Ultimate Fighting Chickenship

Rapport final

Tempête group

May 2019

charlie.brosse@epita.fr
philippe.lefebvre@epita.fr
david.horozian@epita.fr

Table des matières

1 Introduction

1.1 Les racines de Tempête

Le commencement de Tempête

Tempête est un studio de développement de jeux vidéo né fin 2018. Il s'agit l'union de trois étudiants d'EPITA de la même classe INT2 : Philippe, David et Charlie. Nous sommes avant tout un groupe d'amis qui aime imaginer et créer. Créer des choses nous fait plaisir et nous amuse. En temps que membres de Tempête nous sommes toujours curieux de voir dans quelle mesure nous pouvons atteindre nos objectifs atypiques et réaliser nos idées particulières. Nous sommes notre premier public et notre public le plus dévoué. C'est la raison pour laquelle nous croyons que le processus de création fait partie du plaisir. Ainsi, l'objectif du projet est de partager nos jeux avec un public plus large. Le projet représente, à notre avis, une occasion d'améliorer considérablement nos compétences et nos connaissances et quitter notre zone de confort.

À travers cette présentation, nous vous présentons notre premier projet : *Ultimate Fighting Chickenship* .

Tempête Group

Tempête possède trois membres. Chacun est polyvalent, créatif et déterminé : ces qualités permettent la bonne écoute des idées et les pensées de tout le monde à propos du projet. De plus, chacun apporte des spécialités uniques au studio. Les membres sont jeunes pourtant ils sont disposés à s'améliorer en programmation et en travail d'équipe.

1.2 The idea of the game

Ultimate Fighting Chickenship est un jeu PC de combat où deux personnages - des poulets - se combattent jusqu'à ce que la vie de l'un des deux personnages tombe à 0. Le gameplay que l'on vise est similaire

à celui de ***Street Fighter*** et est relativement proche des jeux de combat où deux personnages se font face. Cependant, nous voulons permettre aux joueurs de se déplacer, d'attaquer, et de sauter plus vite que les dans les jeux du meme genre : notre but est d'améliorer les mouvements et l'agilité. Cela serait notre contribution à l'évolution des jeux vidéo de combat.

Un aspect majeur du jeu est qu'il vise à être amusant et plutôt stupide. L'idée de prendre des poulets pour se combattre illustre cet aspect avec brio. Comme nous voulons que notre jeu sois accès sur le multijoueur : nous voulons être sur que notre jeu sois amusant et relaxant pour un groupe d'amis. On ne devrait pas être surpris par les aspects surprenants et les choix artistiques étonnants de notre jeu.

2 But du jeu

Comme dit plus haut, Ultimate Fighting Chickenship est un jeu de 1 vs 1, orienté pour une utilisation multijoueur, que ce soit en LAN ou en ligne.

Avec un environnement amusant and des un gameplay fluide et dynamique, deux joueurs se battent en utilisant des attaques basiques tels que des coups de poings et des boules de feu, et un large éventail de capacités de mouvements pour mettre K-O l'autre.

3 Critique du cahier des charges

3.1 introduction sur les jeux de combats

Le premier jeu de combat est ***Heavyweight Champ*** *(1976)*. Ce jeu pour deux a été sorti pour les arcades. Il a une simple palette de couleur noire et blanche et possède deux personnages affichés sur un plan à deux dimensions. Les joueurs ont un éventail très limités d'attaques à leur disposition : chaque personnage peut faire un coup de poing "haut" et un coup de poing "bas". Le but du jeu est de mettre K-O l'adversaire.

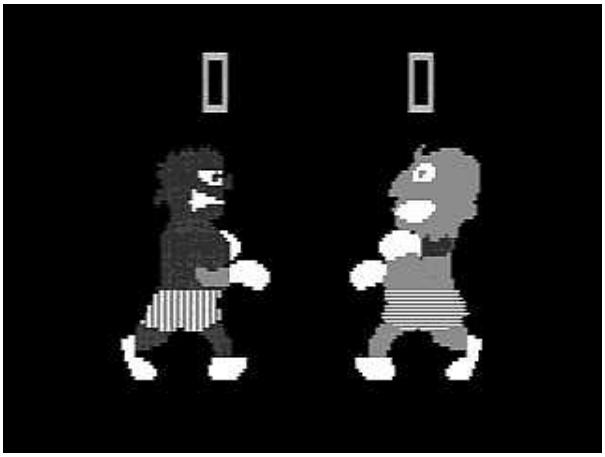

FIGURE 1 – ***Heavyweight Champ***, Le premier jeu de combat

Le genre tel que nous le connaissons aujourd'hui est le résultat de l'évolution des jeux de combats. Le premier jeu à introduire des mouvements plus fluides fut ***Karate Champ*** *(1984)*, marquant ainsi le commencement de la popularisation des jeux de combats dans les arcades. Plus tard, c'est ***Yie Ar Kung-Fu*** *(1985)* qui introduit une différentiation entre les personnages et leurs attaques, leurs style de combats

Une autre étape importante dans l'évolution des jeux de combat est l'apparition de la série des ***Street Fighter***, et surtout de ***Street Fighter II*** *(1991)*, qui est très vite devenu populaire grâce à sa grande variété de

personnages, de mouvements et d'attaques : chaque personnage du jeu possède jusqu'à 30 mouvements et combos qui lui sont propres. Le jeu a aussi gagné de la popularité grâce à sa large disponibilité dans les salles d'arcades ainsi que sur les consoles de salon.

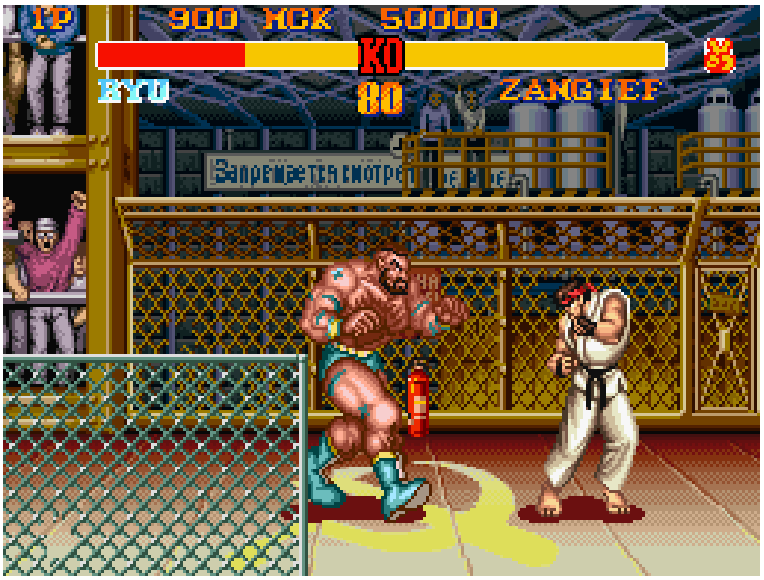

FIGURE 2 – ***Street Fighter II***, le jeu de combat le plus populaire

Au fur et à mesure que la technologie évoluait, des nouveaux jeux similaires à Street Fighter sont apparus, le plus important dans le cadre de ce projet étant :

— ***Mortal Kombat*** *(1995)*, pour son usage de violence et d'attaques spéciales (*"fatalities"*),

— ***Super Smash Bros.*** *(1999)*, Pour son gameplay très dynamique et son mode jeu basé sur des pourcentages.

