HearthStone Battleground

Journal de bord:

* Analyse des besoins :
  + Réunion pour lister les besoins.
  + Analyse des fonctionnalités du jeu Hearthstone Battlegrounds et identifiez les spécifications requises.
* - Définition de l'architecture :
  + Conception de l'architecture globale du système en décomposant le jeu en plusieurs vues (Modèle C4).
  + Identification des relations entre les composants et détermination des interactions(diagramme de classe).
* - Choix des technologies :
  + Sélection des technologies appropriées en fonction des besoins.
  + choisir les moteurs de jeu, les langages de programmation, les bases de données.
* - Conception détaillée :
  + Création de schémas pour chaque composant du système.
  + Définition des liaisons entre les différents modules.
* Développement :
  + Implémentation des fonctionnalités.
  + Vérifier que le code utilise des pratiques et normes de qualité.
* Tests :
  + Procédures de vérification du bon fonctionnement du jeu.
  + Réalisation des tests unitaires, d'intégration et de système.
* Sécurité :
  + Mise en place des mesures de sécurité pour protéger les données des joueurs et prévenir les tricheries.
* Déploiement :
  + Déploiement du jeu sur les différentes plateformes.
  + Vérification des serveurs prêts à gérer le trafic.
* Maintenance et mises à jour :
  + Résolution d’éventuels problèmes et amélioration du jeu.
  + Planification des mises à jour régulières.
* SAV :
  + Formez le personnel pour répondre aux besoins des utilisateurs et maintenir le jeu en bon état.

Analyses des besoins :

Concept du Jeu :

* Jeu de stratégie tour par tour combinant des éléments de cartes à collectionner.

Plateformes :

* PC et mobiles (iOS, Android).

Fonctionnalités :

* Collection de cartes (héros, unités, sorts).
* Interface simple pour gérer les cartes et les parties.
* Matchmaking équilibré pour des duels.
* Système de progression avec récompenses.
* Tournois et classements.

Graphismes et Design :

* Graphismes attrayants et animations fluides.
* Design unique.

Connectivité en Ligne :

* Multijoueur en ligne.
* Sauvegarde en ligne pour jouer sur différentes plateformes.

Monétisation :

* Achats in-app (éléments cosmétiques, packs de cartes).

Support et Maintenance :

* Support client réactif.
* Mises à jour régulières pour contenu et sécurité.

Sécurité :

* Protection des comptes des joueurs et prévention de la triche.

Performance :

* Jeu fluide, chargements rapides, capacité à gérer de nombreux joueurs.

Accessibilité :

* Accessible à un large public avec options pour différents joueurs.

Définition de l'architecture

1. Interface Utilisateur et API :

* UI Client : Affiche et gère les interactions, animations et navigation du jeu.
* API Serveur : Fait le lien entre l'UI et la logique serveur, gère les requêtes et réponses.

2. Logique de Jeu :

* Moteur de jeu : Mécanismes de base, règles, calcul des points, gestion des tours.
* Gestionnaire de cartes : Gère les cartes, leurs interactions et effets.
* Système de matchmaking : Associe les joueurs selon niveau et préférences.

3. Base de Données :

* Joueurs et Profils : Infos des joueurs, statistiques, progrès.
* Inventaire de cartes : Détails des cartes disponibles, caractéristiques, rareté.
* Historique des parties : Enregistrements, résultats, stratégies, statistiques.

4. Serveur et Réseau :

* Serveur de jeu : Gère logique serveur, authentification, synchronisation des parties.
* Gestion de la connectivité : Maintient une connexion stable, gère les problèmes réseau.

5. Sécurité et Conformité :

* Protection contre la triche : Systèmes de détection et prévention.
* Conformité légale : Respect des lois sur la protection des données, monétisation, âge des joueurs.

6. Monétisation et Boutique :

* Gestion des transactions : Gère achats en jeu, transactions, sécurité des infos de paiement.
* Boutique en jeu : Vente de packs de cartes, cosmétiques..

7. Support Multilingue et Multiplateforme :

* Compatibilité multiplateforme : Fonctionne sur PC, mobiles, tablettes.
* Localisation : Traduction et adaptation du contenu pour différentes langues et régions.

