SATIS-KINGDOM

404 Social Life Not Found

Janvier 2025

Sommaire

Introduction

Dans le cadre de notre projet informatique du second semestre de notre année, notre équipe 404 Social Life Not found réalisons un jeu vidéo nommé Satis-Kingdom.

Satis-Kingdom est un jeu de type 2d RPG, s'inscrivant dans un style pixel art et avec une vue du dessus que qualifie de "Top-Down". Le ou les Joueurs en fonction du mode de jeu solo ou multi-joueur incarne un personnage qui évolue dans un univers médiéval-fantastique. les joueurs seront amenés à suivre different quête qui les mèneront à explorer les different aspect du monde que nous avons creer.
En plus de l'aspect d'aventure nous avons pensé un système de combat avec différents ennemis. Et enfin nous proposerons aux joueurs des mécanismes de récolte de ressources qui permettront de faire évoluer à la fois le personnage et le monde qui l'entour.

Récapitulatif des tâches

Tâches	Tristan	Quentin R	Quentin S	Raphaël	Alexandre
Game Design					
Création des assets				R	S
Musiques / sons			S	S	R
Rédaction de l'histoire	R		S		
Map Design	R			S	
Programmation					
Système d'inventaire	S	R		S	
Système de sauvegarde		R		S	S
Déplacement / action Personnage		R		S	
Gestion des ennemis		S	R	S	
Système de dialogue		R		S	S
Tutoriel	S		S	R	
Multijoueur	R	S			
IA			S		R
Site Web					
Page d'accueil	S	S	R		
Téléchargement			R		S
Manuel Utilisateur			R	S	
Autres					
Mise en page Cdc		R	S		

Table 1: Répartition des tâches (**R** : Responsable, **S** : Suppléant)

Tâches	Soutenance 1	Soutenance 2	Soutenance 3
Game Design			
Création des assets	80 %	100 %	100 %
Musiques / sons	50 %	100 %	100 %
Rédaction de l'histoire	100 %	100 %	100 %
Map Design	20 %	100 %	100 %
Programmation			
Système d'inventaire	0 %	100 %	100 %
Système de sauvegarde	0 %	10 %	100 %
Déplacement / action Personnage	0 %	100 %	100 %
Gestion des ennemis	0 %	40 %	100 %
Système de dialogue	0 %	100 %	100 %
Tutoriel	0 %	20 %	100 %
Multijoueur	0 %	30 %	100 %
IA	0 %	40 %	100 %
Site Web			
Page d'accueil	0 %	100 %	100 %
Téléchargement	0 %	30 %	100 %
Manuel Utilisateur	0 %	30 %	100 %
Autres			
Mise en page Cdc	100 %	100 %	100 %

Table 2: Avancement et planification

Comme nous l'avons planifié durant cette première période de travail, nous nous sommes concentrés sur la mise en plus du projet. Tout d'abord avec la création des assets et leur configuration dans unity. Ensuite nous avons commencé à réaliser les premier décors dans lequel le personnage va évoluer, ainsi que l'implémentation de quelques mécaniques fondamentales pour avoir une vue d'ensemble de ce que va être notre jeu. En parallèle de cela nous avons travaillé sur la musique et la rédaction de l'histoire.

Organisation

Pour assurer une bonne collaboration lors de cette première période de travail, nous avons mis en place et utilisé plusieurs outils. Pour avoir une vue d'ensemble de notre projet et regrouper les différents points du développement nous avons un Trello ou nous pouvons chacun avoir accès à l'ensemble des information nécessaire et suivre l'avancement du projet. Pour assurer une bonne communication nous utilisons tous Discord, nous avons un groupe ainsi qu'un serveur dédié à notre équipe qui nous permet d'avoir des réponses rapide à nos questions ou autres informations.

Pour la rédaction nous utilisons Google docs qui nous permet de collaborer sur les différentes tâches que nous devons rédiger comme l'histoire de notre jeux avec les dialogues ou encore les documents pour préparer nos soutenance.

Enfin, un des outils les plus importants que nous avons mis en plus est notre Dépôts Git auto-hébergé. Pour ce projet nous avons décidé de se passer des grandes plateformes traditionnelles comme Github, à la place de cela nous avons mis en place un serveur Git auto-hébergé par nos soins. Nous utilisons le logiciel Gitea qui nous permet grâce à Docker d'avoir très facilement un serveur git avec une belle interface et plusieurs outils pour gérer tous nos Dépôts Git. Nous avons chacun un compte utilisateur qui fait partie d'une organisation et l'ensemble des membres de l'équipe a un accès aux différents dépôts, pour le moment nous avons un dépôt pour le projet Unity et un autre pour le site Web. Cette installation nous permet de travailler en même temps sur le projet grâce aux différentes branches que nous avons créées.

Les Graphismes / Assets

1.1 Les personnages

Notre responsable des graphismes Raphaël s'est occupé depuis le début de réaliser les différents personnages de notre jeu. Nous avons opté pour un style pixel art, ce qui nous permet de donner un style rétro à notre jeu, de plus cela simplifie la réalisation des graphismes. Pour la création des assets ... (a completer par raphael)

Figure 1.1: Assets des personnages

1.2 Graphismes Générales

Un autre point important de l'aspect visuelle de notre sont tous les graphismes qui nous permettent de façonner l'univers dans lequel le personnage va évoluer. Pour créer les "Maps" nous utilisons des graphismes qui ont déjà été réalisés, en effet réaliser les graphismes nous même pour créer de tels décors riches et variés nous aurait pris un temps bien trop important pour les délais du développement de notre jeux. Nous avons pris un pack de graphisme sur Itch.io qui est dans le même style que les personnages que nous avons réalisé.

1.2.1 Les Tiles

Pour la réalisons des “Maps” nous avons opté pour un système de TileMap ou le principe est de placer des Tiles sur une grille. Les Tiles sont des images carrées dans notre cas de 16 pixels par 16 pixels qui peuvent être mises bout à bout pour réaliser des décors.

