Projet Vanguard Fighter's

Projet Shoot Me Up

Table des matières

1 Analyse préliminaire

1.1 Introduction

Nous avons projet de créer un jeu avec un code, une base de données et une interface utilisateur via Figma. Pour ce projet j'ai décidé de créer un jeu dans l'univers des Plateformers (exemple : Smash Bros en 2D) sous le nom de « Vanguard Fighter's » celui-ci à pour finalité au maximum de me faire apprendre comment écrire du code durant toute ma formation, c'est pour cela que j'ai choisi ce projet c'est surtout pour pouvoir le continuer même après que le projet est terminé. Différent de certain de mes camarades j'ai choisi de prendre un autre moteur de jeu qui s'appelle Monogame.
Pourquoi Monogame cela est simple pour un jeu qui je veux, dure sur une longue durée j'ai préféré l'utiliser pour une plus grande efficacité et une plus grande facilité.

1.2 Objectifs

Le projet vise à développer un jeu structuré autour de niveaux interactifs, chacun intégrant des éléments variés pour offrir une expérience de jeu dynamique et immersive. Les objectifs de conception comprennent :

1.3 Conception des niveaux

Chaque niveau sera distinctement identifié par un numéro (ex. : Niveau 1, Niveau 2, etc.) et comportera des éléments de gameplay essentiels :

- **Joueur** :
    - **Déplacements** : Intégration d'une mécanique de déplacement fluide et réactive.
    - **Gestion des vies** : Définition du nombre de vies disponibles pour le joueur.
    - **Capacités de tir** : Gestion avancée des tirs, incluant la direction, le mode rafale, les temps de recharge (cooldown), le décompte des munitions et les mécanismes de recharge.
    - **Apparence visuelle** : Utilisation d'un sprite pour représenter le joueur.
- **Ennemis** :
    - **Points de vie** : Définition de la résistance des ennemis par type.
    - **Minutage d'apparition** : Apparition des ennemis selon un minutage spécifique pour chaque niveau.
    - **Capacités de tir** : Détermination de l'aptitude de chaque type d'ennemi à tirer.
    - **Apparence visuelle** : Attribution d'un sprite propre à chaque type d'ennemi.

- **Obstacles** :
    - **Caractéristiques physiques** : Définition de la taille et de la position (X,Y) des obstacles dans le niveau.
    - **Apparence visuelle** : Utilisation de sprites distincts pour chaque type d'obstacle.
    - **Comportement face aux dégâts** : Réactions des obstacles en cas de tir ou de collision, avec des effets distincts en fonction de leur type.

1.4 Structure de stockage et de gestion des données

Les niveaux et leurs éléments seront décrits et stockés dans une base de données relationnelle pour assurer une organisation et un accès optimisé aux données de jeu.

1.5 Fonctionnalités de gameplay

Au minimum, deux niveaux seront implémentés, chacun incluant :

- **Un joueur** avec toutes les capacités et limitations définies.
- **Des ennemis** intégrés selon un minutage et une répartition adaptée au niveau.
- **Des obstacles** avec leurs comportements en fonction des interactions du joueur et des ennemis.

1.6 Gestion des scores

Un système de gestion des scores élevés (highscores) sera implémenté et stocké dans la base de données pour permettre le suivi et la comparaison des meilleures performances des joueurs.

Gestion de projet

La gestion de ce projet s'est faite depuis Github pour le stockage des fichiers et dossier du jeu pour avoir des suivis continues et régulier. Aussi pour dans le cadre du jeu nous avons utilisés Icescrum pour avoir nos UserStories et tâches quotidiennes. Le JDT se fera via IceTool qui est là pour nous générer automatiquement le jdt et la stratègie de test

2 Analyse / Conception

2.1 Gameplay

2.1.1.1 Le joueur :

Le joueur peut se déplacer avec A et D, il pourra monter et descendre avec W et S. Il peut se déplacer librement sur la map et il a comme spécificité d'avoir une arme (Tirer, Changer, Recharger), des points de vie. Et tout sera compatible dans n'importe quel format de taille de fenêtre « Windows ».

2.1.1.2 Les ennemis :

Les ennemis seront capables de se déplacer librement sur la map avec une arme qui leur seront prédéfinis selon leur niveau, il aura 100hp comme le joueur. Et il saura complètement autonome.

2.1.1.3 Les déplacements :

Les déplacements incluent des mouvements latéraux (gauche/droite) ainsi que des sauts, permettant au joueur d'éviter les obstacles et de naviguer entre les plateformes.

2.1.1.4 Le Tir :

Les tirs sont un mécanisme qui donnerons un objectif. Le joueur et l'ennemi auront une arme prédéfinie pour le commencement du jeu. Les armes auront chaqu'une plein de caractéristique unique qui permet une diversité de gameplay. (Ils pourront tirer, recharger, changer).
La gestion des vies :
La vie se fera très simplement avec un gestionnaire très simple à base de hp max (ici de 100) qui sera enlever selon les dégats mis par l'arme selectionné. Dès que le joueur ou l'ennemi tombe à zéro il est mort.

2.2 Concept

Diagramme de Classe (le lien ne veut pas marcher et je ne comprends pas pourquoi) mais il est dans le dossier racine du PDF.

2.3 Analyse fonctionnelle

2.4 Stratégie de test

Les tests que j'ai décidé de faire au vu du temps accordé était de mettre des Try, Catch, dans mon code pour le permettre de s'il y a un problème durant la génération ou bien durant la partie des parties problèmes. Les tests se mettents dans la console pour me prévenir ou de faire arreter le programme si celui-ci comporte un bug.

3 Réalisation

La réalisation du projet *Vanguard Fighter's* a été organisée autour de plusieurs étapes majeures, chacune apportant des avancées significatives au développement du jeu. Ce chapitre présente les choix techniques, les outils utilisés, les difficultés rencontrées et les solutions apportées.

