

Rapport de la première soutenance - Projet S2

Paul HAAS - Lucas KOTEVSKI - Alexiane LAROYE - Thomas MAURER

promo EPITA 2026



TABLE DES MATIÈRES

Table des matières

1	Introduction	3
1.1	Le jeu: Overwhelmed	3
1.2	Modifications du cahier des charges	3
1.3	Rappel du planning de soutenance	3
2	L'avancement du projet	4
2.1	L'avancement de groupe	4
2.2	L'avancement individuel	4
2.3	L'avancement de chaque partie	5
2.3.1	Model 3D	5
2.3.2	Animation 3D	8
2.3.3	Engine	9
2.3.4	Game Management	9
2.3.5	Interfaces	10
2.3.6	Réseau	10
2.4	L'assemblage du projet	11
3	Récit de la réalisation	12
3.1	Les difficultés rencontrées	12
3.2	La découverte	12
4	Avancement par rapport au cahier des charges	13
5	Attendus pour la prochaine soutenance	13
5.1	Nos objectifs	13
5.2	La répartition des tâches	14
5.3	Le planning pour la deuxième soutenance	14
5.3.1	Site Web	15
6	Conclusion	15
7	Annexe	15
7.1	Map	15
7.2	Menu Principal	18
8	Sources	21

1 Introduction

Cela fait déjà 2 mois que cette aventure a commencé. Nous infligeant déjà des joies et des peines. Cependant, nous avons pris goût à celle-ci et nous nous sommes motivés pour vous présenter dès cette première soutenance une base de projet intéressante. Nous sommes sur la bonne voie pour vous proposer un jeu attractif. Suivez la création de notre jeu : Overwhelmed.

1.1 Le jeu: Overwhelmed

Tout d'abord, nous avions décidé de nous orienter sur la création d'un jeu vidéo, en l'occurrence un FPS, il s'agit d'un jeu de tir à la première personne ou en vue subjective. Notre jeu se joue de 1 à 4 joueurs, le but est de survivre le plus longtemps possible à des vagues de zombies dans un camp militaire abandonné. Des améliorations seront possibles tout au long de l'aventure et de nombreux événements spéciaux occurreront durant différents tours.

1.2 Modifications du cahier des charges

Concernant le cahier des charges plusieurs éléments ont été mis à jour:

- Tableau des répartitions des tâches plus lisible
- Tableau d'avancement des soutenances plus détaillé
- Insertion d'images
- Détail des logos
- Argumentation de l'IA (l'ensemble des techniques que nous allons mettre en œuvre en vue de simuler l'intelligence humaine)
- Argumentation de la partie opérationnel

1.3 Rappel du planning de soutenance

1. Tableau du planning de la première soutenance:

Tâches / Soutenances	Soutenance 1
Model 3D (carte,personnages,ennemis)	Nous avons la base de la Map et de nos personnages
Animation 3D	Quelques programmes basiques (marche,course)
Audio	
Engine (mouvement,shooting,IA)	Pouvoir se déplacer, sauter et tirer
Game Management	Spawn Management et Death Management
Interfaces (menus, inventaire, UI)	Menu principal fonctionnel
Réseau	Application Photon mise en place, possibilité de rejoindre une « room »
SiteWeb	

Nous allons vous expliquer plus en détail le tableau ci-dessus. Tout d'abord, concernant la partie « Model » 3D, pour cette première soutenance nous devions avoir une carte dont la base était finie. Nous voulions pour la prochaine

soutenance simplement à avoir à rajouter des portes, des objets sur des endroits vides, arranger les proportions des objets etc. La carte devait donc être presque complète. Concernant la partie sur l'Animation 3D, pour cette première soutenance il fallait avoir les animations de base (marche, course, saut) mais également les animations corporelles qui vont avec. Pour la tâche « Engine », les collisions devaient être gérées ainsi que le contrôle de la caméra grâce à la souris. Et enfin, il fallait implémenter des armes et la possibilité de tirer. Puis, concernant la partie « Game Management », nous devions gérer les « kills », morts, « spawns » et « respawns » des joueurs, plus précisement il fallait gérer les cas quand l'adversaire meurt, quand le joueur meurt, mais également le cas où le joueur apparaît dans la « map » où réapparaît dans celle-ci. Enfin, pour les interfaces, nous devions créer le menu principal qui se compose du nom du joueur, de créer une « room », de rejoindre une « room » et tous les cas d'erreurs qui peuvent arriver lorsqu'on rejoind une « room ». Et pour finir, pour la partie réseau, il fallait implémenter et paramétriser l'application Photon.