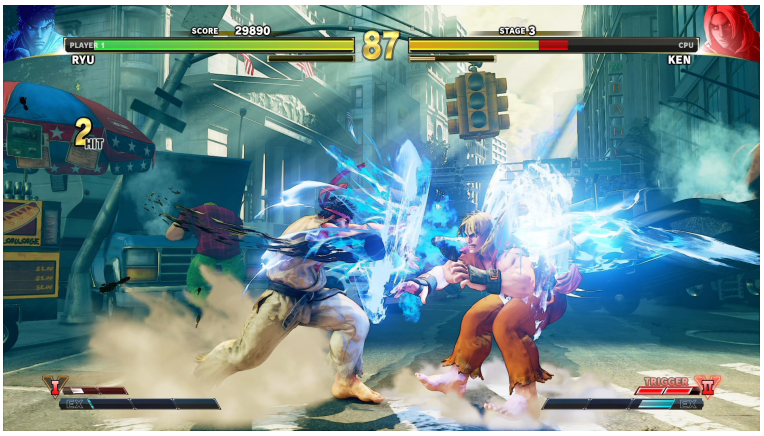

FIGURE 3 – ***Street Fighter V***, Le plus récent jeu de la saga

Critique

Avec de la patience, de l'inspiration par rapport aux grands jeu du genre et beaucoup de documentation, nous sommes fiers d'avoir créé non seulement un jeu qui a les codes d'un jeu de combat, mais aussi un jeu qui a innové.

3.2 Direction artistique et philosophie du projet

La principale ligne directrice que nous souhaitons suivre est de faire en sorte que les visuels restent simples et épurés. Les ressources que nous utiliserons serons composés de couleurs simples avec des touches photo-réalistes. Le but final est d'obtenir un style farfelu, avec des inspirations de ***Lapins Crétins*** mais aussi des films ***Deadpool***. UFC se veut amusant et surprenant. De plus, nous voulons que les sons et les musiques jouent un rôle important dans le plaisir que peux prendre le joueur.

Critique

Jusqu'à la fin du projet, nous avons développé le jeu avec cette philosophie en tête. Cependant, la perte du membre Maxime Mugniot pendant l'année, le développement des personnages 3D et des cartes nous a demandé beaucoup de temps.

Nous n'avons en tout cas pas perdu notre objectif de vue et nous avons continué de créer notre jeu en remodelant nos priorités. Nous avons décidés de ne faire qu'un seul personnage mais de montrer avec celui-ci nos aptitudes et nos capacités en modélisation 3D.

Le gameplay ressemble quasiment à l'identique à ce dont nous avions imaginé à le genèse du projet. Il est fluide et dynamique. De plus, toutes les caractéristiques ont été ajoutées. Bien sûr, n'avoir qu'un personnage ne permet d'avoir tous les différents types de personnages.

Les attaques correspondes également à nos attentes. Ils y a les "Power-up", les attaques de bases, les projectiles et les combos. L'implémentation de ces derniers a été un vrai défi et nous sommes tout de même fiers du résultat.

Moins surprenant, l'aspect auditif du jeu a été la partie la plus gratifiante du projet. L'audio est très important dans un jeu et cela a vraiment apporté de la vie à notre jeu. La pluralité des sons et des voix a été au delà de nos espérances.

3.3 Aspect technique.

Le projet a été réalisé en utilisant les différents logiciels :
- **Blender** : logiciel de modélisation 3D,
- **Unity** : Moteur de jeu et logiciel de développement tout-en-un,
- **Git and GitHub** : plateforme pour le travail collaboratif.
- **Ableton** : Station audionémurique.

Les rapports ainsi que le cahier de charges ont été fait à l'aide de Overleaf. La communication est la clé d'un travail de groupe. C'est pourquoi nous avons aussi créé un serveur Discord.

Pour se documenter, nous avons utilisé les forums Unity, ainsi que des chaînes Youtube telles que Brackeys. D'autres sources telles que Stackoverflow ont parfois été utilisées.

3.4 L'aspect économique

En terme d'économie, nous n'avons pas investi de somme d'argent notables dans le projet, car nous n'avons utilisé que des logiciels que nous possédions au préalables ainsi que des logiciels gratuits. Nous comptons sur le bouche à oreille pour agrandir la communauté de notre jeu.

3.5 Distribution des tâches

	Philippe	**David**	**Charlie**
Mécaniques de jeu			
Mouvements	⊕		✓
Attaques de base		⊕	✓
Combos	✓	⊕	
Design sonore			
Musique	✓	⊕	
Effets sonores	✓		⊕
Voix		⊕	✓
Graphismes			
Design des cartes	⊕	✓	
Design des personnages	⊕		✓
Animations 3D	⊕	✓	
Interface utilisateur		✓	⊕
Autres			
Multijoueur	✓		⊕
Site internet	⊕	✓	
Intelligence artificielle		⊕	✓

✓ pour "responsable", ⊕ pour "suppléant".

3.6 Timeline

Soutenance	**1ère**	**2ème**	**3ème**
Mécaniques de jeu	40%	70%	100%
Mouvements	70%	90%	100%
Attaques de base	25%	60%	100%
Combos	0%	30%	100%
Design sonore	20%	50%	100%
Musiques	30%	60%	100%
Effets sonores	0%	50%	100%
Voix	0%	50%	100%
Graphismes	20%	60%	100%
Design des cartes	50%	75%	100%
Design des personnages	15%	60%	100%
Animations 3D	15%	60%	100%
Interface utilisateur	0%	60%	100%
Autres	40%	70%	100%
Multijoueur	50%	90%	100%
Site internet	30%	60%	100%
Intelligence artificielle	0%	25%	100%

4 Mécaniques de jeu

4.1 Comments

Charlie s'est chargé d'implémenter les mécaniques de jeu dans le projet Unity, recevant à la fois l'aide de Philippe pour le Network et celle de David pour l'assemblage des différents éléments de jeu.

Implémenter les mécaniques de jeu dans le projet fut notre premier contact avec le logiciel Unity et, par conséquent, cette tâche nous prit un long moment avant de rendre quelque chose d'opérationnel. Nous avons eu des difficultés concernant l'apprentissage du C#, en effet la programmation sur Unity est très différente de celle que nous avons l'habitude de pratiquer lors de nos TP.

En effet, programmer les mouvements et les attaques basiques ne fut pas instinctifs mais au contraire très différents de ce que nous avions déjà vu. Cela est en grande partie due à la grande quantité de fonctionnalités que Unity apporte pour la configuration de scènes et l'écriture des scripts.

Ayant eu des difficultés à amorcer le projet, Charlie prit un moment pour réfléchir et se construire une idée précise de

chaque tâche qui lui incombait. C'est de cette façon que lors de la première partie du projet, il s'est établi un bon socle de travail qui lui servit pour ses futures implémentations.

Après s'être familiarisé avec Unity, ses fonctionnalités et la programmation C# sur cette plateforme, Charlie commença le développement du jeu sur la seconde phase du projet avec plus d'assurance et donc moins d'erreurs.

Bien sûr, Charlie avait encore de nombreuses fonctionnalités à découvrir et s'approprier. Par conséquent, il entra dans une phase plus avancée du développement de jeu vidéo.

Après la perte d'un des membres et l'impact que celle-ci a eu sur le travail, nous avions compris que certaines fonctionnalités du jeu serait impossible à mettre en place dans le temps imparti, c'est le cas des attaques ultimes des personnages. Sans se sentir dépassé, nous avons persévéré pour rendre un jeu aussi proche de nos attentes que possible.

Après le réglage de quelques problèmes mineurs, il commença l'ajout de complexité aux mécaniques de jeu. Alternant entre réflexion et code, il cherchait le meilleur moyen d'implémenter chaque mécaniques afin d'optimiser son temps et son programme.

Pourtant, patience et persévérance furent de mise dans ce développement car tout ne marcha pas du premier coup et de nombreuses difficultés sont apparues. Cependant, avec une bonne dose de documentation, la majorité des objectifs furent atteints lors de la deuxième phase.

4.2 Movement- Charlie

1ère soutenance

Pour la première présentation, un joueur de Ultimate Fighting Chickenship était capable d'effectuer deux mouvements différents : le saut (Barre espace) et le déplacement latéral (Touches Q et D).

Afin d'implémenter les mouvements, Charlie créa un script C# nommé Player Controller (voir figure 4). Ce script se charge des lois de la physique s'appliquant sur le joueur comme la gravité et les frottements.