3.1 Environnement de travail

Organisation des dossiers :

- **Doc/** : Contient les documents relatifs à la conception, comme le rapport, la documentation technique, et les spécifications fonctionnelles.
- **Vanguard_Fighters/** : Dossier racine du code source, comprenant tous les fichiers nécessaires à l'exécution et la compilation.

Versions des systèmes et outils :

- **Système d'exploitation** : Windows 10.
- **IDE** : Visual Studio 2022 avec le framework MonoGame.
- **Version .NET** : .NET 8.0.
- **Contrôle de version** : Git et GitHub pour la gestion des versions.

Logiciels de gestion :

- **IceScrum** : Utilisé pour le suivi des user stories et des tâches. Bien que parfois contraignant, cet outil a permis de structurer le projet.
- **IceTool** : Génération automatique de journaux de test et stratégies associées.

3.2 Développement et implémentation

Le développement s'est concentré sur la mise en place de différents modules du jeu :

1. Gestion du joueur :

- **Mouvement** :
  - Le joueur peut se déplacer librement sur la carte avec les touches **A** et **D**, et sauter avec la touche **Espace**.
  - Une gestion fluide des collisions permet d'éviter les bugs (comme la téléportation).

- **Combat** :
  - Implémentation du tir, du rechargement manuel (**R**) et automatique.
  - Possibilité de changer d'arme avec des touches spécifiques.
- **Système de vie** :
  - Affichage d'une barre de vie dynamique.
  - Perte de vie selon les dégâts infligés par les ennemis ou les obstacles.

2. Gestion des ennemis :

- **IA autonome** : Les ennemis se déplacent sur la carte en fonction d'un algorithme simple, avec la capacité de détecter le joueur dans un champ de vision défini.
- **Système de tir** :
  - Les ennemis peuvent tirer lorsqu'ils ont le joueur en ligne de mire.
  - Attribution d'armes spécifiques à chaque type d'ennemi.
- **Dégâts et mort** :
  - Les ennemis possèdent un nombre de points de vie défini (100 HP). Lorsqu'ils atteignent 0, ils disparaissent et un nouvel ennemi peut apparaître.

3. Niveaux et obstacles :

- **Cartes** :
  - Utilisation de MonoGame.Extended pour le rendu des cartes (TiledMapRenderer).
  - Intégration de différents niveaux, chacun avec des obstacles uniques.
- **Obstacles** :
  - Gestion des collisions entre le joueur/ennemis et les obstacles.
  - Apparence visuelle propre à chaque type d'obstacle (barricades, plateformes).

4. Interface utilisateur :

- **Menus** :
  - Un menu principal minimaliste avec les options « Play », « Options », « Quit ».
  - État du bouton « Play » fonctionnel (alternative avec la touche **P**).
- **Accessibilité** :
  - Textes contrastés pour faciliter la lisibilité.
  - Boutons suffisamment grands pour une navigation simple.

3.3 Tests et ajustements

Les tests ont été divisés en plusieurs sprints pour vérifier les différentes fonctionnalités :

Sprint 1 : Tir et Rechargement

- **Tir basique** : Tir d'une balle dans la direction du personnage.
- **Rechargement manuel et automatique** : Fonctionnel, avec un délai adapté à chaque arme.
- **Cadence de tir** : Gestion des tirs en mode rafale ou simple.

Sprint 2 : IA et Ennemis

- **Détection du joueur** : Les ennemis poursuivent le joueur lorsqu'il entre dans leur champ de vision.
- **Tirs ennemis** : Les ennemis tirent lorsqu'ils voient le joueur.
- **Mort des ennemis** : Fonctionnelle, avec apparition d'un nouvel ennemi aléatoire.

Sprint 3 : Interface et Menus

- **Accès au jeu** : Bouton « Play » redirige vers le premier niveau.
- **Gestion des paramètres** : Les boutons pour activer/désactiver le son ne sont pas encore fonctionnels.
- **Retour au menu principal** : Fonctionnalité partiellement implémentée.

3.4 Difficultés rencontrées

1. **Gestion des collisions** :
    a. Les collisions initiales étaient approximatives, causant des bugs comme la téléportation du joueur.
    b. Solution : Implémentation d'un algorithme plus précis basé sur les bounding boxes.
2. **Armes des ennemis** :
    a. Difficulté à attribuer des armes dynamiquement aux ennemis.
    b. Solution : Utilisation d'une bibliothèque (« WeaponsLibrary ») pour centraliser les caractéristiques des armes.
3. **Menus cliquables** :
    a. Les boutons cliquables n'étaient pas fonctionnels à temps.
    b. Solution temporaire : Utilisation de touches alternatives comme **P** pour lancer le jeu.

3.5 Optimisations

1. **Performances** :
    a. Réduction des calculs inutiles pour le rendu des cartes et des sprites.
    b. Gestion efficace des objets en mémoire (balles hors écran supprimées automatiquement).
2. **Lisibilité du code** :
    a. Dispersion des classes avec une architecture orientée objets.
    b. Organisation en modules : **Models**, **View**, **Services**, **Library**.
3. **Expérience utilisateur** :
    a. Design simplifié pour un menu clair et une navigation intuitive.
    b. Textes bien contrastés et boutons accessibles.

La réalisation du projet *Vanguard Fighter's* a permis d'implémenter une base solide pour un jeu de type platformer 2D. Bien que certaines fonctionnalités restent à améliorer (menus avancés, optimisation IA), le projet a atteint ses objectifs principaux tout en ouvrant la voie à des évolutions futures.

