PROJET S2

Nachos Rumble

Rapport final de projet

Los Successos

Paul Berthelot (chef de projet)

Théo Charbonnier

Kilian Hardy

Sam Esber

Juin 2022

Table des matières

1 Introduction **4**

2 Reprise du cahier des charges **5**
2.1 Présentation des membres du groupe 5
2.2 Origine du projet 6
2.3 Histoire de l'art 7
2.4 Nachos Rumble 8
2.4.1 Gameplay 8
2.4.2 Intelligence Artificielle 8
2.4.3 Graphismes 9
2.4.4 Site web 10
2.4.5 Gestion de la caméra 10
2.4.6 Physique 10
2.4.7 Réseau 10
2.4.8 Interface 11
2.4.9 Audio 12
2.5 Réalisation 12
2.5.1 Répartition des tâches 12
2.5.2 Organisation du travail 13
2.5.3 Planning prévisionnel 13
2.5.4 Outils utilisés 13
2.5.5 Aspect financier 14

3 Résumé de la première soutenance **15**
3.1 Multijoueur 15
3.2 Mécanique de tir 15
3.3 Gestion de la partie et des joueurs 15
3.4 Caméra 16
3.5 Physique et mouvement 16

4 Résumé de la seconde soutenance **17**
4.1 Intelligence Artificielle 17
4.2 Modèle des joueurs 17

4.3 Animations 18
4.4 Audio 18
4.5 Menu 18
4.6 Données et statistique des joueurs 18
4.7 Gestion de la partie 18
4.8 Scoreboard 19
4.9 ATH / HUD 19
4.10 Site internet 19
4.11 Système d'arme et de tir 19
4.12 Avancement et retard de la seconde soutenance 19

5 Explication du jeu **20**

6 Développement du projet **21**
6.1 Multijoueur 21
6.2 Interface 22
6.2.1 Menus 22
6.2.2 Interface en jeu 24
6.3 Audio 27
6.3.1 Composition et recherche de l'audio: Sam 27
6.3.2 Implémentation de l'audio : Kilian 27
6.3.3 Implémentation multijoueur des sons : Paul 27
6.4 Système d'armes et de tir 29
6.5 Système de déplacement 30
6.6 Identité visuelle du projet 30
6.7 Site internet 32
6.8 Modélisation 3D 36
6.8.1 Map 36
6.8.2 Point de réapparition 38
6.8.3 Personnage 39
6.8.4 Armes 41
6.9 Animations 42
6.9.1 Animation de mouvement 42
6.9.2 Animation d'armes 42
6.10 Intelligence Artificielle 44

6.11 Gestion du travail du groupe 46
6.11.1 Organisation du travail 46
6.11.2 Soutenances et rapports 47

7 Ressenti **48**

8 Conclusion **50**

1 Introduction

Paul Berthelot, Sam Esber, Théo Charbonnier et Kilian Hardy : voici les noms des quatre membres de l'équipe Los Successos.

Cette équipe a vu le jour dans le cadre du "projet S2" d'EPITA. Son but ? Réaliser un projet d'informatique en 6 mois.

C'est ainsi qu'est né Nachos Rumble, un jeu vidéo de style *FPS* a l'ambiance mexicaine.

Ce rapport de projet retrace l'évolution au fil du temps de la réalisation de ce projet. Il y sera présenté le jeu tel que nous voulions le créer au début du semestre, puis nous expliquerons le développement de chaque spécificités du jeu, en expliquant nos choix, nos peines et nos joies.

2 Reprise du cahier des charges

Le but de cette partie est de reprendre le cahier des charges du projet pour montrer ce que nous envisagions de créer, par quels moyens, et en combien de temps.
Cela nous permettra d'en suite comparer le projet à l'état final avec ce que nous avions en tête il y a plusieurs mois.

2.1 Présentation des membres du groupe

Notre équipe est composée de quatre personnalités se complétant parfaitement pour la création d'un projet si complet.

Kilian Hardy :

J'ai découvert à partir du lycée une passion pour l'informatique qui s'est développée encore aujourd'hui. Grâce à la spécialité NSI en première et en terminale, j'ai pu m'initier à plusieurs langages de programmation. J'ai en effet eu la chance de mener plusieurs projets informatiques en groupe, tels que la création d'un jeu vidéo en python. Sur ce projet je m'occuperais particulièrement de l'aspect de la physique dans le jeu. C'est un côté généralement peu connu des joueurs mais sans doute le plus important car c'est ce qui permet de faire la base d'un jeu en général. Ce projet me permettra vraiment d'apprendre à travailler en équipe et d'acquérir de nombreuses connaissances en développement.

Sam Esber :

Ayant toujours eu un intérêt porté sur les nouvelles technologies, et plus largement sur l'informatique, je m'informe sur ce sujet régulièrement. Et c'est dans cette continuité que j'ai d'abord décidé de prendre l'option NSI au lycée, puis de m'orienter vers une école d'ingénieur en informatique. J'ai déjà réalisé des projets dans ce domaine durant le lycée, allant d'un simple site Web en classe de Seconde, à un jeu 2D complet en Javascript directement disponible depuis un navigateur. J'ai donc déjà des expériences de travail en équipe. Je suis ravi de pouvoir mettre mes compétences dans un projet qui fait sens. Pour ce projet, je me concentrerai principalement sur la réalisation des graphiques, modèles 3D, de l'audio et de leurs implémentations en jeu.

Théo Charbonnier :

Théo Charbonnier : hormis les bases acquises en python en science de l'ingénieur et en mathématiques au lycée, j'ai aussi pu programmer une machine qui lit à haute voix un texte écrit sur une feuille ; sur une carte Raspberry Pi (en python aussi). Je n'ai donc aucune expérience en Unity ou en c#. Mes motivations sont d'apprendre un nouveau langage mais aussi de comprendre la fabrication d'un jeu vidéo. Je m'occupe de la partie réseau du projet.

Paul Berthelot :

Depuis toujours, je suis passionné d'informatique ainsi que des jeux vidéo, et je vois ce projet comme une véritable opportunité qui pourra me permettre d'apprendre, par la pratique, à gérer un travail en équipe ainsi que de développer une grande quantité de connaissances sur le développement et la création de jeux vidéo. J'ai l'avantage d'être très motivé et curieux (peut-être trop), c'est pourquoi le fait de créer un jeu entier par soi-même m'enthousiasme fortement, et j'ai hâte de voir le résultat du travail de mon groupe à la fin de ce projet.

2.2 Origine du projet

Dans le but de trouver l'idée de base pour ce projet, nous nous étions fixés des objectifs précis: réaliser quelque chose d'original, qui nous correspond, et qui, en tant que joueurs occasionnels, nous divertira. En nous inspirant des jeux auxquels nous jouons ainsi que des projets d'années précédentes, nous avons décidé de proposer un FPS en multijoueur.

Le concept étant assez simple, nous avions donc le choix de travailler sur l'aspect gameplay ou l'aspect visuel pour le rendre original, et nous avons opté pour la seconde option. En effet, le FPS étant un genre tellement populaire et repris de si nombreuses fois, nous n'avons pas les capacités de chambouler son gameplay avec nos moyens et nos connaissances.

Nous avons donc cherché un élément permettant de rendre un FPS différent et original, et pour cela quoi de mieux que l'humour ! Le thème du Mexique et de sa nourriture nous est apparu comme la solution évidente pour cela. C'est un univers facile à reconnaître visuellement (sombrero, moustaches...) et qui permet d'avoir une vraie ligne directrice dans le développement artistique du projet.

Le nom du jeu : **Nachos Rumble**, vient des *nachos* (plat mexicain) et du mot *Rumble* qui fait référence au combat. Le nom représente donc parfaitement l'identité du jeu que nous voulons créer : un jeu facile d'accès, divertissant et presque absurde.

2.3 Histoire de l'art

Le jeu de tir à la première personne (en anglais **FPS** pour **F**irst-**P**erson **S**hooter) est un genre de jeu vidéo de tir fondé sur des combats en vision subjective, c'est-à-dire que le joueur voit l'action à travers les yeux du protagoniste.

C'est un sous-genre du jeu d'action qui partage des traits communs avec les autres genres de jeux de tir, tels que le jeu de tir à la troisième personne. Depuis la création du genre, le jeu de tir à la première personne a été marqué par la naissance et l'évolution des graphismes en trois dimensions grâce aux progrès du matériel informatique, ainsi que l'évolution du jeu multijoueur et du sport électronique : le *Esport*.

Les deux premiers jeux de tir à la première personne sont *Maze War* et *Spasim sortis* en 1974. *MIDI Maze*, publié en 1987, introduit le jeu multijoueur en réseau local et devient le premier jeu de tir à la première personne sur console de jeux vidéo. C'est ensuite Microsoft et Bungie qui popularise le genre avec Halo en 2001.

Depuis, c'est l'un des genres de vidéo les plus populaires avec par exemple plus de 30 millions d'exemplaires vendus pour *Call Of Duty BlackOps II*.

Figure 1: Capture d'écran de *Call Of Duty BlackOps II*

2.4 Nachos Rumble

2.4.1 Gameplay

Le gameplay du jeu est commun à la majorité des jeux FPS, mais Nachos Rumble offre des parties bien plus divertissantes que compétitives. Les modes de jeu sont donc choisis en conséquence : nous avons implémenté des parties opposant les joueurs entre eux avec une limite de 20 joueurs. Les parties sont toutes basées sur un affrontement, dans laquelle les joueurs réapparaissent rapidement après s'être fait éliminer. Les points de réapparition seront des points d´définis de la map. La partie se termine lorsqu'un joueur atteint 20 kills (nombre de fois où il a tué les adversaires). Néanmoins, le rendu final n'étant toujours pas passé, ces limites sont susceptibles d'être modifiées.

Pour rendre le gameplay plus actif et moins linéaire, nous avons intégrés des bonus trouvables au sol offrant différentes capacités aux joueurs. Les armes présentes dans le jeu ne seront pas des armes avec un tir précis, au contraire : le hasard est un outil très utile pour rendre un jeu agréable au plus grand nombre. En résumé, le gameplay de Nachos Rumble est destiné au grand public, et le jeu est axé sur le divertissement des joueurs par tous les moyens possibles.

2.4.2 Intelligence Artificielle

L'intelligence artificielle que nous implémenterons dans le jeu sera destinée à remplacer les joueurs réels. Le mode de jeu *solo* sera donc le même qu'en multijoueur, mais les personnages ennemis seront contrôlés par l'IA.
Le rôle de cette IA est donc important si elle doit remplacer de nombreux joueurs d'une partie, et il sera nécessaire de la rendre équilibrée (ni trop forte ni trop nulle) et un minimum réaliste.// L'algorithme sur lequel sera basée l'IA devra prendre en compte sa position sur la map pour contrôler ses déplacements (pathfinding), mais aussi si un joueur ennemi est dans son champ de vision, si l'arme du personnage est rechargée, si les points de vie du personnage sont suffisants, etc.

