Cahier des charges

Delaval-Dupuis Alexandre Groupe A3G1.1

Mamtchourouk Louis

Perrier Gaëtan

Chakroune Younès

I. Introduction
- A. Présentation du projet
- B. Objectif du cahier des charges

II. Contexte du projet
- A. Description du contexte
- B. Analyse des besoins

III. Objectif
- A. Objectifs généraux
- B. Objectifs spécifiques

IV. Périmètre
- A. Plateformes ciblées
- B. Technologies utilisées

V. Fonctionnalités requises
- A. Liste exhaustive des fonctionnalités
- B. Priorisation des fonctionnalités
- C. Interactions entre les fonctionnalités

VI. Contraintes et limitations
- A. Contraintes de temps
- B. Contraintes techniques

VII. Tests et validation
- A. Stratégie de test
- B. Critères de réussite des tests
- C. Procédure de validation du projet

X. Glossaire
- A. Définition des termes techniques utilisés dans le cahier des charges

I. Introduction

A. Présentation du projet

Le but du projet est de recréer le jeu 2048 avec une technologie au choix, en respectant les contraintes imposé, tels que l'utilisation de GitHub et l'écriture d'un code conforme à l'IEEE.

B. Objectif du cahier des charges

Dans le but d'un devoir de niveau 3eme année en BUT informatique, nous allons programmer un jeu vidéo fonctionnel avec des objectifs et des fonctionnalités claires et précises. Le but du cahier des charges est dans un premier temps de lister toutes les fonctionnalités, la technologie utilisée, ainsi que les contraintes auxquelles on va devoir faire face. Dans un second temps, ils nous faut réaliser des tests sur le programme final tels que, pour en citez quelques un des tests unitaire, tests fonctionnels etc...

II. Contexte du projet

A. Description du contexte

Le projet est un devoir donné dans la ressource R5.8, où le but est de concevoir et développer un jeux vidéo fonctionnel, dans une autonomie presque total choisir, en respectant le cahier des charges.

Dans notre cas, le jeu choisi est le 2048. L'objectif est de fusionner des blocs ayant une valeur identique pour atteindre le bloc 2048. De plus, ce projet comporte plusieurs défis, tels que le déplacement des blocs, la gestion de la fusion des blocs... Enfin, le choix de la technologie dépends des goûts et des habitudes des étudiants présent dans le groupe à condition d'utiliser GitHub et d'avoir un code structuré et commenté.

B. Analyse des besoins

Voici les principaux besoins identifiés :

1. **Fonctionnalités du jeu :**
    - Implémentation des règles de base du jeu 2048 : fusion des blocs (couleurs + nombre), gestion des mouvements, et conditions de victoire ou de défaite.
    - Interface graphique simple et intuitive permettant aux utilisateurs de jouer facilement.

- Système de score pour afficher les points cumulés lors de la partie.
- Possibilité de redémarrer une partie ou de quitter le jeu à tout moment.

2. **Technologies :**
   - Choix d'un langage de programmation adapté au développement du jeu (JavaScript).
   - Utilisation d'une bibliothèque ou d'un framework facilitant la création de l'interface graphique (avec JavaScript).
   - Intégration avec **GitHub** pour assurer une gestion du projet en collaboration avec l'équipe, en utilisant le versionnage du code.
3. **Contraintes techniques :**
   - Le code doit être propre, documenté, et conforme aux bonnes pratiques de développement (IEEE).
   - Tests unitaires et fonctionnels à réaliser pour valider la stabilité et la fiabilité du jeu.
     - Ces tests seront réalisés naturellement de par notre façon de coder (TDD)
4. **Contraintes de temps :**
   - Le projet doit être réalisé dans un délai imparti, 10h de TD

III. Objectif

A. Objectifs généraux

1. Créer un jeu 2048 fonctionnel et agréable à jouer
2. Mettre en pratique les connaissances acquises et les normes de codage
3. Travailler en équipe

B. Objectifs spécifiques

1. L'interface doit être facile à comprendre et à utiliser, même pour les joueurs débutants.
2. Les règles du 2048 doivent être respectées à la lettre, et les calculs doivent être précis.

3. Le jeu doit pouvoir gérer les situations de victoire, de défaite et les différentes phases de jeu.
4. Le jeu doit être fluide et réactif, même sur des machines moins puissantes.
5. Permettre aux joueurs de comparer leurs scores et de se classer dans un tableau des meilleurs scores.