3.6 Déroulement

Le projet s'est bien déroulé pour la partie code et rapport mais pour ce qui est de IceScrum qui (à mon avis) nous fait perdre trop de temps ett très peu utile, m'a mis très en retard au vu du travail que j'avais a fournir et que malheuresement, beaucoup de temps de travail se faisant à la maison j'ai souvent oublié de remplir mon IceScrum ce qui m'a valu un relachement au milieu du projet.

3.7 Mise en place de l'environnement de travail

Doc/ : Ce dossier contient les documents relatifs au projet, y compris le rapport, la documentation de conception, les spécifications techniques et les fichiers de gestion de projet.
Vanguard Fighter's/ : Ce dossier est la racine du code source et inclut tous les fichiers nécessaires pour l'exécution et la compilation du jeu

- **Versions des systèmes d'exploitation et des outils logiciels**
    - **Système d'exploitation** : Windows 10
    - **IDE** : Visual Studio 2022 pour le développement C# avec le framework MonoGame.
    - **Version .NET** : .NET 8.0
    - **Contrôle de version** : Git avec GitHub pour le suivi et la gestion des versions du code source et des documents.

4 Description des tests effectués

4.1.1 Sprint 1

4.1.1.1 Tir et Rechargement

Tir Basique	Lorsque le joueur appuie sur la touche "clic gauche de la souris" (ou autre touche configurée pour le tir), le personnage doit tirer une balle dans la direction où il fait face	OK 1 Nov
Nombre de balles	Le joueur peut tirer un nombre limité de balles avant de devoir recharger	OK 1 Nov
Cadence de tir	Si le jeu autorise un tir automatique, le joueur peut maintenir la touche de tir pour tirer en continu, sinon chaque appui correspond à un tir.	OK 1 Nov
Rechargement manuel	Lorsque le joueur appuie sur la touche "R", le personnage doit recharger l'arme.	OK 1 Nov
Rechargement automatique	Si le joueur tente de tirer alors que son chargeur est vide, le personnage doit automatiquement commencer à recharger	OK 1 Nov
Temps de rechargement	Il doit y avoir un délai de rechargement pour chaques armes	OK 1 Nov

4.1.1.2 IA (Bots)

Repérage du joueur	Si l'ennemi voit le joueur (dans un champ de vision défini), il doit commencer à poursuivre le joueur, puis tirer sur le joueur.	OK 1 Nov
L'ennemi (tirs)	Si l'ennemi voit le joueur, Il tire tant qu'il est devant lui	OK 1 Nov
Deplacement	L'ennemi peut bouger tout seul	OK 1 Nov

4.1.1.3 Décors de Jeu

Affichage de la vie	le joueur doit voir une barre de vie ou un indicateur des points de vie de l'ennemi.	OK 1 Nov
Diminution des points de vie de l'ennemi	Lorsque le joueur attaque l'ennemi (avec des balles ou une attaque au corps à corps), les points de vie de l'ennemi doivent diminuer.	OK 1 Nov
Mort de l'ennemi	Mort de l'ennemi (ecran fin de partie)	OK 1 Nov

4.1.1.4 Déplacements des personnages

Deplacement à droite	Lorsque le joueur appuie sur la touche "D" (ou flèche droite), le personnage doit se déplacer à droite.	OK 1 Nov
Deplacement à gauche	Lorsque le joueur appuie sur la touche "A" (ou flèche gauche), le personnage doit se déplacer à gauche.	OK 1 Nov
Arret du personnage	Lorsque le joueur relâche la touche de déplacement, le personnage doit s'arrêter immédiatement.	OK 1 Nov

fluidité	Le personnage ne continue pas de se déplacer une fois la touche relâchée.	OK 1 Nov
Saut	Lorsque le joueur appuie sur la touche "Espace", le personnage doit sauter vers le haut.	OK 1 Nov
Atterissage	Le personnage retombe sur le sol après avoir sauté.	OK 1 Nov

4.1.1.5 Points de Vie et Soin

Affichage des points de vie	Le nombre de points de vie actuels du joueur doit être visible à l'écran à tout moment.	OK 1 Nov
Diminution des points de vie	Chaque fois que le joueur subit des dégâts le nombre de points de vie doit diminuer en conséquence	OK 1 Nov
Points de vie minimum	Si les points de vie du joueur atteignent 0, le personnage doit mourir (perdre la partie)	OK 1 Nov

4.1.1.6 Menu

Accès au jeu	Lorsque le joueur clique sur le bouton "Play", il doit être redirigé vers le début du jeu ou la première scène du jeu	???
Retour au menu	Si le joueur revient au menu, le bouton "Play" doit toujours être fonctionnel.	???
Accès au menu des paramètres	Lorsque le joueur clique sur le bouton "Paramètres", un sous-menu avec différentes options	???
Retour au menu principal	Après avoir modifié les paramètres, le joueur doit pouvoir revenir au menu principal en appuyant sur un bouton de retour ou un bouton "X".	???
Désactiver le son	Lorsque le joueur clique sur le bouton pour enlever le son, toute la musique et les effets sonores du jeu doivent être désactivés.	???
Activer le son	Si le joueur reclique sur le bouton, le son doit être réactivé.	???
Indicateur visuel	Le bouton doit changer d'état ou d'icône (par exemple, une icône de son coupé) pour indiquer si le son est activé ou désactivé.	???
Fermer l'application	Lorsque le joueur clique sur le bouton "Quitter", le jeu doit se fermer correctement.	???

