



Rapport de Projet

The coffee Tales

Tao Brouta Basset C2
Timéo Absalon-Lhéritier C2
Rayan Nouir Mahjoub C2
Luca Macheda C2
Noé Duny-Trublin C2

Mai 2025

Nova Studio

1 L'avancement global

1.1 Les Graphismes

Notre équipe avait initialement établi une direction artistique axée sur l'accessibilité universelle et nos cibles étaient de tout âge. Cette vision s'appuyait sur une philosophie de design inclusif, visant à créer une expérience visuelle intergénérationnel

Le choix d'un style graphique enfantin et accessible s'inscrivait dans une démarche réfléchie. Nous avions étudié les tendances du marché et analysé les retours utilisateurs de jeux similaires pour comprendre que l'accessibilité visuelle constituait un facteur important dans l'adoption d'un titre par un public diversifié. Cette approche nous permettait de toucher aussi bien les enfants découvrant leurs premiers jeux que les adultes cherchant une expérience de détente après une journée de travail.

La palette de couleurs vives et attrayantes avait été soigneusement sélectionnée après de nombreux tests utilisateurs et études psychologiques sur l'impact des couleurs dans les interfaces de jeu. Nous avions notamment privilégié des teintes primaires et secondaires saturées, capables de stimuler l'attention sans provoquer de fatigue oculaire. Chaque couleur avait été choisie pour son impact émotionnel : le bleu pour inspirer la confiance et la sérénité, le rouge pour signaler le danger tout en maintenant l'excitation, le vert pour évoquer la nature et la progression positive, et le jaune pour accentuer les éléments de récompense et de joie.



L'esthétique du jeu puisait son inspiration dans l'univers des productions d'animation contemporaines, particulièrement les œuvres de studios reconnus comme Pixar ou DreamWorks. Cette référence nous permettait de bénéficier d'un langage visuel déjà familier au public, facilitant ainsi l'immersion immédiate. Les formes géométriques simples que nous avions adoptées suivaient les principes du design épuré, où chaque élément visuel servait un propos fonctionnel tout en conservant un aspect esthétiquement plaisant.

Les textures légères constituaient un pilier technique et artistique de notre approche. Contrairement aux jeux AAA qui multiplient les couches de détails, notre stratégie consistait à créer un impact visuel maximal avec un minimum de ressources graphiques. Cette philosophie du "moins c'est plus" nous permettait non seulement d'optimiser les perfor-



mances sur diverses configurations matérielles, mais également de maintenir une clarté visuelle essentielle au gameplay.

L'architecture des environnements avait été conçue selon des principes de lisibilité inspirés des théories de l'architecture cognitive. Chaque zone possédait ses propres codes visuels permettant une navigation intuitive : variations chromatiques pour distinguer les biomes, et utilisation stratégique des contrastes pour guider naturellement le regard du joueur vers les éléments interactifs importants.

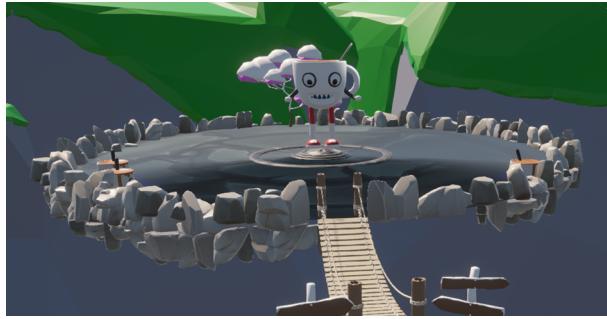
Évolutions Techniques et Artistiques Récentes La dernière phase de développement a marqué un tournant significatif dans notre approche visuelle, nous amenant à affiner et intensifier nos choix artistiques initiaux. Cette évolution s'est appuyée sur les retours des phases de test alpha et bêta, nous permettant d'ajuster notre vision selon les réactions concrètes des joueurs. L'adoption d'une palette chromatique encore plus vive et joyeuse résulte d'une analyse approfondie des métriques. Cette découverte nous a menés à repenser entièrement notre gamme colorimétrique, en poussant l'intensité des teintes tout en maintenant l'harmonie générale.

