

# GodMEL Corporation

## Rapport de Projet

---

---

### Proxima's Quest

---

---

Marco CHAN WITZEL  
Elias BOUARROUDJ  
Godric BOUTELOUP  
Lucile PELOU

Mardi 30 mai 2023



**PROXIMA'S**  
QUEST



---

# TABLE DES MATIÈRES

<b>INTRODUCTION</b>	<b>2</b>
<b>1 Vue générale du projet</b>	<b>3</b>
1.1 Les membres du groupe . . . . .	3
1.2 But du projet . . . . .	5
1.3 Moyens utilisés . . . . .	6
1.4 Répartition des tâches . . . . .	6
1.5 Ressenti général . . . . .	7
<b>2 Avancement du projet</b>	<b>8</b>
2.1 Site Web (Godric) . . . . .	8
2.2 Scénario et cinématique (Lucile et Marco) . . . . .	8
2.3 Mécaniques de jeu . . . . .	10
2.3.1 Le joueur (Elias et Lucile) . . . . .	10
2.3.2 Objets et magasins (Elias et Lucile) . . . . .	12
2.3.3 Ajouts divers et inventaire (Lucile) . . . . .	18
2.4 Interfaces et fonctionnalités . . . . .	23
2.4.1 Menus (Elias, Marco et Lucile) . . . . .	23
2.4.2 Multijoueur (Elias, Marco et Lucile) . . . . .	25
2.4.3 HUD et informations pour le joueur (Elias et Lucile) . . . . .	29
2.5 Intelligences artificielles . . . . .	31
2.5.1 Ennemis (Godric, Marco et Lucile) . . . . .	31
2.5.2 PNJs (Lucile et Marco) . . . . .	34
2.5.3 Boss (Elias et Godric) . . . . .	37
2.6 Graphismes (Godric, Lucile et Marco) . . . . .	40
2.7 Animations (Godric, Lucile et Marco) . . . . .	44
2.8 Musiques et sons (Elias) . . . . .	45
<b>3 Bilan du jeu</b>	<b>47</b>

## TABLE DES MATIÈRES

---

3.1 Les différents niveaux . . . . .	48
<b>Conclusion</b>	<b>50</b>



---

# INTRODUCTION

Dans le cadre de notre projet informatique de second semestre, nous avons décidé de réaliser un jeu vidéo : **Proxima's Quest**.

**Proxima's Quest** est un jeu de rôle de type action-RPG. Le joueur incarne un humain qui évolue dans un monde médiéval-fantastique en vue troisième personne. Ce monde est constitué de plusieurs royaumes, chacun gouverné par un dirigeant. La population vit sous le joug de ce gouvernement et des forces maléfiques qui imprègnent leurs mondes. Le joueur va évoluer dans le royaume le plus important de la planète.

Nous vous proposons ainsi de découvrir la réalisation du projet qui a été réalisé lors de notre second semestre.

---

# Vue générale du projet

## 1.1 Les membres du groupe

Avant d'entamer la présentation du déroulé de notre travail, un rappel des membres du groupe et le ressenti de chacun des membres s'imposent. Le nom du groupe ayant été choisi en fonction de nos prénoms respectifs, soit God pour Godric, M pour Marco, E pour Elias et L pour Lucile. Tous les membres du groupe sont dans la même classe, soit la E1.

### **Marco CHAN-WITZEL :**

Après l'obtention de mon premier ordinateur en février 2020, j'ai de plus en plus apprécié le monde de l'informatique jusqu'au point de vouloir étudier dans cette voie changeant ainsi mes choix d'orientation. C'est donc à travers les jeux soit en tant qu'utilisateur que j'ai découvert ce monde. Ne disposant pas de la spécialité NSI, j'ai appris comme beaucoup à écrire mes premières lignes de programmation durant mon premier semestre à l'EPITA.

Ce projet m'a permis d'apprendre et d'évoluer davantage à travers la création de notre jeu **Proxima's Quest**. Plusieurs éléments cruciaux m'intéressent particulièrement car ils m'ont souvent intrigué lors de mon utilisation personnelle telle que la relation entre le code du jeu et le jeu lui-même exécuté par l'utilisateur, la création d'interface ou encore l'implémentations du rendu graphique. Avec notre projet j'ai su répondre à ces questionnements et mes connaissances ont grandi d'un point de vue que je ne pouvais imaginer.

À l'avenir, je souhaite créer des applications ou même d'autres éléments informatiques pour améliorer le quotidien de chacun. Ce projet, qui est pour moi le premier, m'a permis d'apprendre à manipuler certains outils tels qu'Unity ou encore le C# afin d'accéder à une première ébauche de ce qu'est réellement un projet informatique en groupe.

### **Elias BOUARROUDJ :**

Le jeu vidéo, puis le développement en général, m'ayant attiré depuis tout petit, il était évident que je me tournerais vers une voie vers l'informatique. Ayant d'abord pris la spécialité NSI en première et terminale (où j'ai eu 20 au bac dans cette matière donc c'était rentable), je me suis orienté vers EPITA afin de continuer dans cette voie-là. Cependant, contrairement au reste de mon groupe, ce n'est pas mon premier projet de groupe, puisque j'en ai déjà fait au lycée (Il y avait même déjà un cahier des charges.).

Mais, ce projet est beaucoup plus évolué que le précédent, puisqu'il est en 3d et sur Unity (qui est fait pour cela, contrairement à Python). Ainsi, il m'a permis d'approfondir mes connaissances en C# et sur les projets de groupe, mais aussi d'en apprendre de nouvelles, notamment sur l'utilisation d'Unity, et sur la réalisation d'un jeu en 3d. Enfin, à la fin du projet, je suis un petit peu déçu puisqu'on a pas pu le finir à 100%. Cependant, j'en tire quand même de très bon moment à coder (surtout les boss) et surtout quelques nuits blanches.

### **Godric BOUTELOUP :**

La première console que j'ai reçue en primaire fut ma première expérience avec un jeu vidéo. J'ai commencé à jouer de plus en plus, avoir de nouvelles consoles, différents jeux, tout pour satisfaire ma soif d'aventure. Étant une partie très importante de ma vie, le jeu vidéo est donc devenu mon projet d'orientation pour le début, mais une fois arrivé en terminale, je me suis rendu compte que même si j'adorais ça, ça ne restait qu'une passion. Je me suis donc tourné vers de l'informatique générale qui m'a amené en école d'Ingénieur en informatique. Commencant par essayer de créer un fps sans succès, puis un platformer, je connais un petit peu Unity, même si le C# m'était inconnu jusqu'à maintenant.

Ce projet m'a donc permis d'agrandir le domaine de mes compétences ce qui pourrait être un réel atout pour pouvoir aller en alternance dans le futur. Que cela soit la cohésion d'équipe, le travail d'équipe et plein d'autres ce sont des atouts majeurs.

### **Lucile PELOU : (Chef de projet)**

La création de jeux vidéo m'a toujours attiré, mais venant d'un lycée qui ne possédait pas la spécialité NSI, je n'ai quasiment jamais touché à la programmation en dehors de certains

projets personnels. J'ai commencé il y a deux ans la fondation d'un serveur Minecraft, ce qui m'a permis de regarder le code (en Java) de certains plugins et de manipuler un peu de code pour la première fois. Et hormis une formation python faite pendant ses grandes vacances, je n'avais jamais fait quelconques projets touchant à la programmation avant d'entrer à EPITA.

La création de Proxima's Quest a permis de satisfaire ma soif de création d'un monde dans un jeu vidéo et m'a permis d'acquérir des nouvelles compétences pour les de futurs projets. Je trouve fascinant le fait de créer un univers entier et de le manipuler comme nous voulons qu'il soit. Ce projet m'a permis d'approfondir mes connaissances et surtout d'en acquérir des nouvelles que cela soit dans la façon de travailler individuellement ou en groupe ou concernant C# et UNITY. Je suis maintenant capable de faire la plupart des tâches sans beaucoup de recherches ou de

### **1.2 But du projet**

Quand nous nous sommes mis d'accord sur le type de jeu que nous allons faire, notre principal objectif était de faire un jeu du type Action-RPG avec de multiples fonctionnalités. Le but était de faire endurer au joueur une expérience de jeu difficile mais agréable. Il était question de faire des énigmes assez compliquées à résoudre mais pas trop pour que le joueur puisse quand même avancer sans devenir frustré par le fait d'être bloqué. C'était aussi faire des ennemis assez complexes à combattre mais garder beaucoup de moyens de repli pour le joueur et de façon de diversifier la façon de jeu. En somme le but du projet était de faire une aventure attrayante mais pas non plus quelque chose de confortant pour confronter le joueur à des épreuves. Quand nous allons sur un jeu vidéo, nous voulons du défi et c'est exactement dans cet esprit que Proxima a été développé, pas trop pour avoir l'espoir mais assez pour avoir le goût.

Nous avons donc imaginé au départ un jeu avec deux à trois niveaux ainsi que des boss et des ennemis. Ceux-ci étant muni de plusieurs attaques pour diversifier un maximum l'expérience de jeu du joueur. Le joueur quant à lui a plusieurs moyens pour augmenter ses capacités et

ainsi avoir plus de facilité pour les niveaux compliqués. L'idée de départ était donc un système d'expérience et d'équipement évolutif.

### 1.3 Moyens utilisés

Pour la réalisation de notre projet, nous avons mis en place plusieurs moyens de collaboration et de suivi. Tout d'abord, nous avons utilisé GitHub afin de travailler en collaboration sur les mêmes fichiers et de résoudre les conflits éventuels.

Nous avons également utilisé Discord pour communiquer, échanger des idées et travailler. C'est aussi sur cette plateforme que nous mettons tous les avancements, les problèmes rencontrés et les choses à faire. Discord a été la plateforme principale utilisée car vu l'ampleur de notre projet nous devons être perpétuellement en communication sur les changements. De plus nous étions souvent en vocal lors de nos sessions de travail pour nous aider les uns les autres.