2 L'avancement du projet

2.1 L'avancement de groupe

Vous avez pu sûrement voir que notre groupe se compose en binôme. C'est-à-dire que pour simplifier la communication lorsque quelqu'un est responsable de la tâche, son binôme est suppléant de tâche et inversement, sauf pour certaines parties telles que le site web. Cela nous permet d'avoir une meilleure communication sachant que chaque tâches reliées entre elles n'est pas un choix anodin. En effet, la partie « Model » 3D, se lie avec la partie Animation 3D. Cela nous a permis d'effectuer un travail de groupe organisé, mais surtout de ne pas prendre de retard sur notre projet. Cependant, nous communiquons à quatre pour prendre les décisions ou si nous avons besoin d'aide ou d'avis. De plus, nous communiquons et faisons nos réunions avec l'application Discord, car cela nous facilite la communication, en effet, Discord¹ possède des fonctionnalités telles que le partage d'écran. Nous n'avons pas de fréquence de réunion mise à part chaque soir à l'approche de la soutenance sinon cela dépend de l'avancement et des besoins de chacun.

2.2 L'avancement individuel

Concernant l'avancement individuel, chaque personne du groupe est impliquée dans son travail et essaie de se démener seul avant de demander de l'aide pour progresser. Concernant le travail individuel, aucune tâche n'est en retard, cela montre que nous sommes tous conscients des responsabilités que nous donne le projet. Nous savons que si nous ne faisons pas notre travail, cela aura des répercussions sur tout le projet et pas que sur notre partie. De plus,

1. <https://discord.com/>

pour l'instant, aucun des membres du groupe n'a quitté le navire, c'est un bon point à souligner.

2.3 L'avancement de chaque partie

2.3.1 Model 3D

Concernant la « map », Thomas et Alexiane ont travaillé dessus. Celle-ci a été inspirée de camp militaire issu de jeu vidéo tel que par exemple Call of Duty Warzone² ou d'images trouvées sur le « Web³ ». Pour réaliser celle-ci, nous avons tout d'abord fait des croquis comme vous pouvez l'observer sur l'image ci-dessus, nous avons essayé d'imaginer à quoi pourrait ressembler la map puis nous avons cherché des assets sur l'Asset Store⁴ qui conviendrait à notre projet. Un asset sur Unity est un élément que vous pouvez utiliser dans votre jeu ou votre projet. Un asset peut provenir d'un fichier créé en dehors de Unity⁵, tel qu'un modèle 3D, un fichier audio, une image ou tout autre type de fichier pris en charge par Unity. Il faut savoir que tous les assets que nous avons utilisé

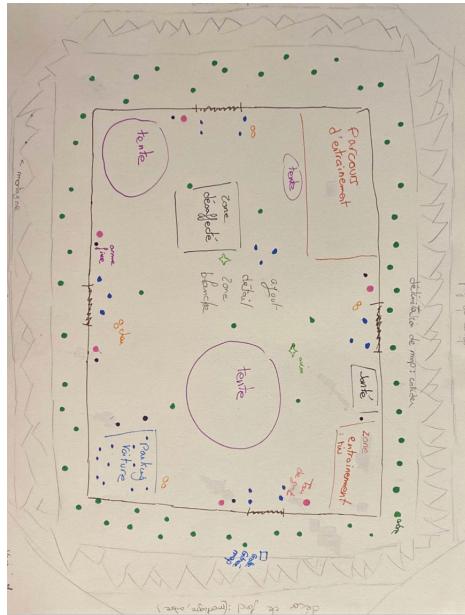


FIG. 1 – Croquis map

sont gratuits. De plus, nous avons certes téléchargé des assets, mais ils étaient tous indépendants, par exemple des planches, des grillages etc. Nous avons donc

- 2. <https://www.callofduty.com/fr/>
- 3. <https://www.google.fr/>
- 4. <https://assetstore.unity.com/>
- 5. <https://unity.com/>

utilisé Unity pour les assembler, les ajuster, et leur ajouter des éléments tels que des « colliders », il s'agit de composants qui permettent d'éviter que notre personnage se retrouve à l'intérieur des éléments qui constituent notre « map ». Nous avons aussi utilisé Blender⁶ pour la création d'objet simple car nous voulons un rendu réaliste et nous ne sommes pas encore assez expert sur Blender pour obtenir un rendu de ce style. C'est pourquoi, la plupart des voitures type char, hélicoptère sont des éléments qui étaient déjà tous assemblés, car comme nous voulons un jeu réaliste, il est trop difficile de créer ce style d'élément avec Blender à notre niveau de compétence. Et cela nous prendrait trop de temps pour un rendu qui ne nous conviendrait sûrement pas. De plus, pour éviter tout problème de chute de personnages, nous avons ajouté sous la « map » un terrain qui bloque tout risque de problème qui nous ferait traverser la « map », nous pouvons le définir comme un terrain de secours. La « map » dans sa globalité est complète, cependant il reste quelques éléments à ajouter ou à ajuster tel que les colliders, les portes, etc. Vous pouvez observer ci-dessous deux images qui représente une zone désaffecté car notre jeu se passe dans un camp militaire abandonné et un exemple de char et de voiture militaire. Vous pouvez voir que les chars sont des éléments très complexes à créer, c'est pourquoi nous ne l'avons pas fait nous même.



