Projet de feedback avec GraphQL et Spring Boot

1)Introduction

Dans le cadre du développement d'applications modernes, la gestion des retours utilisateurs est devenue un élément clé pour améliorer la qualité des produits et services. Les systèmes traditionnels de gestion des feedbacks, souvent basés sur des formulaires web classiques ou des API REST, montrent leurs limites en termes de flexibilité et d'efficacité des requêtes. Par ailleurs, la montée en puissance de GraphQL offre une nouvelle approche permettant d'interroger et manipuler les données de manière plus ciblée et optimisée. Ce projet s'inscrit donc dans cette tendance en développant une application utilisant GraphQL pour gérer les retours clients.

2)Objectif principal

- Permettre aux utilisateurs de donner des feedbacks sur des produits
- Stocker et consulter ces feedbacks via une API GraphQL
- Générer des rapports PDF à partir des données de feedback
- Faciliter l'analyse de la satisfaction utilisateur

3)Problématique

Les solutions existantes pour la gestion des feedbacks souffrent souvent d'un manque de flexibilité dans les requêtes, générant des surcoûts réseau et une surcharge des données non pertinentes. De plus, il est fréquent que les rapports synthétiques demandent un traitement manuel fastidieux, ce qui ralentit la prise de décision. Comment concevoir un système qui optimise l'accès aux données de feedback via GraphQL tout en automatisant la création de rapports personnalisés et facilement exploitables .Ce projet vise à répondre à cette problématique en

combinant une API moderne avec une fonctionnalité de génération de rapports PDF intégrée.

4)Solution proposée

Pour répondre au besoin de l'entreprise SaaS de centraliser et analyser les retours utilisateurs, nous avons conçu un **système de feedback structuré** en utilisant **Spring Boot** et **GraphQL**.

Architecture technique :

- **Backend :** Spring Boot avec Web Services GraphQL.
- **Base de données :** MySQL (ou H2 pour tests).
- **API GraphQL :**
  - `addUser`, `addFeedback` (mutations pour insérer des données).
  - `allProducts`, `feedbacksByProduct`, `averageRating` (requêtes pour afficher les avis).
- **Entités principales :** `User`, `Product`, `Feedback`.
- **Relations :**
  - Un utilisateur peut donner plusieurs feedbacks.
  - Un produit peut recevoir plusieurs feedbacks.

Fonctionnalités clés :

- Ajouter un utilisateur.
- Lister les produits.
- Donner un avis avec note et commentaire sur un produit.
- Calculer la note moyenne d'un produit.
- Afficher tous les avis associés à un produit.

Avantages de la solution :

- Structure flexible avec GraphQL (récupération ciblée de données).
- Code maintenable et bien structuré.
- Extensible pour intégrer des statistiques, notifications, ou tableaux de bord.

5)Spécifications fonctionnelles

- **Gestion des utilisateurs** : ajout d'un utilisateur (nom, email)
- **Gestion des produits** : affichage de la liste
- **Ajout de feedback** (note, commentaire, date, produit, utilisateur)
- **Consultation des feedbacks par produit**
- **Calcul de la moyenne des notes**
- **Génération de rapport PDF des feedbacks**

6)Spécifications non fonctionnelles

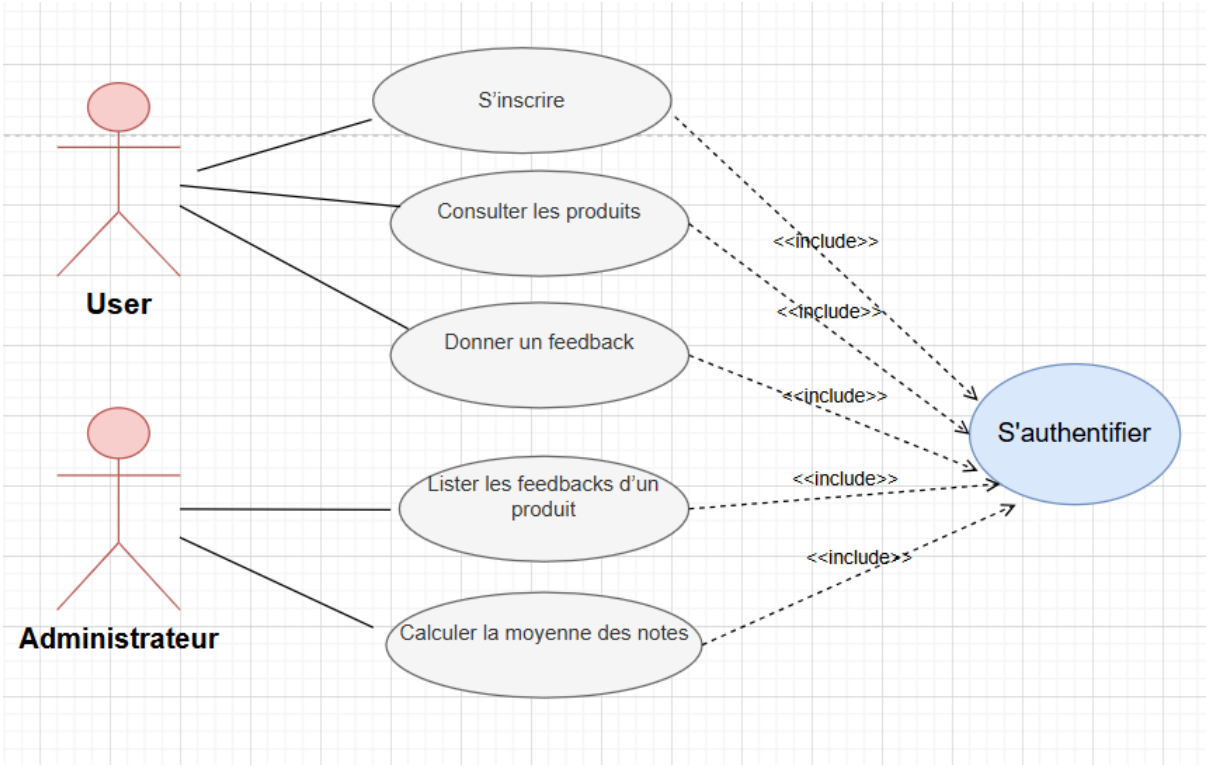
- Application web REST/GraphQL
- Performance : chargement rapide des données
- Sécurité basique (validation, intégrité des données)
- Portabilité : déployable sur serveur Tomcat