En ce qui concerne le suivi de projet, nous avons utilisé Notion pour créer un tableau de bord de l'avancement actuel du projet et des deadlines d'un point de vue général.

Enfin, nous avons utilisé Google Docs pour collaborer sur la rédaction de documents tels que le scénario ou le Game Design.

Pour faire notre projet nous avons utilisé Unity, rider, le langage de programmation C# ainsi que Photon pour faire le multijoueur.

### 1.4 Répartition des tâches

Au départ, nous avons prévu cette répartition des tâches dans le cahier des charges :

Au final nous arrivons :

Le bleu foncé représente les membres principaux qui ont développé dans ces tâches et le bleu clair les membres qui les ont aidés.

Tâches	Marco	Elias	Godric	Lucile
Développement du joueur				
Scénarios/Dialogues				
Musiques et sons				
Multijoueurs				
IA : PNJ				
IA : Ennemis basiques				
IA : Boss				
Graphismes/ Animations				
Site Web				

Tâches	Marco	Elias	Godric	Lucile
Développement du joueur				
Menus				
Interfaces				
Scénarios et Dialogues				
Musiques et sons				
Multijoueur				
PNJs				
Ennemis basiques				
Boss				
Graphismes et Animations				
Site Web				

## 1.5 Ressenti général

Lors de la première soutenance, nous étions globalement en avance sur la plupart des prévisions faites pour celle-ci. Pendant cette deuxième période, nous avons dû faire face à plus de complications d'un point de vue technique et avons mis plus de temps à développer les fonctionnalités. En effet nous avons mis la barre plus haute en matière de fonctionnalités à implémenter et surtout l'implémentation du jeu et l'assemblage de celle-ci prennent du temps. Nous sommes tout de même parvenues à un résultat satisfaisant et les prévisions ont été globalement respectées.

---

## Avancement du projet

### 2.1 Site Web (Godric)

La réalisation d'un site internet pour le projet était un objectif programmé pour la deuxième soutenance. Nous avons fait un site de A à Z en se basant sur du HTML et du CSS, il est entièrement dynamique. Plusieurs pages sont accessibles entre elles, les documents de la première soutenance sont déjà présents sur le site. Le seul élément qui n'est pas encore présent est le lien pour télécharger le jeu, qui ne sera mis en place que pour la troisième soutenance avec la version finale. En ce qui concerne l'esthétique, le rendu ne s'est pas montré à la hauteur de nos espérances, nous avons donc décidé pour la suite de refaire le site en utilisant Bootstrap, qui est une collection d'outils HTML, CSS et Javascript apportant des éléments de site esthétiques et simples à utiliser.

Lors du dernier temps de travail, les changements n'ont pas été conséquents, en effet nous avons opté pour Bootstrap pour rendre le site un peu plus agréable, mais la majorité du site avait déjà codé lors de la deuxième soutenance. Les nouveaux ajouts ne sont que les documents des différentes soutenances, les outils utilisés ainsi que l'installateur de notre jeu.

### 2.2 Scénario et cinématique (Lucile et Marco)

L'histoire est une partie importante de notre jeu, même si sa compréhension reste non nécessaire pour permettre au joueur d'avancer. Lors de notre premier temps de travail nous

avons écrit le fil directeur de l'histoire, afin de situer les ambiances des différents niveaux que le joueur devra parcourir. Nous n'avons pas encore travaillé sur les détails ne jugeant pas cela nécessaire pour le moment. Pour les dialogues, quelques ébauches ont été faites.

Pour la deuxième soutenance, l'histoire ainsi que les dialogues des PNJs ont tous été terminés. Nous avons commencé à écrire les descriptions des artefacts (ceux en lien avec l'histoire) mais nous en avons encore à écrire. Nous avons préféré consacrer plus de temps au développement du joueur et à l'implémentations des niveaux.

Nous avons également implémenté une cinématique qui débute quand le joueur démarre une nouvelle partie, cela lui permet de comprendre le personnage qu'il incarne et d'introduire l'histoire. Pour la réaliser, nous avons utilisé les options de séquence de Unity à l'aide de caméras virtuelles qui ont parfois eu des trajets prédéfinis. Le pack de cinématique de Unity a d'ailleurs été ajouté au projet. Pour gérer les caméras, la barre temporelle a été utilisée. Ces séquences vidéo ont été sauvegardées à l'aide de l'enregistreur Unity puis nous avons réalisé un montage à l'aide de Adobe Première Pro. Le composant permettant de lire des vidéos a ensuite été directement utilisé dans la scène liée à la cinématique.

Une fois la cinématique finie, le joueur apparaît dans un cachot en compagnie d'un autre prisonnier qui va l'accompagner pendant presque tout le long du niveau. Il va lui expliquer comment jouer et les différentes fonctionnalités auxquelles il a accès. Le prisonnier meurt avant la fin du niveau et lui confie la quête principale du jeu.

Une fois le boss du niveau d'introduction vaincue, le joueur a la possibilité de se téléporter au village pour s'évader de la prison. Une fois au village il peut y a différents PNJs et il peut interagir avec certains d'entre eux. Il peut également accéder aux zones suivantes. Il découvrira le reste de l'histoire par l'intermédiaire des différents PNJs, des ennemis et des artefacts.

L'écriture du scénario étant presque fini nous avons principalement fait des ajustements pour plus de facilité sur l'implémentation des niveaux. Les artefacts ainsi que les quêtes ont été peaufiner pour aiguiller le joueur sur le déroulement du jeu. Le but étant toujours de laisser

au joueur une interprétation personnelle mais de lui donner des indices pour qu'il ne reste pas bloqué. Les dialogues des PNJs ont été ajustés pour donner plus d'indices au joueur et coller avec les ajustements du scénarios.

## 2.3 Mécaniques de jeu

### 2.3.1 Le joueur (Elias et Lucile)

Pendant la première partie de travail, nous avons bien avancé le développement du joueur afin d'être plus souple pour la suite. Nous avons donc créé un script pour permettre au joueur de marcher, courir, sauter et de s'accroupir. Celui-ci utilise un système simple au travers des méthodes "transform.translate", "transform.Rotate" et "rigidbody.Addforce" pour le saut. Cependant, il arrivait que le joueur ait encore des problèmes de collisions avec son environnement. Nous avons donc dû baisser la vitesse de celui-ci. En passant sur ces méthodes, nous avons réglé beaucoup de problèmes. Néanmoins, il arrivait que le joueur ait encore des problèmes de collisions, par exemple il est souvent éjecté assez loin si un ennemi ou un PNJ nous pousse. Le joueur peut également attaquer les ennemis (différencié des PNJs grâce au tag "enemy") grâce à son arme, lorsque la boîte de collision touche l'ennemi il prend des dégâts. , celui-ci prend des dégâts. De plus, cette attaque dispose d'un compte à rebours et son arme perd sa capacité à faire des dégâts lorsque celui-ci n'attaque pas pour éviter les collisions non intentionnées.

Notre jeu étant composé d'objets que le joueur peut manipuler, nous avons implémenter une méthode pour qu'il puisse ramasser les objets qu'il trouve au sol, en sachant que ces différents items se placeront directement dans son inventaire si celui-ci n'est pas plein. C'est le script "Pickup" qui s'occupe de cela et vérifie si le joueur n'est pas à proximité d'un objet grâce au Physics.Raycast. Pendant le ramassage de l'objet, le joueur ne peut plus bouger, ni attaquer.

Au commencement, nous voulions que le joueur puisse bouger sa caméra librement autour de lui. Nous avons donc créé un script pour cela. Cependant, nous nous sommes dit que cela simplifierais le jeu si nous donnions la capacité de viser un ennemi au joueur. L'objectif du lock était que le joueur puisse placer sa caméra afin d'avoir une vue d'ensemble lui permettant de se

déplacer tout en observant l'ennemi le plus proche pour simplifier le combat contre ce dernier. Toutefois, il n'est pas totalement opérationnel actuellement.

En ce qui concerne le joueur pour la deuxième soutenance, nous avons changé totalement notre prefab pour le rendre entièrement modulable. Ce changement nous a permis d'ajouter des fonctionnalités plus facilement comme le fait de pouvoir changer les équipements du personnage. La caméra du joueur a quant à elle été modifiée afin de prendre en compte les collisions. Elle ne passe plus à travers les murs et permet au joueur de se diriger dans des espaces plus restreints. Dès que la caméra détecte un objet entre sa future position et le joueur, elle se place un peu avant le point de collisions. De plus, nous avons ajouté les contrôles manettes, de la même manière que ceux du clavier. Cependant, ils ne sont pas encore bien implémentés, ne permettant pas de jouer au jeu seulement avec la manette. Enfin, nous avons refait le saut, qui avant ne se marchait que sur certaines zones. Maintenant, celui-ci fonctionne sur toute la carte. En effet, le jeu vérifie bien si un sol est en dessous du joueur avant de lui donner la permission de sauter.

Enfin pour la dernière partie de notre travail, nous n'avions plus grand-chose à faire sur le développement du joueur. Le joueur peut maintenant gagner de l'expérience et de l'argent en tuant les monstres. Nous avons également ajouté une échelle car nous nous sommes rendu compte que le joueur ne pouvait pas remonter s'il tombait dans certaines zones. Pour ce faire nous avons changé la façon d'avancer quand le joueur appuyait pour monter à l'échelle. Quand le joueur avance alors qu'il agrippe une échelle, un objet en dessous de lui monte comme un ascenseur. Nous avons changé quelques petites autres choses minimales, globalement le travail avait déjà été fait les soutenance précédentes.