4.1.2 Sprint 1

4.1.2.1 Tir et Rechargement

Tir Basique	Lorsque le joueur appuie sur la touche "clic gauche de la souris" (ou autre touche configurée pour le tir), le personnage doit tirer une balle dans la direction où il fait face	OK 1 Nov
Nombre de balles	Le joueur peut tirer un nombre limité de balles avant de devoir recharger	OK 1 Nov
Cadence de tir	Si le jeu autorise un tir automatique, le joueur peut maintenir la touche de tir pour tirer en continu, sinon chaque appui correspond à un tir.	OK 1 Nov
Rechargement manuel	Lorsque le joueur appuie sur la touche "R", le personnage doit recharger l'arme.	OK

		1 Nov
Rechargement automatique	Si le joueur tente de tirer alors que son chargeur est vide, le personnage doit automatiquement commencer à recharger	OK 1 Nov
Temps de rechargement	Il doit y avoir un délai de rechargement pour chaques armes	OK 1 Nov

4.1.2.2 IA (Bots)

Repérage du joueur	Si l'ennemi voit le joueur (dans un champ de vision défini), il doit commencer à poursuivre le joueur, puis tirer sur le joueur.	OK 1 Nov
L'ennemi (tirs)	Si l'ennemi voit le joueur, Il tire tant qu'il est devant lui	ko 1 Nov
Deplacement	L'ennemi peut bouger tout seul	OK 1 Nov

4.1.2.3 Décors de Jeu

Affichage de la vie	le joueur doit voir une barre de vie ou un indicateur des points de vie de l'ennemi.	ko 1 Nov
Diminution des points de vie de l'ennemi	Lorsque le joueur attaque l'ennemi (avec des balles ou une attaque au corps à corps), les points de vie de l'ennemi doivent diminuer.	OK 1 Nov
Mort de l'ennemi	Mort de l'ennemi (ecran fin de partie)	OK 1 Nov

4.1.2.4 Déplacements des personnages

Deplacement à droite	Lorsque le joueur appuie sur la touche "D" (ou flèche droite), le personnage doit se déplacer à droite.	OK 1 Nov
Deplacement à gauche	Lorsque le joueur appuie sur la touche "A" (ou flèche gauche), le personnage doit se déplacer à gauche.	OK 1 Nov
Arret du personnage	Lorsque le joueur relâche la touche de déplacement, le personnage doit s'arrêter immédiatement.	OK 1 Nov
fluidité	Le personnage ne continue pas de se déplacer une fois la touche relâchée.	OK 1 Nov
Saut	Lorsque le joueur appuie sur la touche "Espace", le personnage doit sauter vers le haut.	OK 1 Nov
Atterissage	Le personnage retombe sur le sol après avoir sauté.	OK 1 Nov

4.1.2.5 Points de Vie et Soin

Affichage des points de vie	Le nombre de points de vie actuels du joueur doit être visible à l'écran à tout moment.	ko 1 Nov

Diminution des points de vie	Chaque fois que le joueur subit des dégâts le nombre de points de vie doit diminuer en conséquence	OK 1 Nov
Points de vie minimum	Si les points de vie du joueur atteignent 0, le personnage doit mourir (perdre la partie)	OK 1 Nov

4.1.2.6 Menu

Accès au jeu	Lorsque le joueur clique sur le bouton "Play", il doit être redirigé vers le début du jeu ou la première scène du jeu	OK 1 Nov
Retour au menu	Si le joueur revient au menu, le bouton "Play" doit toujours être fonctionnel.	OK 1 Nov
Accès au menu des paramètres	Lorsque le joueur clique sur le bouton "Paramètres", un sous-menu avec différentes options	ko 1 Nov
Retour au menu principal	Après avoir modifié les paramètres, le joueur doit pouvoir revenir au menu principal en appuyant sur un bouton de retour ou un bouton "X".	ko 1 Nov
Désactiver le son	Lorsque le joueur clique sur le bouton pour enlever le son, toute la musique et les effets sonores du jeu doivent être désactivés.	ko 1 Nov
Activer le son	Si le joueur reclique sur le bouton, le son doit être réactivé.	ko 1 Nov
Indicateur visuel	Le bouton doit changer d'état ou d'icône (par exemple, une icône de son coupé) pour indiquer si le son est activé ou désactivé.	ko 1 Nov
Fermer l'application	Lorsque le joueur clique sur le bouton "Quitter", le jeu doit se fermer correctement.	ko 1 Nov

4.2 Erreurs restantes

Il reste beaucoup d'erreur depuis que j'ai changé mon code récemment pour ajouter l'arme à l'ennemi pour qu'il puisse la changer après X temps. La gestion de collision très simple que je veux pousser pour qu'il n'y ait plus de problème de téléportation vers une plateforme. Le main_menu que je n'ai pas finalisé il manque la partie bouton cliquable pour une meilleur gestion, il y a une alternative avec « P » pour lancer le jeu. Mon ennemi ne contient pas d'arme sur la dernière version du jeu ! (jouable)

5 UX

5.1.1 Éco-Conception

Pour réduire l'impact écologique de *Vanguard Fighter's*, j'ai fait plusieurs choix pour que le jeu consomme moins d'énergie.

- **Palette de Couleurs :** J'ai choisi des couleurs sobres, avec du bleu, du violet, du vert, et du blanc. Ces couleurs ne sont pas trop vives, ce qui aide à économiser de l'énergie, surtout sur les écrans OLED.
- **Optimisation de la Taille des Caractères et de la Lisibilité :** J'ai réglé la taille des textes pour qu'ils soient facilement lisibles sans devoir zoomer ou faire des ajustements. Ça aide aussi à économiser des ressources, car le rendu des textes est plus simple.
- **Design Simpliste :** J'ai opté pour un design simplifié dans ma maquette. En limitant les éléments visuels, l'interface est plus légère et le jeu utilise moins de ressources, ce qui est bon pour l'environnement.