A ce stade les joueurs étaient représentés par deux cubes : un petit représentant la tête et un plus large représentant le torse. En utilisant la classe RigidBody de Unity et en changeant les positions de ceux-ci en fonction des entrées de l'utilisateur : le joueur peut se déplacer librement.

Pour le saut (voir figure 5), Charlie implémenta une variable booléenne vérifiant la position du joueur par rapport au sol : si celui-ci est au sol il peut sauter sinon il ne peut pas. Ce système empêche le joueur de voler.

Tous ces ajouts ont été faits dans un Prefab cependant, l'utilisation d'un tel système a créé un problème : nous devions créér des positions de départs (ou points de spawn). En

effet, au démarrage les deux joueurs commençaient au même endroit. C'est pourquoi nous avons ajouté grâce à la classe Unity Network Start Position : l'hôte du jeu commencera à gauche et le client à droite.

2ème soutenance

En plus des mouvements basiques déjà disponibles, Charlie ajouta un 'dash', la possibilité de faire un double saut ainsi que la rotation des joueurs sur les virages pour la seconde soutenance. Par conséquent le joueur acquit une grande palette de possibilités mouvements lors d'une partie de Ultimate Fighting Chickenship. Tous ces changements furent ajoutés dans le script Player Controller.

A ce stade, le jeu ne fut pas très loin de sa forme finale en termes de possibilités de mouvements. Ces trois nouveaux mouvements furent plus complexes à implémenter que les premiers mouvements tels que les mouvements latéraux.

Le double saut (voir figure 6) fut implémenté en utilisant un compteur de saut en plus de la variable booléenne déjà présente. Chaque fois que le joueur touche le sol le compteur revient à 0. A l'inverse à chaque fois qu'il saute, le compteur est incrémenté de 1. Enfin, le compteur doit rester sous une certaine limite pour que le joueur puisse sauter. C'est pour quoi une condition vérifie que le compteur est bien en dessous

de cette limite (2 pour le double saut). Un autre système aurait été de créer un tableau de booléens contenant un élément par nombre de saut. Cependant, ce système n'aurait pas été optimisé pour un jeu, il aurait demandé trop de conditions et n'aurait pas été adapté.

Une chose importante était de vérifier l'entrée utilisateur non pas dès que la touche fut pressée mais dès qu'elle était 'enfoncée'. Sinon il y a un risque que la touche soit enfoncée trop longtemps et donc que le compteur s'incrémente trop rapidement empêchant le double saut.

De plus, à partir de cette soutenance le joueur se tournait dans la direction vers laquelle il se dirige (voir figure 7). Charlie a ajouté cela en utilisant un booléen portant l'information de la direction prise par le joueur. Alors, pour chaque entrée de mouvement latéral, la rotation est changée si besoin est. Un problème fut que la rotation changeait aussi les axes internes des personnages, c'est pourquoi les vecteurs de mouvements devaient aussi être inversés.

Le dash marche comme une petite téléportation vers n'importe quelle direction. Le dash apporte une nouvelle possibilité mécanique au joueur. Il est assez complexe d'effectuer un dash, cela récompensera les joueurs expérimentés. Enfin, le dash coûte de l'énergie mais nous reviendront à ce système plus tard.

Ces nouvelles capacités de mouvements apporte fluidité et rapidité au gameplay. L'expérience de jeu en résulte plus dynamique.

3ème soutenance

Arrivé à la seconde présentation, nous avions presque totalement fini la création des mouvements. Par conséquent, lors de la dernière phase, nous avons clarifier nos scripts, optimiser l'implémentation et adapté les variables de vitesse et de saut pour respecter nos attentes de dynamique.

Charlie créa aussi une nouvelle capacité de mouvement appelé Protection. En appuyant sur la touche spécifiée, le joueur est capable d'améliorer sa défense. Celui-ci ne pourra ni bouger ni attaquer tant qu'il n'aura pas relacher la touche. Cette capacité apporte une réduction des dégâts encaissés par un pourcentage diminuant au cours du temps (80% puis -15% chaque seconde). Une chose importante est que cette capacité consomme beaucoup d'énergie afin d'éviter que les joueurs ne se transforme en poules mouillées.

L'implémentation d'une telle capacité exigea préparation et réflexion, et cela paya car il n'y eut ni problèmes ni bugs. Charlie montra donc l'expérience acquise pendant ces 6 mois de projet. Plus précisément, cette capacité représente deux variables dans le Script C# Fighter s'occupant des attaques,

une variable booléenne s'activant lorsque la capacité est activée et un nombre flottant représentant le pourcentage de réduction de dégâts. En mettant ce pourcentage dans une variable, Charlie eut la capacité de le faire changer au cours du temps en appelant la fonction Update. Enfin cette capacité supprime les effets d'étourdissement après chaque coup.

4.3 Les attaques - Charlie

1ère soutenance

Pour la première soutenance, Charlie se chargea de l'implémentation des attaques de base : un coup de poing et un coud de pied. La méthode utilisée nécessite un tableau de BoxColliders (boîtes detectant les collisions) ajouté au prefab du joueur ainsi qu'un nouveau script C# Fighter.

Chacune de ces boîtes représente une partie du corps ayant la possibilité de frapper un adversaire. Ce script contenait lors de l'ajout une unique méthode LaunchHit (Lancer l'attaque) prenant en paramètre la partie du corps qui lance le coup. La méthode détecte les collisions avec cette partie du corps (grâce à la méthode OverlapBox) et appelle une fonction complémentaire permettant à ce qui été touché de prendre des dégâts. Cependant, un joueur ne pouvait attaquer qu'en touchant la tête ou le torse, les bras ou les jambes n'étaient là que pour l'attaque. Toucher la tête multipliait les dégâts par 1.5, de plus chaque attaque fait plus ou moins de dégâts.

Bien sûr les entrées furent actualisées dans le script Player Controller : appuyer sur la touche spécifiée permet de lancer un coup de poing ou un coup de pied (K et L).

Par conséquent un script C# s'occupant de prendre les dégâts appelé Health a été créé. Celui-ci instaure un système

de points de vie.

2ème soutenance

Pour répondre à nos attentes en ce qui concerne la deuxième échéance, il y a eu beaucoup de changements dans le système d'attaques. En conservant le système BoxCollider pour les boîtes d'attaque, un nouveau script C# Attack a été créé. Ce script ne contient qu'une structure Attack établissant ses composants et son constructeur. Ainsi, le tableau Box a été remplacé par un tableau d'attaque dans le script Player Controller. Sur l'entrée spécifiée, une attaque est choisie dans ce tableau et appelle la fonction LaunchHit dans le script Fighter. Les Attack ont de multiples variables :

- *name* le nom de l'attack, à ce stade 'punch' ou 'kick'
- *damage* les degâts infligés
- *hitbox* : la partie du corps utilisée
- *knockbackX* et knockbackY : les composants pour la poussée après chaque coup
- *staminacost* : le coût en énergie

Charlie a également implémenté une méthode GetKnocked() appelée avec les composants knockback de l'attaque comme vecteur chaque fois que le joueur est touché. Cette méthode représente la célèbre fonctionnalité mécanique des jeux de combat : le Knock Back ou Push Back. Quand on est touché, on est repoussé dans une certaine direction selon le type

d'attaque.

