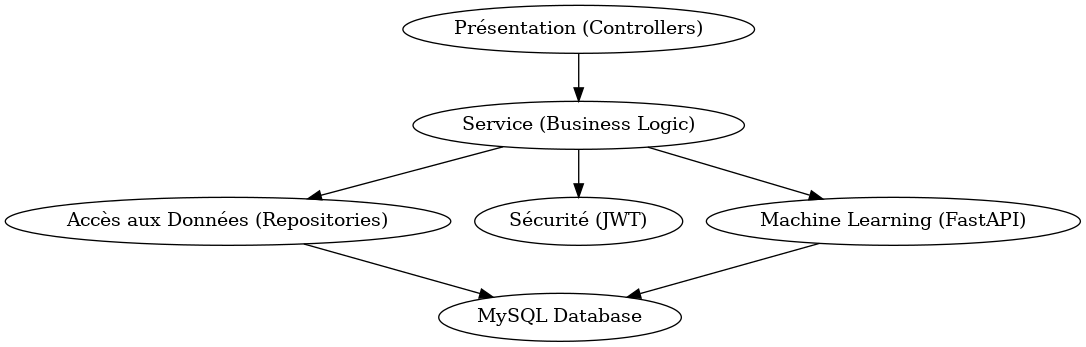
# Documentation du projet Spring Boot - GamesUp

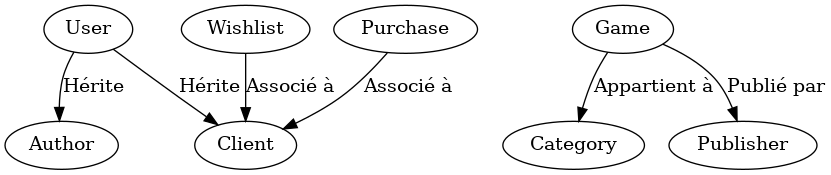
## 1. Diagramme d'Architecture

L'architecture de l'application repose sur un modèle en couches :  
- Présentation : Contrôleurs Spring Boot (ex. AuthController, GameController).  
- Service : Contient la logique métier (ex. GameService, AuthService).  
- Accès aux données : Repositories utilisant Spring Data JPA.  
- Sécurité : Gestion JWT avec JwtUtil et JwtAuthenticationFilter.  
- Machine Learning : API Python pour recommandations.



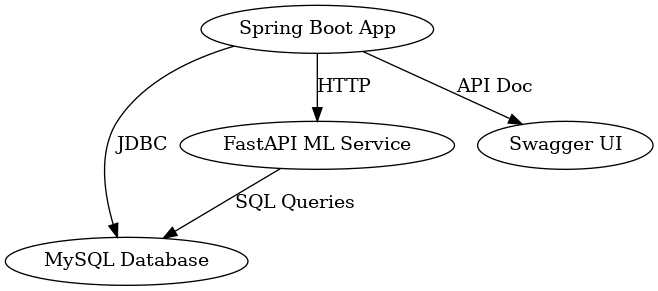
## 2. Diagramme de Classes

Le diagramme de classes présente les principales entités et leurs relations :  
- User : Classe mère avec héritages (Client, Author).  
- Game : Associé à Category, Publisher et Author.  
- Purchase : Associe Client et PurchaseLine.  
- Wishlist : Liée à Client et WishLine.



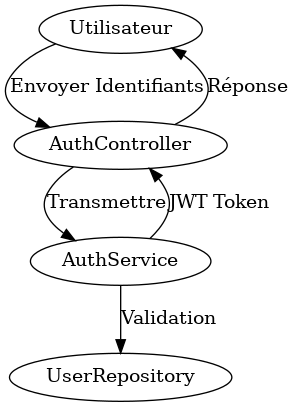
## 3. Diagramme de Composants

Les composants principaux sont organisés de la manière suivante :  
- Spring Boot : Application backend.  
- MySQL : Base de données.  
- FastAPI : API externe pour recommandations.  
- Swagger : Documentation API.



## 4. Diagramme de Séquence (Exemple d'authentification)

Le processus d'authentification fonctionne de cette manière :  
1. L'utilisateur envoie ses identifiants.  
2. AuthController transmet les données à AuthService.  
3. AuthService valide avec UserRepository.  
4. Si succès, génération de JWT.  
5. Réponse avec le token JWT.



## 5. Principes SOLID et bonnes pratiques

L'application respecte les principes SOLID et les bonnes pratiques de développement logiciel :  
- S : Single Responsibility Principle - AuthService suit le principe de responsabilité unique.  
- O : Open/Closed Principle - Extension possible via interfaces CRUDController.  
- L : Liskov Substitution Principle - User et ses sous-classes respectent la substitution de Liskov.  
- I : Interface Segregation Principle - Interfaces spécifiques avec Spring Data JPA.  
- D : Dependency Inversion Principle - Injection de dépendances via Spring.

## 6. Rapports de Couverture de Tests

Les tests de l'application incluent :  
- Tests unitaires : Vérification des services et contrôleurs.  
- Tests d'intégration : Test de l'authentification et des fonctionnalités CRUD.  
Les résultats sont accessibles via des outils comme Jacoco ou Spring Boot Actuator.

## 7. Mise en place du système de recommandation

Le système de recommandation est basé sur un modèle KNN (K-Nearest Neighbors) et utilise :  
- Données issues de la base MySQL.  
- Algorithme de KNN avec la bibliothèque scikit-learn.  
- Communication entre Spring Boot et FastAPI via des requêtes HTTP.  
- JSON pour l'échange de données entre les composants.  
Le modèle est entraîné sur les historiques d'achats et évaluations des utilisateurs.

## Réflexion sur le projet

Durant le développement de ce projet, certaines bonnes pratiques ont été respectées, notamment :  
- Architecture en couches.  
- Respect des principes SOLID.  
- Utilisation de JWT pour sécuriser l'API.  
Cependant, quelques difficultés ont été rencontrées :  
- Gestion d'erreurs perfectible.  
- Optimisation des performances à améliorer.