Le comportement de l'IA sera long et complexe à mettre en place, mais il est nécessaire qu'il soit irréprochable dans un jeu où 10 joueurs s'affrontent, peuvent potentiellement être remplacés par des *"bots"* lorsqu'ils quittent la partie, ou si la partie n'a pas été lancée avec 10 joueurs réels.

2.4.3 Graphismes

Les graphismes doivent être en accord avec le gameplay voulu pour le jeu, c'est-à-dire : décontracté, simple et drôle.
C'est pourquoi l'environnement du jeu, les modèles de joueurs ainsi que tous les objets du jeu seront dans un style *low poly* : leurs traits seront simples. Les couleurs seront aussi plutôt claires et unies.
Il s'agit d'un style de graphisme qui plait beaucoup et qui est très accessible, mais aussi d'un moyen de faciliter la création de modèles 3D pour notre jeu.

Figure 2: Exemple d'arme low poly

Comme Nachos Rumble est un jeu de tir à l'ambiance purement mexicaine, les maps et modèles 3D du jeu auront évidemment un rapport direct avec cette inspiration là (personnages à apparence mexicaine, map dans une ville mexicaine, etc).

Figure 3: Inspiration actuelle pour la map

2.4.4 Site web

Notre site web permettra l'accès simple à notre jeu. Notre projet y sera expliqué en détail avec la mise à disposition de l'avancement de notre travail (rapports de soutenance, premières versions du jeu). Il comportera également une présentation des membres du groupe et de notre ressenti sur le projet. Et évidemment, la version finale de Nachos Rumble sera disponible en téléchargement.

Il s'agit de la vitrine même de notre projet, il est donc important qu'il soit visuellement agréable. Si nos capacités le permettent, nous envisageons de créer un site dynamique pour que les joueurs puissent se créer un compte et potentiellement suivre leurs statistiques de jeu sur le site.

2.4.5 Gestion de la caméra

Notre jeu étant un FPS, littéralement un *jeu de tir à la première personne*, il est logique que la caméra du joueur soit à tout instant positionnée au niveau de la tête du modèle 3D qu'incarnera le joueur. Néanmoins, nous implémenterons sûrement une vue à la troisième personne (derrière le modèle 3D) en guise de caméra *postmortem* sur l'ennemi, car c'est un effet qui revient souvent dans les FPS de ce genre.
Nous aurons aussi un travail d'animation de la caméra lors des mouvements comme la course ou le saut, car cela permet de rendre le jeu plus naturel qu'avec une caméra totalement fixe.

2.4.6 Physique

Il est nécessaire que le jeu possède une bonne physique, pour que les déplacements et sauts des joueurs soient les plus fluides possible, qu'ils soient agréables et pas qu'ils freinent le joueur dans son immersion dans l'ambiance du jeu.

Les potentiels objets à lancer (tels que des *grenades-tacos* par exemple) devront aussi être bien géré pour qu'ils soient utiles et facilement utilisables.

Nous examinerons aussi les possibilités de modifier certains paramètres de la physique en cours de jeu, comme la gravité par exemple, pour que les parties soient plus mouvementées.

2.4.7 Réseau

Pour implémenter la partie multijoueur de notre jeu, nous envisageons d'utiliser le moteur *Photon*. Il est important que la gestion du fonctionnement du multijoueur en réseau soit bien mise en place et qu'elle soit optimisée, au risque que le jeu soit

peu agréable, car les jeux de tir, même non compétitif, sont basés sur des latences faibles. Nachos Rumble envisage des parties avec de nombreux joueurs, il faudra donc mettre en place un système réseau solide permettant de faire marcher de telles parties.

2.4.8 Interface

Les menus et l'interface du jeu sont un aspect visuel très important du jeu, étant donné qu'il s'agit de la première chose que l'on voit de celui-ci. Il s'agit d'un élément trop souvent mis à de côté dans la création d'un jeu, alors qu'il est très important pour que le joueur soit à l'aise avant même d'avoir lancé sa première partie.
L'*ATH* (informations se superposant à l'écran lors du jeu), est un élément-clef du gameplay. Il permettra aux joueurs d'avoir des informations en temps réel sur sa santé, ses équipements, et permet de changer totalement l'aspect d'un jeu. Il ne s'agit donc pas d'un composant du jeu à prendre trop à la légère.

Nous comptons nous inspirer des menus de *Shotgun Farmers* et son l'interface, en y rajoutant évidemment une touche mexicaine

Figure 4: Inspiration pour l'interface : *Shotgun Farmers*

2.4.9 Audio

La partie audio de notre jeu permettra d'affirmer son identité et son lien avec l'esprit mexicain, latino. Il s'agit ici de créer une musique à consonance mexicaine pour les menus et les écrans de chargement, mais aussi d'avoir des bruitages simples mais amusants, et qui permettent surtout au joueur d'être immergé dans le jeu en ayant le sens de l'ouïe stimulé en plus de celui de la vue.

2.5 Réalisation

2.5.1 Répartition des tâches

Tâche	Paul	Sam	Kilian	Théo
Graphisme				
Map		X	X	
Armes		X	X	
Personnages		X		X
Animations	X			X
Audio				
Musiques		X		
Bruitages		X		
Réseau				
Multijoueur	X			X
IA				
Algorithme	X			X
Mécanique de jeu				
Physique	X	X	X	
Caméra	X		X	X
Mécanique de tir	X		X	X
Interface				
Menu	X	X		
ATH	X			
Site web				
Création du site internet			X	X

2.5.2 Organisation du travail

Aucun membre du groupe n'ayant des bases en programmation plus poussées que celles acquises par l'enseignement à EPITA, la charge horaire dédiée à ce projet s'annonce très importante. C'est pourquoi le groupe prévoit de se réunir très souvent que ce soit physiquement ou virtuellement, pour faire part de l'avancement de son travail personnel ainsi que pour permettre de mettre en commun nos avancements et s'entraider.

2.5.3 Planning prévisionnel

Tâche / Date	1re soutenance	2e soutenance	Soutenance finale
Graphisme	30%	70%	100%
Audio	20%	70%	100%
Réseau	20%	70%	100%
IA	10%	60%	100%
Mécanique de jeu	40%	70%	100%
Interface	40%	80%	100%
Site web	25%	75%	100%

2.5.4 Outils utilisés

Dans le processus de création du jeu, notre groupe aura besoin de créer ou de modifier des modèles 3D. Pour cela nous utiliserons **Blender**, qui est un logiciel gratuit et souple qui répondra très bien à nos besoins.

Nous travaillerons sur **Unity** comme moteur 3D de notre jeu, car nous avons quelques bases sur ce logiciel grâce aux travaux pratiques de *NTS*. Unity permet de développer notre jeu en C#, et nous nous aiderons de **Rider** pour écrire tout notre code.

Le logo du groupe, les visuels des menus et HUD du jeu seront réalisés grâce à **Photoshop**.

Évidemment, **LaTeX** nous sera utile pour rédiger de manière formelle les rapports de soutenance (ainsi que ce cahier des charges).

Enfin, notre plus grand allié : **Internet**, nous sera crucial pour ce projet, car toute la documentation des logiciels cités précedemment y est facilement accessible, ainsi que tout ce qui nous est nécessaire pour acquérir les compétences.

2.5.5 Aspect financier

Même si la plupart des outils que nous utiliserons (logiciels, tutoriels) sont gratuits, des dépenses ne sont pas à exclure. Premièrement, acheter un nom de domaine a un coût, tout comme l'hébergement de notre site.
De plus, l'acquisition d'*assets* (modèles 3D) n'est pas nécessaire mais nous permettrait de consacrer plus de temps au code (la physique, le gameplay) tout en ayant un jeu sûrement plus agréable visuellement.
Il ne faut pas oublier de prendre en compte dans l'équation les boîtes finales du jeu et le coût d'impression et de reliure des rapports de soutenance.

D'après nos estimations, notre projet aura un coût approximatif de 40 euros, ce qui est raisonnable (si on ne value pas notre temps de travail).

Nachos Rumble n'est pas destiné à être le prochain *Fortnite*, car nous n'envisageons évidemment pas de le distribuer à grande échelle. Nous avons cependant trouvé intéressant d'aborder ce point. À noter que dans cette éventualité, de nombreuses fonctionnalités devraient être ajoutées (boutique en jeu, personnalisation...), et de nouveaux champs seraient pris en compte dans le budget (publication dans une bibliothèque de jeu, publicités).

Notre jeu tend à toucher un public large, ce qui pourrait grandement être favorisé par le fait d'en faire un **Free To Play**. Le modèle économique serait alors basé sur la vente de cosmétiques dans le jeu (la solution de la publicité étant une atteinte trop importante à l'expérience du joueur). C'est une option adoptée par de plus en plus de jeux, même si elle n'est vraiment efficace que sur les jeux les plus populaires.

3 Résumé de la première soutenance

Le but était d'avoir une base de jeu que l'on pourrait étoffer par la suite. Nous nous sommes donc concentrés sur la jouabilité avec les fonctionnalités suivantes : aspect visuel et aspect jouabilité.

3.1 Multijoueur

Pour gérer les interactions multijoueur de notre jeu, nous avons décidé de se servir du service de Photon, qui permet gratuitement d'utiliser des serveurs avec jusqu''a 20 joueurs en simultanés. Photon est très bien intégré à Unity et comme première tâche du projet, nous avons réalisé des menus très simples permettant aux joueurs se connectant au jeu de créer des salles de jeu ou de les rejoindre. Seule la personne ayant créé la salle de jeu est ensuite en capacité de lancer la partie. Nous avons implémenté la gestion de migration d'hôte (si le joueur ayant la possibilité de lancer la partie quitte la salle ou le jeu) qui redonne les droits sur la salle 'a un autre joueur.

3.2 Mécanique de tir

Nous avons implémenté les bases des mécaniques du tir du jeu. C'est à dire que lorsque le joueur a une arme, le jeu instancie le modèle 3D de l'arme, avec laquelle il peut tirer, et donc infliger des dégâts aux joueurs de l'équipe ennemie. Le tir en lui-même est généré par un script. Il s'agit d'un "Raycast", c'est à dire une ligne droite instantanée, partant du centre de la caméra du joueur. C'est un moyen simple et très utilisé d'implémenter un système de tir dans un jeu, cela évite par exemple plusieurs problèmes lorsqu'on essaie de faire partir le tir du canon de l'arme. Nos armes sont toutes gérées par leurs propres scripts contenant des informations sur leurs fonctionnements (munitions, dégâts, cadence de tir, etc), ce qui nous permet facilement d'implémenter de nouvelles armes. Il s'agit d'ailleurs de quelque chose que nous essayons de généraliser à l'ensemble du projet : faire des scripts qui permettent de gérer d'autres scripts dans le but de rendre la réutilisation du code dans le futur bien plus simple, rapide et sans prise de tête.