8. Analytics et Reporting :

* Suivi des données : Collecte données sur engagement, préférences, performances des joueurs.
* Outils de reporting : Analyse des données pour orienter les mises à jour.

Choix des technologies

### 1. Moteur de Jeu

* Unity : Excellent pour les jeux multiplateformes. Offre un bon support pour les graphismes 2D et 3D.

### 2. Langages de Programmation

* C# (pour Unity) : Langage principal pour le développement avec Unity.

### 3. Développement Côté Serveur

* Symfony : Pour un développement rapide et efficace, adapté pour de nombreux utilisateurs simultanés.

### 4. Base de Données

* AirTable: Base de données, pour stocker des données.

### 5. Gestion de la Connectivité et Réseau

* WebSockets : Pour une communication en temps réel entre le client et le serveur.

### 6. Sécurité

* JWT : Pour la gestion sécurisée de l'authentification et des sessions utilisateurs.
* SSL : Pour sécuriser les communications entre le client et le serveur.

### 7. Infrastructure et Déploiement

* Docker : Pour la conteneurisation et l'orchestration, facilitant le déploiement et la gestion de l'infrastructure.
* Services cloud AWS : Pour héberger le serveur et d’autres service.

### 8. Outils Analytiques

* Firebase : Pour suivre les interactions des utilisateurs et analyser les données de jeu.

9. Abonnement :

* Système d'abonnement : Achat d'abonnement aux joueurs, incluant des avantages exclusifs ou un accès à des contenus premium.
* Gestion des abonnements : Gérer les abonnements des utilisateurs, avec une interface facile pour l'inscription, la mise à jour et l'annulation.

10. Paiement :

* Plateformes de paiement sécurisées : Utilisation de Stripe ou PayPal pour des transactions sécurisées.
* Gestion des paiements : Traiter les paiements, les achats in-app, les abonnements et les microtransactions.

11. Utilisateur :

* Gestion des comptes : Système pour la création, la gestion et la sécurisation des comptes utilisateurs.
* Authentification : Utilisation de JWT pour une authentification sécurisée.
* Interface utilisateur : Créer une interface intuitive pour la gestion du profil, des paramètres, et du support utilisateur.

12. Catalogue :

* Gestion du catalogue de jeu : Un système pour gérer, ajouter ou mettre à jour le catalogue de cartes et d'autres éléments de jeu.
* Mise à jour et équilibre du jeu : Assurer un équilibre constant et des mises à jour régulières du catalogue pour maintenir l'intérêt des joueurs.

Conception détaillée

### 1. Interface Utilisateur (UI) / API

* Schéma UI : Illustration des écrans de jeu, menus, boutons, et autres éléments interactifs.
* Schéma API : Utilisation des méthodes (GET, POST, PUT), pour récupérer au format JSON.
* Interface UI-API : Définition des appels API depuis l'UI pour différentes actions (connexion, gestion de deck).

### 2. Moteur de Jeu

* Diagramme de classe montrant les différentes relations et fonctionnalités du jeu.
* Diagramme de Modèle C4 montrant une vue d’ensemble de la composition du site.

### 3. Base de Données

* Schéma de la Base de Données : Modèle Entité-Relation basée sur le diagramme de classe.

### 4. Serveur et Réseau

* Interface Serveur - Client : Comment les clients communiquent avec le serveur (authentification, échange de données de jeu).

### 5. Sécurité et Conformité

* Interface Sécurité : Intégration des fonctionnalités de sécurité.

### 6. Monétisation et Boutique

* Interface Boutique : Comment la boutique interagit avec le serveur et la base de données pour les transactions.

### 7. Support Multilingue et Multiplateforme

* Compatibilité Multilinguistique : Gestion des différentes langues et adaptations de contenu.
* Compatibilité Multiplateforme : Adaptation du jeu sur différentes plateformes.

### 8. Analytics et Reporting

* Serveur/Jeu : Comment les données sont recueillies et utilisées pour l'amélioration du jeu.

Développement

### 1. Planification et Répartition des Tâches

* Définissez les objectifs clairs : Assurez-vous que chaque membre de l'équipe comprend les objectifs du projet et ses propres responsabilités.
* Répartissez les tâches : Attribuez des tâches spécifiques en fonction des compétences de chaque membre.
* Planification des sprints : utilisations de la méthode agile pour planifier des cycles de développement courts et concentrés (sprints).