Le processus de création des assets 3D combinant Blender et Tinkercad représentait une approche innovante dans notre pipeline de production. Blender nous offrait la puissance et la flexibilité nécessaires pour les modèles complexes, tandis que Tinkercad permettait une approche plus intuitive pour les formes géométriques simples et les prototypes rapides comme les maisons présentes en l'arrière-plan. Cette dualité nous a permis d'optimiser les temps de production tout en maintenant une qualité constante.



La conception du boss final, incarné par une tasse de café géante, illustre parfaitement notre évolution artistique. Ce choix s'inscrit dans une narration visuelle cohérente où chaque élément graphique porte une signification symbolique. La tasse représente simultanément la familiarité rassurante du quotidien et la menace de l'addiction, créant une tension que le joueur ressent instinctivement avant même de comprendre intellectuellement.





Les ennemis secondaires, également modelés selon cette esthétique de café, créent un écosystème visuel cohérent qui renforce l'immersion thématique. Chaque petit ennemi possède ses propres caractéristiques permettant une identification rapide de leurs comportements spécifiques tout en maintenant l'unité artistique globale.

Les décors environnementaux ont bénéficié d'une attention particulière dans cette phase finale. Chaque élément, des plateformes aux arrière-plans, a été repensé pour soutenir la cohérence thématique tout en servant les impératifs du gameplay. L'architecture des niveaux conserve son esthétique de plateforme traditionnelle tout en intégrant harmonieusement les nouveaux éléments thématiques. Les plateformes, murs et structures environnementales maintiennent leur fonction première de support au gameplay, mais bénéficient désormais d'une cohérence visuelle renforcée qui soutient la narration symbolique du boss final.

1.2 Le Gameplay

Avant l'intégration des dernières évolutions, un ensemble complet de fonctionnalités liées au gameplay et à la synchronisation réseau avait été mis en place afin de poser les bases du jeu en solo et en multijoueur.

1. Système de déplacement et d'interaction

Le personnage jouable disposait de l'ensemble des mouvements fondamentaux : déplacement vers l'avant, l'arrière, la gauche et la droite, course, saut et attaque. La logique de déplacement reposait sur un vecteur de mouvement actualisé en temps réel en fonction des entrées du joueur. Ce vecteur était manipulé comme suit :

- Axe Z : déplacement avant/arrière.
- Axe X : déplacement latéral (gauche/droite).
- Axe Y : gestion du saut, avec ajout d'une gravité simulant la chute naturelle du personnage après un saut.



La condition de contact avec le sol était vérifiée pour restreindre certains mouvements (comme le saut), assurant ainsi un comportement physique cohérent. Un système de course permettait d'augmenter la vitesse de déplacement lorsque le joueur maintenait une touche spécifique (comme Shift).

Des interactions avec les ennemis avaient également été mises en place : le personnage pouvait infliger des dégâts via une attaque animée, et inversement, subissait des dégâts lors de collisions avec des entités hostiles.

Le joueur pouvait aussi collecter des pièces disposées dans l'environnement. Lorsqu'une pièce était ramassée, celle-ci disparaissait du niveau et un compteur s'incrémentait pour suivre le score du joueur.

2. Système d'animations

Les différentes actions du personnage étaient accompagnées d'animations spécifiques, assurant une meilleure immersion et une bonne lisibilité visuelle : Animation de marche et de course, déclenchée selon la vitesse du déplacement.

- Animation de saut, activée lors d'une impulsion verticale.
- Animation d'attaque, jouée lorsqu'un coup était porté.
- Animation d'inactivité, jouée par défaut lorsqu'aucune action n'était réalisée.