3.3 Gestion de la partie et des joueurs

Les informations qui sont importantes au déroulement de chaque partie doivent être stockées quelque part. C'est pourquoi nous avons utilisé des scripts pour gérer les rooms ainsi que pour chaque joueur d'une partie, qui permettent ici de stocker les données qui leurs sont liées. Le script RoomManager va être généré une seule fois par partie : à son lancement. C'est ce script qui va permettre d'instancier tout

ce qui se passe durant la partie, comme par exemple de générer un leaderboard (tableau de classement des joueurs selon leurs performances), gérer l'arrêt de la partie, etc. C'est aussi ce script qui instancie pour chaque joueur présent dans la partie leur scripts PlayerManager. Ce script est celui qui va stocker les données de chaque joueur : ses points de vie, son état (en train de courir, marcher, sauter, tomber, etc), mais aussi gérer le respawn (les réapparitions sur la map après être mort), et très important : recevoir et envoyer des informations aux autres joueurs (dégâts, etc). Il va aussi instancier tous les prefabs liés aux joueurs : leurs armes, les colliers utilisés dans la physique et les déplacements, etc.

3.4 Caméra

La caméra des joueurs de Nachos Rumble est évidemment placée au niveau des yeux du personnage joué. Nous avons implémenté la gestion de la caméra, chose basique mais qui doit être bien faite étant donné que l'ensemble du jeu repose sur cette simple fonctionnalité (notamment les mécaniques de tir). La sensibilité de la souris pour faire bouger la vision du joueur est modifiable dans le menu des réglages du jeu. Il était aussi important de limiter la vue du joueur verticalement, c'est à dire de ne pas pouvoir regarder plus bas encore quand on regarde droit vers le sol, ou plus haut quand on regarde droit vers le ciel. Il a aussi fallu gérer le fait de ne plus affecter la caméra lorsque le joueur est dans un menu (par exemple celui de réglages, on ne veut pas que la vue du joueur bouge lorsque celui-ci choisit ses options).

3.5 Physique et mouvement

Nous avons réfléchi et travaillé à la gestion des menus dans l'interface de notre jeu. Pour cela, nous avons tout d'abord créé un script spécialement pour gérer les menus, ce qui nous permet d'ajouter par la suite de nouveaux menus très facilement et rapidement. Pour le moment, les seuls menus que nous avons implémentés sont ceux qui permettent aux joueurs de jouer en ligne en multijoueur, c'est à dire les menus permettant de créer une room, rejoindre une room. Nous avons aussi créé le menu des réglages de sensibilité de la souris et de réglages graphiques, dans lequel il est possible pour le joueur de modifier l'aspect visuel de son jeu (les ombres, la qualité des textures, le niveau de détail, etc), mais aussi de changer le nombre d'images générées chaque seconde par le jeu, ou bien encore la résolution du jeu, s'il est en plein écran ou non, etc. Nous avons réalisé un site web statique à l'aide de PhpStorm en HTML et CSS. Ne connaissant rien à ces langages de programmation, nous avons dû entreprendre des recherches afin d'améliorer peu à peu le site. Pour l'instant, il est assez basique : il comprend les infos du groupe mais aussi un lien vers le PDF de notre premier rapport de soutenance.

4 Résumé de la seconde soutenance

Nos objectifs de travail pour cette seconde soutenance étaient de faire clairement un jeu jouable, avec ses axes de développement. Nous nous sommes focalisés sur la mise en place de compteurs, et statistiques pour pouvoir désigner un vainqueur. Nous avions plusieurs idées, équipe contre équipe, capture the flag etc, mais nous avons décidé de faire un Deathmatch, où le but est de tuer le plus d'adversaire en solo. Dans l'état actuel du jeu, en plus d'avoir des mécaniques de déplacements et un système de tir, nous pouvons finir une partie en la gagnant ou en la perdant. Nous avons également créé un système de noms et de compte. En résumé : Après avoir implanté les bases du jeu lors de la première soutenance, nous avons ajouté le mode de jeu, et son système de victoire ou de défaite.

4.1 Intelligence Artificielle

Paul a commencé à implémenter l'IA sur la base des recherches de Théo. Pour rappel, le but de l'IA que nous voulions créer dans le jeu est de jouer le rôle d'un ennemi en le cherchant, puis en lui tirant dessus à une certaine distance. Nous avons donc dû trouver un moyen de communiquer à l'IA à la fois, où elle peut se déplacer, et dans quelle destination elle doit se rendre.

Le principe, lors de cette soutenance, était de déterminer pour chaque IA une cible (un joueur) en fonction de la distance qui les sépare.
Après avoir créé les principes de base de l'IA, il nous manquait l'implémentation sur les serveurs multijoueur Photon, que nous utilisons pour notre jeu. Nous avons rencontré plusieurs problèmes quant à la synchronisation des IA entre les clients, mais aussi quant à leur capacité à prendre des dégâts, et donc mourir et réapparaitre.

4.2 Modèle des joueurs

Au niveau du modèle 3D des personnages, Sam à également fait une simple maquette de ce à quoi pouvait ressembler pour la troisième soutenance, cependant on hésitait encore sur le personnage, pourquoi pas mettre quelque chose qui a une forme très lisse et basique qui ferait vraiment Cartoon, et low poly. Mais peu importe le style exact, nous comptions mettre des signes distinctifs pour faire penser à ce côté hispanique (Sombrero, moustache, poncho...).

4.3 Animations

Les animations n'avaient pas encore commencé lors de cette seconde soutenance. Étant donné que nous n'avions pas encore choisi de modèle 3D de personnage. Nous avons tout de même listé les différentes animations possibles . Nous n'avions toujours pas d'animations disponibles en jeu, notamment au niveau des armes.

4.4 Audio

Au niveau des bruitages, Sam avait affiné la sélection et assemblé un dossier avec des bruitages précis (coup de feu, bruit de pas...). Cependant nous n'avions encore rien implémenté. Mais pour cette soutenance ce n'était pas la chose sur laquelle on a décidé de passer le plus de temps. Au niveau des musiques, c'était également la même chose.

4.5 Menu

Les menus lors de cette soutenance étaient encore très simples. Nous avions seulement commencé à créer des boutons pour rendre le jeu plus visuel, ce qui n'était vraiment pas le cas avant. L'identité visuelle du jeu était encore en pleine confection par Sam et Paul.

4.6 Données et statistique des joueurs

Pour notre jeu, nous voulions que chaque joueur de notre jeu ait un ensemble de données qui lui est propre, comprenant ses statistiques. Un nouvel Input champ de saisie dans le menu principale qui permet aux joueurs de pouvoir avoir leur propre pseudo dans le jeu. Si, dans le cas où le joueur ne saisit pas de pseudo, un pseudo aléatoire lui sera attribué. Pour que les utilisateurs puissent connaître "l'identité" de chaque joueur dans la partie on a implémenté un texte au-dessus de chaque personnage qui correspond à leurs pseudo respectifs.
Afin de faciliter la connexion de nos utilisateurs, on a veillé à qu'ils puissent rester connectés sur leurs profils même après avoir quitté le jeu.

4.7 Gestion de la partie

Afin d'obtenir une fin à notre jeu, nous avons dû réfléchir par quel moyen une partie pouvait se terminer. Nous avions donc opté pour une fin de partie basée sur le nombre d'éliminations. Autrement dit, le premier à un certain nombre d'élimination gagne et la partie s'arrête.

4.8 Scoreboard

La première version du scoreboard (tableau des scores) lors de cette soutenance était encore très simple et pas forcément opérationnelle à cent pourcent. Il permettait uniquement de voir le nombre d'élimination de chaque joueur dans la partie mais il comportait tout de même de nombreux bugs.

4.9 ATH / HUD

Nous avions créé notre première version de l'ATH afin de rendre le jeu plus agréable et jouable. Il comporte : la vie du joueur, son pseudo, son nombre d'élimination, son nombre d'éliminations, l'icon de l'arme en main, et également le crosshair qui lui permet de visé.
La barre de vie était encore en cours de création.

4.10 Site internet

Pour la seconde soutenance, le site web a été rendu dynamique grâce à un système de compte. Grâce à une base de données (PHPMyAdmin), il est désormais possible de s'enregistrer sur le site en entrant son nom d'utilisateur, email et mot de passe, l'ID servant simplement de repère à l'intérieur de la base de données, pour pouvoir se connecter.

4.11 Système d'arme et de tir

Nous avons légèrement amélioré le système d'armes. En effet, après avoir effectué des tests sur le jeu, il est apparu que la précision des armes (parfaite et égale sur chaque arme) n'était pas très réaliste. Nous avons donc implémenté un système de précision par arme, ainsi qu'une modification lorsque le personnage est en mouvement ou qu'il court.

4.12 Avancement et retard de la seconde soutenance

Beaucoup de travail a été fourni pour créer un jeu fonctionnel et complet lors de cette seconde soutenance, avec de multiples fonctionnalités que nous améliorerons toutes au fil du temps.
Il est possible pour tout joueur de créer sa propre partie et de la lancer, ou bien de rejoindre une partie créée par un autre joueur. Contrairement à la première soutenance, ces parties ont maintenant un but et une fin, ainsi qu'un gagnant.
La partie intelligence artificielle du jeu était en plein développement et était presque fonctionnelle en multijoueur.

En vue de la dernière soutenance, nous voulions enfin rendre le jeu agréable visuellement, en implémentant les animations des joueurs, de tir, de rechargement, mais aussi finir de rendre belle et amusante le map de notre jeu, et pour finir rajouter tous les effets sonores.

5 Explication du jeu

Après avoir lancé le jeu, il faut rentrer un nom d'utilisateur, ce nom sera conservé et vous aurez par la suite des statistiques.
Pour donner suite à cela, vous pouvez cliquer sur "rejoindre une partie" (Find Room) où "créer une partie" (Create Room) si vous créez votre propre partie, vous devrez lui donner un nom, choisir le nombre d'IA (Intelligence Artificielle) et la lancer lorsque vous le souhaitez.

Vous allez alors apparaître sur la carte, pour vous déplacer vous pouvez utiliser les touches *Z Q S D* et *Espace* pour sauter. Vous pouvez courir en maintenant la touche Shift en même temps que vous vous déplacez. Pour changer d'arme il suffit d'appuyer sur le 1 pour le fusil d'assaut 2 pour le pistolet 3 pour les maracas (oui oui), 4 pour le sniper. Le but du jeu est de tuer 20 ennemis. Pour cela rien de plus simple : viser un ennemi et tirez lui dessus avec votre clic gauche. À noter que si vous le souhaitez-vous pouvez également viser avec l'AK-47 et le Sniper en appuyant une fois sur clic droit. Lorsque vous mourrez, vous réapparaissez automatiquement quelque part sur la carte. Le premier à 20 éliminations gagne la partie !
Si vous voulez modifier certains paramètres (résolution, sensibilité de votre souris, volume du jeu etc. . .). Vous avez seulement à faire *Echap* et à modifier. Attention, durant la modification de vos paramètres, vous restez en partie et vous pouvez vous faire tuer !

6 Développement du projet

Dans cette partie, nous allons détailler point par point, comment nous avons géré chaque aspect technique de ce projet. Nous détaillons aussi les problèmes que nous avons rencontré, comment nous avons su les gérer, et par quels moyens.