### 2. Conventions et Normes de Codage

* Conventions de codage : Assurez-vous que tout le monde suit les mêmes standards pour faciliter la lisibilité et la maintenance du code.
* Revue de code : Mettez en place des revues de code régulières pour garantir la qualité et partager les connaissances.

### 3. Utilisation d'Outils et de Technologies

* Choisissez les bons outils : utilisation de GIT pour le versioning.
* Utilisation de bibliothèques et de frameworks : utilisation d’outils déjà testés.

### 4. Assurance Qualité et Tests

* Tests unitaires et d'intégration : Encouragez le développement piloté par les tests (Test-Driven Development - TDD) pour minimiser les bugs.

### 5. Documentation et Commentaires

* Documentez le code : Assurez-vous que le code est bien commenté et que la documentation du projet est à jour.
* Créez une documentation technique pour aider les nouveaux membres de l'équipe.

### 6. Communication et Collaboration

* Réunions régulières : Organisez des stand-ups quotidiens pour suivre les progrès et identifier les obstacles.
* Encouragez la communication ouverte : Créez un environnement où les membres de l'équipe se sentent libres de partager des idées et des préoccupations.

### 7. Gestion des Dépendances et de la Configuration

* Gérez les dépendances de manière efficace : Assurez-vous que toutes les dépendances externes sont bien gérées et à jour.
* Versioning et gestion de la configuration : Utilisez des outils pour gérer les différentes versions du jeu et configurations.

### 8. Réponse aux Retours et Itération

* Tests utilisateurs : Organisez des sessions de test avec de vrais utilisateurs et intégrez leurs retours dans le cycle de développement.
* Itérations rapides : Soyez prêt à faire des ajustements rapides en fonction des retours et des résultats des tests.

Phase de test

### 1. Tests Unitaires

* Objectif : Tester les plus petites unités de code (fonctions, méthodes) indépendamment.
* Mise en œuvre : Utilisez des frameworks de test comme Jest pour JavaScript.
* Exemples de Tests :
  + Fonctions de calcul de score.
  + Mécaniques individuelles de cartes.
  + Algorithmes de matchmaking.

### 2. Tests d'Intégration

* Objectif : Tester les interactions entre les différents modules ou composants du système.
* Exemples de Tests :
  + Communication entre le client et le serveur.
  + Transactions de la boutique en jeu.
  + Chargement et sauvegarde des profils des joueurs.

### 3. Tests de Système

* Objectif : Vérifier le comportement global du jeu et s'assurer qu'il répond aux exigences spécifiées.
* Mise en œuvre : Testez le jeu dans un environnement similaire à la production.
* Exemples de Tests :
  + Performance et stabilité du jeu sous charge.
  + Compatibilité sur différentes plateformes et appareils.
  + Tests de l'interface utilisateur pour s'assurer qu'elle est intuitive et sans bug.

### 4. Tests de Charge et de Performance

* Objectif : Évaluer comment le système se comporte sous des charges élevées.
* Outils : Utilisez de JMeter pour simuler un grand nombre de joueurs.
* Scénarios : Simuler des pics d'utilisation, comme lors de tournois en ligne ou de lancements de nouvelles fonctionnalités.

### 5. Tests d'Acceptation Utilisateur (UAT)

* Objectif : S'assurer que le jeu répond aux attentes et besoins des joueurs finaux.
* Mise en œuvre : Invitez un groupe de joueurs cibles à tester le jeu et collectez leurs retours.
* Focus : Ergonomie, expérience de jeu, équilibre des mécaniques de jeu, et attrait global.

### 7. Automatisation des Tests

* Objectif : Automatiser les tests répétitifs pour gagner en efficacité.
* Outils : Selenium pour les tests d'interface, Postman pour les tests d'API, et des outils spécifiques au moteur de jeu.

### 8. Suivi

* Outils de Suivi : Utilisez des outils comme JIRA ou Trello pour suivre les bugs et les problèmes.

Optimisation

### 1. Analyse des Performances

* Utilisez des Outils de Profilage : Utilisation d’Unity Profiler pour identifier les zones à forte consommation de ressources.
* Surveillez les Métriques Clés : Temps de chargement, fréquence d'images par seconde (FPS), utilisation de la mémoire et du CPU.

