

# Projet Fullstack

<b>Vision Fonctionnelle</b>	<b>2</b>
Motivation	2
Profils utilisateurs	3
Backlog	4
<b>Organisation et Planning</b>	<b>7</b>
Sprint 0 - Cadrage & Setup (19 février - 26 février)	7
◆ Objectifs	7
◆ Livrables	7
Sprint 1 - MVP Fonctionnel (27 février - 27 mars)	8
◆ Fonctionnel	8
◆ Technique	8
◆ Definition of Done	8
◆ DevOps	8
◆ Tests	9
◆ Livrables	9
Sprint 2 Release (28 mars - 24 avril)	10
◆ Angular	10
◆ Sécurité	10
◆ E2E	10
◆ Tests	10
◆ Industrialisation	10
◆ Production-like	10
<b>Architecture et Stack Technique</b>	<b>12</b>
Architecture Logicielle	12
Plateforme DevOps	13

# Vision Fonctionnelle

## Motivation

Notre projet est une plateforme de référencement de jeux où les utilisateurs peuvent rechercher des jeux selon leur genre, nom, ... afin d'obtenir diverses informations sur celui-ci. Il sera possible de se créer un compte afin d'ajouter ou modifier des jeux.

## Profils utilisateurs

- Guest: Visiteur non connecté à un compte -> Readonly
- User: Un utilisateur qui s'est connecté. Peut créer et modifier ses articles. -> Read/Post only
- Reviewer: User pouvant valider/supprimer des ajouts
- Admin: Possède tous les accès

## Backlog

ID	US	Description	Solution	Features
1	(User) Je veux créer un compte	Le user clique sur un bouton Sign-in, arrive sur une page où il doit remplir un input (champs obligatoires) login , mail et password + matching password. Valide son choix avec un bouton. Il reçoit un mail pour la confirmation de création de compte. Puis est authentifié + redirigé vers la page post authentification (cf authentification)	Mettre Sign in	- SignIn - Verification mail - Token - Pass BCrypt
2	(User) Je veux me log	Le user clique sur Log in, est redirigé vers un formulaire contenant login / password. 2 Cas: 1/L'user n'existe pas en Db dans ce cas on retourne un message disant user or password invalid  2/L'user existe en Db on envoie un email avec un code et on passe l'utilisateur sur une vue pour valider le code. Puis on renvoie un jwt token qui devra être vérifié pour les crud	Log in	- Pass BCrypt - Token jwt - 2FA ?
3	(User) Je veux créer une page pour un jeu		Système ajout de nouveau jeu	- Bouton nouvelle page - Formulaire (Titre description, image,...) - Ajout en DB
4	(Visiteur) Consulter la liste des jeux	Sur la page principale du site, le visiteur a accès à une liste de toutes les pages de jeux visibles disponibles.	requete GET	- accès DB - Format vue - render en front - affichage front
5	(Visiteur) Rechercher un jeu	La page principale dispose d'une barre de recherche dans laquelle il est possible d'insérer	Get /{name}	

		des mots clé afin de chercher un jeu en particulier.		
6	(User, Reviewer, admin) Ajouter un article de jeu sur la plateforme	On clique sur créer un article dans la navbar La vue de création d'article est renvoyée. L user peut dans l'en-tête tagger le jeu sur lequel porte l'article. l'utilisateur remplit le contenu de l'article dans une fenetre text et valide avec un bouton de validation qui post l'article dans un état "not approuved" il apparaîtra sur le site après une review	Post/	
7	(User, Reviewer, admin) Modifier un jeu sur la plateforme		Put/	- versionning - accès DB
8	(admin) Supprimer un jeu sur la plateforme		DEL/	
9	(User) Je veux supprimer un jeu que j'ai ajouté, afin de retirer un contenu erroné.	Sur les articles du user, un bouton "modifier" sera disponible afin qu'il puisse corriger son article.	PUT/	
10	(Reviewer) Valider un jeu avant publication définitive	Le reviewer aura une page dédiée où il pourra valider ou refuser les nouvelles pages soumises par les utilisateurs		
11	(Admin) Je veux supprimer ou bannir des Users		DEL/	- Modification DB
12	(Admin) attribuer ou modifier les rôles des utilisateurs, afin de gérer les permissions sur la plateforme.			
13	(Admin) consulter l'historique des événements (audit Kafka), afin de suivre les			

	modifications du catalogue.			
14	En tant que système, je veux publier un message Kafka lors d'une modification du catalogue, afin de notifier les services partenaires.			
15	En tant que système, je veux consommer les événements Kafka, afin de journaliser les actions effectuées.			
16	En tant que Reviewer, je veux pouvoir supprimer un jeu non conforme, afin de maintenir la qualité des données.			
17	En tant que <b>Visiteur</b> , je veux voir la fiche détaillée d'un jeu, afin d'obtenir toutes les informations le concernant.			
18	En tant que utilisateur banni, je ne dois pas pouvoir me connecter au site avec mes identifiants			

# Organisation et Planning

## Sprint 0 - Cadrage & Setup (19 février - 26 février)

### ◆ Objectifs

- Définir les rôles (Visiteur / Utilisateur / Modérateur / Admin)
- Rédiger les User Stories
- Définir architecture
- Mettre en place Git + branches
- Mettre en place la plateforme devops

### ◆ Livrables

- Document de conception
- Schéma architecture
- Planning des sprints
- Repo Git structuré
- CI fonctionnelle

