Soutenance de Projet

SAE-821 Développement d'un moteur de recommandations respectueux de la vie privée

Université de Toulon, La Garde

2 Juin 2025

Sommaire

1. Introduction
2. Problématique
3. Conception UML
4. Fonctionnalités
5. Architecture du système
6. Choix et démarche
7. Déroulement du projet
8. Résultats
9. Conclusion
10. Perspectives

Introduction

Introduction

Les moteurs de recommandation sont devenus omniprésents dans notre quotidien : sur les plateformes de streaming, les sites de e-commerce ou encore les réseaux sociaux. Leur but est de suggérer à l'utilisateur du contenu pertinent en se basant sur ses préférences, son historique de navigation ou les comportements d'utilisateurs similaires.

Pour atteindre cette efficacité, ces systèmes s'appuient sur l'analyse de vastes quantités de données personnelles. Cette collecte, souvent invisible pour l'utilisateur, soulève de nombreuses préoccupations liées à la confidentialité et à la protection de la vie privée.

Dans le cadre de ce projet, nous nous sommes donné pour objectif de concevoir un moteur de recommandations simple mais performant, capable de respecter les données personnelles des utilisateurs, en explorant des approches innovantes de préservation de la vie privée.

Problématique

Problématique

★ **Problème central :** *Comment proposer des recommandations pertinentes sans compromettre la vie privée des utilisateurs ?*

✗ **Limites actuelles :**

- Solutions **centralisées** : profilage, tracking, fuites de données.
- Collecte massive de données personnelles.

▲ **Deux approches de protection :**

- **Differential Privacy** : bruit pour anonymiser.
- **Cryptographie homomorphe** : calcul sur données chiffrées.

✻ **Objectif :** Développer un moteur de recommandation basé sur `MovieLens`, tout en garantissant la confidentialité.

Conception UML

User Story

Le point de départ de notre projet repose sur une vision centrée utilisateur. La user story suivante résume notre objectif principal :

« En tant qu'utilisateur, je veux recevoir des recommandations personnalisées sans que mes données personnelles soient stockées en ligne. »
« En tant qu'utilisateur, je veux pouvoir consulter mes avis sur les films que j'ai vus, sans que mes données personnelles soient stockées en ligne. »
« En tant qu'utilisateur, je veux pouvoir rechercher des films par titre/genres, sans que mes données personnelles soient stockées en ligne. »

Cette phrase illustre une exigence fondamentale : proposer un service personnalisé tout en assurant une protection forte de la vie privée.

Diagramme de cas d'utilisation

Ce qui nous donne le diagramme de cas d'utilisation suivant pour un utilisateur :

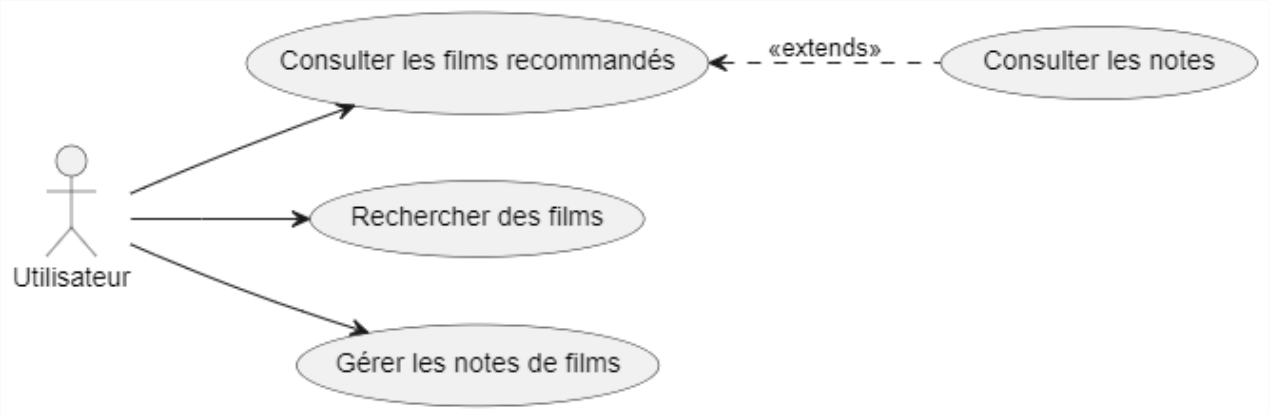

Diagramme de cas d'utilisation de l'utilisateur

Fonctionnalités

Fonctionnalités – Vue générale

Notre moteur de recommandations a été conçu pour offrir une expérience utilisateur complète, tout en respectant les exigences de confidentialité. Il se distingue par quatre caractéristiques principales :