Voici un exemples des Tiles que nous utilisons pour créer nos “Maps” :

Figure 1.2: Tiles

1.2.2 Tile Map

Dans Unity nous avons donc une grille avec plusieurs TileMap chacune ayant un rôle. En effet nous voulons par moment pouvoir superposer différents Tiles ou encore poser des éléments de décorations. C’est pour cela que nous utilisons différents “Layers” Cela nous permet de choisir quels éléments sont au premier plan et lesquels ne le sont pas.

Figure 1.3: Tiles Layers

1.2.3 Tile Pallette

Le prochain outils sont les Tile Palettes elle nous permet d'importer nos différents graphismes de Tile et fournit un ensemble d'outils de pinceaux qui ne permette de "dessiner" sur les TileMap et ainsi de créer des univers de manière efficace et pratique. Nous avons 4 Tile Palette pour le moment que vous pouvez voir ci-dessous, la première pour tout ce qui va être au sol comme la terre, l'herbe, l'eau ou encore pour faire le relief. La seconde pour les éléments de décorations qui se pose par dessous les Tiles deja existant, on y retrouve des arbres, des fleurs, des panneaux, etc... La troisième pour les différents bâtiments. Et enfin la dernière pour l'intérieur des bâtiments.

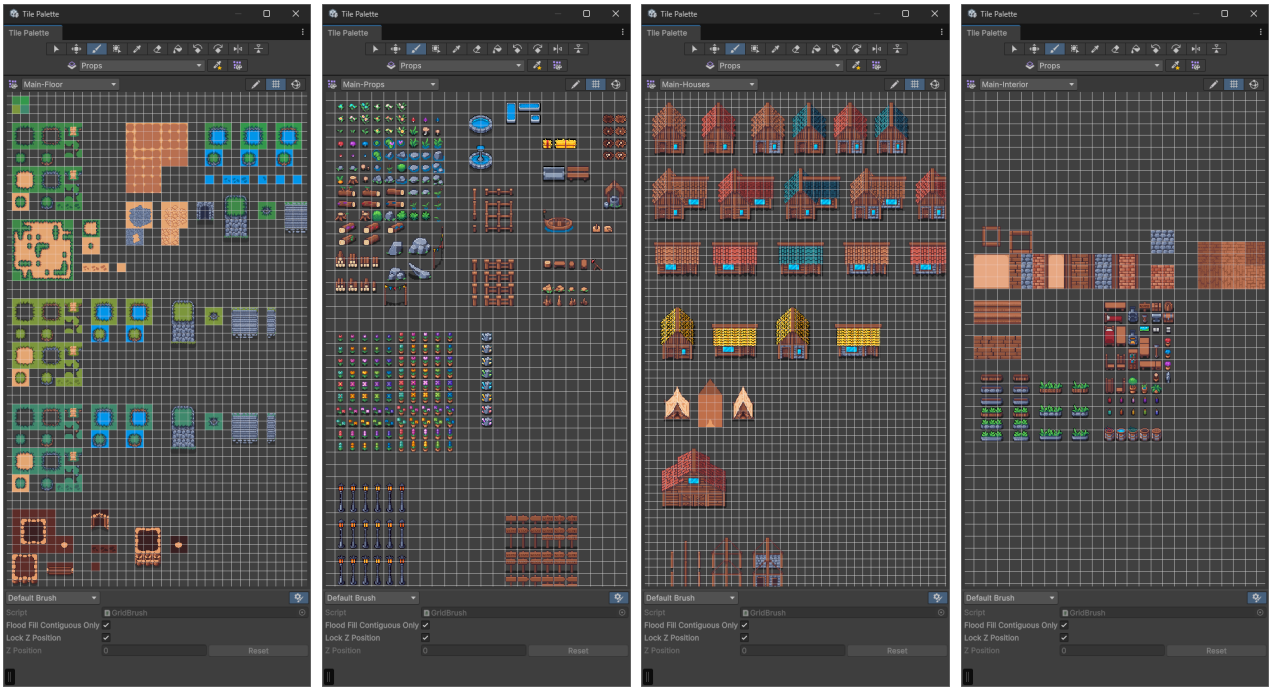

Figure 1.4: Tiles Palettes

Le plus important lors de l'importation des graphismes et la création des Tiles Palette a été de désactiver la compression et échantillonnage sur nos sprites. En effet par défaut unity compresse les ressources importées et utilise une technique d'interpolation linéaire pour adoucir les images. Or dans notre cas les image sont très peu lourde et on une très basse résolution car ce sont des pixels art, il est très important de faire attention à cela pour ne pas perdre la qualité des assets et l'effet pixel art.

De plus nous avons rencontré un léger problème lors de la création de notre monde après avoir placer les tiles sur les TileMap lors de l'exécution du jeu on pouvait remarquer un léger décalage entre les Tile. Pour résoudre ce problème nous avons créé un Sprite Atlas qui permet de regrouper l'ensemble de toutes les images utilisées en une seule et unique image cela améliore aussi la performance en limitant les appels GPU.
Voici un exemple ci-dessous du résultat obtenue avec un sprite atlas :

