Cahier des charges

I. Introduction

A. Présentation du projet

Notre jeu, nommé « Débarquement 1945 » sera un « scrolling shooter » ou « shoot'em up à défilement » soit un jeu où le joueur doit détruire un grand nombre d'ennemis à l'aide de projectiles tout au long du niveau en esquivant les projectiles adverses. Comme son nom l'indique, il se placera dans la période de temps de la seconde guerre mondiale.

B. Objectifs du cahier des charges

Ce cahier des charges a pour but de résumer rapidement le contexte du projet, les objectifs du projet, les fonctionnalités requises, nos contraintes et limitations, ainsi que notre stratégie pour les plans de tests.

II. Contexte du projet

A. Description du contexte

Ce projet a pour but de développer en groupe un jeu vidéo, dans le cadre de notre cours de « Qualité de développement », afin de nous familiariser avec les différentes étapes de la conception et du développement d'un projet .

B. Analyse des besoins

Cette exercice nous demande de développer du code en respectant les normes de codage de notre logiciel de programmation, de rédiger un cahier des charges lisible et succinct, un plan de test et une documentation technique complète, et enfin effectuer des tests unitaires avec un framework de test, pousser le code sur GitHub et effectuer une revue de code sur les pull-request.

III. Objectifs du projet

A. Objectifs généraux et spécifiques

Nos objectifs principaux sur ce projet sont de développer en groupe un jeu vidéo de type « shoot'em up ». Nous tenterons dans ce projet de développer le code en respectant les normes de codage de notre logiciel de programmation, de rédiger un cahier des charges lisible et succinct, un plan de test et une documentation technique complète, et enfin effectuer des tests unitaires avec un framework de test, pousser le code sur GitHub et effectuer une revue de code sur les pull-request.

L'application finale contiendra donc :

Un menu principal avec un bouton "jouer" et "options", un personnage principal qui suit la souris du joueur et tire des projectiles en face de lui, des ennemis qui descendront du haut de l'écran vers le bas, des récompenses périodiques pour avoir vaincu ces ennemis, un système de score qui permettra de créer un boss après avoir vaincu assez d'ennemis, un système de point de vie pour atteindre la fin

du jeu, et pour finir un écran pour afficher le score final et l'enregistrer avant d'être renvoyer au menu principal.

IV. Fonctionnalités requises

A. Liste exhaustive des fonctionnalités

Dans le produit final, nous aimerions que le joueur puisse avoir accès à ces fonctionnalités :

1) Un menu principal contenant :
- Un bouton d'option permettant d'accéder aux paramètres modifiable du jeu
- Un bouton « Jouer » permettant d'accéder au jeu.
2) Le jeu en lui-même contenant :
- Notre personnage principal, le soldat, contrôlé par la souris du joueur, et tirant des projectiles devant lui.
- Des ennemis, descendant du haut de l'écran, et tirant leurs propres projectiles.
- Des bonus, apparaissant rarement lorsque les ennemis sont détruits, après avoir été touchés par assez de projectiles du personnage principal.
3) Une interface de jeu contenant :
- Le score, les points de vie, et les bonus actuels affichés à l'écran
- Un moyen de mettre en pause le jeu (par un bouton, ou bien une touche du clavier)
- Un écran de fin de jeu, lorsque le jeu sera terminé, ou lorsque le joueur est éliminé. Celui-ci permettra la saisie d'un pseudonyme afin d'enregistrer le score du joueur.
- Un fond d'écran avec une image bougeant progressivement au cours du jeu.

B. Priorisation des fonctionnalités

Les fonctions essentielles peuvent être résumées en 7 points :

1) Le jeu doit avoir un menu principal avec deux boutons.
2) Le personnage doit suivre la souris du joueur principal.
3) Le personnage principal doit toujours tirer des projectiles devant lui.
4) Les ennemis doivent bouger du haut de l'écran vers le bas de celui-ci
5) Après 3 contacts avec des ennemis ou leurs projectiles, le jeu doit s'arrêter.
6) Atteindre un certain score doit provoquer un événement de victoire
7) Les ennemis doivent occasionnellement faire tomber des bonus lorsqu'ils sont détruits.

Les fonctions secondaires contiennent :

1) Un menu d'option permettant de changer la taille de la fenêtre, et rappeler les contrôles du jeu
2) Un système de boss, atteint lorsque le score atteint un certain seuil. Ces boss arrêteraient le défilement de l'écran, aurait une barre de point de vie apparaissant sur l'écran, et aurait des projectiles comme les autres ennemis.
3) L'interface de jeu doit être en mesure de présenter correctement les statistiques du joueur : Son score, ses points de vies, etc…

Des fonctions bonus qui pourraient être ajoutés seraient :

1) Ajouter une musique de fond sur le menu principal, dans le jeu, et le menu d'option pourrait réduire ou augmenter son volume.