7)Architecture du projet

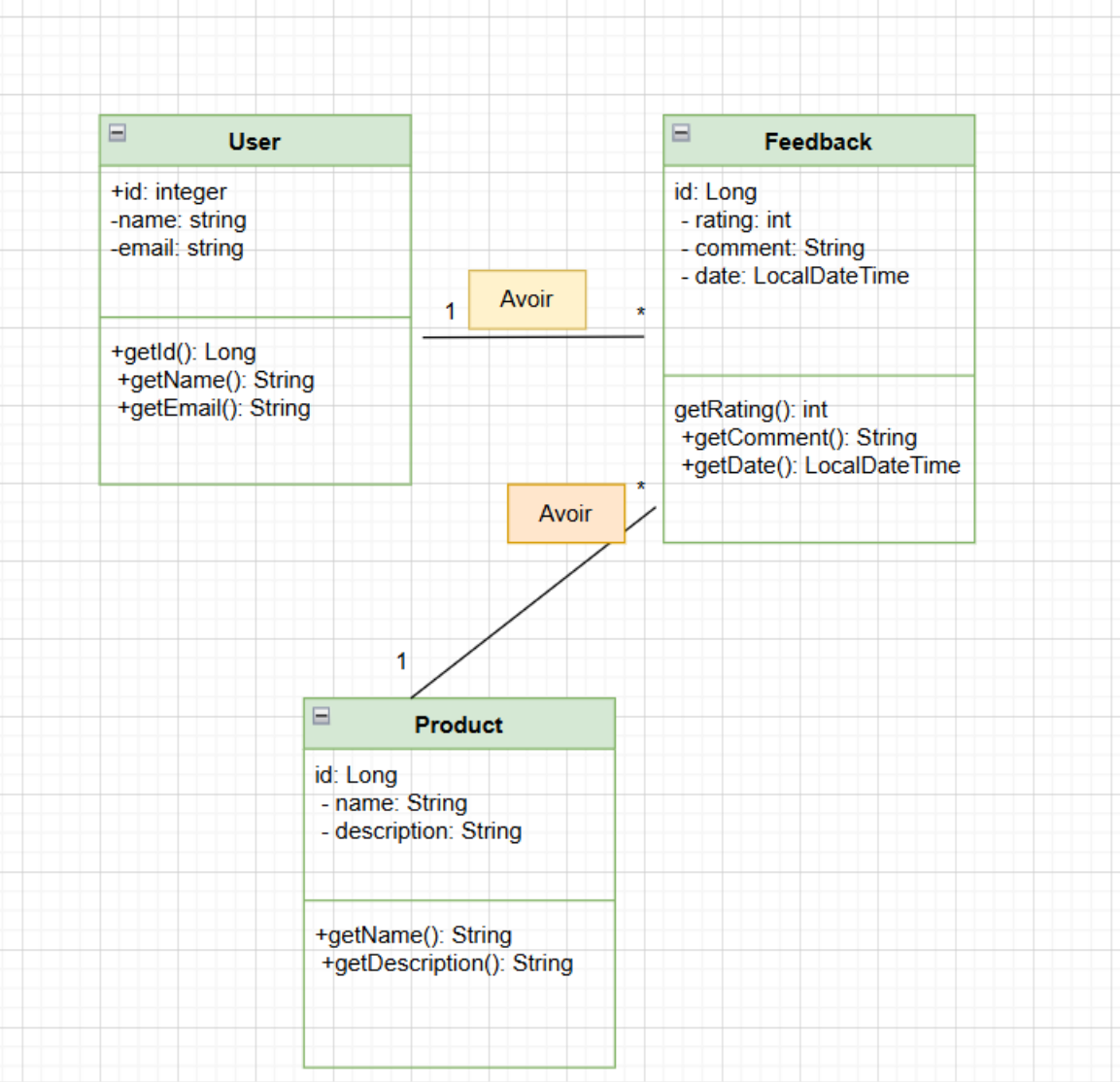
Architecture 3 couches :

- **Contrôleurs GraphQL / REST**
- **Services métiers**
- **Entités / Repositories JPA**
- Diagrammes UML :

✓ Diagramme de cas d'utilisation



✓ Diagramme de classes



8)Technologies utilisées

- Backend : Java 17, Spring Boot
- Base de données : H2 / MySQL
- GraphQL : Spring for GraphQL
- Génération PDF : iText
- Outils : IntelliJ / Spring Tools Suite, Postman, Git, Maven

9)Conclusion

Ce projet vise à résoudre l'absence de système structuré de collecte de feedbacks pour une entreprise SaaS. Grâce à une solution développée avec **Spring Boot** et **GraphQL**, nous avons mis en place une architecture efficace pour recueillir, gérer et exploiter les avis utilisateurs. Le système

est extensible, maintenable, et répond pleinement aux besoins de l’entreprise en matière d’analyse des retours clients.