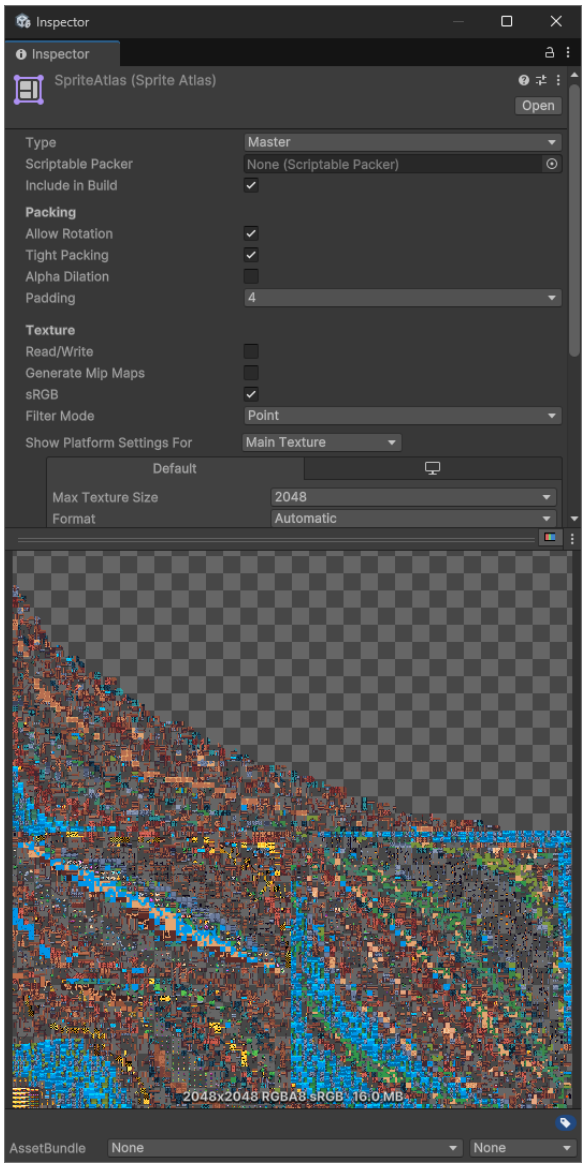

Figure 1.5: Sprites Atlas

1.2.4 Level Design

C'est avec l'ensemble des outils vous précédemment que nous avons commencé à créer notre scène principale. Ceci est une première version qui sera amenée à évoluer et qui n'est pas encore complète. En revanche cela nous a permis de mieux nous imaginer l'ambiance global du jeu.

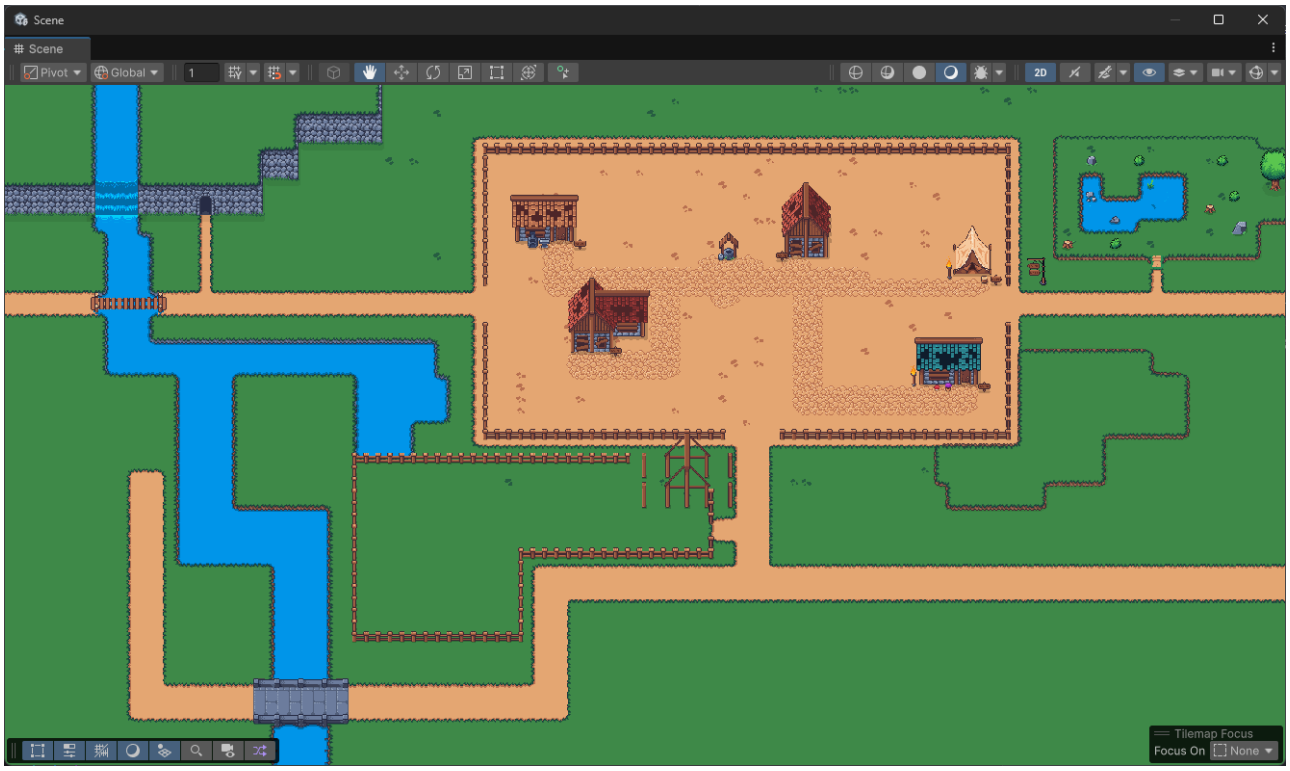

Figure 1.6: Level Design

Comme on peut le voir dans les images qui suivent, la carte principale se décompose en 4 parties majeures.

Nous avons le centre qui est le plus important, c'est le village abandonné que le joueur devra reconstruire et faire évoluer au fil de la partie. C'est ici qu'aura lui la majorité des interactions avec les personnages qui viendront au fur et à mesure s'installer dans le village.

La deuxième zone à gauche va être une vaste forêt, avec beaucoup de ressources ainsi que des falaise ou se cache des grottes. C'est dans cette zone que le joueur sera amené à récolter la majorité de ses ressources et explorer de nouvelles zones pour faire de nouvelles découvertes. Pour le moment nous n'avons pas encore réalisé tout le décors on peut juste y voir une première grotte.

La troisième zone à droite qui va être une vaste plaine plutôt hostile ou ici le joueur rencontre les différent ennemis et sera amené à réaliser plusieurs quête. Pour le moment nous avons seulement un petit exemple pour montrer à quoi vont ressembler les décors de cette zone.