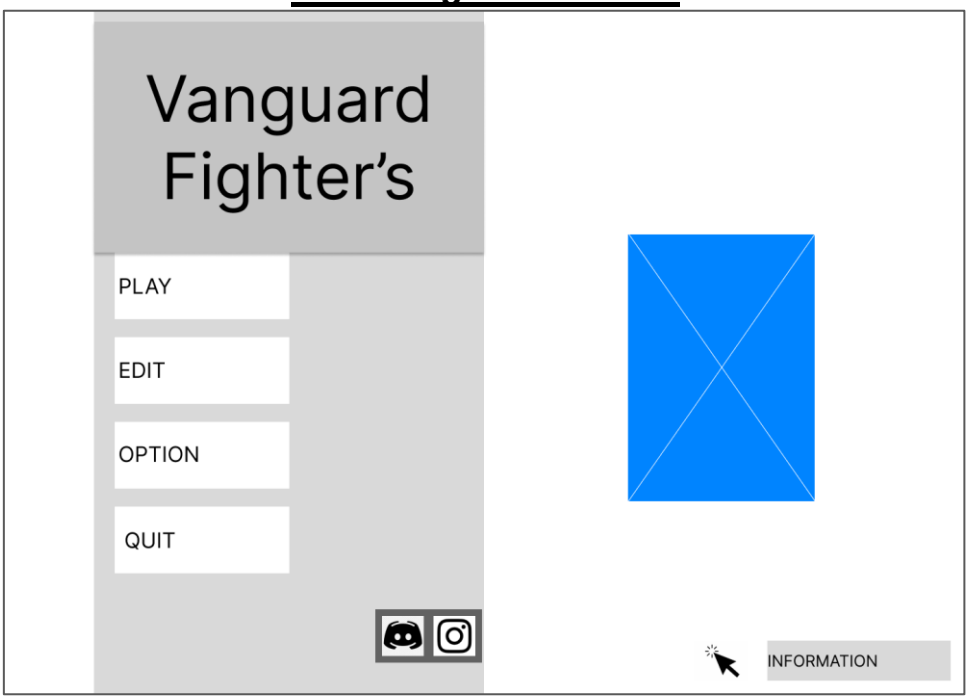
5.1.2 Accessibilité

J'ai aussi pensé à rendre le jeu accessible pour un maximum de joueurs, peu importe leurs capacités.

- **Contraste des Textes et Fonds :** J'ai utilisé des couleurs bien contrastées pour les textes et les fonds, comme le texte blanc sur fond bleu ou rouge. Cela rend les textes plus faciles à lire pour ceux qui rencontrent des difficultés de vision.
- **Navigation et Taille des Boutons :** Les boutons sont assez grands pour être facilement cliqués, même avec une manette. La navigation est aussi simple et peut se faire avec le clavier, ce qui rend le jeu accessible aux personnes rencontrant des difficultés motrices.
- **Maquette Simpliste pour une Meilleure Compréhension :** Le design est simple pour que tout soit clair. Les options sont visibles et il n'y a pas d'éléments superflus. Ça permet aux joueurs de comprendre rapidement comment naviguer dans le jeu.