### 2. Optimisation du Rendu Graphique

* Optimisation des Assets : Réduisez la taille et la complexité des assets graphiques sans compromettre la qualité visuelle.

### 3. Optimisation de la Logique de Jeu

* Réduction des Calculs Inutiles : Évitez les calculs redondants dans les boucles de jeu et les scripts.
* Utilisation efficace des Scripts : Optimisez les scripts pour minimiser l'utilisation de la CPU.
* Gestion des Ressources : Assurez-vous que les ressources sont chargées et déchargées de manière efficace pour éviter les fuites de mémoire.

### 4. Optimisation de la Base de Données et du Réseau

* Requêtes efficaces : Optimisez les requêtes de base de données pour réduire la latence.
* Compression des Données : Utilisez la compression pour les données échangées sur le réseau pour réduire les délais.

### 5. Gestion du Multithreading

* Multithreading : Utilisez le multithreading pour des tâches lourdes afin de ne pas bloquer le thread principal.
* Gestion efficace des Threads : Assurez-vous que l'utilisation des threads est bien gérée pour éviter les problèmes de performance.

### 6. Optimisation pour Différentes Plateformes

* Tests sur Divers Appareils : Testez du jeu sur différents appareils et configurations pour identifier les problèmes spécifiques à certaines plateformes.
* Paramètres graphiques Ajustables : Offrez des options pour ajuster les paramètres graphiques en fonction des capacités de l'appareil.

### 7. Mises à Jour

* Mises à Jour Basées sur les Retours : Faire des mises à jour pour améliorer les performances basées sur les retours des utilisateurs.

sécurité

### 1. Sécurité des Données et de la Confidentialité

* Cryptage des Données : Utilisation du cryptage SSL pour sécuriser les communications entre le client et le serveur.
* Stockage sécurisé des Mots de Passe : Stockage des mots de passe en utilisant des hash sécurisés .
* Protection des Données Sensibles : Vérifiez que les informations sensibles des joueurs sont stockées de manière sécurisée, conformément aux réglementations.

### 2. Authentification et Autorisation

* Authentification Forte : Implémentez l'authentification à deux facteurs (2FA) pour les comptes de joueurs.
* Contrôles d'Accès : Assurez-vous que les joueurs n'ont accès qu'aux fonctionnalités et données qui leur sont nécessaires.

### 3. Lutte contre la Triche

* Détection de triche : Systèmes de détection de triche pour identifier et bloquer les comportements suspects ou les modifications du client de jeu.
* Mises à Jour Régulières : Mettez à jour régulièrement les mécanismes de détection pour contrer les nouvelles méthodes de triche.
* Sanctions : Établissez et appliquez des politiques claires concernant les tricheurs.

### 4. Sécurité du Réseau

* Protection contre les attaques DDoS : utilisez des services de mitigation des DDoS pour protéger votre infrastructure réseau.

### 6. Mises à Jour et Patches de Sécurité

* Mises à Jour Régulières : Fournissez des mises à jour régulières pour corriger les failles de sécurité.
* Réponse rapide aux vulnérabilités : Soyez prêt à déployer rapidement des correctifs en cas de découverte de vulnérabilités critiques.

### 7. Formation et Sensibilisation

* Éduquez l'Équipe de Développement : Assurez-vous que votre équipe est informée des meilleures pratiques en matière de sécurité.
* Sensibilisation des Joueurs : Informer les joueurs sur la manière de sécuriser leurs comptes et les signes de triche.

### 8. Conformité et Réglementations

* Respectez les Réglementations : Assurez la conformité avec les réglementations locales et internationales en matière de protection des données et de sécurité.

Déploiement

1. Préparation de l'Environnement de Déploiement :

* Configuration des Serveurs : Préparez les serveurs avec les logiciels et dépendances nécessaires.

2. Déploiement Multiplateforme :

* Compatibilité des Plateformes : Testez le jeu sur différentes plateformes (PC, mobiles) pour assurer sa compatibilité.
* Optimisation Spécifique à la Plateforme : Ajustez pour chaque plateforme.

3. Automatisation du Processus de Déploiement :

* CI/CD : Mettez en place l'intégration et le déploiement continus pour automatiser le processus.