Enfin la dernière zone sera pour quand le joueur aura débloquer les compétences pour cultiver des ressources ou encore des élevages d'animaux

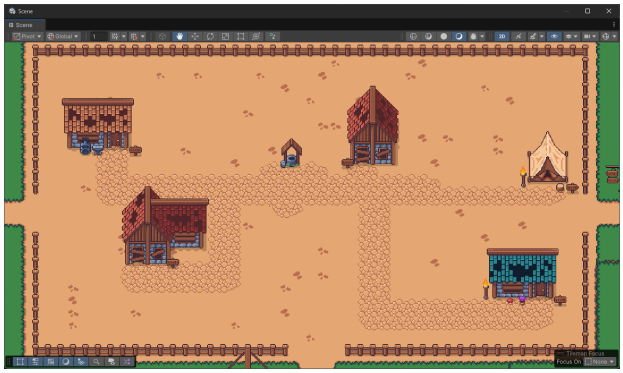

Figure 1.7: Vilage principal

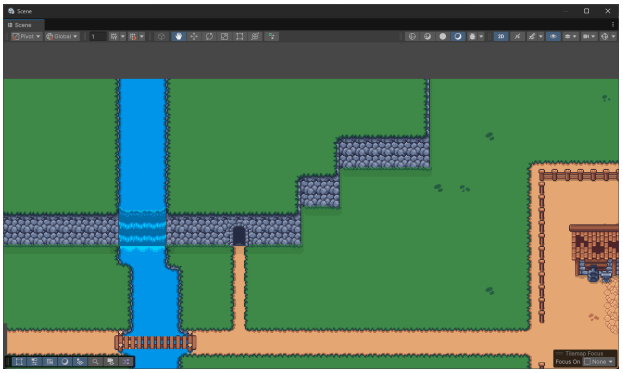

Figure 1.8: Forêt / Grottes

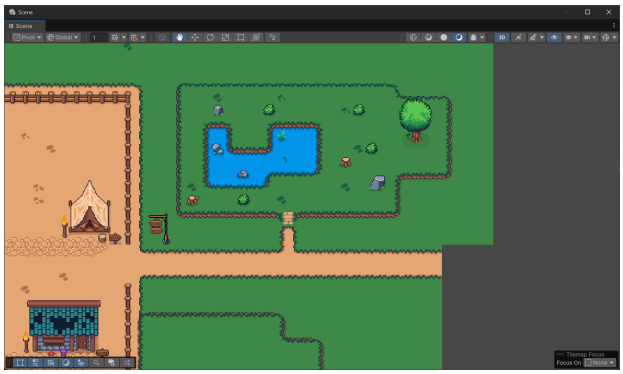

Figure 1.9: Vaste pleine

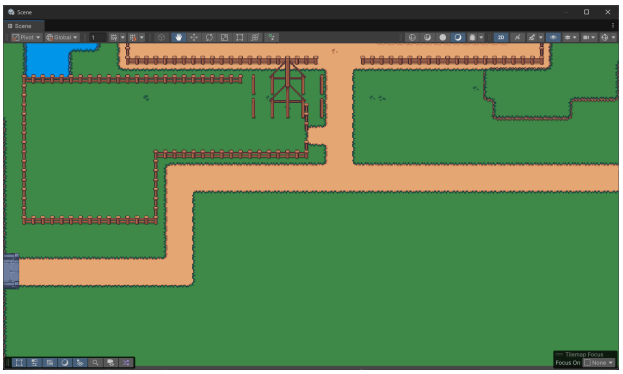

Figure 1.10: Elevage / Culture

Gameplay

2.1 Déplacement

Pour le prototypage de notre jeu notre personnage est une simple capsule cela nous permet de nous concentrer sur la partie technique des différent mécanique et nous implémenterons la partie visuel par la suite.

Notre première mécanique à implémenter a été logiquement le déplacement du personnage. Pour cela il faut d'abord récupérer les différentes entrées du joueur, nous utilisons le nouveau système d'Unity "Input System" cela nous permet de facilement gérer tous les entre que nous aurons besoin dans le futur. On peut facilement créer des action ainsi que d'y lié des touches et on peut très facilement avoir accès au etat de ses actions dans nos script C#.