6.1 Multijoueur

Pour la partie multijoueur de notre projet, qui constitue une très grande partie du jeu, nous avons choisi d'utiliser la bibliothèque *Photon*. Après avoir fait plusieurs recherches sur quelle bibliothèques choisir, qu'est-ce qui avait été utilisé par les groupes des années précédentes, etc, nous avons hésité entre *Mirror* et *Photon*, puis nous sommes partis sur Photon, qui avait l'air d'être plus simple d'utilisation et adapté à notre jeu.

Photon a été très simple d'utilisation quant à l'instantiation de rooms (les parties), ce qui a permis de rapidement s'occuper du multijoueur en jeu. Là encore, Photon nous a permis de développer certains aspects techniques comme sur le *score board* présent dans le jeu, car cette bibliothèque possède un grand nombre de fonction d'*event*. Il était donc moins compliqué de gérer les cas où un joueur quitte la partie, quand un joueur arrive dans la partie, etc.
Photon permet aussi d'associer à chaque joueur dans la partie, des *CustomProperties*, ce qui nous a permis de synchroniser notamment les armes, le nombre d'éliminations et de mort de chaque joueur, etc.

```
Frequently called  0+1 usages  Paul Berthelot +1
public override void OnPlayerPropertiesUpdate(Player targetPlayer, Hashtable changedProps)
{
    if (!PV.IsMine || gameEnded)
        return;

    if (changedProps.ContainsKey("Kills"))
    {
        if ((int) changedProps["Kills"] >= scoreToWin)
        {
            gameEnded = true;
            GameWinner = targetPlayer;
            PV.RPC( methodName: "RPC_EndGame", RpcTarget.All, GameWinner);
        }
    }
}
```

Figure 5: Fonction d'event fournie par Photon

Une particuliarité de Photon est les *Remote Pun Callbacks*, les RPC, qui permettent d'appeler une certaine fonction sur tout les clients connectés à la partie. Là encore nous nous en sommes beaucoup servi, car elle permet d'envoyer tout type d'information sérialisable par le biais de Photon. Cela s'est révélé utile en particulier pour synchroniser les sons entre les joueurs, mais aussi pour le système de tir ou encore synchronisé la fin de partie.

6.2 Interface

6.2.1 Menus

Les menus du jeu constituent le premier élément sur lequel un joueur arrive en démarrant Nachos Rumble.
Il a donc fallu s'assurer qu'ils soient simples et pratiques, tout en étant visuellement beaux pour laisser une première bonne impression au joueur. C'est d'ailleurs pour cela que nous avons totalement retravaillé les menus que nous avions durant les première et deuxième soutenance pour créer ceux actuels.

Figure 6: Menus du jeu

Les menus représentent à la fois des éléments d'UI (*User Interface*) à gérer, mais dans notre jeu, ils ont aussi le rôle de créer et lancer des parties avec Photon. En effet : les menus du jeu sont très simple mais efficaces, ils permettent, lorsqu'on lance le jeu, de choisir entre créer une *room* (une partie), ou en rejoindre une. Dans le menu de recherche de room, on peut voir toutes les parties actives dans lesquelles

se trouvent des joueurs pour ainsi les rejoindre. Dans le menu de création de room, le joueur a la possibilité de tapper le nom qu'il souhaite pour la partie. Enfin dans le menu de la room (lorsqu'on est entré dans une room existante ou qu'on en a créé une), on peut voir la liste des joueurs dans la partie, et si on est le *master* de la room, il est possible de lancer la partie ainsi que de modifier le nombre d'intelligence artificielle que l'on souhaite instancier durant la partie.

Notre inspiration de départ pour les menus du jeu était celle vu en *figure 4*. Nous ne nous sommes pas penché sur la personnalisation des modèles des joueurs, donc leur affichage dans les menus nous a semblé peu utile. Finalement, nous avons préféré opter pour une vue cinématique de la map de Nachos Rumble comme il est fait dans bon nombre de jeu. Cela permet de rendre les menus dynamiques ainsi que de mettre l'eau à la bouche au joueur qui n'a pas encore joué sur la map.

Les menus possèdent aussi des éléments montrant les stats globales du joueur: le nombre d'élimiations qu'il a réalisé en tout, le nombre de fois où il est mort, ainsi que le ratio éliminations/morts.
Le joueur a aussi la possibilité de changer son pseudo, qui sera visible par les autres joueurs durant la partie.
Enfin, nous avons rajouté une musique dans les menus pour rendre le tout un peu plus dynamique.

Paul a créé tous les menus sauf les parties comprenant les statistiques du joueur et la possibilité de changer son pseudo, et les réglages sur la musique.

6.2.2 Interface en jeu

Le HUD (Head-Up Display) est simplement l'affichage des différentes informations sur l'écran du joueur. Par exemple, la barre de vie fait partie du HUD. Dans un jeu du type FPS, un HUD est important car il donne les informations primordiales au joueur Nous avons décidé de rester sur une interface assez basique, rien d'extravagant qui pourrait déranger le joueur pendant sa partie.

Barre de vie

Figure 7: Capture d'écran de la barre de vie en jeu

L'information principale d'un jeu FPS est de savoir, les points de vie dont 'il nous reste, effectivement il est probable qu'un joueur qui a peu d'HP (Heath point = Points de vie) ne va pas courir partout, mais plutôt se cacher et jouer intelligemment. Kilian a donc évidemment implanté un indicateur de vie, de base elle était simplement un compteur de 1 à 100, et lorsque ce nombre était à 0, nous mourrions.

Puis nous avons ajouté un aspect plus intéressant avec une barre de vie, qui est verte quand nous avons beaucoup de vie, et qui devient de plus en plus rouge lorsque nous prenons des dégâts. C'est un effet visuel simple mais important. Car pendant un combat, le joueur a simplement besoin de jeter un coup d'œil pour savoir s'il a beaucoup de vie ou pas. Mais après concertation nous avons quand même décidé de laisser le chiffre pour avoir une indication plus précise si nécessaire. Pour l'aspect Mexique et fun, à la place d'un simple cœur, nous avons mis un petit taco. Bien que moins compréhensible au premier regard, nous pensons que ça vaut le coup. Après discussion, Paul a décidé que les points de vie ennemis ne seraient pas visibles au-dessus de leur tête. Car cela avantagerait trop les personnes qui viennent de réapparaitre. Ils pourraient adapter leur mode de jeu, par exemple courir avec les fusils d'assaut pour finir de tuer une personne qui a peu de points de vie.

Scoreboard

Il est disponible en appuyant sur la touche TAB, il est évident que c'est un élément primordial au jeu, pour savoir si on est en train de dominer une partie, ou au contraire de perdre, pour savoir si un ennemi est sur le point de gagner et pour savoir quel ennemi éviter.

Dans ce menu, il y a plusieurs sections, c'est une sorte de tableau. Tout à gauche évidement, le classement. À noter que plus un joueur est haut, mieux il est classé. Puis dans la deuxième section, le nom du joueur concerné, et à droite toutes les informations qui nous intéressent sur lui, son nombre d'éliminations et son nombre de morts. Étant donné que ce menu est activable en partie, Il a été fait transparent de manière à tout de même voir la partie en cours, même en regardant les statistiques. Pour le rendre fonctionnel, Paul et Kilian se sont servi des fonctions d'évènement fourni par photon, ce qui a permis de modifier les valeurs statistiques affichées à l'écran, cela permet de facilement gérer lorsqu'il y a une élimination ou une mort pour l'ajouter au joueur. Une source de bug a été que lorsqu'un joueur quitte une partie, ses statistiques sont conservées, mais encore une fois, grâce à photon, cela a été résolu assez rapidement. De même manière qu'on ajoute les statistiques à un joueur lorsqu'il rejoint la partie. À noter que pour voir ses statistiques personnelles, pas besoin de regarder le tableau des scores, nous avons mis les statistiques individuelles directement sur l'écran du joueur.

Figure 8: Icon de l'HUD représentant l'AK47

Figure 9: Curseur de souris représentant un nachos

En plus de ses deux principales Features, nous avons implémenté deux petites options supplémentaires, lorsque nous avons une arme dans la main le logo de celle-ci s'affiche, c'est n'est qu'un détail mais il est présent dans tous les FPS. Nous avons également rajouté un petit nacho en tant que pointeur de la souris, c'est vraiment un détail, mais c'est ce genre de petites choses qui font que le jeu est sympathique visuellement et que le joueur est un peu plus immergé dans l'ambiance du jeu.

Le *Kill Feed* est la partie d'UI située en haut à droite de l'écran durant la partie. Elle permet de voir à tout les joueurs de la partie quand un joueur en a éliminé un

Figure 10: Kill feed en action

autre, avec quelle arme, etc.
Cette fois-ci ce sont les *RPC* (Remote Pun Callbacks) qui ont permis de créer cette fonctionnalité. Les dégâts donnés à un joueur passant par des RPC, lorsqu'un joueur meurt, il déclenchera là aussi un appel de RPC correspondant aux informations de l'événement (quel joueur est mort, tué par qui, avec quel arme), permettant d'instancier un prefab dans le Kill Feed.
C'est Paul qui a créé le Kill Feed.

Le *Connection Log* est l'élément d'UI situé en haut à gauche de l'écran, sous les statistiques d'élimination et de mort. Dès qu'un joueur rejoint ou quitte la partie, cette information est affichée sur l'écran du joueur, accompagnée de l'heure précise, pour être le plus au courant possible.

Pour ces deux derniers éléments d'UI, nous nous sommes servi d'un système de Queue, pour gérer le cas où beaucoup d'informations sont à gérer en même temps afin d'éviter un affichage soit buggé, soit moche. Les éléments du Kill Feed et du Connection Log disparaissent au bout d'un certain temps, mais si d'autres éléments attendent dans la Queue qui les gère, alors ils disparaissent pour leur laisser place.

Aussi, il est possible durant la partie d'appuyer sur la touche *Echap* pour faire apparaître le menu des réglages. Il est possible pour le joueur de changer la qualité des graphismes du jeu via des presets (*Ultra*, *Low*, etc), de changer la résolution de son jeu, de le mettre en plein écran. On peut aussi modifier le volume sonore du jeu et la sensibilité de sa souris. Enfin il est possible de quitter totalement le jeu ou de quitter la room pour revenir au menu d'accueil du jeu.
C'est Paul qui a créé l'interface en jeu, avec Kilian qui a lui fait tout ce qui est en rapport direct avec le système de sauvegarde des profils qu'il a aussi créé.

6.3 Audio

6.3.1 Composition et recherche de l'audio: Sam

Sam a sélectionné les différents bruitages, ils sont importants car même si 80% des informations passent par le visuel, il n'en reste pas moins que les parties sont largement influencées par le son (se retourner en entendant un ennemi tirer ou marcher, etc...).

Il ne faut pas que le fusil d'assaut ait le même bruitage que le fusil de sniper par exemple. Car le joueur ne comprendrait pas ce qu'il est en train d'arriver sur la carte, est-ce qu'un ennemi est proche, ou est-ce que ce sont deux autres ennemis qui se tirent dessus entre eux. Nous avons donc rajouté un son pour chaque type d'arme, tous différents et facilement assimilables.
Au niveau de la musique, j'ai moi-même composé une simple mélodie du type hispanique, elle est calme et douce et à la fois quelque peu stéréotypée pour permettre de reconnaître directement l'ambiance du jeu auquel nous jouons.
Avoir une musique indépendante nous tenait à cœur car cela permet d'identifier le jeu, nous sommes les seuls à avoir cette musique, et l'entendre veut dire que nous sommes sur *Nachos Rumble*, et pas sur n'importe quel jeu.

