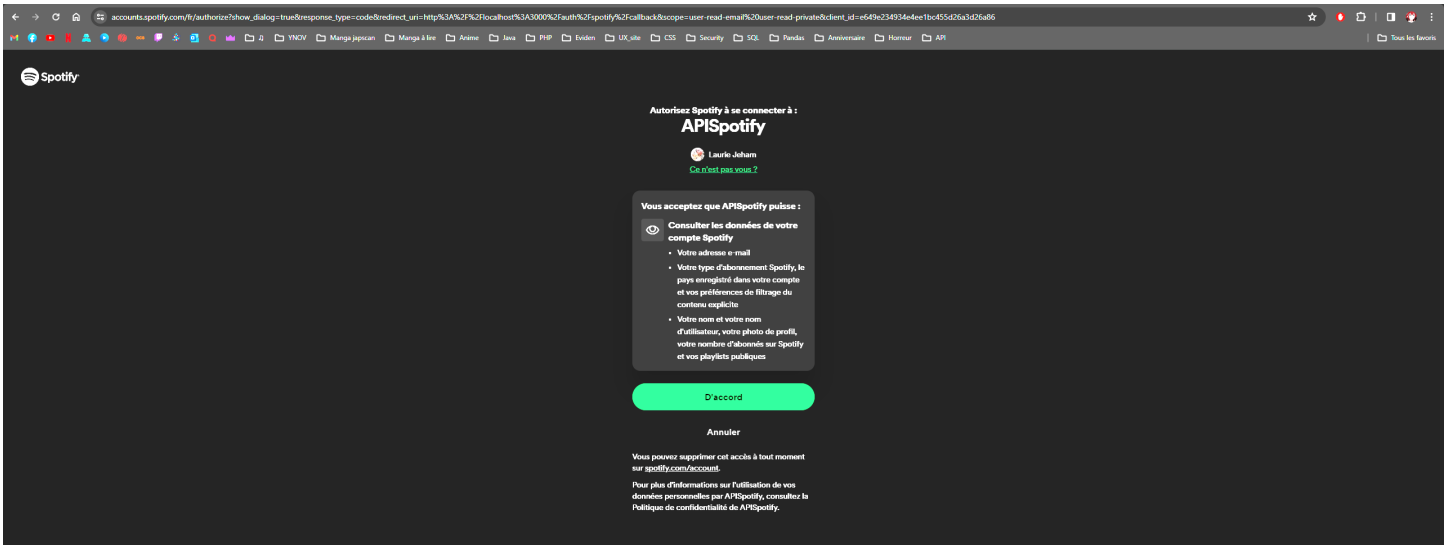
Utilisation de Swagger pour la Documentation de l'API



La documentation de l'API est essentielle pour faciliter l'intégration et l'utilisation de YSpotyAPI par d'autres développeurs. Swagger est utilisé pour générer une documentation interactive et compréhensible de l'API, décrivant chaque endpoint, ses paramètres, et ses réponses possibles.

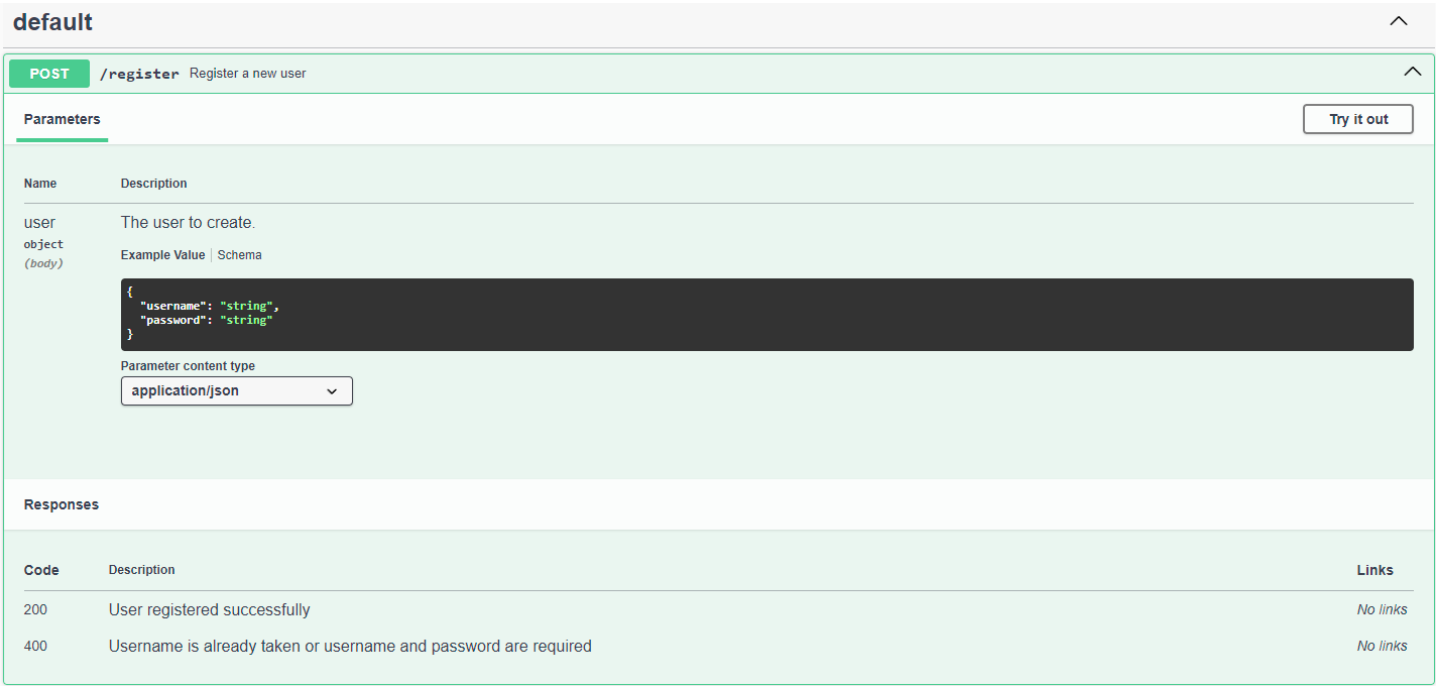
Le fichier swagger.js contient la configuration de Swagger, décrivant les informations de base de l'API, telles que le titre, la version et la description, ainsi que les endpoints définis dans le fichier app.js.

Intégration avec l'API Spotify



L'intégration avec l'API Spotify permet aux utilisateurs d'accéder à leurs informations musicales et de synchroniser leur lecture de musique avec leur groupe. Dans le fichier app.js, des routes sont définies pour gérer l'authentification avec Spotify, la récupération des données musicales, la création de playlists, et la synchronisation de la lecture entre les membres du groupe.

exemple de /register :



Défis et Solutions

Un des défi a été l'intégration avec l'API Spotify, notamment la gestion des tokens d'accès et le rafraîchissement de ces derniers. Nous avons résolu ce problème en implémentant un mécanisme de rafraîchissement automatique des tokens, assurant une expérience utilisateur fluide sans interruptions.
Pour ma part (Laurie), j'avais réalisé mon ancien projet avec le langage JAVA, et le faire en JavaScript m'a demandé plus d'effort de recherche. Ce fut un bon exercice.

Conclusion

L'extension de l'API Spotify que nous avons développée offre des fonctionnalités enrichissantes pour les utilisateurs, améliorant leur expérience d'écoute musicale en groupe. Ce projet a non seulement mis en évidence notre capacité à travailler avec des technologies modernes de développement web, mais aussi notre engagement à fournir des solutions innovantes et de qualité.