L'une des nouveautés majeures a été la création d'un nouveau type d'attaques : les capacités. Nous voulions avoir différents types de capacités. À ce stade, Charlie a implémenté le type de capacités Projectile dans un nouveau script Projectile C#. Les projectiles ont presque les mêmes composants que les Atttacks cependant c'est une classe héritée de NetworkBehaviour. Il tient compte de trois nouvelles variables :

- *VelocityX* : la vitesse constante à laquelle le projectile se déplace
- *Prefab* le préfab à afficher dans le jeu
- *SpawnTransform* : la position de départ du projectile par rapport au joueur

Les projectiles ont leur propre Collider qui détectera la collision et appellera la méthode OnCollisionEnter(). Cette méthode diminue la santé dans le script HealthBar et détruit le projectile lorsqu'il touche un joueur. Les projectiles infligent beaucoup plus de dégâts que les attaques de base et offrent l'avantage de la distance. Cependant, ils peuvent être facilement esquivés et ne peuvent pas être utilisés aussi souvent que les coups de poing et les coups de pied. Le script Player Controller a été mis à jour pour disposer d'un tableau Projectile et pour pouvoir appeler la méthode LaunchProjectile dans le script Fighter sur la bonne entrée. Cette méthode a été mise en œuvre en tant que commande. Cela signifie qu'il

s'agit d'un appel d'un client vers le serveur. En effet, pour s'assurer que le projectile jaillit sur chaque client, le serveur doit être en charge. Pour la deuxième soutenance, Ultimate Fighting Chickenship avait une seule attaque de projectile : une boule de feu (voir figure 8).

3ème soutenance

Pour la soutenance finale nous avions pour objectifs d'ajouter des attaques et de créer des combos.

D'abord, nous avons ajouté de nouvelles capacités. Après avoir introduit les Projectiles dans la deuxième présentation, nous avons créé un nouveau type de capacités : les Power-ups. Les power-ups améliorent une caractéristique et permettent d'avoir de meilleures compétences en mêlée ou en combat à distance.

Tous les Power-Ups ont leur propre façon de travailler, certains travaillent dans le temps, d'autres durent peu d'actions. Chaque mise sous tension a sa propre façon d'être effectuée : elle nécessite une combinaison spécifique de touches. De plus, ils coûtent plus d'énergie qu'une attaque de base mais moins qu'un projectile. Nous avons ajouté plusieurs Power-ups, voici quelques exemples.

— Le *Strength Up* : un puissant Power-Up qui augmente les dégâts, il coûte beaucoup d'énergie et dure 5 attaques.
— Le *Speed Up* : dure une certaine distance, facile à activer, ne coûte pas beaucoup d'énergie. Parfait pour surprendre l'adversaire.
— Le *Jump* : autorise le triple saut une fois. Ne coûte pas beaucoup d'énergie.

Les Power-Ups représentent les premiers combos que nous avons implémentés dans le jeu et exigent une certaine expérience pour les réaliser dans le jeu.

Deuxièmement, nous avons ajouté un nouveau Projectile la boule de glace qui ralentit l'adversaire au toucher. Pour la coder, nous vérifions le contact, réduisons la variable vitesse et bloquons le compteur de saut à 2 pour la durée de l'effet pour empêcher le saut.

Enfin, toutes les hitbox ont été remplacées par le modèle 3D implémenté par Philippe.

4.4 Autres mécaniques - Charlie

Système d'énergie

Un tout nouveau système a été mis en place pour la deuxième échéance : le système d'endurance (ou d'énergie). Certaines actions coûtent de l'endurance et ne peuvent être effectuées que si le nombre actuel de points d'endurance est suffisant. Les joueurs commencent à un montant maximum et ne peuvent pas dépasser cette valeur.

Toutes les attaques et capacités ont un coût d'endurance, le dash coûte aussi de l'endurance. L'endurance est mise à jour en appelant une commande prenant un nombre flottant en arguments : la quantité d'endurance restante est diminuée par ce nombre flottant et la barre d'endurance est mise à jour.

En plus de ces appels ponctuels, cette commande était appelée chaque image par l'événement Update() pour augmenter la quantité d'endurance : de cette façon elle agit comme une régénération. La quantité d'endurance régénérée de chaque image est basée sur le nombre d'images par seconde du jeu pour s'assurer que la quantité de régénération est la même au fil du temps dans n'importe quel environnement :

regeneration = Time.deltaTime*10

Interface Utilisateur et Expérience de jeu

Charlie a mis en place une barre d'énergie (voir figures 9 et 10) qui se met à jour au fil du temps. Il l'a intégré à la barre de santé. L'objectif était de les faire travailler en réseau pour que les deux joueurs puissent voir la barre de santé et d'endurance de l'adversaire. Cela a été fait en utilisant l'attribut SyncVar d'Unité permettant la synchronisation de la quantité de santé et d'endurance et de la mise à jour des barres. Les deux barres ont été implémentées manuellement sur le prefab du joueur pour les faire apparaître au-dessus de celui-ci. Un autre script a été créé pour les deux barres afin de s'assurer qu'elles ne tournent pas avec le joueur à chaque virage.

Un script ExitGame a été créé pour revenir au menu principal. Appuyez sur la touche Echap pour quitter le jeu.

Durée de récupération & étourdissement

Pour la dernière échéance, nous voulions créer des vertiges après chaque coup et attaque. En plus de l'endurance, nous voulions créer une durée de récupération pour chaque attaque afin d'empêcher les joueurs de spammer.

Tout d'abord, nous avons mis en place un rechargement très simple : la durée après une attaque pour qu'il soit à nouveau disponible est le temps de l'animation, rien de plus. Après réflexion, Charlie et ses camarades membres de TEMPÊTE ont

pensé au refroidissement dans les jeux de combat. Il leur est venu à l'esprit que la plupart des jeux de combat ne disposaient pas d'un tel système. De plus, nous avons pensé que le système d'endurance était tout à fait suffisant pour empêcher les joueurs de spammer.

En outre, pour éviter d'avoir un gameplay désordonné, Charlie a ajouté la fonction étourdissements. L'étourdissement représente le temps après avoir pris un coup quand on ne peut rien faire. Chaque fois que l'on prend un coup, son contrôleur est gelé pour une durée qui varie en fonction des dégâts subis et du coup que subit l'Attack ou le Projectile. L'ajout de cette fonctionnalité était pour nous un moyen de rendre le jeu plus difficile et plus réaliste. Il aurait été ridicule de voir un jeu où l'on est insensible aux attaques.

5 Design sonore

5.1 Commentaires

Nous savions depuis le départ que le design sonore serait une des parties les plus gratifiantes du projet car cela apporte de la vie à notre jeu. C'est un bon moyen pour apporter de la personnalité au jeu et partager nos émotions, ainsi que faire en sorte que le joueur ressente des émotions. Par chance, l'implémentation des sources audio dans notre jeu fut l'une des plus faciles que nous avons vécu. C'était une des étapes les plus importante pour l'établissement de l'atmosphère burlesque dans notre jeu.

5.2 Musique

1ère soutenance

Philippe a eu le rôle principale en design sonore, et plus particulièrement dans le domaine de la production musicale. Il a utilisé ses précédentes expériences de création musicales pour créer le thème principale de Ultimate Fighting Chickenship. Il a utilisé Ableton (voir figure 11) et le micro de son iPhone pour enregistrer les effets sonores.

Le thème principale est essentiellement une boucle de 1min20 qui se répète. Elle est assez longue pour avoir des variations intéressantes mais assez courte pour assurer une taille de fichier relativement faible.

L'élément principal de la musique est les cris de poules qui se répètent de manière rythmées. Avec cela, des percussions épiques avec des violons rapides s'ajoutent à la musique pour améliorer le rythme. Philippe a ajouté des Charleston rapides pour apporter une atmosphère Hip-Hop à la musique. Une flûte peut aussi être entendue. Dans la seconde partie de la boucle, les percussions épiques sont échangées avec des batteries Trap, qui aident à casser la monotonie de la boucle et renforcent l'aspect Hip-Hop de la musique que nous avoir.

Avec un thème principal qui est pleinement défini, le taux de travail nécessaire pour produire le reste du travail a été grandement réduit.

2ème soutenance

Durant la seconde période, Philippe a continué a fabriqué des sons pour fournir aux joueurs une experience de jeu plus satisfaisante. Il a créé la musique qui allait être entendue durant les combats, la musique qui allait accompagner le mieux la plupart du temps des joueurs passés dans le jeu. Le processus a été le même que pour créer le thème principal : il a créé une boucle de 1min20 à l'aide de Ableton.

