Projet Fullstack

Nom du projet : Cristal

Nom du groupe : Cristaline

1. Description du projet

Cristal est une plateforme web permettant de référencer des jeux vidéo et de suivre son expérience de joueur, inspirée d'un concept similaire à MyLetterboxd mais appliqué aux jeux vidéo.

L'objectif du MVP est de proposer un hub simple où les utilisateurs peuvent :

- Consulter un catalogue de jeux
- Noter des jeux
- Ajouter des jeux en favoris
- Voir les favoris d'autres utilisateurs
- Importer des jeux depuis une source externe gratuite
- Démontrer une communication distribuée via Kafka

Le périmètre est volontairement limité afin de garantir la qualité, la stabilité et la reproductibilité du projet.

---

2. Architecture technique

Frontend

- Angular
- Architecture MVC

- Communication avec le backend via une REST API

Backend

- Java Spring Boot
- Architecture 3-tiers :
    - Controller
    - Service
    - Repository
- REST API
- Kafka pour la communication asynchrone via des topics

Base de données

- H2 (base embarquée pour le développement et les tests)

DevOps et Build

- GitHub pour la gestion du code source (main + branches de fonctionnalités)
- GitHub Actions pour l'intégration continue (build + tests + rapport de couverture)
- Docker pour la conteneurisation
- docker-compose pour exécuter la stack complète (backend + Kafka + frontend)

---

3. Fonctionnalités du MVP

Catalogue

- Liste des jeux

- Détail d’un jeu
- Recherche par titre
- Filtrage par genre ou année

Notes

- Un utilisateur peut attribuer une note à un jeu
- Calcul et affichage de la note moyenne

Favoris

- Ajouter un jeu à ses favoris
- Voir la liste de ses favoris
- Voir les favoris d’un autre utilisateur

Administration

- Ajouter un jeu au catalogue
- Modifier les informations d’un jeu
- Supprimer un jeu
- Importer un jeu depuis une API externe gratuite

Kafka

- Publication d’un événement lors :
    - De l’ajout d’un jeu
    - De la modification d’un jeu
    - De l’import d’un jeu externe

- Possibilité d’utiliser un topic dédié pour les demandes d’import, consommé par un service backend

---

4. User Stories

En tant qu’utilisateur :

Je veux voir la liste des jeux disponibles.
Je veux consulter le détail d’un jeu.
Je veux rechercher un jeu par son titre.
Je veux filtrer les jeux par genre ou année.
Je veux ajouter un jeu à mes favoris.
Je veux voir la liste de mes jeux favoris.
Je veux voir les jeux favoris d’un autre utilisateur.
Je veux noter un jeu.

En tant qu’administrateur :

Je veux ajouter un jeu au catalogue.
Je veux modifier les informations d’un jeu.
Je veux supprimer un jeu du catalogue.
Je veux importer des jeux depuis une source externe gratuite.
Je veux que chaque ajout, modification ou import publie un événement via Kafka.

Planning du projet

Sprint 1 – 1 semaine

Objectif : Mise en place du socle technique et du catalogue

- Initialisation du repository GitHub
- Mise en place du backend Spring Boot
- Mise en place du frontend Angular
- Configuration de la base H2

- Création du modèle Game
- Implémentation du CRUD complet des jeux
    - Liste
    - Détail
    - Création
    - Modification
    - Suppression
- Recherche par titre
- Filtrage par genre ou année
- Premiers tests unitaires (service + repository)
- Dockerisation du backend

Livrable attendu en fin de sprint :
API fonctionnelle pour la gestion du catalogue et frontend permettant d'afficher la liste et le détail des jeux.

Sprint 2 – 2 semaines

Objectif : Fonctionnalités utilisateurs et intégration Kafka

Fonctionnalités supplémentaires

- Création d'un modèle User simplifié
- Système de favoris
- Système de notes
- Calcul de la note moyenne
- Import d'un jeu depuis une API externe gratuite

Kafka

- Mise en place du cluster Kafka
- Implémentation d'un Producer
    - Publication d'événements lors de l'ajout, modification ou import d'un jeu
- Implémentation d'un Consumer
    - Traitement d'une demande d'import via un topic dédié

Qualité et configuration

- Ajout de tests unitaires complémentaires
- Tests d'intégration
- Mise en place d'une pipeline GitHub Actions
    - Build
    - Exécution des tests
    - Rapport de couverture
- docker-compose complet (backend + frontend + Kafka)
- Documentation Swagger
- Endpoint Actuator /health
- README expliquant le lancement du projet

Livrable attendu en fin de sprint :
MVP complet avec gestion des favoris et des notes, import externe fonctionnel, Kafka démontré et application exécutable via docker-compose.