4. Gestion de la Configuration et Versioning :

* Gestion de la Configuration : Utilisez des outils comme Ansible pour la gestion des serveurs.
* Versioning : Suivez les versions pour faciliter le suivi et le rollback.

5. Stratégie de Déploiement :

* Déploiement de secours : Ayez un plan B en cas de problème avec le déploiement initial.

6. Surveillance Post-Déploiement :

* Support Technique : Ayez une équipe prête pour les problèmes techniques.

7. Communication avec les Joueurs :

* Annonce du Déploiement : Informer les joueurs des détails du déploiement.
* Retours des Utilisateurs : Soyez attentif aux retours pour des ajustements éventuels.

Maintenance et mises à jour

1. Planification de la Maintenance :

* Maintenance Régulière : Programmez des maintenances pour les mises à jour et optimisations. Informez les joueurs à l'avance.
* Équipe de Support : Ayez une équipe technique dédiée pour la maintenance et les urgences.

2. Gestion des Mises à Jour :

* Cycle de Release : Calendrier régulier pour les mises à jour majeures et mineures.
* Mise à jour du Contenu : Ajoutez régulièrement du nouveau contenu (nouvelles cartes, modes de jeu, événements).
* Mises à Jour de Performance et Sécurité : Améliorez la performance et la sécurité.

3. Correction des Bugs :

* Suivi des Bugs : Utilisation de JIRA pour suivre et gérer les bugs.
* Déploiement rapide des Correctifs : Réagissez vite pour corriger les bugs critiques.

4. Mises à Jour Basées sur les Retours des Joueurs :

* Collecte de Retours : Recueillez les avis des joueurs via les forums, les sondages, les réseaux sociaux.
* Adaptation en fonction des Retours : Utilisez ces avis pour améliorer le jeu.

5. Tests Avant Déploiement :

* Tests compréhensifs : Testez soigneusement chaque mise à jour avant déploiement.
* Bêta Testing : Testez les mises à jour majeures avec un groupe de joueurs bêta.

6. Documentation et Formation :

* Mise à jour de la Documentation : Gardez la documentation technique et les guides pour les joueurs à jour.
* Formation des équipes : Formez les équipes de support et de développement sur les nouveautés.

7. Compatibilité et Support Multiplateforme :

* Assurez la Compatibilité : Testez les mises à jour sur toutes les plateformes supportées.
* Support à Long Terme : Planifiez un support continu pour différentes versions et plateformes.

8. Communication Transparente :

* Informez les Joueurs : Communiquez clairement sur les mises à jour, correctifs, et maintenances.
* Engagement de la Communauté : Utilisez les mises à jour pour engager la communauté et maintenir l'intérêt.

Formation

1. Formation Technique :

* Gestion des Outils : Formez sur les outils de surveillance, gestion des incidents et maintenance.
* Procédures de dépannage : Apprenez les méthodes de diagnostic et résolution de problèmes courants.

2. Formation sur la Base de Données et le Réseau :

* Gestion de la Base de Données : Formez sur l'accès et la modification sécurisée des données.
* Compréhension du réseau : Connaissances de base sur la gestion du réseau, surveillance du trafic, résolution de problèmes de connectivité.

3. Formation en Sécurité Informatique :

* Sensibilisation à la Sécurité : Compréhension de l'importance de la sécurité informatique et meilleures pratiques.
* Gestion des Incidents de Sécurité : Procédures en cas de faille de sécurité ou attaque informatique.

4. Support Client et Communication :

* Compétences en Service Client : Communication efficace avec les utilisateurs, gestion des demandes de support.
* Gestion des Réclamations : Techniques pour gérer les plaintes et problèmes des utilisateurs.

5. Formation sur les Mises à Jour et les Changements :

* Gestion des Mises à Jour : Connaissance des dernières mises à jour du jeu et de leurs impacts.
* Changements de Politiques ou Fonctionnalités : Formation sur les modifications des politiques de jeu ou nouvelles fonctionnalités.

6. Formation Continue :

* Sessions Régulières : Sessions de formation pour maintenir les compétences à jour.
* Ressources d'Auto-Formation : Manuels en ligne, tutoriels vidéo, forums de discussion.

7. Exercices Pratiques et Simulations :

* Scénarios de simulation : Exercices pratiques dans un environnement contrôlé.
* Retours et Évaluations : Feedback constructif après les exercices de formation.