L'idée de la musique est venue en écoutant "Praise The Lord" par A$AP Rocky. Le son de flûte fascinant est resté dans nos têtes. Philippe a donc réutilisé ce même son pour

créer son propre "remix" de la chanson originale. Dans la première partie de la boucle, il a utilisé des percussions similaires à la chanson originale pour faire une référence. La deuxième partie de la musique, est complètement différente de la première. Elle contient quasiment exclusivement des sonorités de musique électronique.

3ème soutenance

La troisième et dernière musique du jeu utilise encore une fois les même techniques que les deux premières. Cette musique sera celle entendu en deuxième par le joueur. Elle couvre la deuxième scène du jeu : quand les joueurs se connecteront entre eux pour jouer une partie.

Pour mieux rentrer dans le contexte, Philippe a décidé que la musique serait composée d'un boucle simple. Elle est principalement composée d'éléments rythmiques semblables à ceux du thème principal. La boucle d'une minute, bien que minimaliste, est censée apporter aux joueurs de l'anticipation avant leur match et servir de transition entre le menu principal et la scène de combat. La reverbation sur les percussions font croire aux joueurs qu'ils sont dans une gigantesque arène comme des gladiateurs.

5.3 Effets sonores

1ère soutenance

Nous savions que les effets sonores prendraient place plus tard le développement du jeu parce qu'il est difficile d'attacher des sons à seulement des squelettes de personnages. Sachant cela, nous n'avons pas travaillé sur les effets sonores pendant les premiers mois, mais les membres ont tout de même échangé leurs idées sur comment les sons seraient enregistrés et quels types de sons seraient absolument nécessaires. Depuis le début, nous savions que nous voulions enregistrer des sons de manière amateur et ne les traiter que légèrement.

2ème soutenance

La deuxième soutenance marqua une réussite en terme de design sonore avec l'apparition dans le jeu de courts effets sonores.

A l'aide de sa voix et de celle de Charlie, Philippe a créé une grande banque de sons, contenant près d'une centaine de sons. On s'attendait à ce que ce nombre augmente au fur et à mesure que nous ajoutions des fonctionnalités au jeu. Parfois parsemé de réverbération ou de distorsion pour mieux s'adapter au contexte, presque toutes les actions du jeu ont leur propre fonction de production sonore dans leur script. Il utilise la fonction aléatoire de Unity pour choisir un clip mp3

dans le tableau de clips que chaque action possède, permettant au jeu d'être ultra dynamique et constamment surprenant.

3ème soutenance

Philippe a fait encore plus de sons pour les actions du jeu et les a ajoutés au tableau sonore déclenché par le script aléatoire.

6 Graphismes et Interface utilisateur

6.1 Commentaires

Le développement de l'interface utilisateur et de l'aspect graphique du jeu est celui qui a été le plus bouleversé par la perte d'un de nos membres au début du projet. Nous avons dû réduire les objectifs que nous nous étions fixés pour la première soutenance, dans le design des personnages - modélisation des personnages aussi bien que dans le design des cartes. Ce délai a persisté tout au long du développement, et pour cette raison, ces aspects du développement sont en deçà des objectifs que nous voulions atteindre.

6.2 Interface utilisateur et design

David était responsable de l'interface utilisateur ainsi que du design du jeu. Il a créé le menu principal et les différents éléments de l'interface utilisateur, ainsi que les logos du jeu. Le progrès de ces parties sont décrites ci-dessous.

1ère soutenance

L'interface utilisateur est un point qui a beaucoup évolué au fil du développement du jeu. Pour la première soutenance, l'interface utilisateur n'était composée que d'un simple menu principal au fond gris avec un bouton Play et un bouton Exit, avec le thème musical principal du jeu en boucle joué

en arrière-plan (voir figure 12). Le jeu n'avait pas encore de logo à ce stade.

2ème soutenance

Pour la deuxième soutenance, le menu principal a été modifié de manière drastique. Les boutons Play, Options et Exit se fondent dans un nouveau décor fait à la main à partir d'une photo d'un restaurant fast-food américain du milieu du XXe siècle dénommé Burger Chef. Tout contenu susceptible d'être protégé par des droits d'auteurs a été remplacé avec Photoshop (voir figure 13). Un son spécifique se joue lors de l'appui sur chacun des boutons.

Avec le menu principal, David a aussi designé un logo pour le jeu, sous deux variantes :

— Le premier est un design en 2 lignes, qui apparaît sur la bannière du site et dans le menu principal du jeu (voir figure 14),

— Le second est un design en 3 lignes, qui a été conçu pour être utilisé dans des endroits moins larges, sa résolution étant plus proche d'un carré (voir figure 15).

Le logo a été conçu pour être frappant et amusant par le choix des polices d'écriture et des couleurs.

En outre, des touches ont été assignées pour retourner au menu principal depuis l'écran de jeu, mais leur implémentation n'était pas finale.

3ème soutenance

Pour la dernière soutenance, un menu d'options a été ajouté pour modifier la résolution du jeu ainsi que le niveau de volume. Pour fluidifier la navigation entre les différents écrans du jeu, un menu pause a été ajouté avec deux boutons "Resume" (reprendre la partie) et "Back to main menu" (pour retourner au menu principal). Etant donné que le jeu est principalement joué sur le réseau, le menu pause ne stoppe pas le déroulement du jeu.

6.3 Modélisation 3D

1ère soutenance

Très peu de travail a été fait au cours de la première période de notre projet parce qu'il a d'abord été attribué à Maxime. Notre temps n'a pas été consacré à cette partie du projet dans les premiers mois. Évidemment, c'était le meilleur choix puisqu'il n'était peut-être pas aussi important d'avoir un modèle 3D complexe dans la première présentation que d'avoir un personnage qui peut se déplacer et une fonctionnalité multijoueur.

2ème soutenance

Philippe s'est occupé de la modélisation 3D du personnage. Il a appris à utiliser Blender, un logiciel de montage 3D gratuit. Il a d'abord conçu un personnage humanoïde en 3D (voir

figure 16). Il voulait utiliser un style de caractère humanoïde, parce que cela faciliterait l'utilisation du modèle dans Unity, parce que Unity offre des presets pour les modèles humanoïdes. Il voulait toujours que le modèle ressemble à un poulet, alors il a agrandi la cuisse mais a gardé des os de jambe minces, a remplacé les pieds humains par des pieds de poulet et a mis un mélange poulet-humain pour la tête. Une fois ce premier personnage terminé, Philippe a frappé un grand coup, parce que faire les personnages suivants serait un jeu d'enfant puisque nous ne ferons que changer ses textures.

3ème soutenance

Philippe a ensuite importé le modèle dans Unity et l'a simplement mis à l'échelle à la bonne taille pour que le Player Prefab ait un modèle de forme et de taille parfaites. Le joueur peut maintenant admirer et contrôler un personnage de base dans son jeu.

6.4 Animations 3D

1ère soutenance

Sans modèle 3D, il était logique qu'aucune animation n'ait été faite pendant la première période du projet, si ce n'est des recherche sur la façon de réaliser les animations des mouvements et des attaques du personnage.

2ème soutenance

Quant aux animations, Philippe a d'abord construit et placé les os du personnage. Il a ensuite procédé à "pondérer" (voir figure 17) le modèle 3D pour que les animations déforment le personnage d'une manière naturelle. Il a réalisé deux animations de base : une pose debout où les bras du poulet bougent légèrement, et une animation de marche latérale, où les poulets font des pas de côté d'une manière un peu ridicule pour améliorer le côté amusant du jeu.

3ème soutenance

Comme pour la dernière période de travail, Philippe a implémenté dans Unity les animations du modèle 3D réalisées avec Blender. Il a utilisé le système d'animations fourni par le logiciel pour faire des transitions entre les différents états d'animations. Lorsque le modèle s'arrête, L'animation du personnage passe en animation de repos.