6.3.2 Implémentation de l'audio : Kilian

L'implémentation de l'audio a commencé par la musique réalisée par Sam dans le menu. Je n'ai pas rencontré vraiment de difficulté pour cette partie, il faut tout simplement lancer la musique quand l'utilisateur est dans le menu puis l'arrêter dès qu'il lance ou rejoint une partie. Pour ajouter plus de possibilité à l'utilisateur et de lui offrir plus de libertés, j'ai ajouté un onglet dans le menu qui permet de gérer la musique, c'est-à-dire : un slider qui permet de baisser ou augmenter le volume et une case à cocher qui permet de couper ou non la musique.

Après m'être occupé de la musique, j'ai implémenté le bruitage des armes à feu. Chaque arme a son propre bruitage afin de bien les distinguer. J'ai donc ajouté pour chacune d'elles un composant *"Audio Source"* afin que le son provienne bien de l'arme en question et non d'un tout autre objet. Le son du coup de feu va donc s'activer aussitôt que le joueur tire.

6.3.3 Implémentation multijoueur des sons : Paul

L'implémentation des sons telle que l'a fait Kilian marchait très bien, mais elle ne permettait pas aux différents joueurs dans la partie d'entendre les autres joueurs tirer, ni réapparaître, etc, car tout ces sons étaient produits localement.

Pour résoudre ce problème, je me suis encore une fois servi de ces bien pratiques RPC (Remote Pun Callbacks) fournis par Photon. En effet j'ai créé pour chacun des clients un *Audio Manager* qui permet à la fois de recevoir et d'envoyer des sons par le biais de Photon.

Les objets que sont les sons dans Unity d'étant pas sérialisable, j'ai donc dû sélectionner quels attributs du son envoyer par le biais d'une fonction RPC. Cela m'a pris un peu de temps, car la documentation Unity sur les *Audio Source* est plutôt vague, mais j'ai réussi au final à mettre en place cet Audio Manager.

Les sons sont donc joués en local lorsqu'on tire par exemple, et une fonction RPC se charge d'envoyer ce sont aux autres clients (uniquement si on souhaite qu'il soit entendu par les autres joueurs de la room évidemment). Le sons reçu par les autres joueurs est donc spatialisé et 3D, c'est à dire qu'il n'est pas mono (on peut savoir d'où provient le son en l'entendant) et qu'il est produit au même endroit sur la map.

Kilian :

Si on pense que c'est un détail c'est en fait tout le contraire, dans un jeu de tir il est primordial de comprendre d'où on se fait tirer dessus pour pouvoir riposter.

Les bruits de tir ne sont pas les seuls, j'ai aussi rajouté des bruits de pas pour entendre les ennemis se déplacer, mais aussi un son de réapparition pour ne pas se faire surprendre lorsqu'un joueur réapparaît dans notre dos par exemple.

On aurait voulu essayer de varier les sons en fonction du type de sol sur lequel on marche par exemple quand on marche sur du sable ou sur le toit d'un bâtiment le son n'aurait pas été le même. Mais nous n'avons pas eu le temps car il aurait fallu retravailler sur la map afin de distinguer chaque type de sol.

Pour finir sur l'audio, pour accentuer une l'aspect un peu compétitif du mode de jeu, j'ai ajouté une musique de fin de partie qui se lance quand un joueur à gagné la partie en même temps que l'écran de fin de partie.

6.4 Système d'armes et de tir

Quant aux armes, le système de tir a été développé par Paul, Kilian et Théo. Nous sommes en fait revenus de multiples fois sur l'implémentation des objets dans nos scripts. En effet, il est important de commencer le développement d'une fonctionnalité de telle manière à ce qu'elle soit facilement adaptable pour des potentielles modifications dans le futur. Malheureusement ce n'était pas le cas pour la première implémentation des armes que nous avions effectué dans notre jeu car elle ne permettait pas d'ajouter le , c'est pourquoi nous avons repris le système d'arme presque de zéro avant la troisième soutenance pour pouvoir améliorer notre système de tir.

En l'état final du jeu, notre système d'armes est assez complet. Le joueur apparaît sur la map avec plusieurs armes qu'il peut équiper. Les armes sont : l'AK47, le Desert Eagle, le Dragunov (un sniper) et les fameuses maracas, qui servent d'armes de mêlée.
Tout les tirs des armes sont calculés par un raycast qui part du centre de la caméra. Si ce raycast touche une IA / joueur, il lui infligera des dégats.
Nous avons implémenté un système de dispersion pour rendre les armes plus réalistes et le jeu plus agréable. Le raycast est en fait dévié de manière aléatoire dans un certain rayon.
Nous avons aussi implémenté dans le jeu des modifcations de vitesse du déplacement lorsque le joueur est en train de viser, et inversement, nous avons réduit la précision des armes des joueurs lorsque ce dernier est en train de courir.

Au tout début du développement du projet, ous avons commencé l'implémentation des objets en tant qu'*Items*, en C# une classe parente des armes. Cela est très pratique car cela nous permettra dans le futur, si l'envie nous vient, d'implémenter des items autre que des armes (exemple : un nachos à manger pour reprendre des points de vie) de manière très simple.

6.5 Système de déplacement

Paul :

Le système de déplacement de notre jeu est un autre point clé de son gameplay, en effet beaucoup de jeux se démarquent de par leur système de déplacement original et fluide. Le déplacement permet de changer totalement le gameplay du jeu auquel on joue, et peut rendre un jeu incroyable comme il peut le rendre très ennuyeux.

C'est pourquoi nous avons essayé de faire un système de déplacement le plus propre et fluide possible, pour que cela ne soit pas quelque chose qui porte atteinte au rythme du jeu.
Au début du développement de notre jeu, les mécaniques de physique et de déplacement du jeu étaient fonctionnelles, mais elles ne nous ont pas pour autant satisfaites. Elles étaient en fait trop simples, et elles rendaient le jeu mou. Et surtout un des problèmes que l'on a rencontré était le fait que le mouvement était saccadé. On a longtemps cru que cela venait de Photon et d'une potentielle latence due au serveur. C'était en fait un problème lié à l'appel des fonctions d'événements d'Unity (Update et FixedUpdate).
Au final, notre système de mouvement utilise les lois de newton pour calculer à chaque instant la vitesse de chute d'un objet. Quant aux déplacements du joueur, nous avons utilisé un *CharacterController*, car l'implémentation était la plus facile et surout celle produisant le moins de bug.

C'est Paul qui s'est occupé de rework le système de déplacement.

6.6 Identité visuelle du projet

En tant que groupe pour ce projet de second semestre, nous nous devions d'avoir une identité visuelle qui tient la route, et il en va de même pour notre jeu. C'est pour cela que nous pris un peu de temps pour s'occuper de la création de logo pour le groupe et pour le jeu.

Le nom de notre groupe : *Los Successos* (qui ne veut malheureusement rien dire en espagnol), est directement lié à l'ambiance de notre jeu : le Mexique. C'est pour quoi nous avons choisi de créer un premier logo très simple, en 2D, comportant la forme d'une moustache et d'un sombrero, deux attributs que tout bon mexicain possède évidemment. Nous avons choisi les couleurs grises et jaunes car c'était assez original, et c'est d'ailleurs les mêmes couleur que l'on retrouve sur le site de notre projet, car nous voulions avoir une identité visuelle reconnaissable.
Nous avons aussi fait un autre logo, encore plus simple, que nous avons d'ailleurs

affiché au démarrage du jeu. C'est un logo simple mais efficace, et nous avons eu du mal à choisir entre les deux, donc nous avons décidé de garder les deux.

Figure 11: Premier logo *Los Successos*

Figure 12: Second logo *Los Successos*

Quant au jeu en lui-même, nous avons choisi de faire quelque-chose d'assez cartoon et simple, mais cette fois-ci dans l'ambiance mexicaine. C'est pourquoi nous avons opté pour un logo en simuli-3D, dans un dégradé jaunde-orange. La police d'écriture est droite et épaisse, et le détail qui a son importance : la lettre *N* du logo a un sombrero qui la couvre.

Nous avons fait plusieurs versions du logo, notamment pour l'icon de notre jeu, et d'autres pour la jaquette de la boîte de jeu, etc.

Paul s'est occupé des logos sauf le premier (*figure ?*) qui a été fait par Sam.

Figure 13: Variante du logo

Figure 14: Icon du jeu

Figure 15: Logo Nachos Rumble complet

6.7 Site internet

Théo :

Pour expliquer le cheminement qui nous a emmené à concevoir le site web tel qu'il l'est aujourd'hui, j'ai jugé bon de résumer ce qui avait été fait pour les deux premières soutenances.

Première soutenance :

N'ayant aucune connaissance en site web, je décide de m'orienter vers un site statique, en HTML / CSS, pour commencer. Le but ici est d'avoir un site sur lequel publier nos avancés et renseigner les gens sur la nature du projet. Le site, codé sur PHPStorm, comprend donc trois onglets : une page d'accueil sur laquelle sont présents les informations du jeu : les modes de jeu disponibles ; une deuxième page recensant des informations sur le groupe : nom, photo ; puis une troisième page avec les liens téléchargeables au format PDF : le cahier des charges et le premier rapport de soutenance.

Pour l'instant le site n'est pas disponible sur internet, nous n'avons pas encore jugé pertinent de l'y mettre étant donné qu'il ne contient presque pas de contenu. Nous pensons le mettre en ligne seulement pour la soutenance finale.

Je n'ai pas rencontré beaucoup de difficultés sur cette partie. Il existe en effet de très bons sites internet bien documentés et surtout bien détaillés sur la prise en main de PHPStorm.

Deuxième soutenance :

Etant plus familier du langage HTML, j'ai enrichi, et surtout embelli, le site avec de nouvelles informations, des photos et surtout une plus belle fonte. On a ensuite eu l'idée d'implémenter un système de compte permettant le partage et la consultation des données du jeu de sorte à rendre celui-ci plus intéressant. Pour cela, il faudrait lier le jeu a une base de données, elle-même relié au compte sur le site pour permettre le transfert des donnés. Il a donc fallu rendre le site dynamique, grâce au langage PHP. Le bon point était que c'était directement possible sur PHPStorm. Malheureusement, pour vois les avancés du site, il n'était plus possible de l'ouvrir sur un navigateur web comme cela était possible pour les sites statiques. Il a donc fallu installer Xampp. C'est une application permettant de mettre en place un serveur web local pour pouvoir accéder PHPMyAdmin : une application permettant la création d'une base de données pour pouvoir stocker les informations récupérées du jeu.

