# **Cahier des Charges pour HearthStone Battleground**

## Objectif du Projet

L’objectif du projet est de développer un nouveau mode de jeu "Battleground" pour HearthStone qui est une nouveauté majeure du jeu. Le jeu est développé en utilisant Unity pour le moteur de jeu, avec une base de données pour gérer les Assets, une base de données pour la facturation et une base de données pour la gestion de la progression des joueurs. Les bases de données communiqueront avec des API (micro-service).

## Choix des Technologies

### Moteur de Jeu

* Unity est un moteur de jeu multiplateforme (smartphone, ordinateur, consoles de jeux vidéo et Web) développé par Unity Technologies. Son autre avantage est son interface utilisateur plus facile d'accès et un service plus approprié pour les créations indépendantes.
* Son développement est basé sur le langage C#.

### Base de Données

* Asset Database : Pour gérer les éléments statiques tels que les cartes, les héros, et les serviteurs.
* MySQL : Pour gérer toutes les transactions financières, les achats in-app et les données utilisateur liées à la facturation.
* Apache Cassandra : Pour le stockage et la gestion des données relatives à la progression des joueurs, en raison de sa scalabilité et de sa performance.

### API

C# : Choisi pour le développement des API en raison de sa compatibilité native avec Unity.

## Modèles et Principes de Développement

### Principes SOLID

* Veiller à l'application des principes SOLID pour un code orienté objet propre et maintenable.

### Software Craftsmanship

* S'engager dans l'artisanat logiciel pour promouvoir un haut niveau de qualité et une amélioration continue du code.

### Design Patterns

* Utiliser des design patterns appropriés pour résoudre des problèmes de conception et structurer le code de manière optimale.

## Gestion de Projet

### Outils

* Trello : Utilisé pour la gestion des tâches et la planification du projet.
* GitHub : Pour le contrôle de version, la gestion du code source et le suivi des problèmes.

### Gestion des Branches avec GitHub

* Adopter une stratégie de branches qui inclut des branches principales pour la production, des branches de développement pour les nouvelles fonctionnalités (chaque tâches aura une branche dédiée).
* Implémenter des revues de code et des pull-requests pour maintenir la qualité du code.

## Livrables

### Documentation

Fournir une documentation complète pour le moteur de jeu, les API (swagger), la base de données et les procédures de déploiement.

### Code Source

Le code source complet avec des commentaires détaillés et une documentation intégrée.

### Tests

Des tests unitaires et d'intégration pour assurer la fiabilité et la robustesse du jeu.

## Méthodologie

### Agile

Suivre une approche Agile pour le développement, avec des sprints bi-hebdomadaires et des réunions de rétrospective.

### CI/CD

Mettre en place l'intégration continue et le déploiement continu pour automatiser les tests et la livraison des nouvelles fonctionnalités.

## Sécurité

### Conformité

Assurer la conformité avec les standards de sécurité de l'industrie et les réglementations relatives à la protection des données.

### Cryptage

Utiliser le cryptage pour protéger les transactions et les données sensibles des utilisateurs.