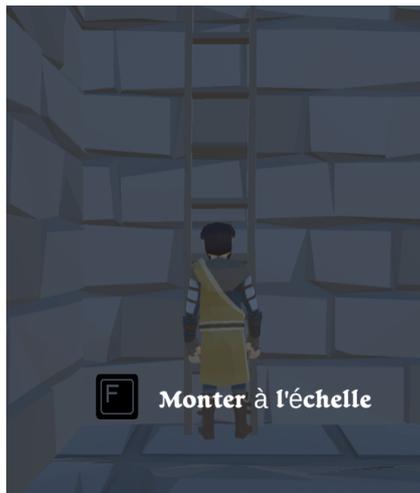


FIGURE 2.1 – Le joueur devant l'échelle



FIGURE 2.2 – Joueur montant à l'échelle

### 2.3.2 Objets et magasins (Elias et Lucile)

Lors de notre premier temps de travail, nous avons commencé à implémenter des objets. Nous avons donc créé une classe Item héritant de la classe ScriptableObject. Il y a également différents types d'items dans le jeu, c'est pourquoi nous avons fait un script pour chaque type d'items héritant de la classe Item.

La classe Item possède les attributs suivant :

- Un nom
- Une description
- Le visuel de l'objet dans l'inventaire
- Le modèle 3D de l'objet
- Un booléen vérifiant si l'objet est stackable
- Le nombre maximum d'objets de même type stackable
- Une autre version du modèle 3D
- Le prix de l'objet

En effet, certains objets nécessitant une manipulation du joueur, tel que l'épée ou encore le bouclier, possèdent deux modèles 3D afin de faciliter le rendu en jeu. Lors de cette première partie de travail nous avons implémentés trois types d'objets en jeu : les équipements, les armes et les potions. Pour les prochaines soutenances, nous implémenterons plusieurs autres types comme la nourriture ou encore les artefacts.



FIGURE 2.3 – Potion de force



FIGURE 2.4 – Bouclier

Le joueur peut donc ramasser des objets sur le sol, mais ce n'est pas la seule méthode pour obtenir des objets. En effet, afin de permettre l'expérience de jeu plus agréable, le joueur peut également se procurer les objets de base en les achetant aux boutiques du village. Pour cela, nous avons inclus un système monétaire dans le jeu. Cette monnaie, appelée "Écu", pourra être trouvée par le joueur en tuant les différents ennemis et boss. Comme dit plus haut, le joueur peut voir sa monnaie en tout temps dans un emplacement spécifique. Cette statistique se met à jour quand le joueur achète des objets au magasin et prochainement lorsqu'il récupère de l'argent en tuant des monstres. Pour le moment il n'a aucun moyen de récupérer de l'argent. De plus, en passant sa souris sur un des objets vendus au magasin, les informations le concernant apparaissent, c'est-à-dire son nom et sa description.

Pour faire le magasin nous avons créé une classe Slots avec les variables suivantes :

- L'item
- Le visuel de l'item
- Le prix de l'item
- Un booléen vérifiant si les informations concernant l'item sont ouvertes

Ce script est attribué à tous les emplacements. Cela nous permet de rendre le magasin dynamique si nous voulons changer les items présents ou si nous voulons créer d'autres boutiques.



FIGURE 2.5 – Boutique de Potions

Dans un premier temps nous avons donc ajouté plusieurs types d’objets tels que les armes, les armures ou encore les potions. Les potions ont subi un changement au niveau de leurs attributs et sont désormais munies d’un entier qui va donner le niveau de la potion et de deux méthodes pour utiliser (donner son effet au joueur) et annuler l’effet de la potion. Lorsque le joueur utilise une potion, un minuteur apparaît sur son écran pour l’informer de la durée restante de l’effet de la potion, la potion de vie n’étant pas concernée puisqu’elle permet au joueur de se régénérer instantanément.



FIGURE 2.6 – Utilisation d’une potion



FIGURE 2.7 – Compteur d’effet

Nous avons implémenté deux nouveaux types d’objets, soit les fleurs et la nourriture. Les fleurs ont été ajoutées en tant qu’objet, car nous en avons besoin pour une quête particulière, mais le reste des objets non particulier sera quant à eux dans le type d’objet artefact que nous implémenterons par la suite. De même que les potions, la nourriture est considérée comme un consommable.

Un autre type d’objet a été ajouté : les sorts. Les consommables (les potions et la nourriture) ainsi que les sorts ont été ajouté à l’inventaire rapide pour que le joueur puisse les équiper. Les consommables peuvent être équipés facilement à l’aide de la flèche droite comme les armes et

les boucliers dans leur propre emplacement. Une fois équipé le consommable peut être utilisé par le joueur à n'importe quel moment (sauf lorsque l'inventaire est ouvert). Les sorts ne sont obtenables qu'une seule fois et ne sont accessibles qu'à partir de l'inventaire principal. Ils peuvent ainsi être équipés à partir de là. Une fois que le joueur obtient un sort, il se colore dans son inventaire. Pour le moment, nous n'avons ajouté que deux types de sort : la boule de feu et le sort de téléportation. Le sort de la boule de feu est un sort qui se projette sur la carte, se détruit dès qu'il rencontre un objet, et fait des dégâts si ce qu'il touche est un ennemi. Si le joueur clique une fois, le joueur se prépare à la lancer, ce qui devra être montré par le biais d'un curseur ou d'une animation, puis réappuyer pour lancer le sort. Enfin, celui-ci, après le lancer, disposera d'un temps d'attente de 50 secondes afin d'éviter l'utilisation en continu. Le sort de téléportation permet de revenir dans les lieux déjà parcourus. Il ouvre un menu, et le joueur fait son choix parmi les scènes présentes sur le menu.



FIGURE 2.8 – Inventaire rapide



FIGURE 2.9 – Inventaire des sorts



FIGURE 2.10 – Boule de feu

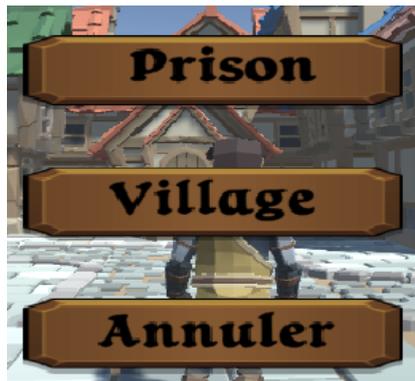


FIGURE 2.11 – Menu de téléportation

Avec l'arrivée, des fleurs et de la nourriture, nous avons également implémenté des magasins correspondants. Concernant les magasins, nous leur avons ajouté plusieurs pages qui peuvent être débloquées en fonction des boss déjà battus. Cet ajout permet d'augmenter la qualité des objets vendus en fonction du niveau du joueur.



FIGURE 2.12 – Armurier

Lors de notre dernier temps de travail nous avons ajouté un nouveau type d'objet : les artefacts. Ces objets nous permettent d'implémenter les scénarios pour donner des indices et des informations au joueur. Les types d'objets ont été peaufiner, par exemple les épées infligent maintenant des dégâts différents en fonction de leurs raretés et prix. Pour diversifier les objets disponibles, nous avons implémenté plusieurs nouveaux objets, comme des armures, des épées,

des boucliers ainsi que des potions de niveaux supérieurs. Les potions ont subi un ajustement étant donné qu'elle donnait trop d'avantage au joueur. Le joueur peut également obtenir 5 sorts pour permettre de diversifier ses attaques et mettre plus de dégâts à ses ennemis. Les magasins ont été améliorés en conséquence des changements mais étant donné qu'ils sont dynamiques, cela n'a pas été une tâche compliquée. Pour finir tous les objets ont vu leurs prix, visuels et descriptions mis à jour. Les variables des objets ont été transformées en tableau de variable et nous avons ajouté la variable pronom.

Concernant les sorts ajoutés lors de ce dernier temps de travail, nous retrouvons un sort de soin, un sort qui lance une décharge électrique dans une certaine zone et un sort qui fait apparaître des petites flammes autour du joueur afin de faire des dégâts aux ennemis. Ces sorts ont été faits de telle sorte qu'ils apportent un soutien au joueur tout en ne l'avantageant pas trop, au travers des temps de blocage du sort.

Les visuels ont globalement été tous retravaillés, voici quelques images illustrant les propositions précédentes avec leurs nouveaux visuels :



FIGURE 2.13 – Un artefact



FIGURE 2.14 – Nouvelle armure



FIGURE 2.15 – Des indices sur l'histoire



FIGURE 2.16 – Inventaire des sorts



FIGURE 2.17 – Cercle de feu



FIGURE 2.18 – Sort de soin

### 2.3.3 Ajouts divers et inventaire (Lucile)

#### Inventaire principal

Pour gérer tous ces objets, le joueur possède un inventaire. Dans celui-ci se trouve 12 emplacements. Certains objets peuvent être empilés, comme les potions par exemple, avec un maximum de 5 par pile. Il peut manipuler chacun des objets en cliquant dessus ou en utilisant les touches associées à chaque action. Les actions sont différentes selon le type de l'item, pour les équipements et les armes, le joueur peut les équiper, tandis que pour les potions, il peut les

utiliser. Il peut également jeter les objets au sol ce qui les fera apparaître à côté de lui. Pour le moment, il peut seulement équiper les armes et les boucliers. Pourtant, il ne peut pas encore utiliser le bouclier et les divers consommables mais uniquement les armes.

### **Inventaire rapide**

Le joueur possède également une barre de trois emplacements ou chaque emplacement est réservé à un type d'objet spécifique. On l'appelle une barre d'inventaire rapide. Les objets qui sont actuellement équipés sur lui seront affichés dans cette barre d'inventaire rapide. Le premier emplacement est réservé pour une arme, le deuxième pour le bouclier et le troisième pour un consommable, que cela soit des potions ou de la nourriture. Cette barre d'inventaire rapide est toujours visible par le joueur. En appuyant sur la flèche du haut, il changera son arme actuelle par la suivante dans son inventaire. De même pour la flèche du bas qui lui permettra de changer son bouclier par le suivant dans son inventaire. Une fois l'épée équipée, il peut l'utiliser pour attaquer. Tout comme les magasins, en passant la souris sur un item d'un des deux inventaires, le joueur peut voir les informations sur l'item en question.

C'est le script "inventory" qui gère la plupart des choses qui sont liés à l'inventaire.