6.5 Design des cartes

1ère soutenance

Pour la première soutenance, David a réalisé la première carte, la carte de Krispoul. Le travail a été fait avec Photoshop, en utilisant comme ressources une vidéo Ionisx et le site flaticon.com. Le but principal en récréant l'arrière-plan du Mimo était de lui rester fidèle mais en obtenant une image

au final non compressée et en haute résolution. Ainsi, une attention spéciale a été donnée à la récréation du dégradé vert et au placement des icônes à gauche, dont certaines ont dû êtres refaites de zéro puisqu'elles n'étaient pas disponibles en ligne. (voir figure 18)

2ème soutenance

Pour la deuxième soutenance, puisque nous devions avancer plus profondément dans les autres aspects du développement du jeu, le design des cartes a été limité à de la recherche et au test de nos idées sur Photoshop et Unity.

3ème soutenance

Pour la dernière soutenance, la dernière carte a été ajoutée, la carte de Git Poule. Elle est constituée d'un arrière-plan imitant l'apparence d'un shell. Encore une fois, le travail a été réalisé sur Photoshop de façon minutieuse pour assurer la meilleure qualité possible. Cela donne au jeu deux cartes jouables.

7 Autres

7.1 Commentaires

Dans les premières semaines de réalisation du projet, nous avons pensé que le multijoueur serait assez facile à intégrer à notre jeu et ne changerait pas tant que ça par rapport à un jeu solo. Nous avions tort. Non seulement toutes les fonctionnalités de notre jeu devaient être codées pour fonctionner en multijoueur et nécessitaient une compréhension complète des fonctionnalités réseau d'Unity, ce qui prenait la plupart de notre temps, mais ces fonctionnalités étaient devenues obsolètes lors de la réalisation de nos projets. Notre dépendance envers Unity a eu un impact négatif sur notre projet, en particulier sur la partie multijoueur.

Pourtant, nous sommes heureux d'avoir mis en place le réseau très tôt, cela nous a permis d'avoir une idée du résultat final. De plus, l'implémenter trop tard aurait été une erreur, cela aurait complètement changé notre projet et nous aurait obligé à réécrire et réévaluer chaque script.

7.2 Site internet - David

1ère soutenance

David a commencé a travaillé sur le site dès la première soutenance, en utilisant des connaissances acquéries dans le passé en HTML et en CSS. Ainsi, dès le début du projet, le

site avait déjà une page d'accueil et évoluait rapidement. Le site a commencé par être un simple lot de pages vides, qui a été transformé en une vraie disposition :

— *Home*, où figurent quelques paragraphes pour introduire le site, le projet et l'équipe,
— *Project*, une page où nous postons des articles à propos de l'avancement du projet au fil du temps avec des captures d'écran,
— *Download*, une page comportant des liens de téléchargement pour le jeu et les divers documents liés à chaque soutenance,
— *Links*, une page comportant des liens vers tous les médias créés pour le jeu ou qui ont été utilsés pour créer le jeu.

Pour la première soutenance, seules les deux premières pages étaient prêtes.

2ème soutenance

Pour la deuxième soutenance, David a continué à travailler sur le site, qui atteint rapidement le stade final. Le site comportait toutes les pages précisées dans le cahier des charges. A partir de ce point, la seule chose qui reste à faire est d'alimenter les pages avec du contenu, un procédé qui a déjà commencé. Avec la parole de chacun des membres du groupe, David a écrit des articles destinés à informer la communauté du progrès de l'équipe dans le développement du jeu.

3ème soutenance

A l'approche de la dernière soutenance, le site était déjà réalisé dans sa totalité. Plus de contenu a été ajouté au fil du progrès de l'équipe, et quelques coquilles techniques ont été réparées. (voir figures 19 à 22)

7.3 Multijoueur

1ère soutenance

La première chose que Philippe a faite concernant l'implémentation multijoueur de notre jeu est de faire de grandes recherches sur toutes les façons possibles de l'implémenter. Il s'est également penché sur les concurrents d'Unity et sur la société qui avait la meilleure offre gratuite. Il a ensuite choisi le plan gratuit de unity parce qu'il était le plus utilisé dans le monde et donc beaucoup de tutoriels et d'informations pouvaient être trouvés en ligne.

De plus, l'implémentation du multijoueur dès le début du projet est la meilleure solution en ce qui concerne le temps qu'il aurait fallu pour convertir un jeu solo en un jeu multijoueur, car la façon dont les informations sont transmises est totalement différente dans un jeu multijoueur. Nous avons donc commencé par faire un jeu multijoueur, et si quelqu'un souhaite jouer en solo, il peut héberger son propre serveur local nativement.

Notre scène de jeu contient un GameManager Prefab qui gère les capacités réseau du jeu. Ce Prefab contient le composant Network Manager ainsi que le composant Network Manager HUD. Le composant Network Manager gère le multijoueur et le composant HUD nous permet d'avoir un menu prédéfini avec des fonctionnalités de base telles que LAN Host et LAN Client.

Ainsi chaque objet dans la scène qui a un comportement lié au serveur doit avoir un Network Identity Component et un Network Transform Component.

— Network Identity va definir si l'objet est basé sur le serveur ou si le client aura l'autorité sur l'objet
— Network Transform synchronise l'objet avec le serveur. Ce Component a des paramètre pour la fréquence à laquelle l'information est envoyée et quel seuil de mouvement est utilisé.

Avant chaque action, l'objet doit vérifier si le client à l'autorité sur lui. De cette manière, le joueur peut uniquement bouger son personnage.

2ème soutenance

Philippe a fait peu de travail sur la structure générale de la fonctionnalité multijoueur du jeu dans la deuxième période du projet. De concert avec Charlie, Philippe a fait de petits ajustements. Des caractéristiques telles que la barre de santé

ont été corrigées dès la première présentation. Le jeu a toujours l'interface graphique du réseau d'origine de l'Unity que nous avions l'intention de changer pour une interface personnalisée au moment de la troisième présentation.

Nous avons également corrigé quelques problèmes de synchronisation des mouvements entre les personnages survenus avant la première présentation. Charlie et Philippe ont également rencontré des bugs concernant l'interface utilisateur : la barre de santé et d'endurance ne diminuerait pas de la même manière pour le client et le serveur. Heureusement, nous avons pu résoudre ces problèmes avant la date limite en utilisant les commandes (voir la partie sur la mécanique de jeu).

3ème soutenance

Parce que la fonctionnalité multijoueur était déjà pleinement opérationnelle dans la deuxième période de notre projet, ce mois-ci a consisté uniquement en des recherches sur l'efficacité du réseau ainsi qu'en un gros travail de résolution de bugs.

8 Production du projet

8.1 Rythme de travail et difficultés

Au cours de ces six mois de projet, nous avons rencontré plusieurs difficultés. En effet, des nuits ont été passées à essayer de corriger un bug sur Unity ou à trouver comment implémenter une mécanique de jeu... Par exemple, la partie la plus difficile à comprendre a été la mise en place du réseau sur notre projet Unity. Ces difficultés ont modifié notre rythme de travail.

Notre rythme de travail a été intensif, chaque membre a tout donné pour créer des fonctionnalités au jeu. Pourtant, passer des heures à s'arracher les cheveux à cause d'un bug n'était pas toujours la solution. En dehors de la période des rushes, nous avons généralement passé du temps à rassembler de nouvelles façons de programmer, des idées sur les jeux de combat et de nouvelles idées. Par conséquent, il n'y a pas eu de perte de temps. L'établissement d'un tel rythme de travail a été un grand pas vers l'achèvement de notre projet.