C. Interactions entre les fonctionnalités

Les fonctionnalités suivantes interagissent ensemble :

1) Les projectiles créés par le personnage principal interagissent avec les ennemis pour faire descendre leurs points de vie interne. Lorsqu'un projectile touche un ennemi et retire son dernier point de vie interne, il disparaît de l'écran. Rarement, cela générera une entité de bonus.
2) Les projectiles créés par les ennemis interagissent avec le personnage principal lorsqu'ils rentrent en contact avec lui, réduisant ses points de vies. Similairement aux ennemis, si un projectile touche le personnage principal et lui retire son dernier point de vie, cela arrêtera la partie, et relancera la partie si jamais le joueur possède encore des vies.
3) Les projectiles quittant l'écran seront détruits pour éviter d'avoir trop d'entités en même temps en jeu.

V. Contraintes et limitations

A. Contraintes de temps

Nous avons 6 séances de deux heures et du temps libre en dehors des cours, pour réaliser ce projet, ce qui nous donne jusque mi-novembre pour terminer le projet. Nous tenterons d'avoir un livrable acceptable à la fin de la 5ème séance, afin d'avoir une dernière séance où nous pourrons rajouter du bonus, ou perfectionner les détails.

B. Contraintes techniques

Nous utiliserons le moteur de jeu Godot afin de réaliser notre projet : Nous utiliserons donc son langage approprié « GDScript », un langage conçu spécifiquement pour Godot. Bien évidemment, nous suivrons l'évolution du projet grâce à l'outil de gestion GitHub, son tableau Kanban, et ses fonctionnalités de « pull requests » et « Issues », afin de documenter et valider les fonctionnalités ajouter dans notre projet.

VI. Tests et validation

A. Stratégie de test

Pour s'assurer de la qualité de notre projet nous réaliserons plusieurs types de tests :

Les tests unitaires vérifieront le bon fonctionnement des fonctions individuelles du code (ex. gestion des collisions, calcul du score, déplacement du joueur). Les tests fonctionnels quant à eux, permettront de s'assurer que les différentes fonctionnalités du jeu (déplacement, affichage, interaction avec les éléments) fonctionnent correctement ensemble. Pour finir, les tests manuels

seront réalisés par les membres de l'équipe pour essayer le jeu et détecter d'éventuels bugs, problèmes d'affichage ou comportements inattendus.

B. Critères de réussite des tests

Un test sera considéré comme réussi lorsque :

- La fonctionnalité testée fonctionnera conformément aux spécifications du cahier des charges.
- Aucune erreur ou bug bloquant n'apparaîtra lors de l'exécution du jeu.
- Le comportement observé sera conforme aux règles du jeu (ex. défaite lorsqu'un ennemi touche le joueur, victoire quand les conditions sont remplies).
- Les performances du jeu resteront stables (pas de ralentissement ou plantage).

C. Procédure de validation du projet

La validation du projet se déroulera en plusieurs étapes :

Tout au long de notre projet, nous ferons des revues de code : un ou plusieurs membres du groupe reliront le code de ses coéquipiers lors des pull-requests sur GitHub pour détecter les erreurs ou incohérences. Nous ferons une exécution complète des tests unitaires et fonctionnels pour vérifier la stabilité du jeu, et pour finir, nous aurons une session de test finale, lors de la dernière séance où tous les membres testeront le jeu dans son intégralité pour valider la jouabilité, la fluidité et l'absence de bugs majeurs.

Le projet sera considéré comme validé lorsque toutes les fonctionnalités principales seront opérationnelles, les tests tous réussis, et que le jeu respectera les critères fixés dans le cahier des charges.

VII. Glossaire

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

Voici quelques termes techniques pouvant nécessiter une description :

1) Normes de codage

Les règles et bonnes pratiques à respecter lors de l'écriture du code pour le rendre lisible, cohérent et facile à maintenir.

2) Framework

Un ensemble d'outils et de bibliothèques facilitant le développement d'un logiciel ou d'un jeu, en fournissant une structure de base.

3) GitHub

Une plateforme en ligne permettant d'héberger du code, de collaborer en équipe et de suivre l'évolution d'un projet à l'aide du système de versionnement Git.

4) Pull request

La fonctionnalité de GitHub qui permet de proposer des modifications du code afin qu'elles soient revues et validées par d'autres membres avant d'être intégrées au projet principal.

5) Revue de code

Le processus de vérification du code par un ou plusieurs membres de l'équipe pour détecter d'éventuelles erreurs, améliorer la qualité et assurer la cohérence du projet.

6) Tests unitaires

Les tests automatisés qui vérifient le bon fonctionnement d'une petite partie précise du code (par exemple une fonction ou une méthode).

7) Tests fonctionnels

Les tests qui valident le comportement global du programme pour s'assurer que les différentes fonctionnalités fonctionnent correctement ensemble.

8) GDScript

Le langage de programmation utilisé par le moteur de jeu Godot, conçu spécialement pour développer des jeux de manière simple et rapide.

9) Moteur de jeu (Game Engine)

Un logiciel servant de base au développement de jeux vidéo. Il gère des éléments essentiels comme les graphismes, les sons, la physique ou les entrées du joueur.