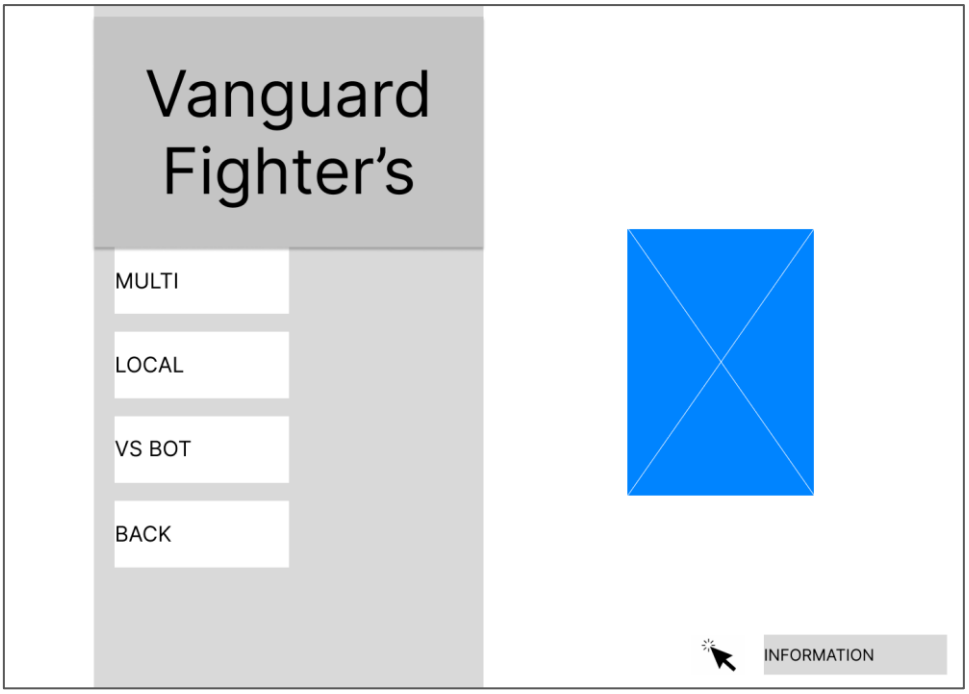
5.1.3 Conception

HomePage-Wireframe



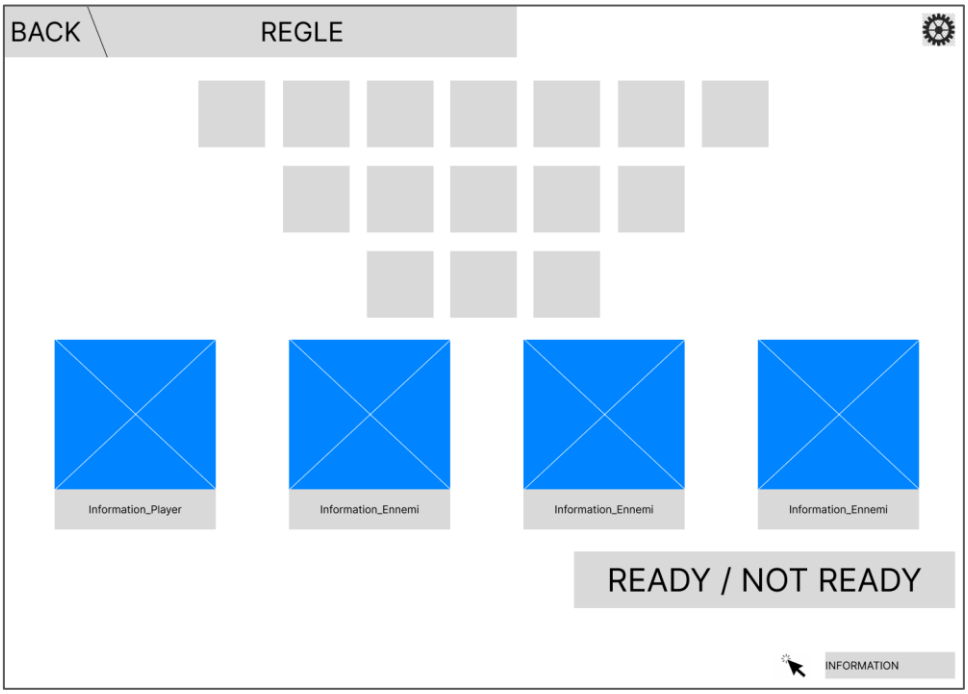
Ceci est le Home page qui permet d'aller sur les boutons suivant (« Play », « Edit », « Option », « Quit ». La page est minimaliste pour bien comprendre la page ou elle vous emmène par la suite) Le carré bleu sert de démonstration où se mettra l'image de votre personnage.

Play-Wireframe



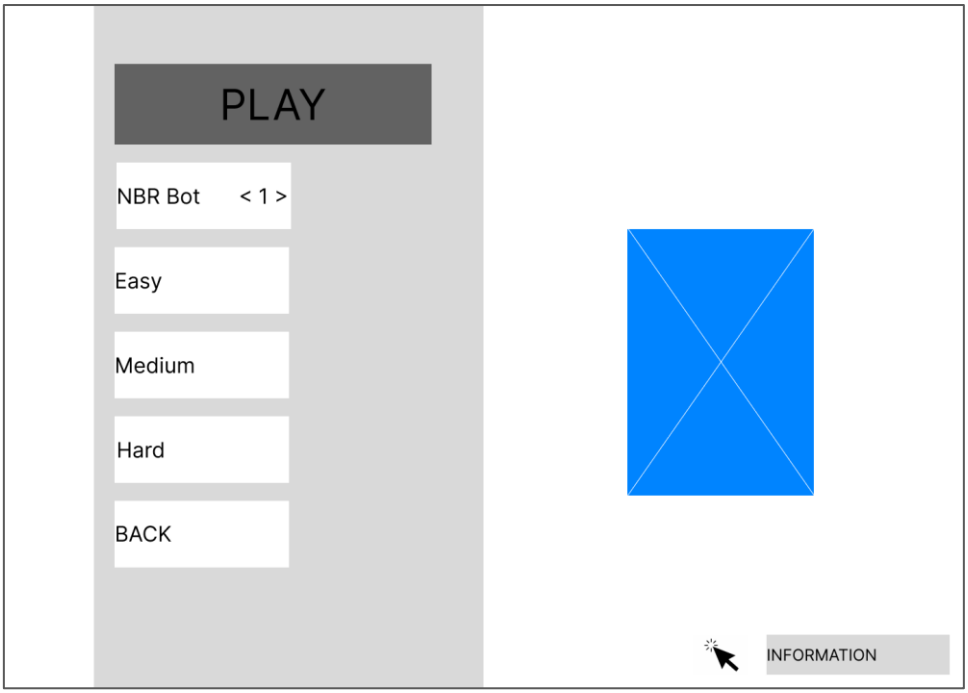
La page Play vous emmène dans les bases du jeu et de ce qu'il y aura, donc du multi pour jouer en ligne, un local pour jouer avec ses amis chez-soi et pour la fin juste un mode pour s'entraîner avec un bot ou plusieurs.

Multi-Wireframe



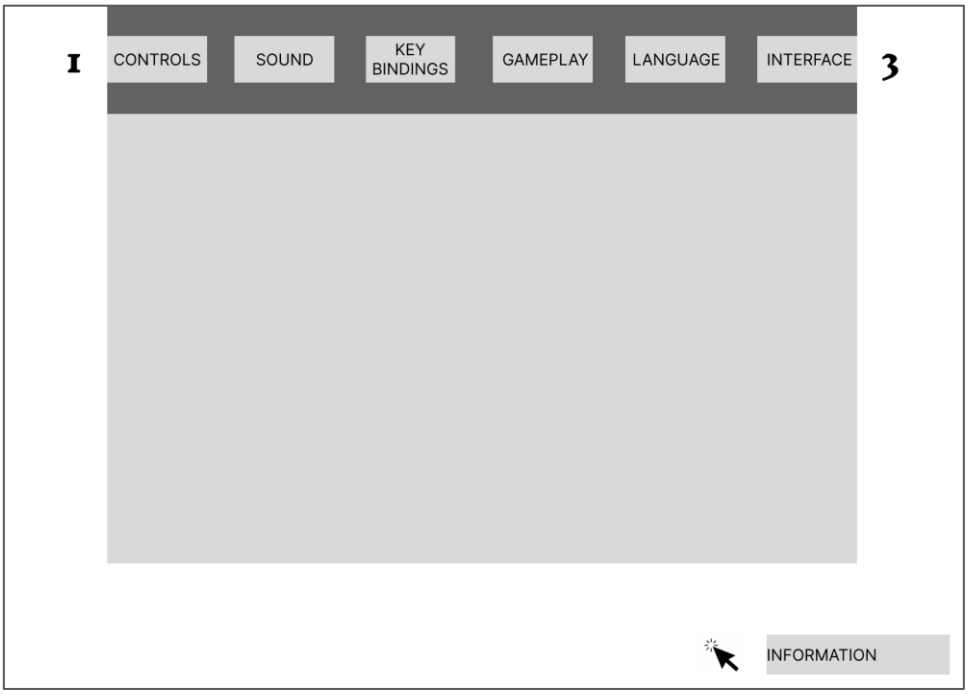
Cette page vous montre que le jeu permettra de joueur jusqu'à 4 joueurs et avec plusieurs possibilités de choix de personnages.