La synchronisation entre les entrées du joueur, les actions du personnage et les animations était soigneusement gérée pour garantir un retour visuel cohérent.

3. Caméra

La caméra suivait dynamiquement le personnage sur l'axe horizontal. Elle pouvait être orientée autour de l'axe vertical à l'aide de la souris, permettant de contrôler la direction du personnage de manière fluide et intuitive, tout en maintenant une vision optimale de l'environnement de jeu.

4. Adaptation au multijoueur (réseau)

Avec l'implémentation du réseau, toutes ces mécaniques de déplacement, d'action et d'interaction avaient dû être adaptées pour fonctionner dans un environnement multi-joueur. L'objectif principal était d'assurer une synchronisation en temps réel des états et actions de chaque joueur entre les différents clients connectés.

Pour cela :

- Chaque action (déplacement, saut, attaque, course) était envoyée et reçue via le réseau afin de refléter les mouvements des autres joueurs sans latence excessive.



- Le système d'animations avait été modifié pour que celles-ci soient synchronisées sur toutes les machines. Lorsqu'un joueur effectuait une action, les autres voyaient son animation correspondante en temps réel, assurant ainsi une cohérence visuelle globale.

5. Nouvelles animations et enrichissement du gameplay

Trois nouvelles animations avaient été ajoutées afin de renforcer le réalisme des mouvements du personnage :

- Une animation spécifique pour la marche vers la gauche.
- Une animation dédiée à la marche vers la droite.
- Une animation de mort, qui s'activait lorsque le personnage perdait toutes ses vies.

Ces ajouts avaient permis d'améliorer la fluidité des enchaînements visuels et d'accroître l'immersion.

Par ailleurs, une nouvelle mécanique de déplacement avait été introduite pour diversifier le gameplay : le déplacement sur des surfaces glacées. Ce système appliquait un effet de glisse qui réduisait le contrôle du joueur sur ses mouvements, rendant la navigation plus imprévisible et augmentant la difficulté sur certaines zones du niveau. Cette mécanique apportait un défi supplémentaire et nécessitait une adaptation constante du joueur selon le type de terrain.



6. Amélioration de la caméra

La caméra avait été retravaillée afin d'offrir un contrôle étendu sur deux axes : en plus de la rotation verticale (axe Y), le joueur pouvait désormais ajuster l'angle horizontal (axe X), permettant une meilleure visibilité et une liberté accrue dans l'exploration de l'environnement.

7. Synchronisation de la collecte d'objets

Le système de collecte de pièces avait été entièrement révisé pour être compatible avec le mode multijoueur. Désormais, lorsqu'une pièce est ramassée par un joueur, son état est immédiatement mis à jour et synchronisé sur tous les clients. Cela garantissait que la pièce disparaissait bien sur l'ensemble des écrans et que chaque joueur disposait d'un compteur exact reflétant la situation en temps réel.

Ensuite plusieurs améliorations ont été apportées au gameplay afin d'offrir une expérience de jeu plus fluide, plus agréable et plus adaptée aux mécaniques de déplacement imposées par notre level design.

L'un des changements majeurs concerne la fluidité des mouvements du personnage. À l'origine, le personnage ne pouvait se déplacer que lorsqu'il était en contact avec le sol. Cette contrainte rendait les déplacements rigides et peu adaptés aux phases de plateforme. Nous avons donc supprimé cette restriction : le personnage peut désormais se déplacer librement en l'air, ce qui permet d'ajuster sa trajectoire lors d'un saut ou après une chute.

Cependant, pour conserver une cohérence de gameplay, le saut reste uniquement possible au sol. Ce changement a considérablement amélioré la jouabilité, en rendant les déplacements plus naturels et en facilitant le franchissement des obstacles.

Une autre amélioration importante a été l'ajout d'un système de barre de vie visible à l'écran. Lorsqu'un joueur subit des dégâts, la barre se décrémente automatiquement, offrant un retour visuel clair sur l'état de santé du personnage.