Figure 16: Base de donnée du site

C'est dans cette partie que j'ai rencontré le plus de difficultés car le passage de site statique à site dynamique entraine bon nombre de changements. Il a fallu apprendre à utiliser Xampp et à gérer une base de données (comprendre ce qui est en local, ce qui est en hébergement distant, comment passer de l'un a l'autre etc.). Mais c'est surtout la prise en main du langage PHP (trop peu documentée sur internet à mon gout) qui a été compliquée. En effet, il existe énormément de balises différentes, et donc plusieurs manières d'aller d'un point A à un point B et

le langage n'étant pas très intuitif, on peut facilement se perdre entre les méthodes. Cette partie m'a aussi bien aidé en langue car la majorité des blogs que j'ai visité était écrit en anglais.

Troisième soutenance :

Pour le rendu final, il a fallu encore améliorer l'aspect visuel du site. Il a aussi fallu ajouter au site un moyen permettant de télécharger le jeu : un "installeur". Pour ce faire, j'ai utilisé *InnoSetUp Controller*, une application permettant de créer des installeurs. Le point fort de cette application est que la création de l'installeur est automatique, il suffit juste de bien indiquer les dossiers donc on a besoin (Le cœur du jeu en exécutable et les fichiers annexes). J'ai donc pu ajouter au site un lien permettant, le plus facilement possible, d'installer le jeu sur son ordinateur pour y jouer.

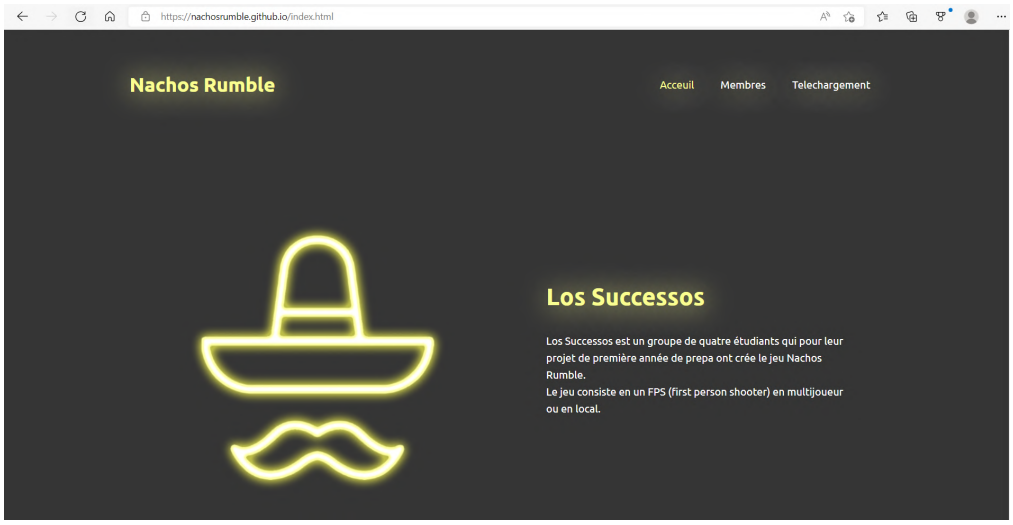

Figure 17: Site du projet (*https://nachosrumble.github.io*)

Pour ce qui est du système de compte, des changements de plan ont eu lieux et il a été décidé d'abandonné ce projet (du retard ayant été pris dans l'avancement du jeu, nous avons préféré nous focalise sur la jouabilité plutôt que sur les stats). J'ai ensuite mis le site en ligne à l'aide de GitHub. Pour ce faire, j'ai dû uploader tous les fichiers sur un projet (nommé 'NachosRumble') afin de les mettre en ligne sur GitHub.

J'ai perdu pas mal de temps sur la partie de l'installeur car celui-ci ne peut pas déchiffrer des dossiers de fichiers. En fait, il faut créer un autre dossier du même nom dans lequel on met l'actuel dossier contenant les fichiers annexes pour

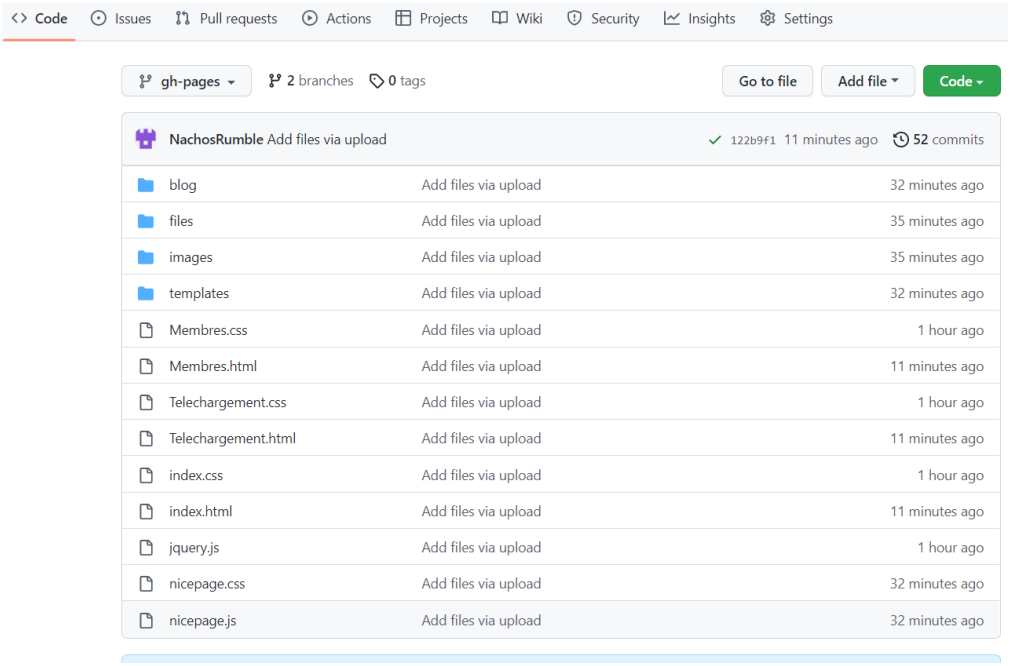

Figure 18: Projet github pour le site

que l'installeur comprenne ce qu'on lui envoie car il ne déchiffre que les dossiers de dossiers de fichier (de 'SetUp\jeu.ddl' à 'SetUp\SetUp\jeu.ddl')...

6.8 Modélisation 3D

Sam :

6.8.1 Map

Dans le cahier des charges, l'idée de base était de créer une seule map, dans le thème Mexique. Et très rapidement, j'ai commencé à faire mes recherches sur comment faire une map intéressante et compétitive, avec différentes positions, différentes manières de jouer (en restant caché ou en courant partout) et sur le papier la carte me paraissait parfaite avec l'idée de ce que l'on se faisait de notre jeu.

Et c'est dans cette optique là que j'ai continué dans mes démarches de création de carte pour la deuxième soutenance, sans m'aider d'aucune *assets*, en partant de zéro. J'ai donc modélisé la carte prévue. J'ai dû apprendre à utiliser de manière efficace *Blender* pour monter cette carte de toute pièce, et j'ai passé énormément de temps à faire, refaire, et optimiser les différents objets de la carte. Lorsque je l'ai envoyé aux autres membres du groupe, elle fut validée. La conception de la carte me paraissait donc en grande partie terminée.

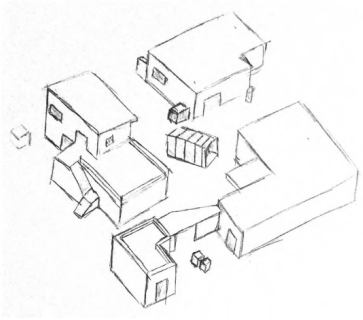

Figure 19: Croquis de la carte (1ère soutenance)

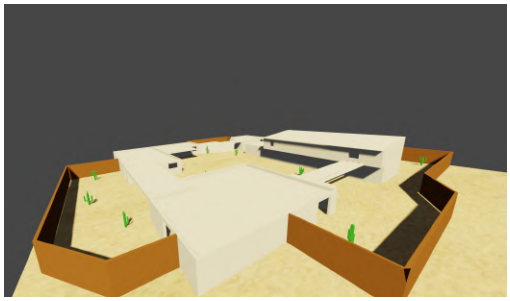

Figure 20: Map issue de la deuxième soutenance

Malheureusement, nous avons eu quelques problèmes de collisions lors de la première tentative d'importation en jeu, et même après de nombreux fixs, la map restait complètement injouable en partie. La conclusion pour la deuxième soutenance fut donc que nous devions la retravailler pour fixer ses bugs. Et c'est ce que nous avons fait, j'ai modifié les objets de la carte pour qu'ils soient indépendants et que les *collider* (les objets générant les collisions) fonctionne. Et nous avons alors réussi à mettre cette carte en jeu.
Cependant après plusieurs tests, alors que je pensais le travail sur la carte terminée, les autres membres du groupe m'ont fait remonté que malheureusement,

cette carte ne correspondait finalement pas à ce que nous attendions du jeu, pas assez mexicaine, trop compliquée, alors que nous recherchons quelque chose d'ouvert et d'amusant. Au début frustré que ma carte ne suffise pas, j'ai ensuite repris le travail et ai suivi les consignes du groupe à la lettre, carte plus grande, moins d'espace intérieurs, plus réaliste, encore plus esprit Mexique, etc. J'ai donc commencé par faire un désert simple avec des endroits plats pour rajouter par la suite des constructions.

J'ai ensuite rajouté une ville basique au centre du décor. On ne peut rentrer dans les maisons, mais on peut monter sur les différents toits, de manière à avoir une vision de jeu plus grande, et un jeu rapide, qui va vite avec beaucoup de combats en peu de temps. Un nouveau problème s'est alors posé, la taille de la carte, elle me paraissait beaucoup trop grande et vide, une ville et des arbres c'est joli, mais ça ne fait pas beaucoup d'endroits ou être à couvert. Pour régler ce soucis, j'ai cherché de nombreux *assets* disponible gratuitement dans l'esprit désert et low poly, et j'ai donc rajouté notamment un chariot de mineur. Je l'ai mis dans un coin de la carte (proche d'une falaise pour que cela ait l'air plus réaliste). Je lui ai donné une couleur de bois sombre, comme ancien, je l'ai à moitié enterré dans le sable. J'ai ensuite disposé des pépites d'or et des rubis dans le chariot, mais également dans le sol à côté, toujours pour ce côté réaliste. J'ai rajouté les outils du mineur, eux aussi plantés dans le sable, ou posé sur le chariot, comme abandonnés. Et pour finir j'ai donné une couleur réaliste et métalliques aux pépites, bien jaune pour l'or, et rose pour le rubis. Le but ici est de rentrer dans le cliché pour que cela soit rapidement assimilable par le joueur.

J'ai par la suite rajouté un cimetière, puis une petite oasis asséchée, un mirador, etc. Et j'ai soigné chaque détail, de la même manière que pour le chariot du mineur. C'est ainsi que petit à petit la carte s'est formée, la base était un simple désert auquel j'ai rajouté tout un univers tournant autour d'une ambiance mexicaine.

Pour arriver à un résultat aussi satisfaisant j'ai modifié certains assets, notamment au niveau des couleurs pour avoir quelque chose qui correspond à la carte et au décor. Cette carte fut alors validée à l'unanimité par les membres du groupe. Il est vrai qu'aujourd'hui avec le recul je pense que cette carte correspond beaucoup mieux au jeu, et j'en suis bien plus fier.