Nous avons créer dans ce script une classe "ItemInInventory" qui contient trois variables :

- L'item (de type Item)
- Le type d'inventaire
- Le nombre d'item

L'inventaire est géré par une liste d'ItemInInventory. Nous avons crée plusieurs méthodes pour faciliter son utilisation :

La méthode AddItem() qui va ajouter un item à l'inventaire, cette méthode est utilisée quand le joueur ramasse un item ou en achète un. La méthode RemoveItem() qui va supprimer un objet de l'inventaire, est utilisée lorsque le joueur jette un item au sol. La méthode RefreshContent() qui va mettre à jour le visuel de l'inventaire. Pour finir, des méthodes sont présentes pour vérifier si l'inventaire est rempli, ouvrir le panel d'action, pour utiliser, équiper ou encore jeter les objets.

Pour les emplacements de l'inventaire, nous avons utilisé la même méthode que pour ceux des magasins.



FIGURE 2.19 – Information concernant la potion de force

Une modification mineure, mais essentiel a été faite pour l’inventaire. Lorsque celui-ci est ouvert, le reste du jeu est mis en pause. Dans un premier temps, nous avons implémenté les armures, mais pas la possibilité de pouvoir les équiper, c’est maintenant le cas. Le bouclier peut maintenant être utilisé une fois qu’il est équipé, ce qui va augmenter la résistance du joueur. Concernant l’utilisation, l’équipement ou autres actions, tout est maintenant accessible à partir d’un panel d’action en cliquant sur l’objet ou avec les touches correspondantes, que cela soit dans l’inventaire rapide, l’inventaire d’équipement ou l’inventaire principal.



FIGURE 2.20 – Inventaire d’équipement



FIGURE 2.21 – Joueur entièrement équipé



FIGURE 2.22 – Panel d’actions

Le joueur possède plusieurs compétences, soit : la force, la hauteur du saut, la vitesse, la vie et la résistance. Tous ces attributs peuvent être augmentés temporairement grâce aux potions. Afin de permettre au joueur de les augmenter définitivement, nous avons ajouté un système d'expérience. Quand le joueur gagne un niveau, il gagne ce qu'on appelle un "point de compétence" qu'il peut répartir parmi les différentes compétences dans un inventaire accessible dans l'inventaire principal.



FIGURE 2.23 – Inventaire d'expérience

Pour améliorer l'expérience de jeu et rajouter du défi au jeu, nous avons implémenté des quêtes. Elles se débloqueront tout au long de l'aventure et permettront aussi au joueur de revisualiser des indices qui lui ont été donnés. Cela va aussi être un moyen pour récompenser le joueur que cela soit en matière d'expérience ou en matière d'argent. Le joueur peut accéder à ses quêtes dans un inventaire accessible dans l'inventaire principal. Il s'adapte en fonction du déroulement des quêtes et du nombre. Le joueur a aussi la possibilité de trier les quêtes en fonction de leur état (non débloquées, en cours, terminées, toutes).



FIGURE 2.24 – Inventaire de quêtes



FIGURE 2.25 – Déroulement d'une quête

Pour cette dernière soutenance nous avons refait totalement la façon d'accéder à l'inventaire pour un rendu beaucoup plus esthétique et pratique. Le joueur a maintenant le choix d'accéder à l'inventaire qu'il souhaite peu importe sur lequel il est. Au niveau des quêtes nous les avons terminés, elles sont maintenant totalement opérationnelles et toutes implémentées. Il y en a cinq et une dernière qui est une quête bonus donnée par un des PNJ au village. Pour permettre au joueur de repérer plus facilement les objets que nous voulons lui donner nous avons ajouté des coffres. Ils ne sont pas toujours utilisés car il ne faut pas que le joueur puisse trouver tous les objets dispersés sur les cartes aussi facilement. Ce coffre fonctionne à l'aide d'une animation et de la méthode habituelle d'ajout dans l'inventaire.



FIGURE 2.26 – Inventaire



FIGURE 2.27 – Coffre



FIGURE 2.28 – Ouverture d'un coffre

## 2.4 Interfaces et fonctionnalités

### 2.4.1 Menus (Elias, Marco et Lucile)

Concernant les menus du jeu, nous avons développé le menu principal, permettant de lancer une nouvelle partie, de charger une sauvegarde précédente, seulement en solo, ou de quitter le jeu. Nous avons aussi fait un menu permettant de se connecter à une partie multijoueur. Ensuite, si nous lançons le jeu, nous disposons d'un menu pause, d'où nous pouvons revenir au menu principal, changer ses options, ici certaines touches et le plein écran du jeu, ou encore sauvegarder. Les données sauvegardées sont mises dans un fichier texte et le jeu se charge de les lire pour bien repositionner le joueur ou remettre son inventaire correctement. Actuellement, celle-ci est accessible dans un fichier texte, permettant de modifier les sauvegardes à n'importe quel moment du jeu. Dans le futur, les sauvegardes seront également munies d'un système d'encryptage. Enfin, un écran de Game Over est mis en place si le joueur meurt.



FIGURE 2.29 – Menu Principal du Jeu



FIGURE 2.30 – Menu Pause



FIGURE 2.31 – Menu de Sauvegarde



FIGURE 2.32 – Menu des options

Pour la deuxième soutenance, nous avons ajouté la possibilité d'augmenter et de baisser le son dans le menu des options. Nous avons aussi rendu le menu de changements de touches déroulant et avons ajouter plus de possibilités de changements de touches. De plus, si vous associez une touche déjà associée à une action, et se désassocie de la première, se met sur la nouvelle et envoie un message d'avertissement au joueur. Nous avons aussi mis en place une barre de chargement lors des changements de scènes longs afin de rendre l'attente plus sympathique. Enfin, les sauvegardes, qui avant étaient modifiables très simplement, sont dorénavent encryptées à l'aide de la méthode d'encryptage du chiffre de Vernam. La clé de celui-ci est sélectionnée aléatoirement avant la sauvegarde puis encrypte.

Pour finir nous avons finalisé les menus en modifiant quelque peu comment ils étaient implémentés en jeu. Nous avons ajouté un script qui gère les options que choisit le joueur et il est maintenant en "ne pas détruire au démarrage". Cette modification nous a permis de faire en sorte que le joueur puisse modifier ces options dans le menu principal et non d'abord de devoir se connecter dans la partie. En matière d'options nous en avons ajouté une supplémentaire, la traduction. En effet tout le jeu peut être traduit en anglais si le joueur le souhaite. Au niveau des options déjà implémentées, le joueur peut maintenant modifier toutes ces touches. Les sauvegardes ont aussi été modifiées puisque le script d'encryptage provoquait des récursions infinies et des problèmes de pertes de données. Nous avons donc changé le système pour qu'il ressemble plus à un code Cesar. Malgré ce problème, le jeu peut dorénavant sauvegarder toute une partie, sur trois fichiers différents.



FIGURE 2.33 – Menu des options

## 2.4.2 Multijoueur (Elias, Marco et Lucile)

En ce qui concerne le multijoueur, nous avons posé les bases en utilisant Photon comme il était prévu. En essayant de connecter deux joueurs, certains problèmes sont survenus car les explications que l'ont suivait portaient sur un jeu en deux dimensions. Par exemple, la caméra du joueur client étaient celle du joueur hôte et inversement pour la caméra du joueur hôte. La solution était de gérer correctement l'appartenance des points de vues dans le script du joueur intervenant dans le multijoueur. Ainsi, grâce aux composants liés au "Photon View", notre multijoueur permet à deux joueurs de se connecter sur la map "Village" impliquant la synchronisation des positions et des animations des joueurs. Toutefois, tout n'est actuellement pas implémenté, par exemple les menus ou encore les IAs.

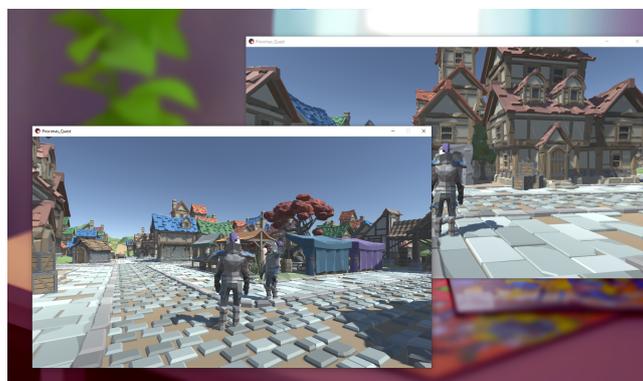


FIGURE 2.34 – Illustration du Multijoueur

L'ajout de plusieurs modifications a été fait dans le multijoueur pendant la deuxième séance de travail. En effet, nous avons implémenté un écran d'attente de joueurs, car nous souhaitons que la partie coopérative de notre jeu soit bloquée à deux joueurs empêchant l'avancée de l'histoire lorsqu'un seul joueur est connecté.

À l'inverse, nous avons empêché la possibilité de se connecter à plus de deux joueurs affichant un message d'erreur lorsqu'un troisième joueur essaie de se connecter. Le joueur essayant de se connecter ne passe pas par le menu en attente et arrive directement dans la partie.

Nous avons dissocié l'apparence des deux joueurs, le maître de la session est représenté par l'apparence du joueur principal de l'histoire solo et le client par un autre personnage féminin. Nous avons aussi dissocié les inventaires des deux personnages afin que chaque joueur ait son propre inventaire. L'inventaire étant créé directement dans la scène Unity, il est directement séparé en fonction des joueurs facilitant la résolution du problème.

Nous avons ensuite essayé de ramasser des items au sol de manière à ce qu'ils disparaissent de la scène pour les deux joueurs et apparaissent dans le bon inventaire. Des difficultés ont été rencontrées, car la destruction de l'item se fait facilement lorsque le client maître souhaite détruire l'objet. Or, lorsque le client souhaite détruire un objet, celui-ci n'a pas la permission de le détruire. Nous avons donc essayé plusieurs méthodes comme par exemple de transférer la requête au maître lorsque le client souhaite détruire un objet. Ainsi, des recherches devront être faites pour que cette fonctionnalité marche pour le client maître et pour le client.