Enfin, dans le cadre de l'amélioration de l'expérience utilisateur et de la continuité du gameplay, un système de sauvegarde a été implémenté au sein du projet. Ce système permet de conserver l'état du joueur au moment de la sauvegarde et de restaurer cet état lors d'un futur chargement de la partie. Concrètement, lors de la sauvegarde, plusieurs données sont enregistrées :

- La position du joueur dans l'environnement de jeu (coordonnées X, Y, Z),
- La valeur actuelle de sa barre de vie,
- Son score,
- Les pièces déjà collectées,



- Et le nom de la scène active.

Ces informations sont regroupées dans une structure de données, puis sérialisées au format JSON avant d'être écrites dans un fichier de sauvegarde local. L'utilisation du format JSON rend les données à la fois faciles à manipuler, lisibles, et persistantes entre les sessions de jeu.

Lors du chargement d'une partie, le fichier est lu et les données sont appliquées dès le démarrage du jeu. Le joueur retrouve ainsi :

- Sa position exacte au moment de la sauvegarde,
- Le même niveau de vie et de score,
- Et un environnement cohérent où les pièces déjà récupérées ne sont plus présentes.

L'intégration de cette fonctionnalité représente une avancée importante dans la qualité globale du jeu. Elle renforce l'immersion, permet une gestion flexible des sessions de jeu.

1.3 Le Game design

Notre approche du game design s'enracinait dans une philosophie d'accessibilité progressive, inspirée des grands classiques du genre plafformer comme Super Mario, Rayman ou encore SackBoy. L'objectif central consistait à créer une courbe d'apprentissage naturelle où chaque mécanisme de jeu s'introduit organiquement, permettant au joueur de développer ses compétences sans frustration excessive.

Nous avions étudié les principes de l'affordance en design d'interaction, appliquant ces concepts aux éléments de jeu. Un piège dangereux devait "paraître" dangereux avant même que le joueur n'en comprenne le fonctionnement exact. Cette approche visuelle préventive permettait d'éviter les morts injustes qui brisent l'immersion et génèrent de la frustration.

Le système de pièges constitue le point principal de notre gameplay, chaque type étant conçu pour enseigner une compétence spécifique au joueur. Les sols équipés de pics, par exemple, introduisent la notion de timing dans les sauts, forçant le joueur à observer les patterns avant d'agir. Les marteaux oscillants développent la perception spatiale et la patience, tandis que les lames rotatives exigent une coordination précise entre observation et exécution.

Les pièges à déclenchement automatique représentaient notre première incursion dans les mécaniques contextuelles. Ces systèmes utilisent des zones de détection invisibles qui analysent la position, la vitesse et même la direction du joueur pour déterminer le moment optimal d'activation. Cette technologie nous permettait de créer des défis qui s'adaptent au style de jeu individuel, générant des expériences uniques même lors de parties répétées.





Les mécanismes à basculement introduisaient une dimension psychologique dans le game-play. Contrairement aux pièges automatiques, ceux-ci créent une tension d'anticipation où le joueur sait qu'il déclenche lui-même le danger. Cette mécanique développe la réflexion stratégique et la prise de décision sous pression, compétences transférables à d'autres aspects du jeu.

Le système de pièges séquentiels représentait notre approche la plus sophistiquée de la conception de puzzles d'action. Ces mécanismes enchaînent plusieurs actions dans un ordre pré-déterminé, obligeant le joueur à comprendre non seulement chaque élément individuel, mais également leurs interactions. Cette complexité croissante prépare naturellement aux défis du boss final.

Les pièges combinés synthétisent tous les apprentissages précédents, créant des défis multidimensionnels qui testent simultanément réflexes, planification et adaptabilité. Ces sections servent de "examens pratiques" validant la maîtrise des compétences acquises précédemment.