FIG. 2 – zone désaffecté

6. <https://www.blender.org/>



FIG. 3 – *char et voiture*

Nous voulions avoir quasi fini la « map » pour la première soutenance, nous avons donc atteint notre objectif et nous sommes même allés plus loin que nos prévisions. En effet, il s'agit de la base du jeu, donc c'est un élément indispensable à réaliser le plus tôt possible.

De plus, concernant les personnages, nous avons utilisé Mixamo pour pouvoir télécharger le « skin » de nos personnages. Le « skin » représente l'apparence des différents personnages. Cependant, le « skin » de nos zombies est issu de l'Asset Store. En outre, certes nous avons téléchargé nos « skins » sur Mixamo⁷, mais nous allons sûrement envisager de les ajuster avec l'application Adobe Fusse CC⁸. De plus, nous avons quatre « skins » de personnages car comme nous implémentons un multijoueur de 1 à 4 personnes il nous faut donc 4 « skins » distincts.

7. <https://www.mixamo.com>

8. <https://www.adobe.com/fr/wam/fuse.html>

2.3.2 Animation 3D

Les premières animations 3D ont vu le jour ces dernières semaines. Alexiane et Thomas ont utilisé un simple personnage 3D afin d'avoir les premiers aperçus du fonctionnement de l'animation sur Unity. D'abord, il était important que le personnage puisse être mobile, qu'il puisse se déplacer dans toutes les directions. Nous nous sommes donc appuyés sur les premiers TP d'AR/VR que nous avons eu en début d'année. Ces TPs qui entrent dans le cadre de notre parcours en Nouvelles Technologies et Société nous ont permis d'acquérir les bases d'Unity comme par exemple pouvoir déplacer un objet.

Ainsi, nous avons réussi à faire en sorte que le personnage se déplace en avant, en arrière, à gauche et à droite. Nous avons également ajouté la fonctionnalité de saut, et tout le code qui va avec afin que le personnage ne puisse sauter si et seulement s'il est au sol. Une fois toutes ces possibilités ajoutées, nous devions nous occuper du visuel, car jusque-là, le personnage va là où on lui demande, mais en gardant la posture initiale. Ainsi, nous nous sommes aidés de Mixamo, pour télécharger les animations de marche, de course, de saut, et de respiration (le joueur est inactif, il reste sur place). Vous trouverez sur la *Figure 4*, le schéma des composants de l'animateur de notre joueur. Nous avons mis

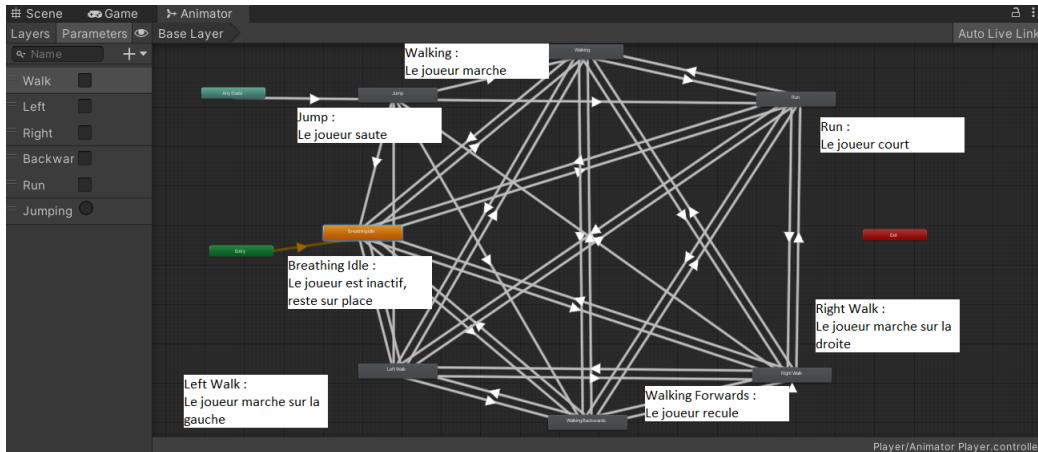


FIG. 4 – Animator Player

des annotations sur cette capture d'écran pour la lisibilité. Comme vous l'aurez constaté, le schéma n'a pas l'air très bien organisé en raison des nombreuses flèches qui lient les différentes animations. Pourtant, ces flèches sont le cœur de notre fonctionnement. Dans l'onglet à gauche, l'on retrouve les paramètres, qui sont principalement des booléens dans notre cas, et un trigger : « Jumping ». Ces paramètres sont gérés dans notre script C#, par exemple, lorsque l'on est à l'arrêt, et que l'on clique sur la touche « W », le booléen « Walk » devient « true », le personnage avance et son animation de marche s'active. Ainsi, le joueur marche tout en étant animé. Tout ceci est un bon début de l'animation

visuelle. Pour la prochaine soutenance, les objectifs sont d'animer les zombies, ce qui ne semble pas très compliqué puisque nous allons seulement mettre des animations de marche, d'attaque, et de mort. En ce qui concerne les personnages, nous allons devoir ajouter l'arme dans la main, mais cela paraît moins compliqué maintenant que nous savons comment nous y prendre.