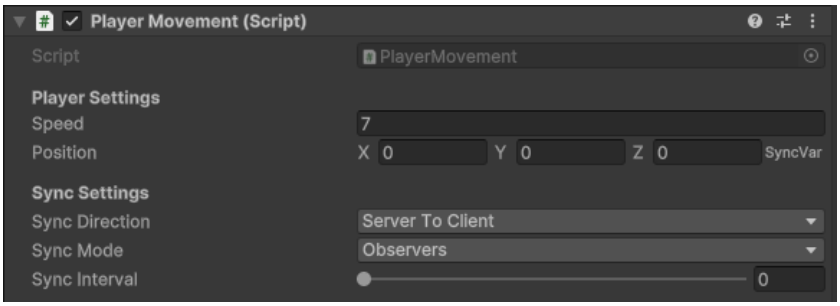

Figure 2.1: Player Movement Script

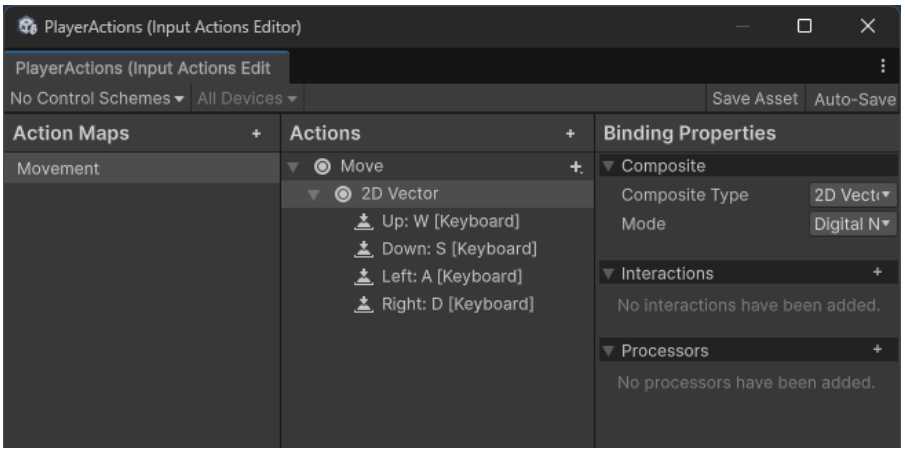

Figure 2.2: Player Input System

C’est ainsi qu’à chaque frame du jeu nous récupérons un Vecteur 2D normalisée de l’entrée du joueur avec les touches WSAD. Par exemple, si seulement la touche W est appuyé, notre vecteur vaut (1,0), pour S (-1,0) et si deux touches sont enfoncés en même temps comme W et A alors notre vecteur vaut (1, -1).

Il nous suffit par la suite d’utiliser la méthode MovePosition du rigidbody2D qui est directement mis à disposition par Unity on multiplie juste la valeur par une variable speed configurable pour régler la vitesse de déplacement et sans oublier d’aussi multiplier par Time.fixedDeltaTime pour ne pas dépendre du nombre de FPS, en effet on ne veut que notre vitesse depende de la puissance de l’ordinateur. Le rigidbody2D est le moyen mis à disposition par Unity pour simuler la physique.

2.2 Stats du personnage

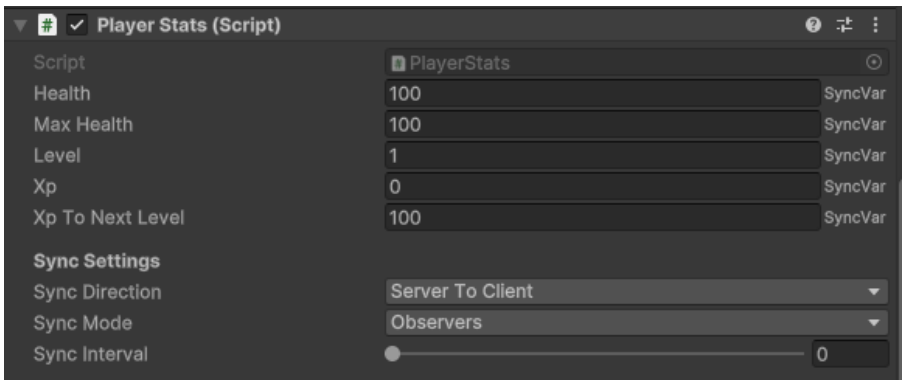

Figure 2.3: Player Stats Script

En plus de cela nous avons créé un moyen de donner des attributs à notre personnage comme la santé ou encore son niveau. Pour rendre cela utilisable nous avons différentes méthodes liées à ce script qui nous seront utiles par la suite.

Nous avons une fonction TakeDamage qui sera utile quand le personnage sera en combat, cette fonction enlève simplement une quantité de vie donnée en paramètre et si la vie du personnage descend a 0 on appelle on fonction Die qui existe mais n’est pas encore implémenté.

De plus, on dispose d’une fonction GainExp qui ajoute de l’expérience au joueur cette fonction gère le système de niveau, si l’expérience dépasse le prochaine pallier alors on augmente de niveau et le conteur est remis à 0, de plus le nombre d’expérience requis pour le prochain niveau augmentera en fonction d’un coefficient que nous pouvons contrôler.

MultiJoueur

Pour le multijoueur de notre jeux nous avons décidé d'utiliser le librairie Mirror Networking qui est open source est simple d'utilisation. Le multijoueur est sûrement le point le plus important de notre projet car tout va dépendre du multijoueur, même le script de déplacement du personnage. C'est pourquoi dès le départ nous avons donné de l'importance à cet aspect.

La mise en place de Mirror a été assez facile, il suffit de placer une GameObject "Network Manager" qui va s'occuper de gérer tout ce qui dépend du réseau. C'est sur cet objet qu'on définit calque paramétrage comme la position d'apparition du personnage ou le protocole qu'on veut utiliser, pour le protocole nous avons choisis Telepathy qui est très léger et efficace.

Chaque Objet qui dépend du réseau doit avoir un composant NetworkIdentity, et chaque script dependent doit hériter de NetworkBehaviour au lieu de MonoBehaviour habituellement.

Pour les différents scripts comme le déplacement un attribut très utile est "isLocalPlayer" cela permet de vérifier si le script doit agir comme si c'était le personnage du client actuel ou s' il doit ignorer le script car le personnage représente alors un hôte distant.

De plus, pour synchroniser la position des 2 joueurs, il faut utiliser un NetworkTransform qui va permettre de mettre à jour la position sur les différentes instances.

Pour le moment nous utilisons l'UI mis à disposition de base par Mirror pour se connecter ou héberger le serveur comme on peut le voir sur l'image ci-dessous. Nous avons le choix entre héberger un serveur et d'être client en même temps ce qui va permettre au joueur de jouer sur son monde et d'être rejoint par un autre joueur ou rejoindre un joueur.