- **Recommandations personnalisées** basées sur les préférences de l'utilisateur, calculées localement.
- **Stockage local sécurisé** : les données personnelles ne sont jamais centralisées, et les mots de passes sont **hachés** dans la base de données pour éviter toute identification directe.
- **Confidentialité préservée dès la conception**, grâce à l'intégration de techniques de **Differential Privacy** dans les communications.
- **Interface claire et intuitive** via `React`, assurant une navigation fluide et rapide.

Fonctionnalités – Techniques

D'un point de vue technique, notre solution repose sur une architecture modulaire et des technologies robustes :

- **Filtrage collaboratif par SVD (Singular Value Decomposition)** : permet de réduire la dimensionnalité et d'identifier les similarités entre utilisateurs.
- **Base de données noSql** : pour faciliter le stockage et la récupération des données d'interaction utilisateur.
- **Differential Privacy** : ajout de bruit contrôlé aux données lors des échanges avec le serveur de recommandation, afin de protéger les informations sensibles.
- **Communication sécurisée par endpoints** : les échanges entre le back-end (serveur Quarkus) et le front-end sont encapsulés dans des requêtes REST sécurisées.
- **Interface avec React** : permet d'afficher dynamiquement les recommandations dans un environnement web léger.

Architecture du système

Diagramme de classes

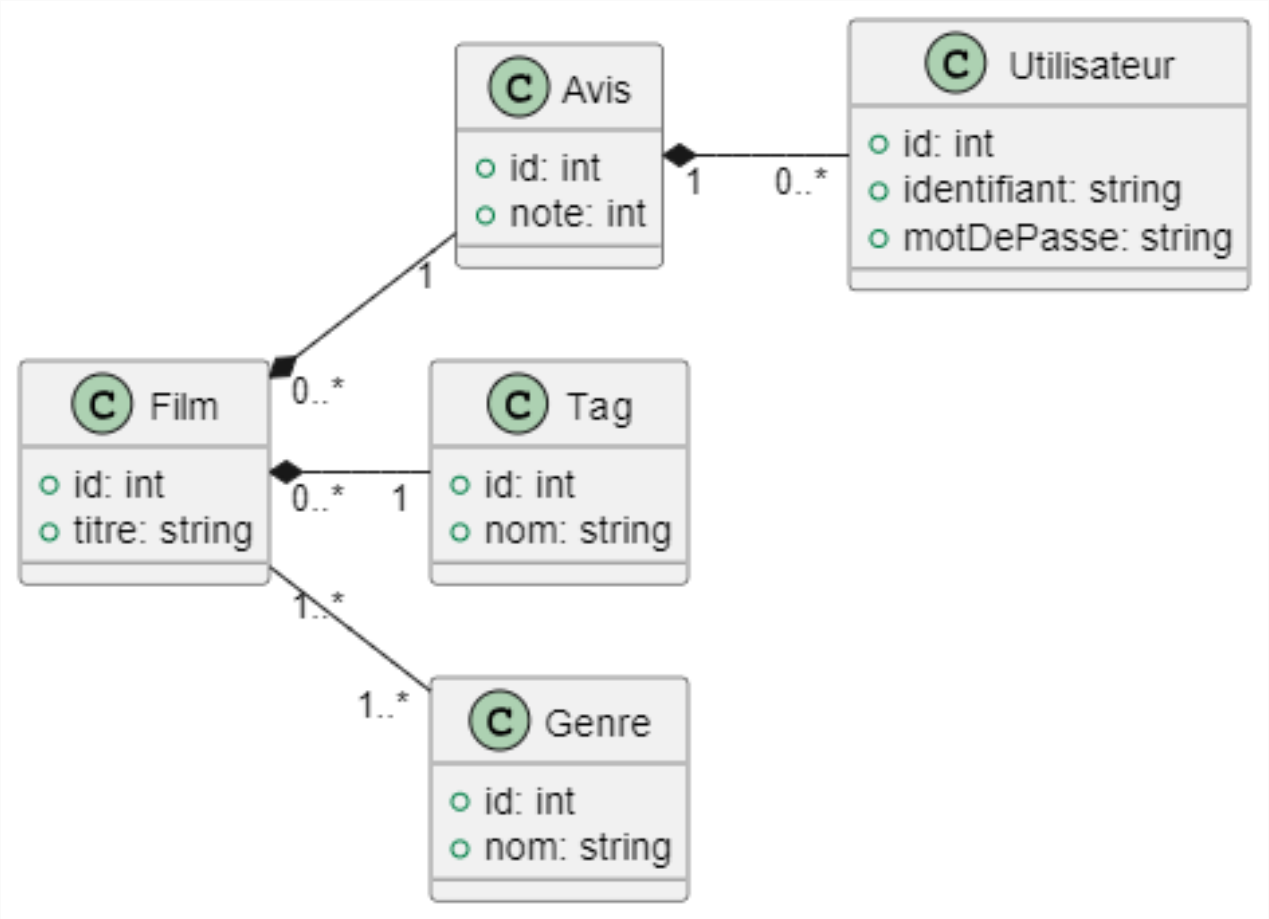

Figure 1: Diagramme de classes du système

Diagramme de deploiement

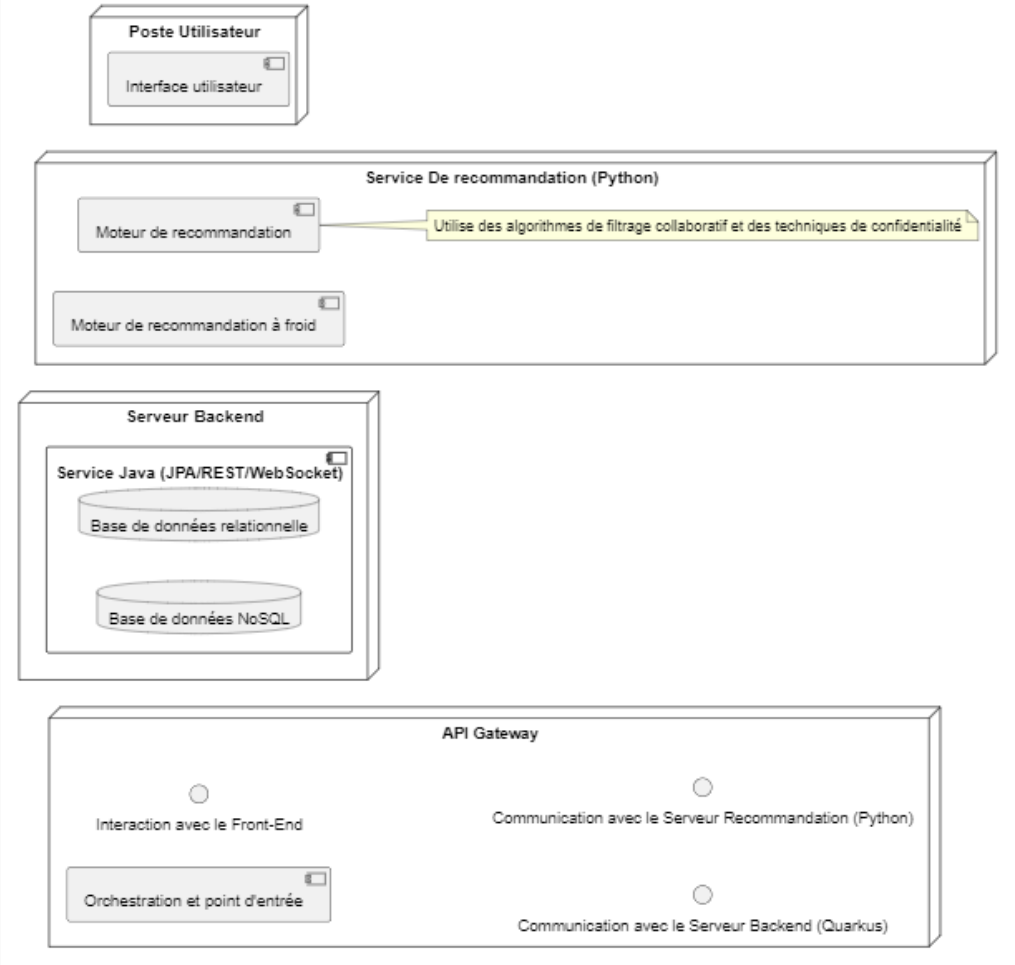

Figure 2: Diagramme de déploiement du système

Modules principaux

- **Interface utilisateur** : Streamlit pour l'interaction
- **Moteur de recommandation** : filtrage collaboratif local
- **Base de données locale** : stockage des interactions
- **Système d'anonymisation** : confidentialité des échanges

Choix et démarche

Système de recommandation à chaud

Contexte : Utilisé lorsque l'utilisateur a un historique de notes ou d'interactions avec le système.