2.3.3 Engine

Lucas a jusqu'à présent développé un engine se décomposant en 3 parties fonctionnant autour de la class PlayerController. Un Engine autrement appelé moteur de jeux est l'ensemble des logiciels permettant de calculer et simuler la physique d'un jeu vidéo.

- Premièrement, les déplacements du personnage ayant déjà été développé pour l'animation 3D, il ne manque plus qu'à implémenter les collisions afin d'empêcher le personnage de traverser un mur ou un autre joueur. Cela a été fait en utilisant la même méthode que pour le TD d'AR/VR.

- Deuxièmement, le control de la caméra grâce à la souris est au cœur même des FPS. Afin de l'implémenter, l'utilisation des classes et méthodes de la librairie UnityEngine⁹ fut nécessaire. Pour permettre un certain réalisme il a fallu empêcher la caméra de pouvoir faire une rotation entière sur l'axe Y. Elle est donc bloquée au minimum au niveau des pieds du personnage et au maximum au niveau de sa tête.

-Troisièmement, l'implémentation des armes et de la possibilité de tirer. Pour permettre cela, la création de 6 nouvelles classes fut requise. Item et Gun, deux classes abstraites permettant une création plus simple d'armes différentes. ItemInfo et GunInfo, deux classes stockant le nom et les dégâts de l'arme. La class IDamageable est une classe dont hérite tous les objets pouvant prendre des dégâts. Et enfin SingleShotGun, la classe définissant notre arme. Dans celle-ci, la fonction Use() est utilisé que lorsque PlayerController détecte l'utilisation du clic droit et pose un impact sur la position du premier objet touché et lui retire les dégâts de l'arme s'il peut en prendre. Dans cette rubrique, malgré le fait que cela n'était pas encore nécessaire à la soutenance, l'ajout d'une seconde arme ainsi que de la possibilité de changer d'une arme à une autre a déjà été implémenté.

2.3.4 Game Management

Pour pouvoir gérer les éliminations , morts, apparition et réapparitions des joueurs. L'initialisation des joueurs par une classe RoomManager et une autre PlayerManager a paru opportune. En effet la classe RoomManager permet d'initier une instance de la class PlayerManager par joueur. Cette instance initialisera un GameObject Player et une instance de la classe PlayerController permettant de le contrôler. Player et PlayerController seront détruits dès que les points de vie seront à 0, puis réinitialisé par PlayerManager. La classe SpawnManager permet de gérer les points d'apparition et sa méthode d'en obtenir un aléatoirement

9. <https://docs.unity3d.com/ScriptReference//>

parmi ceux disponibles. Ils sont initialisés avec la class `Spawnpoint`. Il fut aussi nécessaire de synchroniser certaines informations entre les différents joueurs, tel que les dégâts encaissés et produits. Pour faire cela les méthodes de la librairie Photon furent utilisées.

Pour l'instant l'Intelligence Artificielle n'ayant pas encore été développé, le jeu s'oriente essentiellement sur un FPS de type Match à mort. A terme les ennemis seront les IA et des conditions permettant de réapparaître seront définies.

2.3.5 Interfaces

Avant le menu principal, Paul a d'abord dû insérer un écran de chargement pour éviter les erreurs lorsqu'un utilisateur essaye de créer ou rejoindre trop rapidement une « room » (salon virtuel permettant à différents utilisateurs de se rejoindre). Ensuite, il a fallu créer un script « `MenuManager` » permettant de gérer l'ouverture et la fermeture des menus (l'écran de chargement, le menu principal, le menu où l'on crée la « room », etc.). Nous avons aussi fait un menu d'erreur lorsque l'on n'arrive pas à créer une « room ». Nos menus sont composés de « `Button` » permettant différentes actions et de « `InputField` » pour récupérer les entrées textuelles de l'utilisateur.

L'utilisation des classes préfabriquées de PUN (le package associé à Photon ; plus de précisions dans la partie réseau ci-dessous) furent aussi très utiles tel que la classe « `Player` », notamment pour donner un pseudonyme aux joueurs ou encore « `RoomInfo` » pour identifier et distinguer chacunes d'entre elles.

La partie « Interfaces » est donc à jour avec l'objectif fixé pour la première soutenance : on a bien un menu principal qui nous permet d'effectuer les actions dont on a besoin. Il resterait juste à le rendre un peu plus esthétique puisqu'il est assez terne pour l'instant (on peut s'occuper de cela plus tard, voire même pour la dernière soutenance puisque cela est de moindre importance). De plus, nous avons pu prendre de l'avance sur cette partie puisque nous avons déjà pu intégrer les pseudos qui s'affichent au-dessus des joueurs. Pour cela, il y a simplement besoin de faire que le texte créé au-dessus du joueur s'oriente constamment dans la direction de la caméra.