Enfin, un problème persiste dans notre multijoueur malgré de nombreuses recherches de débogage. Lorsque l'on passe d'une scène solo à multijoueur, il est impossible de créer une session. Il est dit que le type de serveur est bien un client maître comme demandé, mais le type ne correspond pas malgré tout.



FIGURE 2.35 – Menu Attente



FIGURE 2.36 – Erreur : Session Pleine

Les avancées précédentes et la deuxième soutenance nous ont fait réaliser notre retard et nous avons, lors de la période dont nous disposons, essayés d'intensifier nos efforts sur ce domaine. Premièrement, pour ce qui est du menu en attente, nous avons implémenté dynamiquement les diverses erreurs facilitant à la fois l'utilisation mais aussi la traduction de celles-ci. Les diverses erreurs sont donc la session pleine, un nom vide et une session inexistante ou déjà créée pour rejoindre ou réaliser une nouvelle partie. L'instanciation des joueurs a aussi été revue.

Nous n'avons malheureusement pas pu implémenter les différents niveaux comme prévu en multijoueur. Cependant, l'ensemble des divers spécificités le sont. Les ennemis basiques ont donc été modifiées pour pouvoir fonctionner sur notre multijoueur. La gestion des deux joueurs a donc été faites en fonction de la distance de ceux-ci par rapport à l'ennemi. De la même manière, les boss ont dû gérer la gestion des deux joueurs accompagnés de leurs diverses phases tout en étant synchronisés à travers photon pour la jouabilité des deux joueurs.

À l'image des ennemis, les PNJs tel que les marchands ont aussi récupéré les deux joueurs pour pouvoir interagir avec eux lorsqu'ils se rapprochent. Ainsi, leurs positions sont synchronisées tandis que les menus d'interactions ne le sont pas afin de créer des boutiques et interactions indépendantes aux joueurs.

Les spécificités des joueurs sont celles qui ont causé le plus de difficultés et provoqués de nombreux problèmes. En travaillant avec les RPC de photon, nous sommes parvenus à récupérer des items et avons implémentés la suite de cela comme le fait de pouvoir l'équiper ou le jeter.

Les requêtes de suppressions et de créations d'objets se fait donc par l'hôte tandis que le client les transfert à l'hôte. Au départ, des soucis d'échelle étaient rencontrés lorsqu'on équipait une arme ou un bouclier. Cela est dû au fait que l'instanciation photon sous un parent ne fait pas la différence d'échelle entre coordonnées dans le monde et sous un parent. Pour que les objets soit fonctionnels en multijoueur, les scripts de l'inventaire et l'inventaire rapide ont été modifiés pour qu'ils soient désynchronisés entre les joueurs.

Pour ce qui est de l'utilisation des objets tel que les potions ou les divers types de nourritures, cela est directement lié aux propriétés du joueur que nous pouvons développer ci-dessous :

En effet, les diverses propriétés telles que la vie, l'expérience ou encore l'argent ont été à la fois séparées entre les deux joueurs car leurs attributs ne sont pas les mêmes mais aussi synchronisés grâce aux RPC de photon pour pouvoir détecter les attributs des autres joueurs.

La mini carte a aussi été reprise de l'univers solo et a été désynchronisée grâce à l'affichage de Unity. La modification du script et de son emplacement dans la scène a été nécessaire pour que la carte soit fonctionnelle.

En ce qui concerne les problèmes rencontrés, le fait de s'inspirer des scènes déjà faites en solo nous a grandement aidé mais ont nécessité de nombreuses modifications puisque les résultats voulus diffèrent. Les problèmes de synchronisation ou ceux qui concernent photon nous ont globalement retardé dans l'objectif final que nous avons de ce projet cela étant dû à la difficulté de correction de ceux-ci et à la reconstruction de l'exécutable à chaque nouvel essai de correction. En effet, certains scripts ou objets étaient présents dans la scène à l'avance puisque il était facile de récupérer le joueur qui était seul dans la scène. Pour remédier à la difficulté d'obtention des joueurs dans le multijoueur, nous avons parfois associé directement le script au préfabriqué du joueur.

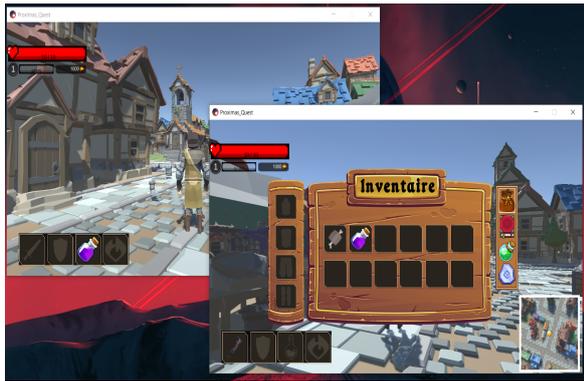


FIGURE 2.37 – Inventaires et Items

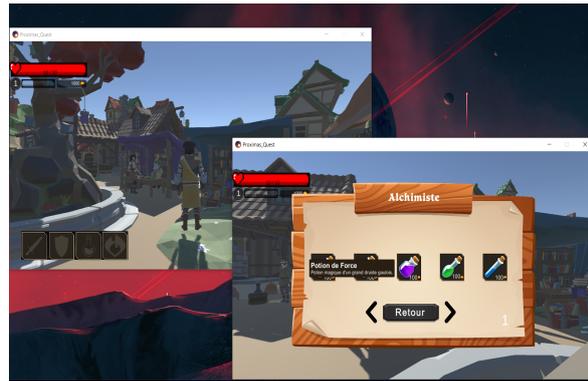


FIGURE 2.38 – Boutique indépendante

### 2.4.3 HUD et informations pour le joueur (Elias et Lucile)

A l'écran le joueur peut voir plusieurs informations essentielles. Tout d'abord sa barre de vie, fonctionnant par progression, ses points de vie exacts n'étant pas visibles. Il peut également voir sa monnaie actuelle et son inventaire rapide.

Le joueur étant capable d'interagir avec son environnement, nous avons donc mis en place une boîte d'informations dynamique qui s'affiche si nécessaire. Cela lui donne des informations concernant la touche à appuyer et l'action qu'il veut effectuer. Une deuxième boîte d'informations, elle aussi dynamique, a été mise en place pour l'informer des actions effectuées ou de l'impossibilité de certaines d'entre-elles.



FIGURE 2.39 – Action possible



FIGURE 2.40 – Action réalisée par le joueur



FIGURE 2.41 – Interface HUD

En ce qui concerne l'HUD nous avons rajouté deux emplacements dans l'inventaire rapide afin que le joueur puisse voir son consommable et son sort équipé, comme dit précédemment. De plus, nous avons rajouté son niveau actuel et sa barre d'expérience ainsi qu'une mini carte. Nous avons également modifié la barre de vie pour afficher plus clairement sa vie actuelle et nous avons fait de même pour la barre d'expérience. Concernant la mini carte, elle se situe en bas à droite de l'écran du joueur et se désactive lors de certaines actions tel que le fait de parler avec des PNJs. Elle fonctionne simplement avec une caméra au-dessus du joueur qui filme la scène, ce qui n'est pas la chose la plus optimale, mais permet une lisibilité plus grande de la scène, ce qui pourrait possiblement être amélioré dans le futur.



FIGURE 2.42 – Changement de l'HUD

Lors de notre dernière phase de travail nous n'avons pas ajouté de nouveaux affichages. Cependant nous avons changé les graphismes de ceux déjà présents. Nous avons modifié l'affichage du texte d'information et rendu dynamique l'affichage de l'image du bouton pour indiquer au joueur l'interaction qu'il peut effectuer. Pour faire cela nous avons fait un masque sur le bouton tout noir avec un texte représentant la touche choisie par le joueur pour les interactions. Globalement l'HUD reste pareil à la soutenance précédente vu qu'il n'y avait pas d'ajout à faire mais il y a eu des ajustements légers pour un rendu plus propre.



FIGURE 2.43 – Affichage du bouton d’interaction dynamique

## 2.5 Intelligences artificielles

### 2.5.1 Ennemis (Godric, Marco et Lucile)

Pour le moment, nous avons créé un type d’ennemi, le soldat. Nous lui avons implémenté un comportement de patrouille, lors de celui-ci, des temps de pauses régulier lui ont été intégrés pour un air plus naturel.

Ainsi, les ennemis se déplacent de manière autonome dans l’environnement en suivant un chemin prédéfini et en utilisant des nœuds placés à des points clés de la carte. Lorsque le joueur est assez proche, les ennemis détectent sa présence et activent leur comportement d’attaque. Ils poursuivent le joueur jusqu’à ce qu’il soit hors de portée ou jusqu’à ce que l’un des deux soit vaincu. Bien que les ennemis basiques soient programmés de manière relativement simple, leur comportement ajoute une couche d’imprévisibilité et de difficulté au jeu, offrant ainsi une expérience plus immersive pour les joueurs.



FIGURE 2.44 – Soldat

À travers nos recherches réalisées avant la première soutenance, nous savons comment implémenter nos ennemis selon l'idée que nous avons d'eux dans l'univers de notre jeu. Grâce à cela, il devient très abordable pour nous de les implémenter simplement lorsque nous en aurons besoin. Ainsi, nous avons ajouté, pour exemple, une araignée capable de marcher de manière aléatoire entre plusieurs points, d'attaquer, de traquer, de prendre des dégâts et de mourir. Néanmoins, nous avons fait face à une problématique d'animation due à sa forme. En effet, lors de l'animation de marche, le corps de l'araignée se dirigeait dans la mauvaise direction. En modifiant la position de celui-ci, nous avons pu la réorienter dans le sens voulu. Un deuxième ennemi que nous avons implémenté est le Gobelin, il sera présent dans le niveau 0, pour que le joueur apprenne à se défendre. Son fonctionnement est similaire aux premiers ennemis que nous avons implémentés (le soldat), sauf que celui-ci ne se déplace pas en suivant un chemin précis pour simuler une ronde, il se déplace à des endroits aléatoires.