## Sprint 1 - MVP Fonctionnel (27 février - 27 mars)

- ◆ Fonctionnel
  - Entité Game
  - Repository JPA
  - Service
  - Controller REST
  - DTO + validation
  - Gestion erreurs globale
  - Swagger
  - Producer Kafka (GAME\_CREATED / UPDATED / DELETED)
  - Topic : game-events
  - Consumer Kafka
  - Table Audit (optionnel mais conseillé)
  - Recherche / filtre simple
- ◆ Technique
  - Tests unitaires Service
  - Tests Controller
  - Couverture > 50%
  - Base PostgreSQL Docker
- ◆ Definition of Done
  - CRUD complet
  - Tests passent en CI
  - Documentation Swagger accessible
- ◆ DevOps
  - docker-compose complet :

- backend
  - frontend
  - postgres
  - kafka
- CI améliorée (rapport coverage)

- ◆ Tests

- Tests intégration API
- Tests Kafka
- Couverture > 60%

- ◆ Livrables

- MVP fonctionnel
- README clair
- Stack lancerable via docker-compose

## Sprint 2 Release (28 mars - 24 avril)

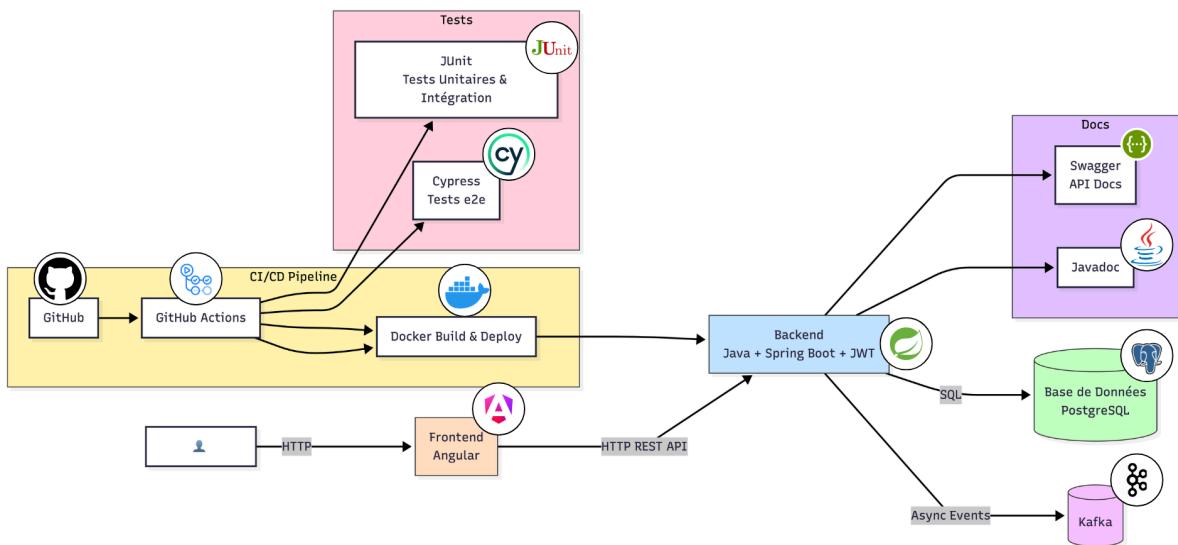
- ◆ Angular
  - Page liste jeux
  - Page détail
  - Formulaire création
  - Filtrage simple
  - Gestion erreurs
- ◆ Sécurité
  - Protection endpoints écriture (clé API ou auth simple)
- ◆ E2E
  - Cypress :
    - Scénario consultation
    - Scénario création
- ◆ Tests
  - Couverture backend > 70%
  - Tests E2E en CI
- ◆ Industrialisation
  - CI complète :
    - Build
    - Tests
    - Coverage
    - Build image Docker
  - Release tag 1.0
  - Artefact repository
- ◆ Production-like

- Profil prod
- Variables d'environnement
- Logs propres
- Actuator /health
- Documentation finale

# Architecture et Stack Technique

## Architecture Logicielle

Catégorie	Techno utilisée
Frontend	Angular
Backend	Java
Backend	Spring Boot
Backend	Kafka
Devops	Docker
Devops	GitHub
Devops	GitHub Action
Tests (Unitaire + Intégration)	Junit
Tests (e2e)	Cypress
Documentation	Javadoc
Documentation	Swagger
DB	PostgreSQL



# Plateforme DevOps

## Workflow de collaboration

Pour ce projet, nous utiliserons la plateforme GitHub pour la gestion du code, la pipeline, les merge requests et la gestion des tickets. En effet, tout cela est regroupé sur une seule page ce qui permet d'avoir toutes les informations essentielles au bon déroulement du projet à un seul endroit.

Chaque ticket engendrera la création d'une branche feature sur laquelle sera fait le développement correspondant. Une merge request devra être faite pour permettre de la fusionner à master et nécessitera l'accord d'un autre membre du groupe. De plus, la branche devra passer la pipeline pour être merge ce qui implique de rajouter des tests pour avoir un coverage suffisant et de ne pas rétrograder sur ceux déjà présent.

Des branches de hotfix pourront également être créées pour les correctifs de bug et devront suivre le même processus pour la fusion.

## Pipeline CD

Pour la pipeline CD, nous utiliserons GitHub Action qui est outil très flexible prenant en charge toutes les technologies que nous utilisons et permettant également de faire le déploiement. Ce dernier se fera en tant qu'image Docker publiée sur Docker Hub.

La pipeline effectuera les actions suivantes:

- Build du frontend et backend
- Lancement des tests unitaires du frontend et backend
- Lancement des tests d'intégration
- Lancement des tests e2e
- Check du coverage
- Création et déploiement de l'image Docker (uniquement si sur main)

Si une de ces étapes échoue, la pipeline est stoppé.