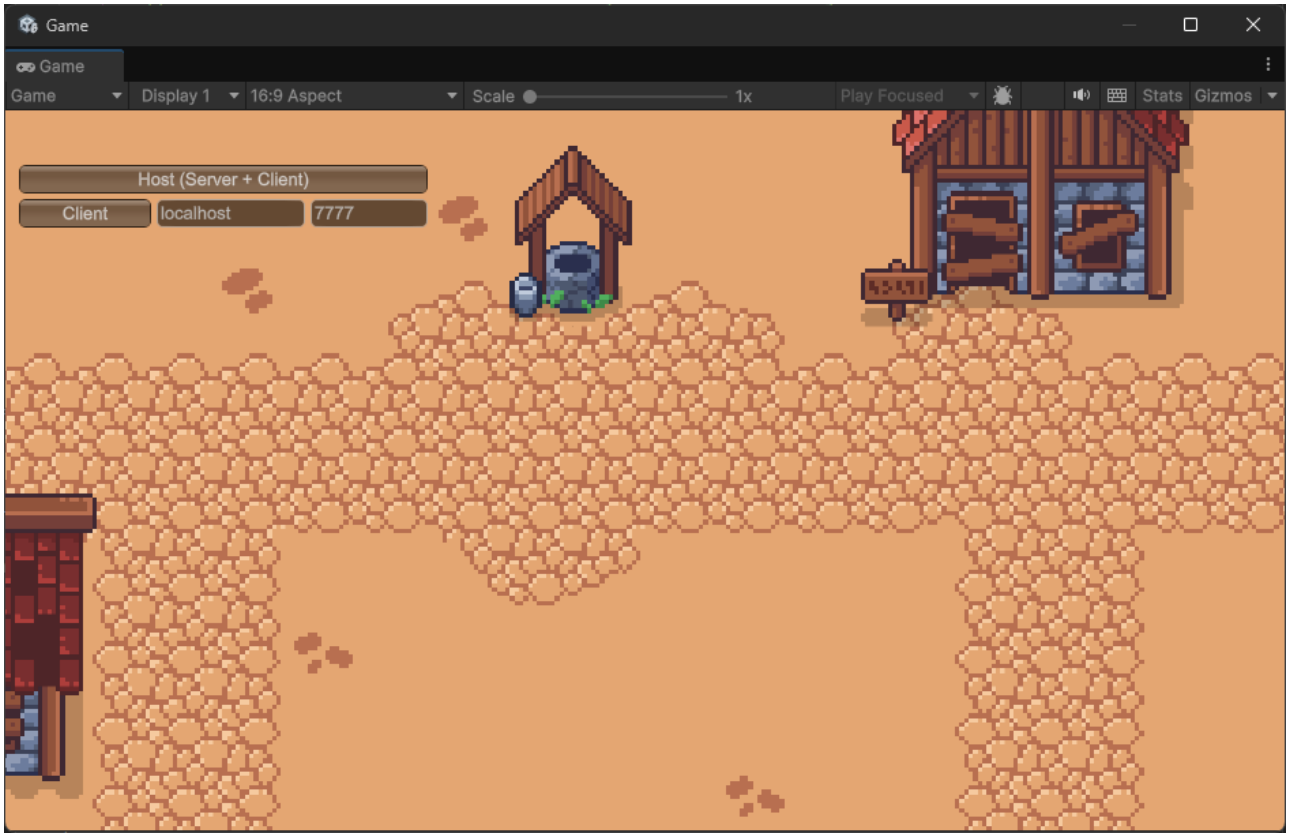

Figure 3.1: Multijoueur UI

HUD / Menus

4.1 HUD

Nous avons créé un HUD assez simple pour le moment permettant d'afficher au joueur les informations utiles comme la santé, le niveau actuel et la progression de l'expérience. Nous avons un Script qui s'occupe de mettre à jour l'interface en temps réel avec les informations du Script PlayerStats que nous avons vu dans la partie précédente sur le gameplay.

De plus ce menu s'adapte au mode multijoueur ou la bar de santé est bien celle du joueur local. Quant au bar d'expérience et le niveau actuel nous avons décidé que ces statistiques allaient être la même pour tous les joueurs et dépend donc de l'avancement de la partie de celui qui héberge le serveur. Ainsi la barre d'expérience est synchroniser entre les 2 joueurs, de plus les 2 joueur peuvent gagner de l'expérience et ainsi augmenter la bar commune.