2.3.6 Réseau

La partie réseau, faite par Paul, peut être décomposée en deux parties :

- Tout d'abord il a fallu implémenter et paramétriser l'application Photon. Sur leur site web¹⁰, on peut créer une nouvelle application Photon à partir de son dashboard (cf. Fig 5). On obtient ainsi une « `App ID` », une sorte d'adresse IP propre à notre serveur Photon. Ensuite, on a dû importer le package « `PUN 2` », une sorte de boîte à outils (des fonctions, des classes, des méthodes, ...) permettant de mettre en relation notre projet Unity avec son serveur Photon dédié (grâce à son `App ID`). Le fichier `PhotonServerSettings` nous permet alors de gérer quelques paramètres, tel que la possibilité de fixer la région en EU pour

10. <http://www.photonengine.com>

éviter tout problème de latence.

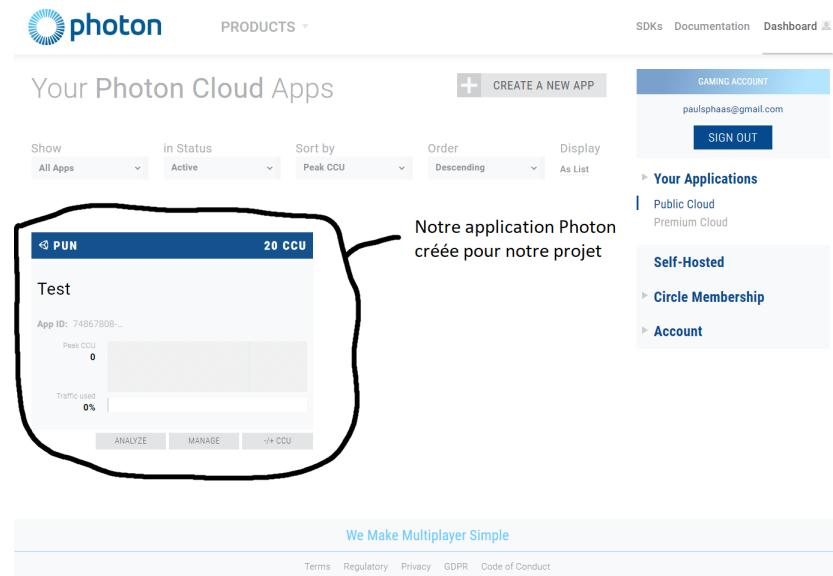


FIG. 5 – Dashboard Photon

- L'autre partie, une fois notre application en place, consiste à utiliser les fonctions de PUN 2 pour s'occuper du réseau (en ajoutant « using Photon.Pun; » au début des fichiers). Nous avons fait que dès le lancement du jeu, on est directement connecté au server puis ajouté dans le « lobby », espace qui nous permet de créer des « room » et d'en rejoindre. Ensuite, pour les « rooms », on a pu utiliser les fonctions « PhotonNetwork.JoinLobby() » et « PhotonNetwork.CreateRoom() ». Cette partie est donc bien prête pour la première soutenance puisque nous avons réussi à utiliser les services de Photon pour faire fonctionner l'aspect multijoueur en ligne de notre jeu. En effet, on arrive bien à rejoindre la « room » de n'importe quel autre joueur.

2.4 L'assemblage du projet

Concernant la mise en commun du projet, pour l'instant chacun a effectué ses parties de son côté et nous avons décidé de ne pas mettre en commun de suite nos parties pour éviter tout problème avant la première soutenance. Il y a juste les parties de Paul et Lucas qui sont depuis le début combinées vu qu'ils ont travaillé sur un même projet Unity pour pouvoir utiliser les mêmes variables et méthodes. Nous avons comme objectif de mettre en commun toutes nos parties pour la deuxième soutenance comme l'assemblage du jeu avec la map et les

animations.

3 Récit de la réalisation

3.1 Les difficultés rencontrées

Nous avons rencontré plusieurs difficultés ce qui est très courant dans un projet. Tout d'abord, concernant la map, le partage de la map issue de Unity3D, nous a posée de nombreux soucis car le transfert via Github ne voulait pas se faire car notre map était trop volumineuse. Nous avons réussi à régler le souci après de nombreuses recherches. Cependant notre map reste tout de même assez volumineuse malgré le fait que nous avons par exemple descendu la qualité des textures. Ce souci est dû à nos assets car nous voulons un jeu réaliste donc ceux-ci sont volumineux dû à leur niveau de qualité. Cela est un problème que nous devons encore résoudre.