8.2 Satisfactions

La première satisfaction du groupe était de pouvoir travailler dans un environnement de travail positif, en se serrant les coudes et en trouvant des idées qui ont aidé à mener à bien le projet. Aujourd'hui, nous sommes heureux de présen-

ter un jeu fonctionnel et qui ressemble bel et bien à un jeu. C'est une grande victoire pour chacun d'entre nous, qui, dans l'ensemble, étions des novices à chacune des tâches qui nous étaient assignées.

Une chose qui nous rend évidemment fiers est l'abilité que nous avons eu à finir le projet à temps, malgré le défi que cela représentait au début. Nous étions des débutants complets, mais en apprenant de nombreuses abilités en programmation, design de jeu, graphismes et même en développement de site internet, nous avons pu réaliser un produit fini. Bien qu'il ne soit pas parfait, c'est un jeu totalement jouable.

Nous sommes aussi heureux que, malgré la perte d'un membre du groupe, nous avons quand même été aptes à délivrer tout en temps et en heure pour chaque soutenance, et compenser la charge de travail pour faire avancer le projet sans obstacles au fil des présentations.

Un de nos principaux succès est d'avoir été aptes à réaliser un jeu de combat fluide et rapide dans son gameplay, entre les modèles de Super Smash Bros. et de Street Fighter. Un joueur est libre dans ses mouvements mais ne se fait tout de même pas éjecter de la carte après avoir été touché.

Tout l'équipe est fière d'avoir pu établir cette ambiance absurde et burlesque dans le jeu. Les voix et efets sonores ont

beaucoup aidé, et bien sûr le fait que les personnages soient des poulets humanoïdes a beaucoup aidé. Créer quelque chose qui nous fait rire et que l'on aime n'a que contribué à la motivation du groupe.

8.3 Déceptions

La sélection des personnages, pour nous, est essentielle au genre du jeu de combat. L'une de nos déceptions était peut-être de ne pas avoir mis notre attention au bon endroit au bon moment. Nous savons que si nous nous y étions concentrés quelques heures de plus, nous aurions peut-être eu plus de personnages et une meilleure diversité de cartes.

9 Conclusion

En tant que membres de TEMPÊTE , nous avons été unis et nous nous sommes investis pleinement dans le projet. Avec de grandes ambitions et une forte croyance en nos capacités, nous avons créé un jeu qui vaut cher à nos yeux. Nous avons tous joué à ***Street Fighter*** ainsi qu'à ***Super Smash Bros.*** étant plus jeunes. En s'inspirant de ces classiques qui font du jeu de combat ce qu'il est aujourd'hui, ***Ultimate Fighting Chickenship*** est l'oeuvre de notre propre jeu de combat.