Les comportements des ennemis avaient déjà bien été travaillés lors des temps de travail précédent. Cependant nous avons décidé d'implémenter des nouveaux ennemis avec des comportements totalement différents. En effet les ennemis tels que les soldats, gobelins et araignées étaient presque tous basé sur le même type d'intelligence artificielle et surtout ils attaquaient le joueur à proximité. Avec l'arrivée des sorts pour le joueur lors de la dernière soutenance, nous avons décidé de rajouter cette fonctionnalité pour les ennemis également. En effet, il y a maintenant plusieurs nouveaux ennemis qui envoient des sorts sur le joueur et donc attaque le joueur à distance ce qui permet une bonne diversification des ennemis. En premier le Mage noir, cette nouvelle IA est le tout premier ennemi de type "mage", comme pour le soldat il fonctionne sur un système de patrouille dans la carte jusqu'à que le joueur soit à portée, il lancera donc un sort de base sur le joueur. Ce sort est très important car il suit le joueur peu importe ses mouvements. Vient ensuite le Mage de glace est le deuxième ennemi de type "mage", comme le premier il fonctionne sur un système de patrouille jusqu'à rencontre avec le joueur pour lui jeter un sort de glace qui lui aussi suivra le joueur jusqu'à le toucher ou heurter un élément du décor.

Nous avons également ajouté un géant qui est un ennemi bien plus imposant que les autres, sa carrure fait de lui un ennemi totalement différent à combattre par rapport aux autres. Pour

le toucher avec une arme le joueur doit être très proche de lui et doit faire attention à ne pas se prendre les coups assez conséquents du géant. L'arme du géant n'étant pas une épée, une massue ou une arme dans ce style, cela fait de lui un ennemi redoutable malgré sa lenteur qui pourrait faire croire le contraire.

En ce qui concerne le dernier type d'ennemis, nous avons affaire à un chevalier, cette IA est la plus complexe de tous les ennemis, en effet lui aussi fonctionne sur un système de patrouille, mais son attaque se distingue des autres, en effet il a trois attaques différentes pour rendre les combats plus immersif.

Au final il y a en tout sept types d'ennemis sachant que certains n'ont pas le même comportement. Par exemple pour les soldats, nous les avons diversifié pour répondre à nos besoins, en effet certains sont maintenant directement en recherche de l'adversaire et ne patrouillent pas. Ils sont conçus uniquement pour attaquer directement l'adversaire dès qu'ils apparaissent. L'araignée a quant à elle subi un petit changement pour l'une d'entre elles. En effet, conformément à notre scénario, le PNJ accompagnant le joueur dans le premier niveau doit mourir. Pour cela nous avons donc fait une araignée qui doit d'abord attaquer le PNJ et le joueur une fois celui-ci mort.

Lors des soutenances précédentes, nous avons des problèmes avec la méthode LookAt() qui permet aux ennemis de regarder le joueur. Celle-ci les faisait bouger d'une manière anormale sur l'axe Y. Nous avons enfin trouvé la cause du problème et l'avons réglé en bloquant cet axe dans la méthode. Un autre problème que nous avons résolu est l'animation de mort qui n'était pas au sol. En effet les ennemis (ainsi que le joueur quand il mourrait) se retrouvaient à 2 mètres du sol mort. Au départ nous avons pensé à des problèmes de NavMesh ou de capsule de collision mais c'est en vérité l'animation qui posait problème. Pour régler cela nous avons ajouter le "Motion" en paramètre dans l'animation, ce qui a pour effet d'autoriser les animations de mouvements sur le transform des ennemis. En faisant cela les ennemis meurent bien au sol, ce qui est même beaucoup plus naturel et réaliste.



FIGURE 2.45 – Mage noir



FIGURE 2.46 – Géant

## 2.5.2 PNJs (Lucile et Marco)

En accord avec la recherche sur le fonctionnement des ennemis, nous avons aussi fonctionné avec la notion de "State Machine". Nous avons donc créé un PNJ qui se promène dans le village à l'aide de deux comportements dont celui nommé "Idle" et l'autre "Promenade". Les scripts associés à ces états lui permettent de marcher de point en point donnant l'impression qu'il se promène dans le village. De plus, des retours en Idle sont programmés pour un air plus naturel.

En découle le PNJ interactif, le prêtre, dont les scripts ont été modifiés par rapport au simple PNJ pour que celui-ci s'arrête et nous regarde lorsque l'on s'approche de lui à l'aide d'une "Box Collider". Par la suite, à l'aide du Canvas, une proposition pour dialoguer avec lui s'affiche. En appuyant sur "F", la boîte de dialogue accompagnée de son script permettant le défilement du texte s'affiche. Lorsque le dialogue est fini, après la fermeture de la boîte, il est possible de recommencer l'opération. Pour le moment, les dialogues ne sont pas dynamiques car la boîte de dialogue du prêtre nous a, en premier lieu, permis l'apprentissage du fonctionnement de la boîte de dialogue.

Enfin, deux marchands ont été créés, l'alchimiste et l'armurier. Nous pouvons interagir avec eux en entrant dans leurs "Box Collider". À l'aide de Canvas et de scripts, ces shops deviennent fonctionnels.



FIGURE 2.47 – PNJ : Marchand



FIGURE 2.48 – PNJ



FIGURE 2.49 – PNJ : Prêtre

Dans la même idée que pour les ennemis, les recherches faites auparavant nous ont permis d'implémenter relativement aisément, de nombreux PNJs permettant d'animer comme bon nous semble notre village. Nous avons implémenté différents modèles de PNJs. Tout d'abord un premier statique qui n'interagit pas avec le joueur, il est juste muni d'un Animator pour donner vie au jeu. Un deuxième modèle, aussi statique, mais avec qui le joueur peut interagir et certains d'entre eux sont aussi capable de donner des objets au joueur. Le troisième modèle sera mobile et non interactif. Pour finir le dernier sera mobile et s'arrêtera si le joueur est assez proche pour parler avec s'il le souhaite.

Le premier PNJ avec lequel le joueur interagit est un peu particulier, car il accompagne le joueur pendant presque toute la durée du niveau. Il est donc muni d'un navmesh et suit le joueur

jusqu'à une certaine distance. Le premier niveau étant un niveau d'introduction, le joueur est obligé de faire certaines actions pour continuer à avancer, c'est sur cela que se base le prisonnier pour avancer dans son dialogue et mourir.

Enfin, dans le scénario l'ensemble des dialogues en lien avec l'histoire ou ajoutant des détails sur l'univers du jeu ont été écrits et seront retranscrits à l'aide d'une boîte de dialogue faite dynamiquement. En effet, les lignes de dialogue peuvent être mises dans le script lié au PNJ que l'on veut implémenter.

Les PNJs étaient censés être terminés pour la deuxième soutenance mais ils étaient plus à 90 % en vérité. Nous avons donc complété cela en ajoutant des choses non prévues d'origine pour permettre au joueur une meilleure expérience de jeu. Tout d'abord les PNJs qui donnent des objets au joueur ne les jettent plus au sol mais les donnent bien dans l'inventaire du joueur (sauf si celui-ci est plein). Près de cinquante PNJs ont été placés dans le village avec certains interactifs. Si nous faisons le compte il y a dans le village : 21 PNJs non interactifs et statiques, 10 PNJs non statiques et non interactifs, 5 PNJs non statiques et interactifs et 10 PNJs statiques et interactifs.

Les PNJs non statiques sont munis d'un script qui leur permet de se déplacer aléatoirement dans le village à des endroits prédéfinis. Quant à ceux qui sont statiques et non interactifs ils sont là pour donner plus de vie au village. Il y a donc 15 PNJs avec lequel le joueur peut interagir (en dehors des marchands). Ils sont là pour donner des indices et des informations. Nous avons également comme dit précédemment un PNJ donnant une quête mystère, ce qui permet au joueur de débloquent un nouveau sort. Si le joueur veut espérer résoudre les énigmes pour accéder aux différents niveaux, il est important qu'il parle aux PNJs afin de mieux comprendre l'histoire de Proxima. Ces échanges faciliteront son avancé et sa compréhension du monde.

Pour finir nous avons également mis à jour le personnage de notre niveau Tutoriel.

### 2.5.3 Boss (Elias et Godric)

Nous avons implémenté un premier boss nommé "Noopy", qui serait le boss d'introduction du jeu. Il s'agit d'un chevalier muni d'une masse, expliquant son paterne d'attaques lourdes et lentes. Cependant, seule l'attaque de base a été implémentée actuellement. Les Boss se trouveront dans une salle, qui, à l'entrée du joueur, affichera la vie et le nom du boss et fermera la salle derrière lui. Pendant le combat, une fois que le boss a atteint la moitié de sa vie, il rentre dans une seconde phase, ses dégâts et sa résistance sont alors doublés et il inflige un malus au joueur, ici une division par 2 de ses dégâts.

En ce qui concerne son IA, elle est similaire à celle des ennemis basiques, puisque le boss sera en "Idle" en attendant que le joueur entre dans la salle, et se mettra à pourchasser le joueur si celui-ci entre. Cependant, cette IA n'est pas la version finale. En effet, ce n'est pas celle indiquée dans le cahier des charges. Ainsi, plus tard, celui-ci aura 3 attaques en plus.



FIGURE 2.50 – Barre d'HP du premier Boss

Pour la deuxième soutenance nous avons amélioré notre ancien boss, avec un système de "phase", en effet, arrivé à la moitié de sa vie le boss enchante son épée avec un sort de feu, ses dégâts seront donc amplifiés. Si le joueur est trop loin du boss, celui-ci lance une boule de feu.

Lors de ce dernier temps, nous avons rattrapé notre retard sur ce domaine, en améliorant le premier boss et en ajoutant 4 nouveaux adversaires, plus durs les uns que les autres. Les boss utilisent en très grande majorité des effets spéciaux, tout comme les sorts, permettant de diversifier leurs attaques et les animations. De plus, tous les boss suivent la gestion de la distance et ajoutent de nouvelles attaques en fonction de la vie.