Concernant la partie Animation 3D, l'Animator n'est pas de tout repos. En effet, cet outil peut devenir une source de problème s'il est mal organisé. De plus, son fonctionnement nécessite une certaine quantité de connaissances car il est rempli de fonctionnalités diverses qui le rend d'autant plus complexe car pour faire une chose il peut y avoir plusieurs possibilités de réalisation. Cependant, avec du recul et de l'aide, nous arrivons à nous en sortir et à obtenir les résultats attendus.

Pour ce qui est des parties réseau et interfaces, cela a pris du temps de comprendre les différentes classes et méthodes de Photon et de savoir quand et comment les utiliser. Malgré que la plupart des méthodes porte des noms assez transparents, le principe n'est souvent pas très intuitif ou clair. De plus, la documentation Photon n'est pas des plus pratiques à utiliser.

En ce qui concerne l'Engine et le Game Management, il fut compliqué pour Lucas de comprendre comment différencier un joueur d'un autre et d'utiliser les méthodes et classes de la librairie UnityEngine car la documentation n'est pas très claire.

3.2 La découverte

En ce début de projet, nous avons déjà pu découvrir plusieurs choses. En effet, concernant la partie « Model » 3D, nous avons pu manipuler des outils comme Blender et Unity, cela nous a permis d'acquérir de l'expérience, mais également de se perfectionner dans l'utilisation de ces outils.

La partie Animation 3D nous a également permis de manipuler l'outil Animator de Unity qui est plus complexe qu'il ne paraît. En effet, cela n'a pas été évident de comprendre son fonctionnement, et l'organisation du schéma n'est

5 ATTENDUS POUR LA PROCHAINE SOUTENANCE

pas géniale, puisque plus on ajoute d'animation, et plus il faut lier les cases avec les flèches, ce qui complique la lisibilité de notre schéma. Il faut également être rigoureux au niveau du script C# et manipuler correctement les paramètres afin que les animations soient fluides et s'activent en temps voulu. Mais une fois que tout est bien réglé, c'est agréable de voir que les animations sont bien synchronisées et sont fluides.

Le fait de faire un jeu multijoueur en ligne nous a permis de découvrir comment cela est possible. On a découvert comment on peut communiquer avec le serveur Photon dédié mais aussi, on s'est rendu compte de la nécessité de synchroniser tous les éléments avec le serveur pour qu'ils soient communs à tous les joueurs.

4 Avancement par rapport au cahier des charges

2. Tableau du planning de la première soutenance identique à celui issu du cahier des charges:

Tâches / Soutenances	Soutenance 1
Model 3D (carte,personnages,ennemis)	Nous avons la base de la Map et de nos personnages
Animation 3D	Quelques programmes basiques (marche, course)
Audio	
Engine (mouvement, shooting, IA)	Pouvoir se déplacer, sauter et tirer
Game Management	« Spawn Management » et « Death Management »
Interfaces (menus, inventaire, UI)	Menu principal fonctionnel
Réseau	Application Photon mise en place, possibilité de rejoindre une room
SiteWeb	

Comme vous pouvez l'observer, nous ne sommes en retard dans aucune de nos parties. Cependant, nous sommes en avance dans certaines telles que la partie Animation 3D. Cependant il s'agit d'une légère avance, en effet pour cette partie nous avons juste implémenté une ou deux animations en plus.

5 Attendus pour la prochaine soutenance

5.1 Nos objectifs

Nous avons plusieurs objectifs pour la prochaine soutenance. Le premier est de garder cette cohésion de groupe et cette entraide. Le deuxième est de rester sur cette bonne lancée et de garder notre avance pour pouvoir régler tout problème qui viendrait à se montrer. Et enfin, notre dernier objectif est de continuer à acquérir de l'expérience et des connaissances, donc ne pas hésiter à ajouter des éléments intéressants que nous pourrions trouver lors de nos recherches internet. Cela nous permettra de compléter notre jeu et de le rendre encore plus divertissant.

5.2 La répartition des tâches

Vous trouverez ci-dessous notre répartition des tâches. Ce tableau est issu de notre cahier des charges et comme celui-ci nous convient parfaitement, il restera ainsi pour la deuxième soutenance.

3. Tableau de la répartition des tâches du projet:

Tâches / Personnes	Alexiane LAROYE	Thomas MAURER	Paul HAAS	Lucas KOTÈVSKI
Model 3D (carte, personnages, ennemis)				
Animation 3D				
Audio				
Engine (mouvement, shooting, IA)				
Game Management				
Interfaces (menus, inventaire, UI)				
Réseau				
SiteWeb				

Légende:

Responsable de la tâche = ■

Suppléant de tâche = □

5.3 Le planning pour la deuxième soutenance

Vous trouverez en dessous notre planning pour la deuxième soutenance. Celui-ci est issu du cahier des charges. Certes nous sommes en avance sur certaines tâches, cependant nous ne voulons pas le modifier car nous préférons garder l'avance que nous avons plutôt que de se fixer des objectifs trop haut et de ne pas pouvoir les atteindre et donc d'être en retard sur notre planning. De plus, nous ne pouvons pas deviner à l'avance les problèmes que nous devrons résoudre. Toutes les tâches ne sont pas du même degrés de difficulté donc il se peut qu'en plus des suppléants de tâche d'autres personnes devront aussi aider.