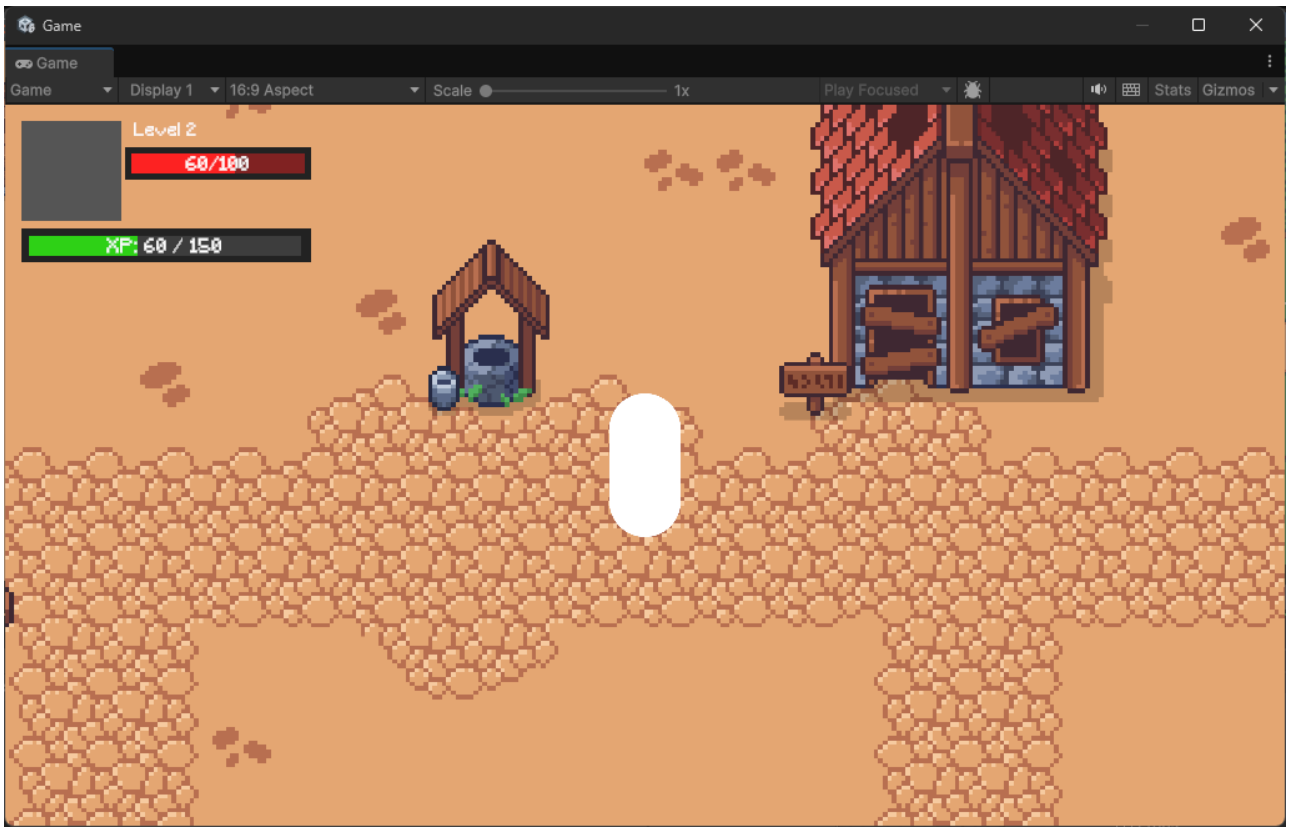

Figure 4.1: Player HUD

4.2 Menu

Nous avons créé un premier prototype de menu principale qui permet d'accéder à différents modes de jeu, au paramètre et de quitter le jeu.
Depuis les paramètres nous pouvons modifier la qualité graphique, la résolution et le plein écran. Dans le futur nous implémenterons d'autres possibilités pour le son, la gestion des sauvegarde ou peut être la configuration de touche.
Pour le moment ce menu ne fonctionne pas encore avec le mode multijoueur, mais il le sera prochainement.

Figure 4.2: Menu Principal

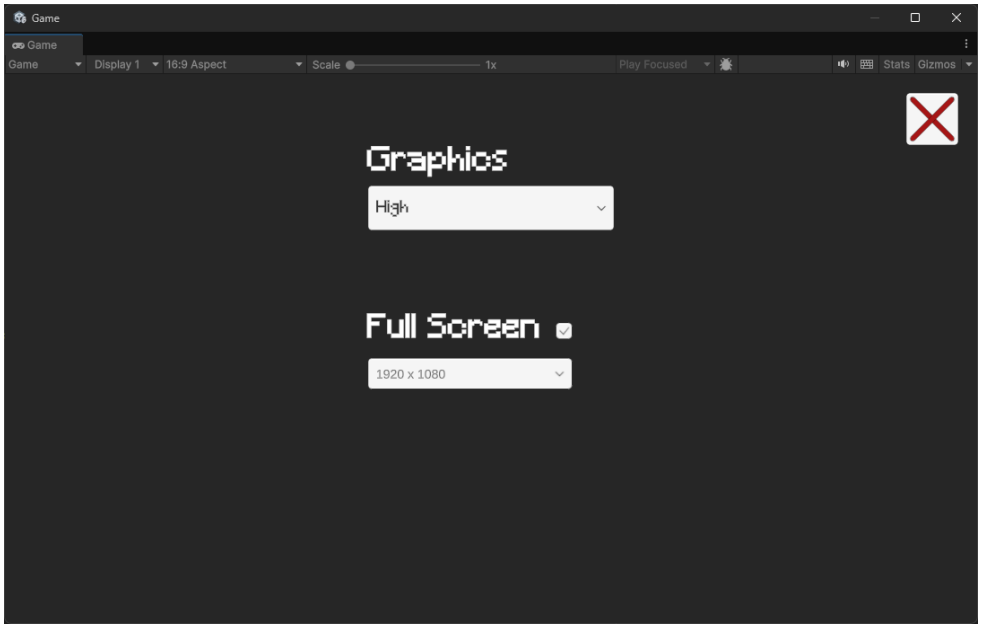

Figure 4.3: Menu Paramètre

Musique / sons

Histoire

La rédaction de l'histoire a bien progressé. Nous avons réussi à mettre des mots sur les idées que nous avions pour le jeu. L'objectif n'est pas de créer quelque chose d'extrêmement innovant, mais de concevoir un jeu avec une histoire à la fois immersive et complète.

Lors de la rédaction, nous avons décidé de structurer l'histoire non pas sous forme de paragraphe classique, mais plutôt comme un manuel détaillant les étapes concrètes de sa mise en œuvre dans le jeu. Plus concrètement, nous avons séparé les actions, les quêtes et les dialogues afin de faciliter son intégration lors de la phase de développement.

L'histoire est donc séparée en 3 parties distinctes :

1. le tutoriel qui permettra au joueur de se familiariser avec son environnement ainsi que les mécaniques de jeu.
2. La phase d'aventure ou le joueur suivra des quêtes qui lui permettront de débloquer de nouvelle fonctionnalité ou d'améliorer ses statistiques.
3. Le dénouement marquera la fin de l'histoire avec la victoire du joueur contre l'antagoniste.

Bien que l'histoire soit centrée sur l'accomplissement de cet objectif principal, elle inclura également des quêtes secondaires, offrant au joueur l'opportunité d'explorer davantage notre univers et d'enrichir son expérience.

Site Web

Bien que lors de cette première soutenance, le site web n'était pas notre priorité, dû aux dernières instructions que nous avons eues, notre équipe a malgré tout réussi à fournir un premier prototype de page d'accueil du site internet représentant notre jeu.

Figure 7.1: Site internet

(Les images correspondant au trailer et aux slides sont à titre indicatif et subiront des modifications lorsque nous aurons des images du jeu satisfaisante pour pouvoir apparaître sur notre site web.)