Nous avons rencontré quelques difficultés sur de nombreux assets importés car certains avait déjà des réglages d'éclairage bien à eux (*lightmaps* et *UV maps*), ce qui a posé un problème lorsque nous avons ajouté de la lumière directement sur Unity. La résolution de ce problème est simple, mais pas évidente au premier coup

Figure 21: Capture d'écran de la map durant le développement

d'œil. Après avoir effectué nos recherches nous avons trouvé comment régler ce soucis sans difficulté. Encore une fois le problème n'était pas de régler le bug en lui-même, mais de savoir d'où il venait et pourquoi.

6.8.2 Point de réapparition

Une des (nombreuses) choses qui influe également sur la rapidité de jeu et sur la manière de jouer, sont sur les points de *spawn* (de réapparition).
Effectivement, apparaître au centre de la carte, en haut d'un bâtiment, dans un endroit où il n'y a pas de protections, etc, peut drastiquement modifier le cours de la partie. La carte étant déjà étudiée pour avoir un combat rapide sans longue phase de recherche, Sam a choisi les points de spawn en fonction des différents obstacles.
Il a été décidé de faire apparaître les joueurs dans des endroits calmes, où ils ne peuvent se faire directement tuer. Cela pour principalement deux raisons, la première n'est évidemment que pour bien comprendre la situation dans laquelle nous sommes, pour avoir le temps d'analyser l'endroit de la carte ou nous sommes apparus, et pour pouvoir adapter son style de jeu (jouer en utilisant un fusil de sniper car nous sommes en hauteur, ou au contraire passer au fusil d'assaut car nous sommes dans la ville...) apparaître caché nous laisse le temps de faire ses choix. La deuxième raison est celle du spawn-kill (se faire tuer en boucle par un ennemi lorsque l'on réapparaît). C'est quelque chose de très frustrant lorsque cela nous arrive. Pour éviter cela, les points d'apparitions sont cachés, éloignés les uns des autres et tous protégés par au moins un obstacle.

Figure 22: Exemple de bug causé par une mauvaise UV map

Une dernière chose qui a été prisée en compte, est le fait que, lorsque l'on apparaît, il y a un léger bruit. Pour encore une fois diminuer les phases de recherches, les points de spawn ont été placés pour que si quelqu'un est dans le secteur, il entende la personne apparaître. C'est un habile mélange entre, laisser le temps, et ne pas en perdre inutilement à courir en cherchant des ennemis.

6.8.3 Personnage

L'objectif initial quant au modèle 3D du personnage que l'on incarne dans le jeu était d'avoir un personnage très basique, et même assez mignon pour rendre le jeu drôle et agréable visuellement. Un personnage trop réaliste aurait fait tâche dans un jeu *fun* comme le nôtre, nous nous sommes donc dirigés sur un personnage aux proportions plus *cartoon*, avec des clichés mexicains. Partant de ce postulat j'ai trouvé un modèle 3D qui me paraissait réunir toute ces caractéristiques.

Mais nous nous sommes rapidement rendu compte qu'il était trop complet, trop dans son univers, avec notamment l'arme en guise de main et l'écran à la place du visage. J'ai donc commencé, notamment pour prendre l'outil *Blender* en main, à faire moi-même un personnage.
Pour la première soutenance, nous avions donc un modèle 3D en construction. Il s'agissait uniquement d'une ébauche, deux bras, deux jambes, une tête, un som-

brero et une moustache. Je l'ai ensuite peaufiné jusqu'à arriver à une ébauche assez satisfaisante au yeux des membres du groupe. Cependant, même si cette création m'a apprise beaucoup de choses, notamment dans l'utilisation de l'outil de modélisation *Blender*, le personnage créé était trop basique, pas assez atypique, trop sérieux, pas assez cliché.

Figure 23: Inspiration initiale

Figure 24: Second modèle 3D

J'ai donc commencé à chercher un asset 3D de personnage qui pourrait être éditable pour arriver à un cliché d'homme mexicain, et j'ai assez rapidement trouvé mon bonheur. J'ai donc modifié le personnage en question pour en faire un vrai mexicain, et en plus de cela, un personnage drôle qui divertira les joueurs. Je lui ai rajouté un gros sombrero, et des moustaches énormes, cela ne laisse pas de place pour son visage, mais c'est le but. Le rendu final est satisfaisant, on a un personnage hispanique un peu en colère, et qui est agréable visuellement.

Figure 25: Modèle 3D final des personnages du jeu

6.8.4 Armes

Assez rapidement, nous avons trouvé un pack d'armes qui correspondait à ce que nous voulions. Dès la première soutenance nous avions les armes définitives. Il est important d'avoir des armes propres et reconnaissables au premier regards pour que le joueur ne soit pas perdu lors de la prise en main du jeu. Nous avons donc implanté au début les armes les plus basiques, Pistolet (*Desert Eagle*), Fusil D'assaut (*Ak-47*), sniper (*Dragunov*). L'avantage est que, après avoir implémenté le système de tir, nous pouvons avoir différent style de jeu. Jouer en courant partout avec son arme automatique et tirer constamment, ou au contraire jouer de manière plus posée avec son sniper qui lui est plus précis. Le sniper est un cas particulier car on doit viser avec lui, nous avons donc dû rajouter un viseur. C'est un des principe de base dans les *FPS*, les fusils à lunettes sont plus précis et il suffit de viser pour tirer de loin. C'est donc ce que nous avons fait.

De plus nous avons rajouté une arme de mêlée, au corps-à-corps, mais au lieu de rajouter un simple couteau, nous avons choisi de rajouter des maracas pour plus d'amusements et moins de "violence". Cela rajoute encore une fois une nouvelle façon de jouer, jouer silencieusement pour surprendre son adversaire dans le dos.

6.9 Animations

Kilian :

Les animations ont été quelque chose de très nouveau pour moi lors de cette dernière période de projet. Elles sont très intéressantes à faire mais peuvent être tout autant agaçantes. Le principe de base pour faire des animations (celles que j'ai utilisées), est d'enregistrer la position d'un objet d'un moment à un autre et de faire la transition entre les positions avec ou non des conditions.

6.9.1 Animation de mouvement

La première animation que j'ai faite a été celle de la respiration. Autrement dit, j'ai réalisé un mouvement sur les armes qui vont se mettre à bouger légèrement de haut en bas dans le but de simuler la respiration de notre personnage pour ajouter du réalisme.
Ensuite, une animation quand on prend des dégâts : un léger mouvement de caméra pour que les joueurs s'aperçoivent qu'ils ont pris des dégâts sans pour autant que ce soit dérangeant pour jouer.

6.9.2 Animation d'armes

Pour les armes j'ai commencé par les animations de visé sûr les armes.
Ces animation se font en plusieurs étapes:

Tout d'abord, enregistrer les positions de l'arme par rapport au joueur lors d'une période qui dure quelques millisecondes. La première position est celle classique, quand le joueur tient son arme, et la seconde position est celle quand il vise.

Figure 26: Arme en animation de repos

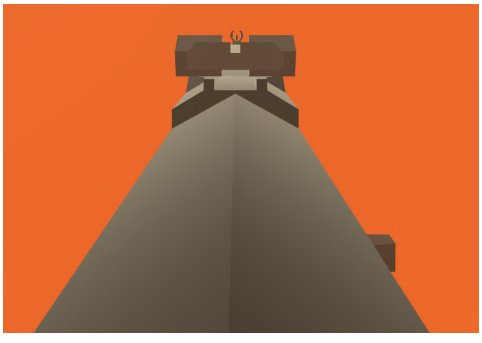

Figure 27: Arme en animation de visée

Cette transition qui va créer l'animation va se lancer sous une certaine condition: s'active sur le clic droit de la souris. Quand le clic droit va être pressé, une

animation va se créer entre la position de départ et la position de visée.

On a souhaité instaurer la visée que sûr certaines armes, cependant diverses armes comme le fusil de sniper possède une visée un peu plus particulière. En effet, un fusil de sniper possède une lunette, donc pour créer cette illusion, en plus de l'animation, j'ai ajouté une image qui s'affiche juste après l'animation. De plus, à ne pas oublier, la visée permet l'augmentation de la distance de vision afin de voir plus loin.

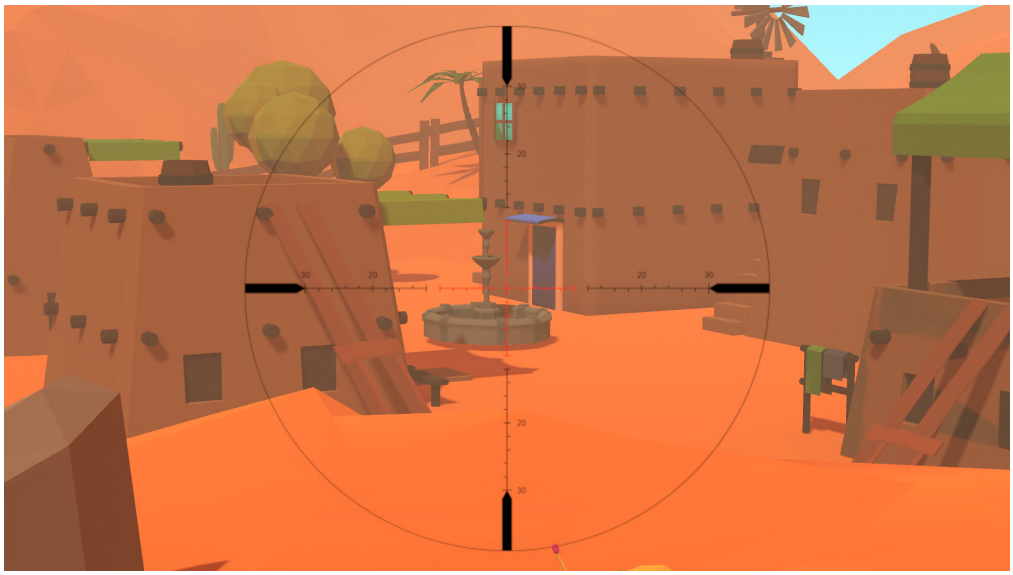

Figure 28: Sniper en animation de visée

6.10 Intelligence Artificielle

L'intelligence artificielle de notre jeu a d'abord été envisagée comme un joueur géré par l'IA, qui ferait tout comme un joueur normal, c'est à dire : viser un ennemi, tirer, se déplacer, etc.
On s'est vite aperçu que ceci serait réalisable, mais que cela ne serait pas une intelligence artificielle des plus intéressante pour le gameplay de notre jeu. Nous avons donc décidé de simplifier son fonctionnement. En fait, le comportement de l'IA se rapproche de celui d'un joueur dit 'infecté' dans les modes de jeu de type infection (exemple : le mode de jeu 'infection' dans le jeu *Halo*, où lorsqu'un joueur est infecté, son but est d'infecter les autres joueurs en courant vers eux).