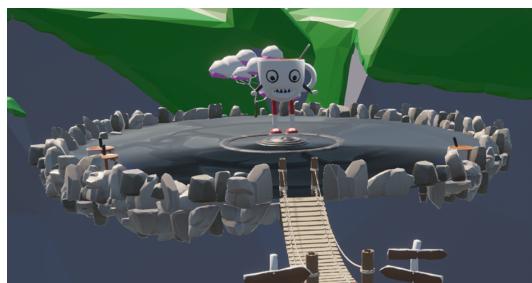
La conception des ennemis (bots) s'appuyait sur des principes d'IA comportementale différenciée. Chaque type d'ennemi possède son propre arbre de décision, créant des patterns reconnaissables mais non prévisibles. Les ennemis agressifs utilisent des algorithmes de poursuite directe, tandis que les stratégiques emploient des tactiques d'embuscade et de positionnement territorial. Les variations selon le positionnement introduisent une dimension tactique dans les affrontements. Un même ennemi peut adopter des stratégies différentes selon qu'il se trouve en hauteur, dans un espace confiné, ou près d'autres ennemis. Cette adaptabilité force le joueur à développer un différentes techniques d'approche.



Le système de collection de pièces dépasse la simple accumulation de points pour s'intégrer dans une économie ludique complète. La distribution spatiale des pièces guide subtilement la progression du joueur, l'encourageant à explorer des zones optionnelles et à prendre des risques calculés. Certaines pièces sont volontairement placées dans des positions qui testent les limites des compétences acquises.

Les passages secrets créent une couche de jeu supplémentaire pour les joueurs curieux et expérimentés. Ces zones cachées contiennent non seulement des récompenses exclusives, mais également des défis uniques qui poussent les mécaniques de base dans leurs retranchements. Cette approche à deux niveaux permet de satisfaire simultanément les joueurs occasionnels et les perfectionnistes.

L'implémentation de la tasse de café géante comme antagoniste principal représente l'aboutissement de notre réflexion narrative et ludique. Ce choix iconographique porte une charge symbolique profonde : la tasse incarne les habitudes quotidiennes qui peuvent devenir addictives et limitantes. Le joueur ne combat pas simplement un ennemi, mais métaphoriquement sa propre dépendance.



La mécanique de combat indirect, où le joueur doit poursuivre et détruire les petites tasses fuyardes, inverse les codes traditionnels du boss fight. Au lieu d'affronter directement un adversaire statique et puissant, le joueur devient le poursuivant face à des cibles agiles et évasives. Cette inversion crée une dynamique de jeu unique qui requiert patience, stratégie et persistance.

Le comportement de fuite des petites tasses utilise des algorithmes d'évitement qui simulent une intelligence rudimentaire mais crédible. Chaque tasse analyse en temps réel la position du joueur, les obstacles environnants, et la position de ses congénères pour calculer la trajectoire d'évasion optimale. Cette complexité technique invisible crée une expérience de jeu fluide et naturelle.

L'agrandissement de la carte répond à une nécessité à la fois technique et ludique. Les nouvelles zones spécifiquement conçues pour les ennemis mobiles utilisent des principes d'architecture de niveau inspirés de l'urbanisme et de l'éthologie. Ces espaces offrent suffisamment de liberté de mouvement pour permettre des comportements d'IA complexes tout en maintenant des contraintes qui empêchent les situations de blocage.

La continuité visuelle et fonctionnelle entre l'ancienne et la nouvelle zone nécessitait



une attention particulière aux transitions. Nous avons développé des "zones tampons" qui opèrent graduellement la transition entre les différents biomes, évitant les ruptures visuelles brutales qui briseraient l'immersion.

1.4 L'Intelligence artificielle

Avant l'intégration des dernières évolutions, un ensemble complet de fonctionnalités liées à l'intelligence artificielle et au site web avait été mis en place afin de poser les bases du jeu en solo et en multijoueur.