10 Appendix

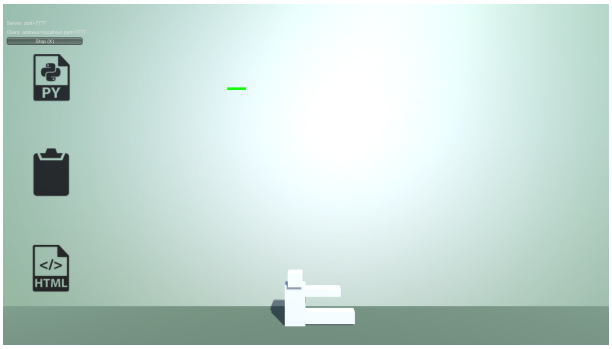

FIGURE 4 – Position normale

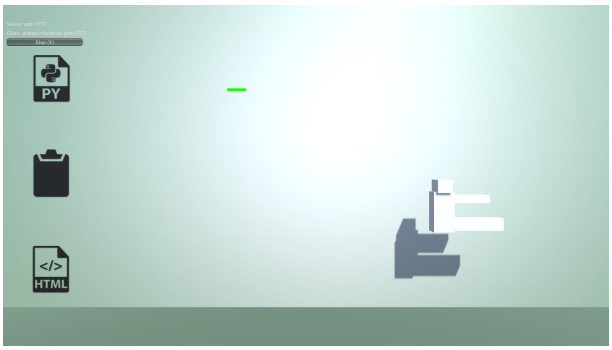

FIGURE 5 – Personne qui saute

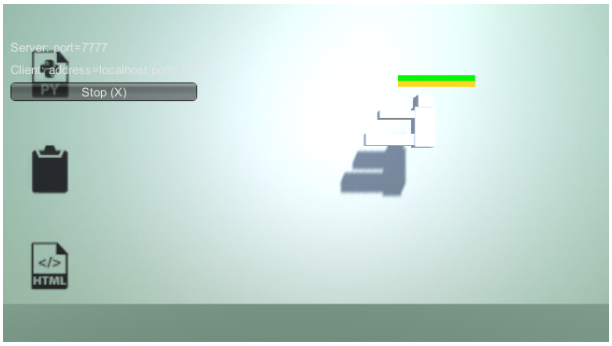

FIGURE 6 – Nouvelle hauteur de double saut

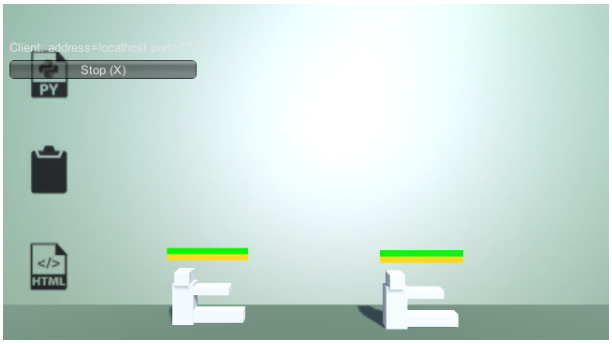

FIGURE 7 – Les deux personnages se font face grâce à la rotation

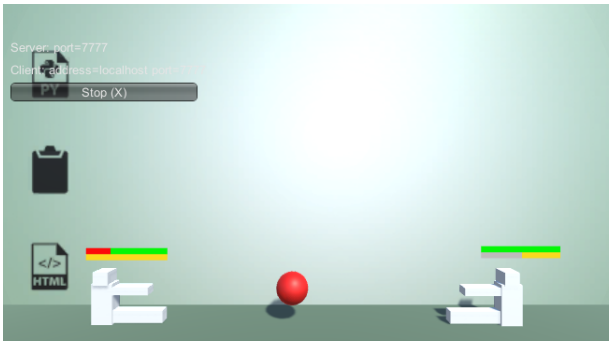

FIGURE 8 – Une boule de feu qui applique des dégats

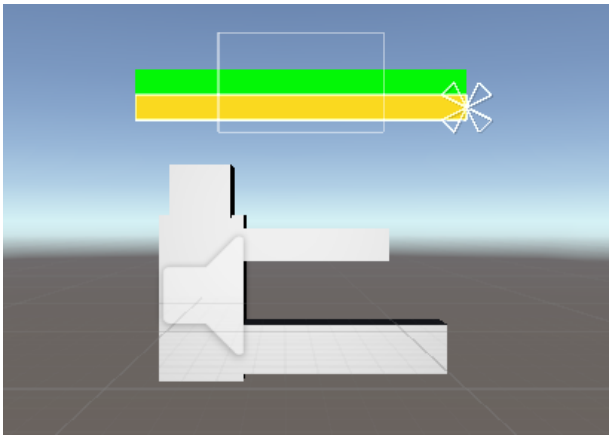

FIGURE 9 – Une barre d'endurance complète

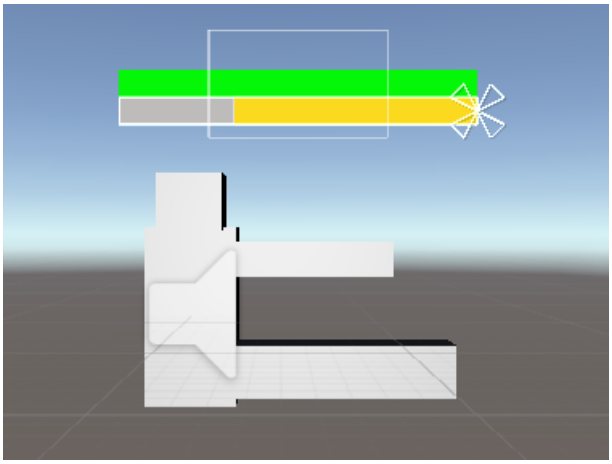

FIGURE 10 – Une barre d'endurance qui se recharge

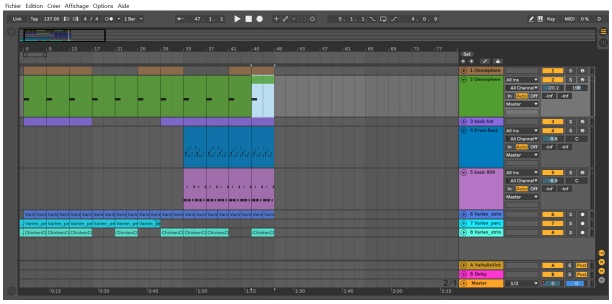

FIGURE 11 – La création de la musique dans Ableton

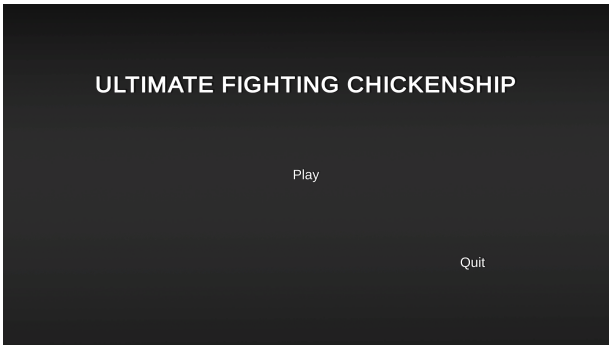

FIGURE 12 – Une capture d'écran de la première version du menu

FIGURE 13 – Une capture d'écran du menu final

FIGURE 14 – Le logo à deux lignes

FIGURE 15 – Le logo à trois lignes

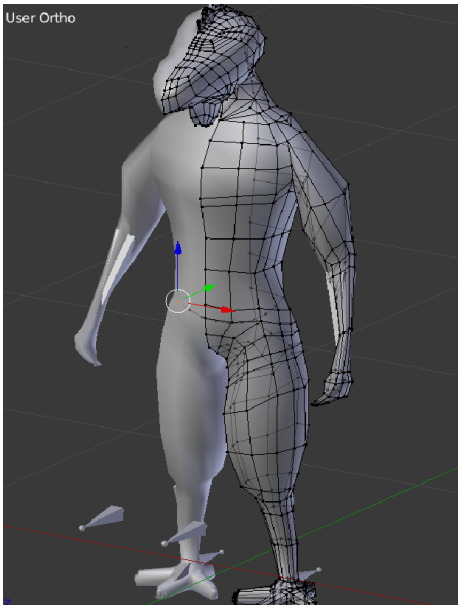

FIGURE 16 – La création du modèle 3D dans Blender

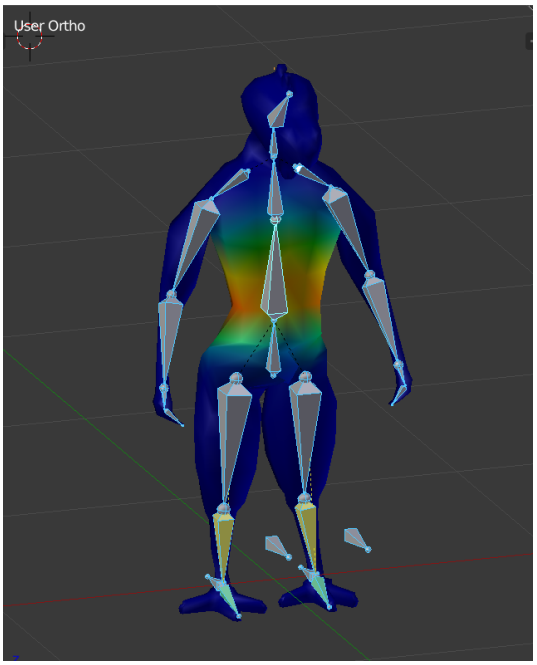

FIGURE 17 – Le pondérage en 3D

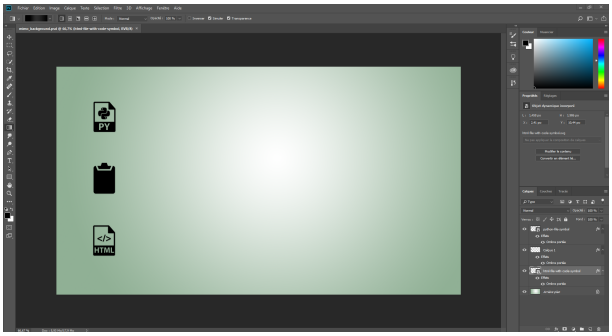

FIGURE 18 – La création de l'arrière-plan du Mimo dans Photoshop

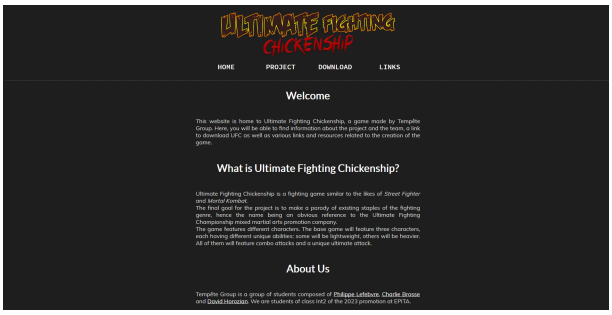

FIGURE 19 – Une capture d'écran de la page d'accueil - Avril 2019

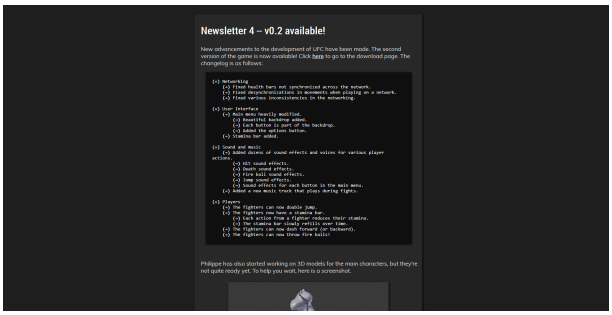

FIGURE 20 – Une capture d'écran de la page du projet - Avril 2019

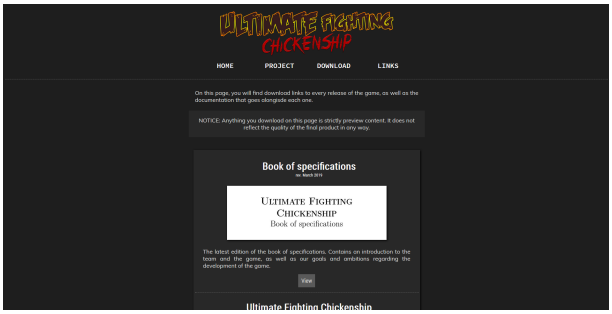

FIGURE 21 – Une capture d'écran de la page de téléchargements - Avril 2019

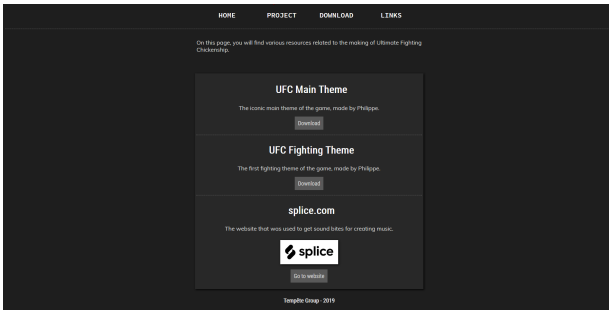

FIGURE 22 – Une capture d'écran de la page des liens - Avril 2019