4. Tableau du planning de la deuxième soutenance:

Tâches / Soutenance	Soutenance 2
Model 3D (carte,personnages,ennemis)	Finalisation de la map et des personnages
Animation 3D	Animation des personnages et des ennemis
Audio	Musique de fond et début des bruitages
Engine (mouvement,shooting,IA)	Implémentation des zombies + nouvelles armes
Game Management	Création des « rounds »
Interfaces (menus, inventaire, UI)	Ajout d'éléments sur l'UI (argent, n°round, compteur de mort, etc)
Réseau	Synchronisation des zombies avec les autres joueurs
SiteWeb	Création du site Web

Pour la deuxième soutenance, nous continuerons la conception de la map et des personnages, mais nous implémenterons aussi de nouvelles animations. Nous ajouterons également la musique de fond sachant qu'il y en aura des différentes pour le jeu, le menu. Nous implémenterons également certains bruitages. Puis nous implémenterons les zombies et de nouvelles armes. Sans oublier, que nous devrons créer également les « rounds ». Il faudra aussi ajouter des éléments sur l'UI comme par exemple l'argent, compteur de mort etc. Et enfin il faudra synchroniser les zombies avec les autres joueurs. Et bien évidemment il faudra commencer la création du site « Web ».

5.3.1 Site Web

Pour cette deuxième soutenance, nous allons commencer la création d'une autre base de notre jeu : le site « Web ». Alexiane s'occupera de sa création avec l'aide de Lucas.

Celui-ci se composera tout d'abord d'une page d'accueil, où les logos de notre jeu et de notre groupe apparaîtront. Celle-ci nous permettra d'accéder à une page pour la présentation du projet. Cette page rassemblera l'histoire, les membres, la chronologie de la réalisation, les problèmes rencontrés, les solutions envisagées etc. Et vous trouverez également une autre page qui contiendra les liens sur les sites.

6 Conclusion

En conclusion, nous avons pris goût à la création de ce projet et à ce travail de groupe. C'est pourquoi, nous attendons avec impatience de vous présenter la suite du développement de notre projet à la deuxième soutenance.

7 Annexe

7.1 Map

Vous trouverez ci-dessous quelques images de notre carte.



FIG. 6 – *camp*



FIG. 7 – *végétation et fond de carte*

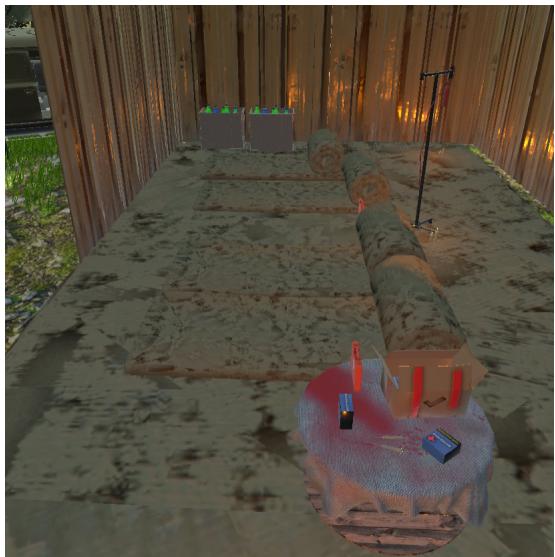


FIG. 8 – *sante*



FIG. 9 – *tente et tour*

7.2 Menu Principal

Voici des captures d'écrans des différentes interfaces du menu principal.