En ce qui concerne le premier boss Noopy, nous lui avons ajouté deux nouvelles attaques, la première, un coup de poing dragon, se déclenche quand le joueur est à mi-distance du boss. La deuxième est son attaque spéciale, en effet quand arrive à la fin de sa vie, il déclenche une vague de météorite jusqu'à ce qu'il meure.

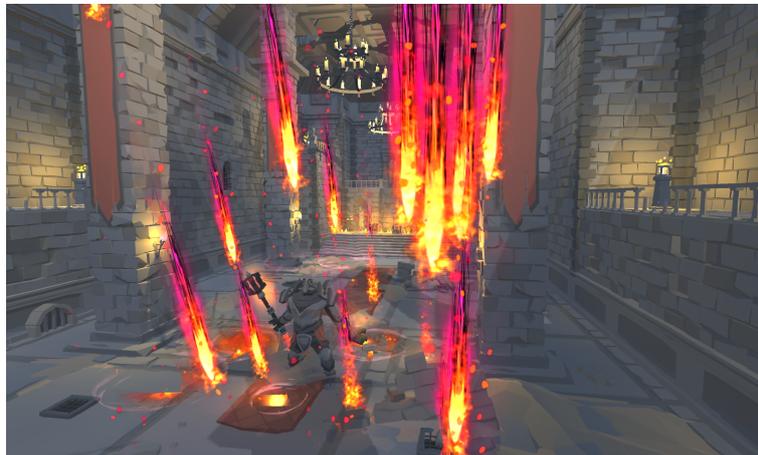


FIGURE 2.51 – Dernière attaque de Noopy : vague de météorite

Le premier boss est un fantôme, nommé Buster (oui ghost buster), qui dispose d'attaques basiques, mais qui lorsqu'il perd la moitié de sa vie, commence à embêter le joueur, en se rendant intouchable, en mettant un écran noir devant le jeu ou en volant petit à petit la vie du joueur. Son IA est différente de celle du premier boss, puisque celui-ci se déplacera à des endroits prédéfinis tout en attaquant le joueur.



FIGURE 2.52 – Boss Niveau 1 : Buster

Le second boss est un ancien PNJ, qui se retourne contre le joueur (plot twist!). Celui-ci dispose d'une IA similaire au fantôme, lui permettant de se déplacer d'un endroit à l'autre,

donnant l'impression de fuir le joueur. En effet, le prêtre ne lance que des attaques à distance et a littéralement pour but de vous faire quitter le jeu. De plus, lorsqu'il arrivera à la moitié de sa vie, il baissera de moitié toutes les statistiques du joueur (sauf le saut) et si vous arrivez à le blesser assez pour qu'il soit aux portes de la mort, il régénérera toute sa vie, vous forçant à le recombattre depuis le début.

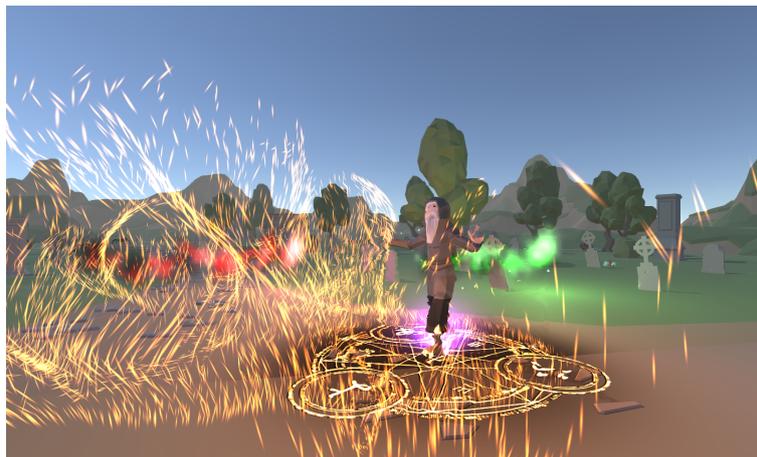


FIGURE 2.53 – Boss Niveau 1 : Prêtre

Le boss de la zone de la forêt s'appelle quant à lui Joe Froid. Celui-ci est immobile mais dispose d'attaque de zone permettant de ne pas en faire un désavantage. De plus, celui-ci aura une puissance d'attaque assez importante. L'attaque finale fera en sorte qu'il puisse tuer en un coup le joueur.

Le Chevalier Noir, aussi appelé Soron, est le boss final de notre jeu, situé dans une salle de boss que nous aborderons dans la partie graphique. Le boss est de loin l'IA la plus développer de notre jeu en ce qui concerne les ennemies, composé de 3 phases de combat, avec plus de 8 attaques différentes entre combats à distance et au corps-à-corps.



FIGURE 2.54 – Boss Niveau 3 : Chevalier Noir

## 2.6 Graphismes (Godric, Lucile et Marco)

Au tout départ, nous voulions apprendre à utiliser Blender pour faire les différentes maps et objets composant le jeu, mais nous nous sommes vite rendu compte que ce n'était pas si facile et que nous allions manquer de temps. Nous avons donc décidé d'acheter des packs d'assets pour nous aider avec la conception des différentes cartes. Nous avons ensuite rassemblé ces éléments pour pouvoir créer des cartes entières représentatives du monde médiéval-fantastique dans lequel se déroule le jeu. Pour ce qui est du graphisme des interfaces, nous nous sommes aidé d'images correspondants à nos besoins récupérés sur internet et avons utilisés le type TextMeshPro pour un rendu visuel plus satisfaisant. Vous avez pu voir les graphismes en dehors des cartes plus haut.

Notre jeu est donc composé de plusieurs zones, 3 niveaux et une zone annexe qui sert comme intermédiaire entre les différents niveaux et qui permet au joueur un peu de repos. Nous avons donc prévu de faire un premier niveau qui se situera dans des cachots et dans des grottes. Ce niveau n'est pas encore fini mais est en bonne voie. Concernant le niveau 2, nous avons décidé de le faire dans une forêt et pour le niveau 3, un château. Nous avons quelques morceaux de ces deux derniers mais nous ne les avons pas encore assemblés. Le niveau 1, qui se déroule dans un cimetière, n'est quant à lui pas du tout commencé. Pour finir, la zone annexe, qui est un village,

est totalement finie même si il faudrait rajouter quelques arbustes et des décors afin de le rendre plus vivant.

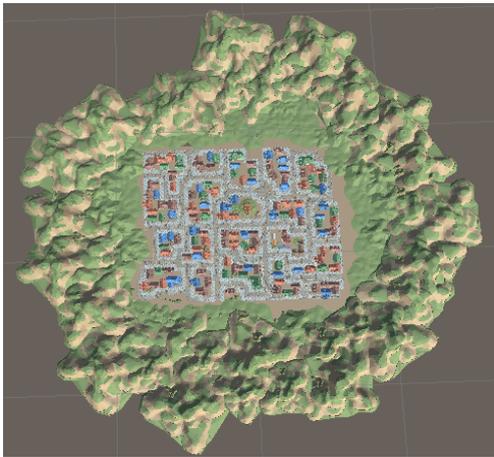


FIGURE 2.55 – Carte : Village



FIGURE 2.56 – Boss : Noopy

Concernant les graphismes généraux nous en avons globalement parlé dans leurs parties respectives. Nous avons changé l'écran de mort pour l'harmoniser avec le reste du jeu. Les tailles des polices ont aussi été réadaptées pour être en accord avec le reste. Globalement, nous avons harmonisé la quasi-totalité des graphismes pour avoir un produit visuellement cohérent et agréable.

Pour les cartes, nous avons bien avancé. Les premiers niveaux sont finis, il nous reste uniquement à rajouter des objets d'environnement tel que des arbustes ou autres pour rendre le paysage plus attrayant.

<b>Carte</b>	<b>Avancement</b>
Donjon (niveau 0)	100%
Village	95%
Cimetière (niveau 1)	90%
Caveau (niveau 1)	90%
Forêt (niveau 2)	70%
Château (niveau 3)	60%



FIGURE 2.57 – Donjon partie 1

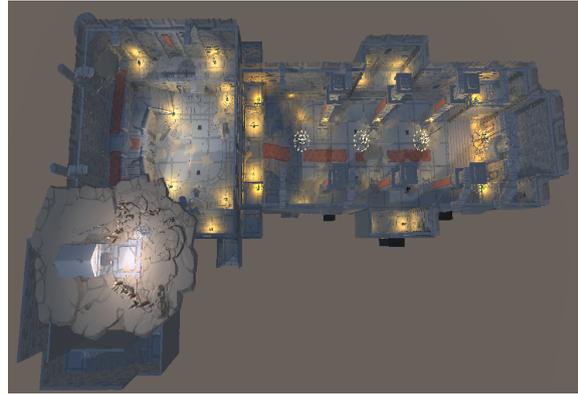


FIGURE 2.58 – Donjon partie 2

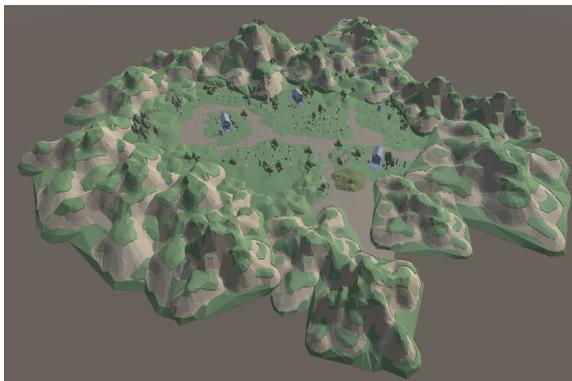


FIGURE 2.59 – Cimetière

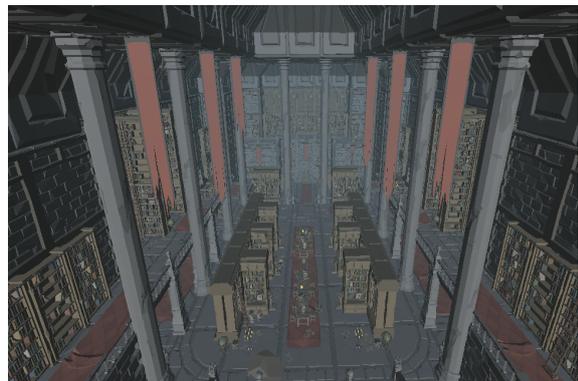


FIGURE 2.60 – Bibliothèque du caveau



FIGURE 2.61 – Niveau 3

Pour le dernier temps de travail nous avons peaufiné tous les graphismes du jeu pour arriver à un rendu cohérent. Le but étant de voir un bout du jeu et de reconnaître directement le jeu, nous voulions une identité visuelle propre à Proxima. En conséquences tous les menus, informations

et autres ont été mis à jour. Cela a déjà été dit précédemment mais nous avons totalement refait l'interface de l'inventaire et nous sommes assez satisfaits du résultat. Étant donné que celui-ci l'accompagne partout il était nécessaire qu'il soit agréable et pratique pour l'utilisateur. Le principal objectif était d'allier l'utile à l'agréable. Tous les menus ont aussi été revu pour être ajusté. Nous avons également pris en compte les remarques des soutenances précédentes et fais quelques changements d'ordre au niveau de l'affichage.