Méthodes utilisées :

- **Filtrage collaboratif** : basé sur les préférences d'utilisateurs similaires.
- **Matrix Factorization (SVD)** : extraction de facteurs latents pour modéliser les préférences.
- **Approche hybride** : combinaison user-based et item-based.

Choix justifiés :

- Fortes performances avec un dataset riche (MovieLens).
- Algorithmes bien adaptés à la personnalisation dynamique.
- Intégration fluide avec la Differential Privacy pour les échanges sécurisés.

Système de recommandation à froid

Contexte : Utilisé lorsque l'utilisateur n'a pas d'historique de notes ou pour les nouveaux utilisateurs.

Méthodes utilisées :

- **Filtrage collaboratif sur métadonnées** : exploitation des attributs de films ou d'utilisateurs.
- **Apprentissage supervisé** : prédiction des préférences à partir de profils (ex : genres, âge, popularité).
- **Modèles hybrides** : combinaison de règles et modèles ML pour améliorer la couverture.

Choix justifiés :

- Meilleure robustesse face au problème du démarrage à froid.
- Possibilité d'intégrer des données sans historique personnel.
- Complémentarité avec le système à chaud pour une couverture complète.

Déroulement du projet

Déroulement du projet

Le projet a été mené selon une méthodologie **agile** structurée en **sprints hebdomadaires**. Chaque lundi, une réunion de planification permettait de définir les tâches à réaliser, accompagnées d'un **chiffrage en points de charge**. Un **rapport journalier textuel** était rédigé tout les jours pour assurer un suivi continu de l'avancement.

Outils :

- **Python** : langage principal pour le développement du moteur de recommandation
- **GitHub** : collaboration, versionnement, documentation
- **Quarkus** : architecture backend
- **React** : front-end pour la visualisation locale

Tous les éléments (code, docs, rapports) sont sur GitHub pour une architecture micro-service et une collaboration efficace.

Résultats

Résultats obtenus : SVD

Notre système a permis d'atteindre plusieurs objectifs clés, tant sur le plan fonctionnel que technique :

- **Prototype opérationnel** : système complet de bout en bout avec SVD.
- **Qualité des recommandations** :
  - Précision : 72.2% / 71.8%
  - Rappel : 69.1% / 68.6%
- **Respect de la vie privée** :
  - Échanges bruités par Differential Privacy
- **Mesure des performances (temps)** :
  - Initialisation des données : 0.14 s
  - Entrainement du modèle : 0.72 s
  - Temps de réponse moyen : 0.03 s

Résultats obtenus : Deep Learning

Notre système a permis d'atteindre plusieurs objectifs clés, tant sur le plan fonctionnel que technique :

- **Prototype opérationnel** : système complet de bout en bout avec un modèle d'apprentissage profond.
- **Qualité des recommandations** :
  - Précision : 72.6 % / 71.3%
  - Rappel : 69.3% / 67.5%
- **Respect de la vie privée** :
  - Échanges bruités par Differential Privacy
- **Mesure des performances (temps)** :
  - Initialisation des données : 0.08 s
  - Entrainement du modèle : 0.68 s
  - Temps de réponse moyen : 0.03 s

Conclusion

Conclusion

- Projet mené avec rigueur et efficacité
- Objectifs respectés
- Confidentialité au cœur du développement
- Enrichissement personnel et technique

Perspectives

Perspectives d'évolution

- **Explorer d'autres approches de recommandation à froid** : améliorer la rapidité et la précision, notamment en situation sans historique utilisateur.
- **Repenser l'architecture microservices** : réduire les fortes dépendances entre les modules pour gagner en modularité, maintenabilité et résilience.
- **Automatiser l'initialisation des données** : remplacement des appels manuels aux endpoints par des scripts d'initialisation ou des pipelines de données.
- **Intégrer un modèle de langage (LLM)** : générer des recommandations contextuelles plus riches grâce à l'exploitation de texte ou de préférences complexes.
- **Améliorer l'interface utilisateur** : créer un front-end plus moderne, esthétique et responsive pour renforcer l'engagement utilisateur.

UNIVERSITÉ DE TOULON

Merci pour votre attention !

N'hésitez pas à poser vos questions !