DifficultBot_Menu-Wireframe

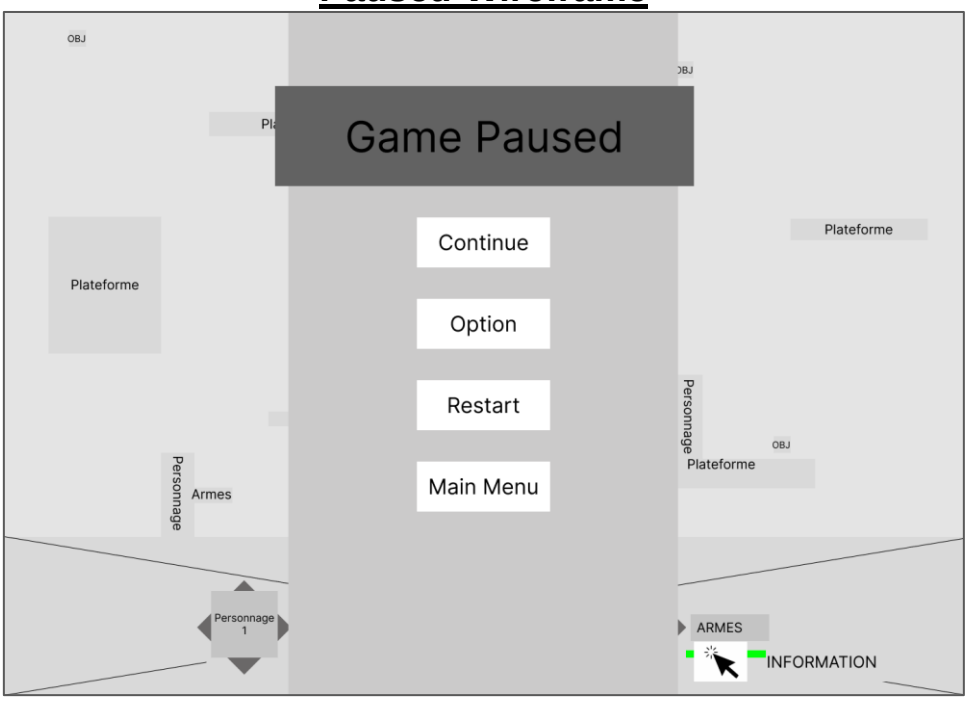


DifficultLocal_Menu-WireFrame

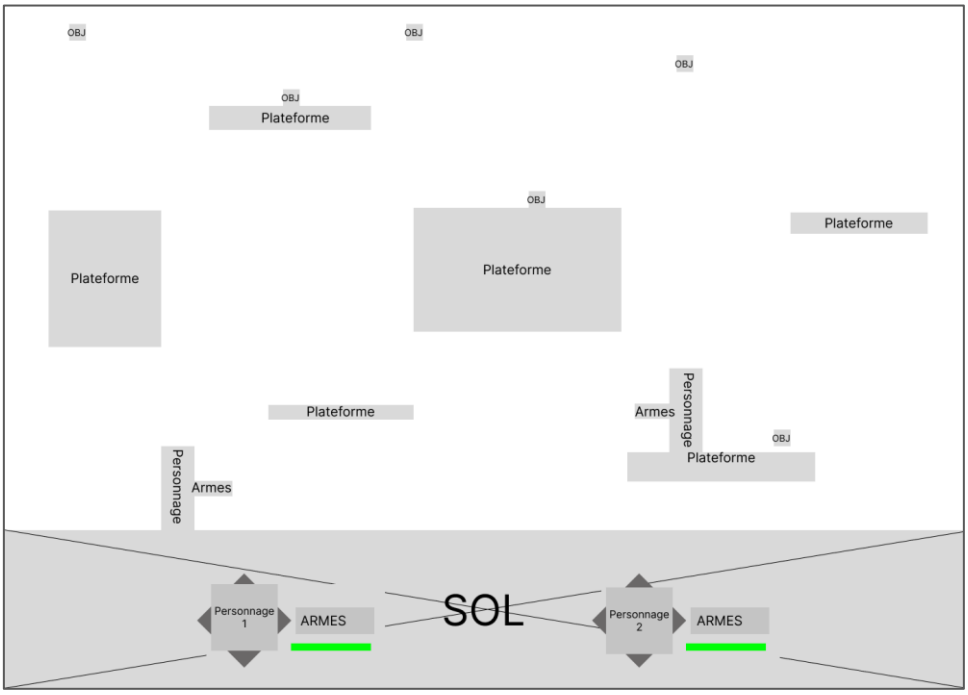
Option-Wireframe



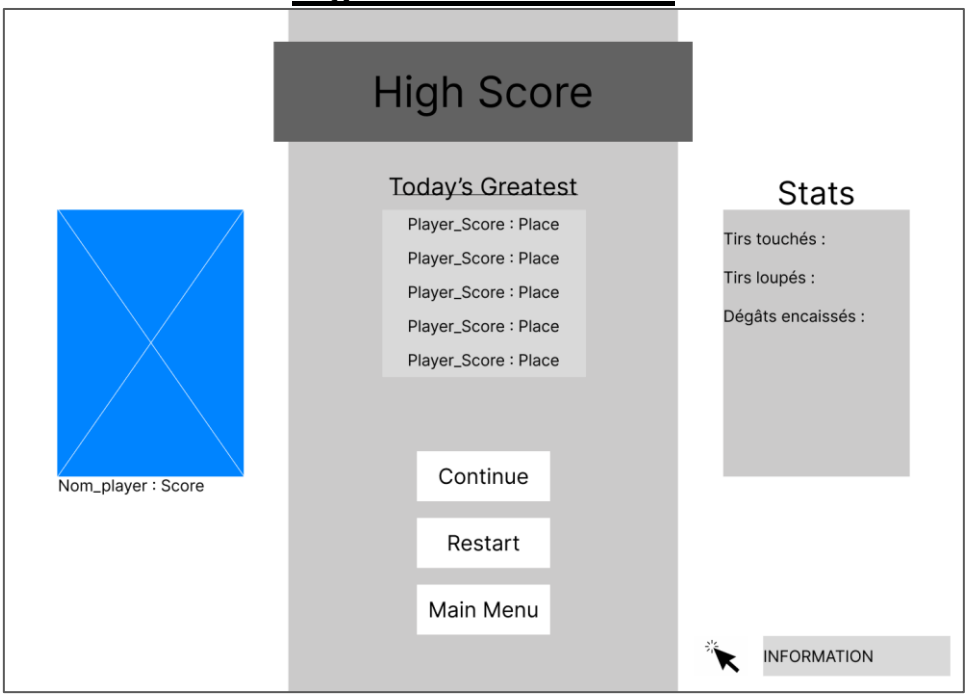
Paused-Wireframe



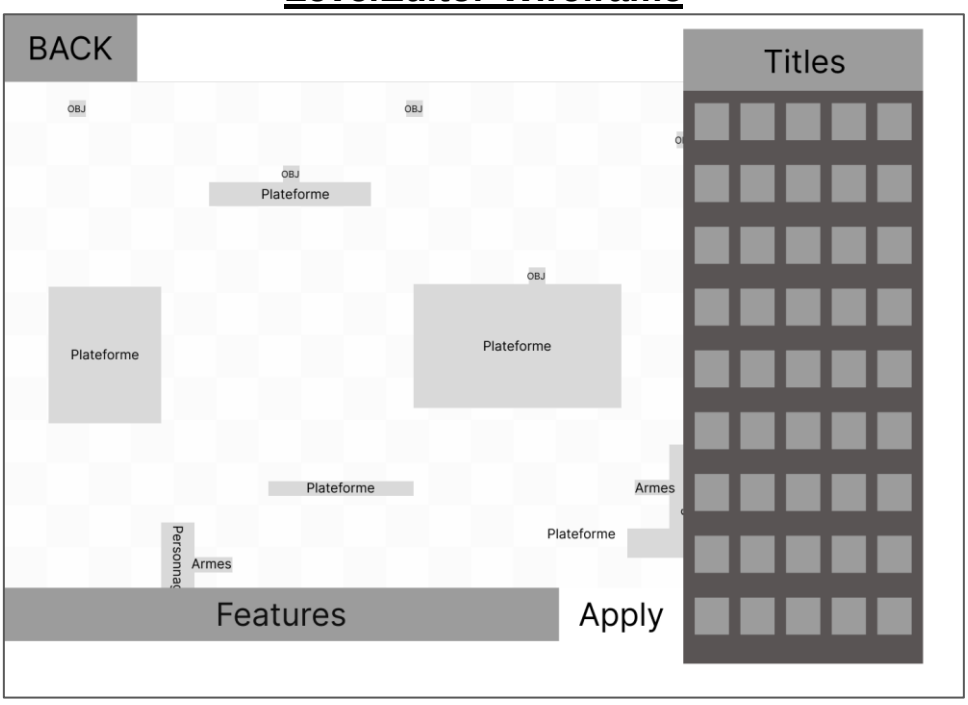
InGame-Wireframe



HighScore-Wireframe



LevelEditor-Wireframe



LevelEditor-High_Fidelity

Le LevelEditor est très simple mais cela est dû que je voulais que la page soit au maximum compréhensible pour l'utilisateur tout en ayant l'essentiel pour s'amuser à faire des maps. Le menu des TileSet(les blocs de textures) se visualise comme un téléphone pour bien touché le maximum de personnage et de comment utiliser ce champ. Les boutons sont grands et facile d'accès pour une meilleur utilisation.

6 Conclusions

Mes objectifs personnels sont atteint juste dû faite que j'ai appris plein de chose durant le projet même si les calculs complexe que je ne sais pas faire se sont ChatGPT ou bien Internet qui me les a fournis. J'ai tout de même aimer faire ce projet que bien entendu je continurais, c'est le but de ce projet c'st que même si je ne le finisais pas, que je pourrais le continuer assez facilement avec de la motivation d'où mon écartement de space invaders. Un peu Complexe tout de même, j'aurais du me fixer un plafond pour, ne pas trop, tout de même avoir un PA à la fin du projet. Les difficultés que j'ai aperçu durant le projet c'est surtout l'arboresence du projet pour bien s'y retrouver tout du long et aussi les fonctionnalités à implenter qui font casser tout le reste du code tout le temps, donc fallait revenir dessus et ça prenait beaucoup de temps.

7 IA et interactions

La création et la gestion de l'intelligence artificielle ont été réalisées avec une approche itérative. Pour chaque fonctionnalité complexe, j'ai écrit le code de manière autonome et j'ai sollicité des corrections et des explications pour résoudre les erreurs. Cette collaboration m'a permis d'améliorer ma compréhension et d'implémenter des solutions efficaces.