Le fonctionnement de l'intelligence artificielle reste proche de celui imaginé par Théo au début du projet : l'IA apparaît tel un joueur, puis se dirige vers le joueur le plus proche dans le but de le 'frapper'.

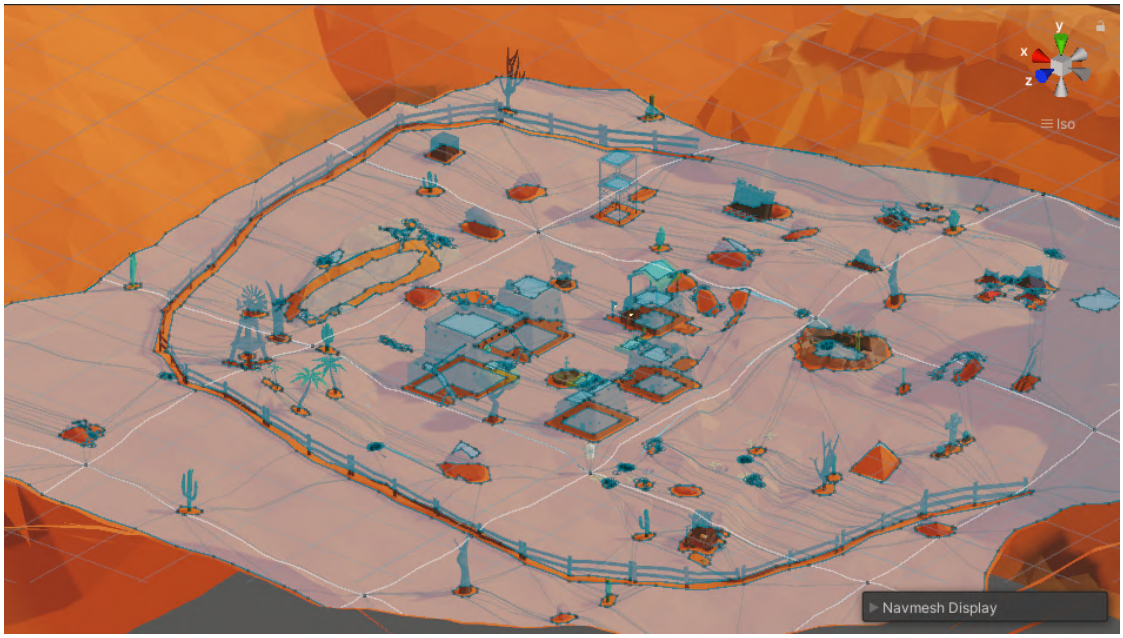

Figure 29: NavMesh de la map du jeu

Le déplacement sur la map de l'IA a été géré grâce au *NavMesh* d'Unity. Il s'agit d'une mesh qui correspond aux endroits sur le décor sur lequel l'IA pourra se déplacer.
L'algorithme lié à l'IA est simple : elle va déterminer à un intervalle de temps donné (nous avons choisi ici un intervalle de 0,1 seconde) le joueur le plus proche.

Son déplacement est ensuite géré par un algorithme de *PathFinding* en lien avec la NavMesh, ce qui fait que l'IA se déplacera sur la map en fesant le chemin le plus court vers sa cible : le joueur le plus proche. Enfin lorsque l'IA se trouve à la distance requise pour 'frapper' le joueur cible, elle le fait, lui infligeant ainsi des dégâts

La partie la plus compliquée du développement de l'intelligence artificielle a été le fait que nous voulions pouvoir jouer avec l'IA durant les parties en ligne.
Il a donc fallu gérer leur instantiation dans Photon, et repenser totalement le système de dégats qui ne peut marcher de la même manière que celui des dégats entre joueurs 'réels', étant donné qu'une IA n'est pas un joueur en lui-même.
Pour cela, seul le joueur 'master' de la partie a la possibilité de choisir le nombre d'IAs qui seront créées durant la partie avant que celle-ci ne commence. Les IA lui 'appartiennent' d'un point du vue de Photon, et il a donc aussi fallu gérer le cas où le joueur master quittait la partie, pour transmettre les IA au nouveau master afin qu'elle reste dans la partie.

La seule arme que possède l'IA est donc l'arme de mêlée de notre jeu : les fameux *maracas*.
Visuellement, l'intelligence artificielle est similaire aux joueurs de la partie à un détail près : la couleur de sa salopette est différente. Cela permet pour les joueurs d'identifier si un ennemi est un joueur réel ou une IA.
Il a aussi fallu implémenter les animations déjà existantes sur les joueurs cette fois-ci sur les IA, pour bien comprendre que l'IA ne court droit dessus lorsqu'on est sa cible.

C'est Paul qui s'est occupé de développer l'intelligence artificielle.

6.11 Gestion du travail du groupe

6.11.1 Organisation du travail

L'organisation des tâches a changé depuis le cahier des charges, étant donné que nous avons préféré nous mettre d'accord en choisissant chacun les tâches sur lesquelles on était plus à l'aise pour travailler.
Voici un tableau représentant la répartition des tâches.

Tâche	Paul	Sam	Kilian	Théo
Graphisme				
Map		X		
Armes		X		
Personnages		X		
Animations	X		X	
Audio				
Musiques		X		
Bruitages	X	X	X	
Réseau				
Multijoueur	X			
IA				
Fonctionnement	X			X
Mécanique de jeu				
Physique	X	X	X	
Caméra	X		X	X
Mécanique de tir	X		X	X
Interface				
Menu	X		X	
ATH	X		X	
Site web				
Création du site internet				X

6.11.2 Soutenances et rapports

Comme ce document, tous les rapports de soutenances ont dû être écrits en *LaTeX*. Pour créer ces documents, nous avons utilisé Overleaf, qui est un site qui propose un éditeur de document LaTeX et qui est assez pratique.
C'est Paul qui s'est chargé d'écrire les rapports de soutenance en LaTeX avec l'aide des autres membres du groupe.

Quant aux soutenances en elle même, qui font pleinement partie du projet, nous nous sommes réunies à chaque fois en amont pour préparer au mieux notre oral, ainsi qu'un *Powerpoint* qui permet de mettre des images sur nos mots durant la présentation.

7 Ressenti

Théo :

Ce projet a été mon deuxième grand projet après le TPE au lycée. Je me pensais donc prêt à travailler en équipe mais le projet étant plus complexe et radicalement différent, cela n'a pas été aussi facile que prévu. Il a fallu en effet apprendre à communiquer sur ses avancées, comme faire en sorte que les autres comprennent nos codes par exemple. Il a aussi fallu apprendre à manipuler *Git* pour mettre en commun nos différents travaux.
J'ai également découvert les mécaniques de création des sites web avec les langages HTML, CSS et PHP ainsi que les diverses applications en lien(PHPStorm, Xapmm). Il a aussi fallu que j'apprenne à gérer une base de données avec PHPMyAdmin. J'ai aussi appris à coder sur Unity (en C).
Au final, ce projet représente une partie importante de notre année et je suis assez fier de voir où on en est arrivés. Et, même si je n'ai pas travaillé sur toutes les parties, j'ai quand même pu découvrir au travers de mes amis l'utilisation de pleins d'autres applications comme Blender par exemple. En conclusion, le fait de travailler en mode projet m'a vraiment permis de comprendre ou du moins de découvrir le quotidien d'ingénieur dans le domaine de l'informatique.

Sam :

Travailler sur ce projet a été un vrai plaisir, tout d'abord pour la liberté que celui-ci apporte, notre seule limite concrète pour rajouter des choses dans le jeu, était notre imagination. Mettre un instrument de musique en guise de couteau. *Why Not ?*
J'ai également apprécié le fait de travailler en groupe, l'ambiance au sein de l'équipe *Los Sucessos* fut vraiment motivante, et voir les avancements du projet jour après jours, cela fut réellement stimulant. L'apprentissage dû à ce projet fut également appréciable, contrairement à une fonction basique en C#, les bugs ne sont pas simplement des *IndexOutOfRangeException* mais de réels problèmes complexes à résoudre. Réussir à manier Blender a également été un grand enjeu pour moi qui me suis chargé, entre autres, de la modélisation. En conclusion ce projet a été pour moi une belle aventure et surtout, une belle occasion d'apprendre.

Paul :

J'ai beaucoup apprécié le temps passer à travailler sur ce projet. Même si j'y ai parfois laissé mon sommeil, ce projet m'a vraiment premis d'apprendre comment marche Unity et C#. Mais en plus de ça, j'ai appris en étant chef de groupe, ce qui est nouveau pour moi. J'ai donc du apprendre à travailler mieux en groupe, mieux communiquer pour que les choses se fassent plus simplement, et quoi de mieux que de le faire dans le groupe des *Successos*!
J'ai eu de nombreux moments où une tâche qui était censé être rapide s'est avérée plus dure que prévue, de nombreuses fois où je ne trouvais aucune documentation à mes problèmes sur internet, de nombreuses fois où j'ai recommencé le développemment d'une fonctionnalité à zéro, mais j'ai bizarrement beaucoup apprécié travailler sur ce projet, de par la liberté qu'il m'a offert dans mes choix.
Ce projet a été une belle aventure en soit et je suis très content du résultat final de nos nombreuses heures de travail.

Kilian :

Le développement de ce projet à été très enrichissant en tout point, notamment du fait qu'il est quasiment libre et qu'on a très peu de restrictions ce qui permet de laisser place à notre imagination et de créer en quelque sorte notre jeu "idéal". J'ai également pu mettre en œuvre mes connaissances en programmation, que j'avais acquises par le passé, mais aussi à l'école, dans un projet spécifique avec une portée plus large que le travail pratique hebdomadaire. J'ai également pu élargir ces connaissances car ce projet nécessitait beaucoup de recherches. Ce projet m'a permis de me former à un aspect qui me semble fondamental dans mon développement, à savoir le passage de l'idée au code. En effet, cette année, tous les travaux pratiques étaient très ciblés et ressemblaient plus à du remplissage qu'à une profonde réflexion, ce qui est normal pour les débutants. J'ai donc réussi à m'entraîner à conceptualiser une idée en quelque chose de concret qui peut être fait en code.

C'est avec beaucoup de travail et de communication qu'aujourd'hui je suis heureux d'avoir fini ce projet. J'ai vraiment apprécié travail avec mon groupe car même si j'étais affecté à certaines tâches, j'ai pu apprendre et découvrir leurs travaux. J'ai essentiellement codé en C# pour notre jeu mais j'ai pu apprendre comment créer un environnement de jeu, ajouter des animations ou encore intégrer des sons au sein du jeu.

8 Conclusion

Ce projet a été pour nous une réelle source d'apprentissage. Apprendre à travailler sur un projet informatique commun est très formateur et nous sommes fiers du résultat obtenu.
Nous nous sommes écartés en quelques points de notre cahier des charges initial, mais le jeu est très largement similaire à ce que nous avions imaginé.
Le développement du projet a pu s'avérer compliqué lorsque les dates de rendu approchaient, mais l'entente au sein du groupe n'aurait pu être meilleure. Ce projet marque la fin de notre première année au sein de l'EPITA, et l'équipe *Los Successos* est très heureuse d'avoir partagé sa passion à travers ce projet !