IV. Périmètre

A. Plateformes ciblées

1. Web
    a. Jeu accessible depuis un navigateur web (desktop et mobile).

B. Technologies utilisées

1. Langage de programmation principal
    a. Javascript
2. Framework
    a. Nuxt.js

V. Fonctionnalités requises

A. Liste exhaustive des fonctionnalités

1. Grille de jeu
    a. Création d'une grille de jeu de taille personnalisable (4×4, 5×5, etc.)
    b. Affichage des tuiles avec leurs valeurs
    c. Animation des mouvements des tuiles
2. Mécaniques de jeu
    a. Génération aléatoire de nouvelles tuiles (2 ou 4) à chaque mouvement
    b. Déplacement des tuiles dans les quatre directions (haut, bas, gauche, droite)
    c. Fusion des tuiles de même valeur

    d. Détection de victoire (atteindre une tuile de valeur 2048)
    e. Détection de défaite (plus aucun mouvement possible)
3. Interface utilisateur
    a. Affichage du score actuel
    b. Affichage du meilleur score
    c. Bouton pour rejouer
    d. Options de personnalisation (Dark Mode)
    e. Menu principal avec bouton "Nouvelle partie” et choix des paramètres du jeu.
4. Autres fonctionnalités
    a. Mode infini (continuer à jouer après avoir atteint 2048)
    b. Système de classement

B. Priorisation des fonctionnalités

1. Fonctionnalités essentielles
    a. Grille de jeu de base (4×4)
    b. Mécaniques de jeu de base (génération, déplacement, fusion, victoire, défaite)
    c. Interface utilisateur minimale (score, bouton rejouer)
2. Fonctionnalités améliorant l'expérience utilisateur
    a. Options de personnalisation (thèmes)
    b. Statistiques de jeu (meilleur score)
3. Fonctionnalités avancées
    a. Modes de jeu supplémentaires (infini)
    b. Système de classement

C. Interactions entre les fonctionnalités

1. Grille de jeu et mécaniques de jeu
    a. La grille est le support des mécaniques de jeu.

    b. Les mouvements des tuiles sont effectués sur la grille et les fusions se produisent entre les tuiles adjacentes.

2. Mécaniques de jeu et interface utilisateur
    a. L'interface affiche l'état de la grille, le score et les autres informations pertinentes.
    b. Les actions de l'utilisateur (glisser, cliquer) déclenchent les mécaniques de jeu.
3. Options de personnalisation et interface utilisateur:
    a. Les options de personnalisation permettent à l'utilisateur de modifier l'apparence du jeu et sont accessibles via l'interface utilisateur.

VI. Contraintes et limitations

A. Contraintes de temps

1. Délai de livraison
2. Temps selon les cours

B. Contraintes techniques

1. Complexité du jeu
    a. Certaines fonctionnalités peuvent être plus complexes à implémenter que d'autres.
2. Optimisation des performances

VII. Tests et validation

A. Stratégie de test

1. Tests unitaires
    a. Vérifier le fonctionnement de chaque fonctionnalité isolément (génération de tuiles, déplacements, fusions, etc.).
2. Tests d'intégration

    a. S'assurer que les différentes parties du jeu interagissent correctement (interface utilisateur, moteur de jeu, etc.).
    b. Simuler des scénarios de jeu pour vérifier le tout.
3. Tests de performance
    a. Mesurer les temps de réponse du jeu, notamment lors de mouvements rapides ou de grandes grilles.

B. Critères de réussite des tests

1. Fonctionnalités
    a. Toutes les fonctionnalités listées précédemment doivent être implémentées et fonctionner correctement.
2. Performance
    a. Le jeu doit être fluide et réactif, sans ralentissements ni bugs.
3. Compatibilité
    a. Le jeu doit fonctionner correctement sur les différentes plateformes ciblées (web, mobile).

C. Procédure de validation du projet

1. Tests unitaires et d'intégration
    a. Exécuter les tests régulièrement pour détecter les régressions.
2. Tests de performance
    a. Utiliser des outils pour identifier les parties du code les plus lentes.
3. Validation finale
    a. S'assurer que tous les critères de réussite sont remplis.

X. Glossaire

A. Définition des termes techniques utilisés dans le cahier des charges

- **GitHub :** Plateforme en ligne permettant de gérer des projets informatiques en collaboration. Elle permet de versionner le code, de suivre les modifications et de travailler en équipe.
- **IEEE :** Institut des Ingénieurs Électriciens et Électroniciens.
- **Framework :** Ensemble d'outils et de bibliothèques qui facilitent le développement d'applications.
- **Nuxt.js :** Framework JavaScript
- **Régression :** Apparition d'un bug dans une partie du code qui fonctionnait précédemment.
- **Tests unitaires :** Tests qui vérifient le fonctionnement correct de petites portions de code (fonctions, méthodes).
- **Tests d'intégration :** Tests qui vérifient que les différentes parties d'une application interagissent correctement entre elles.
- **Tests de performance :** Tests qui mesurent les performances d'une application (temps de réponse, consommation de mémoire, etc.).