Pour finir, au niveau des cartes, elles sont entièrement terminées. Le niveau 3 ayant été revu car il ne collait pas au scénario. Au final nous arrivons à un total de 6 cartes. Tout d'abord le donjon dans lequel le joueur pourra suivre le tutoriel du jeu. Le village qui est la place principale ou il peut trouver des indices, informations sur le déroulement de l'histoire et se réapprovisionner. Viennent ensuite le cimetière et le caveau que le joueur pourra parcourir lors du niveau 1. Et la forêt ainsi que la salle du boss finale qui sont respectivement le niveau 2 et 3.



FIGURE 2.62 – Niveau 2



FIGURE 2.63 – Salle du Boss Final

## 2.7 Animations (Godric, Lucile et Marco)

En ce qui concerne les animations, nous avons utilisé le site Mixamo pour les générer, afin d’animer les joueurs, les PNJs et les ennemis. Bien que ces animations ne soient pas encore parfaitement fluides, elles permettent déjà au joueur de se mouvoir avec aisance dans l’environnement du jeu. Nous envisageons d’effectuer des ajustements supplémentaires en modifiant les animations déjà mises en place pour améliorer l’aspect visuel du jeu.

Les animations sont globalement finies pour les personnages, il manque quelques transitions pour le joueur, mais l’ensemble est bien présent. Pour ce faire, nous avons utilisé des modules d’animations avec des paramètres que nous activons ou augmentons selon leurs natures et nos besoins. Nous n’avons pas rencontré de problème majeur concernant l’animation des personnages.

Plusieurs modifications ont été effectuées sur les contrôleurs des PNJs et ennemis. Ils ont parfois été dédoublés pour faire de légères modifications selon ce dont on attendait des personnages. En effet, les scripts qui héritent d’un état de machine ont parfois besoin d’un parcours aléatoire des points définis pour les ennemis ou encore d’une interruption pour se mettre en position fixe et regarder le joueur lorsque celui-ci s’approche.

Pour le premier niveau, le joueur doit sortir d’un cachot en enfonçant une porte, il doit également pouvoir ouvrir et fermer une autre porte. Nous avons donc dû faire une animation

pour les différentes portes. Nous avons rencontré des problèmes pour bien placé la porte à la fin de sa position et rotation finale de l'animation. En effet, il arrivait parfois qu'après l'animation la porte revienne à son emplacement d'origine. Pour régler ce problème nous avons mis un événement avant la fin de l'animation qui désactive le module d'animation de la porte. Pour la porte qui s'ouvre et se ferme, il arrive parfois qu'elle n'arrive pas à sa destination, ceci est dû au personnage qui peut parfois bloquer la porte avec son corps. Nous avons donc rajouté un corps à la porte, elle n'est plus gênée par le joueur, mais lui par contre se fait éjecter par celle-ci s'il l'a gêne dans son ouverture ou sa fermeture.

Lors de la dernière phase de travail, nous avons résolu beaucoup de problèmes concernant les animations. Comme dit plus haut les animations de morts se déroulent maintenant bien au sol, là où elle devrait être. Au niveau des animations des objets, tel que les portes ou encore les coffres, nous avons fait une avancée majeure car les problèmes sont entièrement résolus. En effet il arrivait que la position finale dans l'animation d'un objet ne soit pas prise en compte et qu'il revienne à l'état d'origine. Nous avons maintenant trouvé la méthode pour résoudre ce problème, à chaque état d'animation nous mettons un autre état ne nécessitant aucune condition. Cet état est l'état final de l'objet ce qui permet si problème il y a sur la fin de l'animation précédente quant au placement final de l'objet, d'assurer qu'elle soit respecter. Dans un dernier temps nous avons aussi réglé certaines animations sur le mouvement du personnage, comme par exemple un PNJ qui se tranche le torse avec une épée.

Une partie des contrôleurs d'animation ont été dupliquées et modifiées pour le multijoueur car beaucoup dépendent des joueurs. Ainsi, quelques tests ont été ajoutés dans les scripts afin de dissocier le premier joueur et le second.

## **2.8 Musiques et sons (Elias)**

Pendant la deuxième phase de travail nous avons ajouté des musiques (une pour le village, une pour le donjon), qui pourront être baissées et augmentées dans le jeu. Nous avons aussi ajouté pour le moment un son pour les villageois, qui s'active quand on s'approche d'eux.

Cependant, ces sons venant des licences The Legend Of Zelda de Nintendo et Minecraft de Mojang, nous n'avons plus les droits de publication du jeu.

Une musique a aussi été mise lors de la cinématique d'introduction permettant de créer l'ambiance voulue, celle-ci étant présente dans la vidéo réalisée par montage, il fut aisé de gérer sa bande sonore.

Lors de cette dernière phase de travail nous avons ajouté quelques sons et musiques pour finaliser le tout.

---

## Bilan du jeu

Tâches	Prévu	Réalisé
Développement du joueur	100%	100%
Menus	100%	100%
Interfaces	100%	100%
Scénarios et Dialogues	100%	100%
Musiques et sons	100%	100%
Multijoueur	100%	70%
PNJs	100%	100%
Ennemis basiques	100%	100%
Boss	100%	100%
Graphismes et Animations	100%	100%
Site Web	100%	100%

Nous n'avons pas réussi à accomplir toutes les tâches, c'est surtout le multijoueur qui nous a retardé car nous avons rencontré beaucoup de problèmes.

Pour le développement du joueur, toutes les fonctionnalités prévues ont été implémentées. Que cela soit les quêtes, les objets ou encore le déplacement du joueur, elles sont toutes maintenant fonctionnelles.

Pour ce qui est du scénario et du dialogue, il est totalement fini et nous avons bien une fin à l'histoire. Le joueur doit donc combattre 5 différents boss pour réussir à finir le jeu. L'histoire n'est pas totalement dévoilée pour laisser le joueur une interprétation personnelle.

En ce qui concerne les musiques et le son, nous avons ajouté quelques sons et musiques, les principales actions font du sons. Comme c'était initialement prévu nous n'avons pas non plus trop appuyé sur cet aspect car ce n'était pas notre volonté.

Pour le multijoueur, beaucoup de choses ont été faites mais malheureusement ce n'est pas suffisant par rapport à ce qui était prévu. Les joueurs peuvent maintenant jeter des objets, équipés des armes, des armures, utiliser des consommables. Les vies, expériences et argent

ont synchronisé pour les deux joueurs. Les magasins et les dialogues sont aussi fonctionnels. Nous avons réussi à implémenter un ennemi et un boss. Il nous reste donc l'implémentation de tous les niveaux et certains ajustements sur les choses déjà faites.

Nous avons donc réussi à implémenter tous les comportements prévus pour les PNJs ainsi que tous ceux prévue dans le village.

Pour les ennemis basiques, plusieurs ennemis ont été ajoutés.

En ce qui concerne les boss, nous avons ajouté 4 nouveaux boss.

Les graphismes ont harmonisé mais globalement nous n'avons pas fait de changements très marquants.

Au final nous avons fini le solo mais le multijoueur n'est donc pas au niveau auquel il devrait être.

### **3.1 Les différents niveaux**

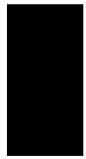
Notre jeu est composé de quatre niveaux si nous comptons le tutoriel.

Le niveau 0 a été le plus compliqué à mettre en place étant donné qu'il fallait implémenter un PNJ qui accompagnait le joueur et devait mourir afin de le laisser continuer seul son aventure. Lors de se niveau le joueur va pouvoir affronter ses premiers ennemis, récupérer ses premiers objets et c'est le prisonnier qui sera chargé de lui donner ses deux premières quêtes.

Le village n'est pas considéré comme un niveau mais n'a pas été facile à implémentée étant donné le nombre important de PNJs à placer. Les spécifications concernant les PNJs ont été précisés plus haut. C'est par cette zone que le joueur peut accéder aux prochains et précédents niveaux.

Concernant le premier niveau c'est un cimetière et un caveau. Le joueur va devoir résoudre une énigme pour pouvoir accéder au caveau et résoudre la deuxième quête que lui a donné le prisonnier. Pendant ce niveau le joueur va pouvoir affronter deux boss.

Le niveau 2 est une forêt dans laquelle le joueur devra trouver certaines choses pour pouvoir faire apparaître le boss. Pour finir le niveau 3 est tout simplement le boss final du jeu.



---

# Conclusion

Pour conclure ce rapport de projet, nous avons réussi à faire un jeu solo de qualité même si nous aimerions encore plus l'améliorer. Le multijoueur n'est quant à lui pas dans nos objectifs, nous nous sommes rendu compte que nous avons eu trop d'ambitions et qu'il nous était difficile de le terminer avec le temps imparti. Malgré tout nous sommes contents du résultat et ce projet nous a permis d'approfondir nos connaissances.