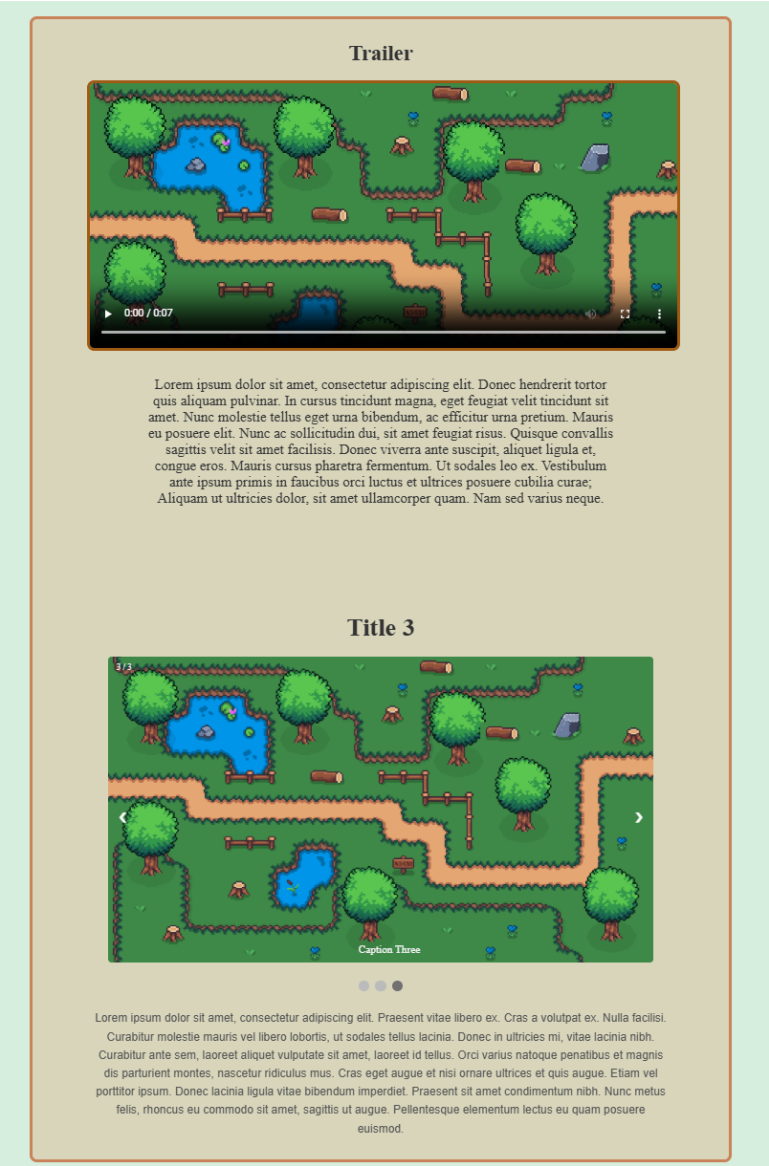

Figure 7.2: Site internet

Le site web restera dans les couleurs de notre jeu avec des couleurs rappelant la nature, là où le joueur passera la majorité de son temps. Un menu en haut de la page permettra à l'utilisateur d'en apprendre plus sur le jeu, mais aussi sur l'équipe, tout en lui laissant la possibilité de le télécharger. La vidéo en arrière-plan du logo du jeu Satis-Kingdom présente le style graphique que le joueur rencontrera en installant le jeu, suivi des trois mots résumant au mieux Satis-Kingdom : Récolter, construire et combattre ! Si cette première partie de la page d'accueil a plus au client potentiel, la suite de cette page se compose d'un trailer présentant une courte vidéo sur le jeu avec une phase de gameplay en complément des slides présentés au bas de la page d'accueil.

Etat d'avancement

Tâches	Prévisions	Avancements
Game Design		
Création des assets	80 %	80 %
Musiques / sons	50 %	40 %
Rédaction de l'histoire	100 %	90 %
Map Design	20 %	20 %
Programmation		
Système d'inventaire	0 %	0 %
Système de sauvegarde	0 %	0 %
Déplacement / action Personnage	0 %	20 %
Gestion des ennemis	0 %	0 %
Système de dialogue	0 %	0 %
Tutoriel	0 %	0 %
Multijoueur	0 %	15 %
IA	0 %	0 %
Site Web		
Page d'accueil	0 %	15 %
Téléchargement	0 %	0 %
Manuel Utilisateur	0 %	0 %
Autres		
Mise en page Cdc	100 %	100 %

Table 7.1: Etat d'avancement

Dans l'ensemble jusqu'à présent nous avons réussi à suivre notre planning, nous avons pris légèrement de l'avance sur la partie technique et le site web. Pour la rédaction de l'histoire et la musique ils nous manque seulement quelques petits détails à finaliser qui n'ont pas d'importance majeure, pour la rédaction ils nous reste a finir la structure des quêtes secondaires par exemple.

Conclusion

En conclusion, nous sommes globalement satisfaits de l'avancement de Satis-Kingdom. Durant cette première phase de travail, nous avons posé les bases nécessaires à la construction d'un jeu vidéo RPG riche et bien structuré. Cela inclut la mise en place d'outils de gestion efficaces comme notre serveur Git auto-hébergé, l'utilisation de Trello pour le suivi des tâches, et un serveur Discord dédié à la communication au sein de l'équipe.

Sur le plan technique, nous avons réalisé les premières mécaniques fondamentales, comme les déplacements, les statistiques du personnage, et une version simple du multijoueur grâce à la bibliothèque Mirror Networking. Par ailleurs, nous avons avancé sur les graphismes, la musique, et la rédaction de l'histoire cela permet de donner une identité claire à notre univers médiéval-fantastique.

Cette phase a également permis de valider notre organisation et notre méthode de travail en équipe. Les fondations posées durant cette période vont nous permettre de continuer à développer davantage la partie technique. En effet pour la seconde phase nous prévoyons de mettre en place une version solide de la plupart des mécanique de jeu, en plus de cela nous avons pour objectif de finir l'ensemble du Game Design.

Satis-Kingdom est encore au début de son aventure, mais grâce au travail collectif et à notre passion, nous sommes déterminés à faire évoluer ce projet vers une version complète et dont nous sommes fiers.