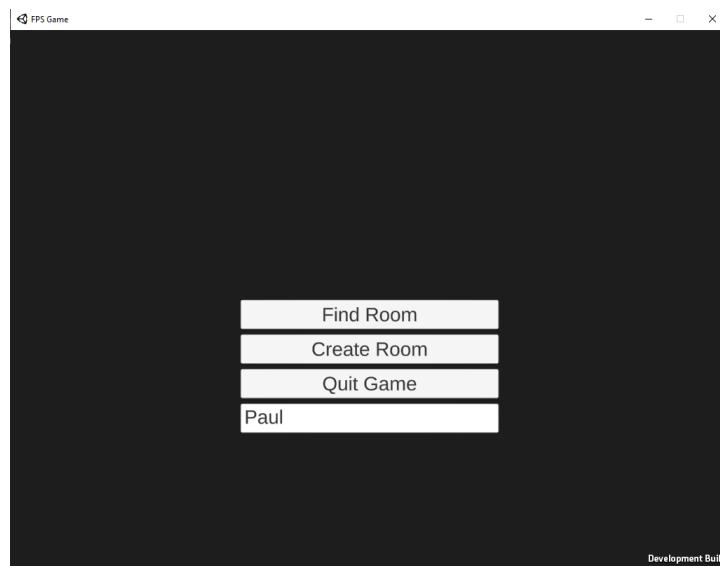


FIG. 10 – *Menu principal*

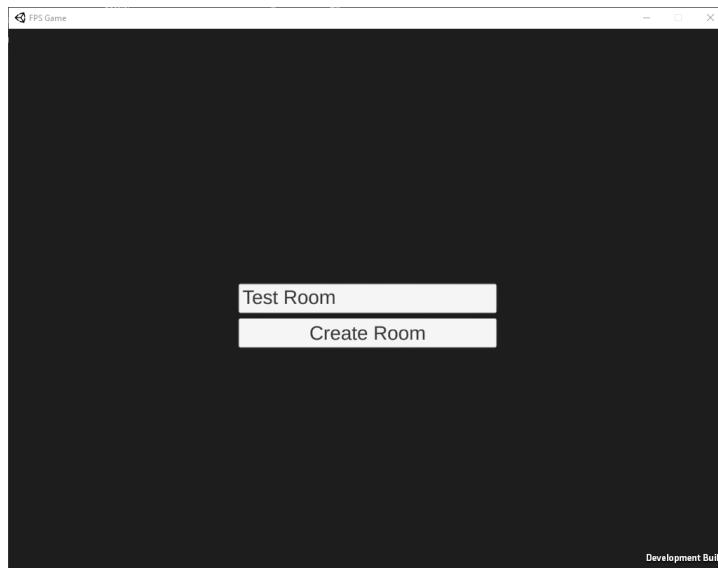


FIG. 11 – *Création d'une room*

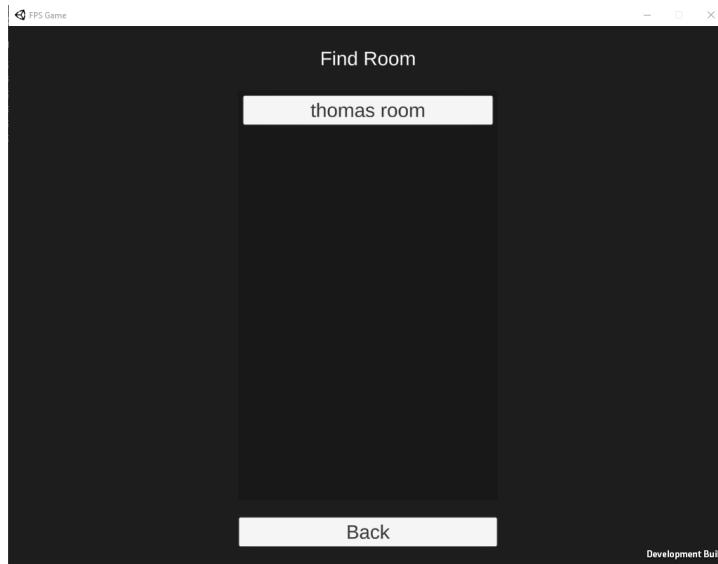


FIG. 12 – *Liste des rooms disponible*

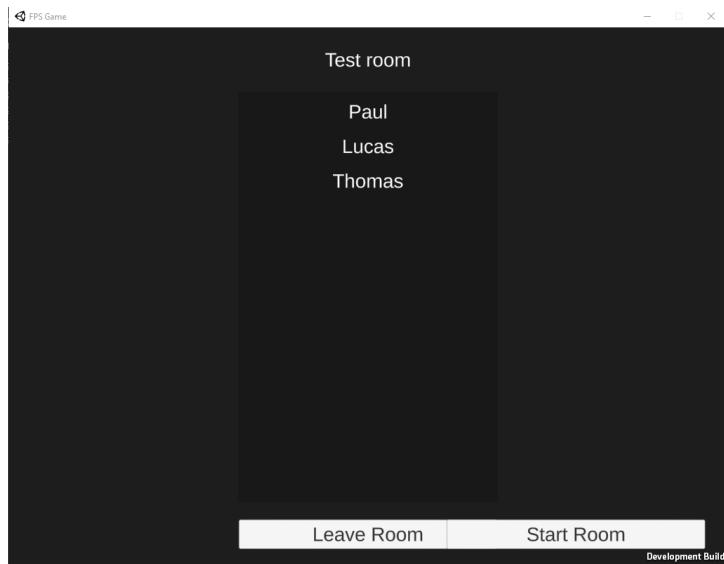


FIG. 13 – *Dans une room*

8 Sources

Dans cette section, vous trouverez toutes les références des sites que nous avons utilisé, ainsi que d'autres éléments dont nous nous sommes inspirés:

- Discord
- Google
- Asset Store
- Unity
- Blender
- Mixamo
- Abode Fuse CC
- Photon
- Call of Duty Warzone