Dès les premières étapes du projet, une réflexion approfondie a été engagée pour concevoir des systèmes d'IA variés, dynamiques et capables de s'adapter à différents types de comportements, tout en assurant une cohérence complète avec les mécaniques de gameplay et les contraintes du multijoueur. Le développement de ces intelligences artificielles n'a cessé d'évoluer au fil des versions du jeu, tant sur le plan technique que fonctionnel.

Initialement, plusieurs prototypes d'ennemis ont été créés afin de représenter les différents types de dangers que le joueur pouvait rencontrer. Chaque IA répondait à un schéma comportemental spécifique, allant de la simple poursuite à distance à des attaques déclenchées par des conditions précises.

C'est ainsi que plusieurs types d'ennemis ont vu le jour, chacun ayant été conçu avec un comportement unique, adapté à son rôle dans le jeu, et pensé pour diversifier les situations de combat et renforcer l'immersion du joueur.

- Le premier type d'ennemi que nous avons mis en place est l'ennemi volant. Ce dernier évolue à une certaine hauteur au-dessus du sol, ce qui lui permet de survoler les obstacles et d'avoir une vision dégagée sur l'environnement. Son comportement repose sur un système de détection basé sur les tags : dès qu'un joueur entre dans son champ de vision et est identifié par le tag « Player », l'ennemi réagit immédiatement en lançant un projectile. Il s'agit ici d'une boule de feu, générée dynamiquement et dirigée automatiquement vers la position actuelle du joueur au moment du tir. Le fonctionnement de ce projectile est conditionné à plusieurs cas : si la boule entre en collision avec un objet quelconque du décor, elle se détruit automatiquement à l'aide de la commande Network.Destroy. Il en va de même lorsqu'elle atteint le joueur, avec en plus l'appel de la fonction TakeDamage, qui inflige les dégâts correspondants. Enfin, pour éviter que la boule ne reste indéfiniment active dans la scène, un système de désintégration automatique est prévu : si la boule ne touche rien après un certain temps ou une certaine distance parcourue, elle se détruit d'elle-même.
- Un autre type d'ennemi que nous avons implémenté est la tourelle, un ennemi stationnaire mais redoutable à moyenne portée. Son fonctionnement est relativement similaire à celui de l'ennemi volant, bien qu'elle ne se déplace pas. Placée stratégiquement dans certains endroits de la carte, la tourelle analyse son environnement et, dès qu'un joueur entre dans sa zone de détection, elle engage immédiatement une séquence d'attaque. Là encore, un projectile en l'occurrence une boule de feu est lancé en direction du joueur. Cette attaque suit exactement



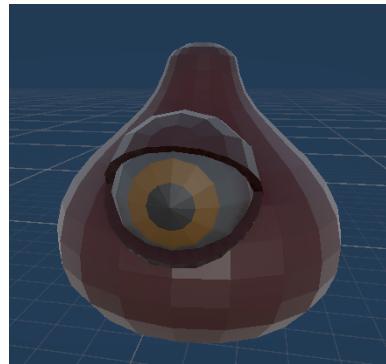
la même logique que celle décrite précédemment : collision avec un obstacle ou avec le joueur destruction du projectile, et dans le cas du joueur, application de dégâts via TakeDamage. Si le projectile ne touche aucune cible, il est supprimé automatiquement après un certain temps ou à partir d'une distance prédéfinie. Ce type d'ennemi force le joueur à être vigilant à son environnement et à anticiper les angles d'attaque, ce qui contribue à instaurer une tension constante dans les zones couvertes par des tourelles.

- Nous avons également mis en place un ennemi d'un tout autre type : le bloc tombant. Celui-ci ne se déplace pas à proprement parler, mais attend passivement qu'un joueur s'approche pour agir. Il est équipé d'un système de détection verticale. Dès que le joueur passe directement en dessous de lui, et donc entre dans sa zone de déclenchement (dont la taille correspond approximativement à celle du bloc), l'ennemi engage une attaque : il tombe brutalement vers le sol dans le but d'écraser le joueur. Si ce dernier parvient à esquiver à temps, le bloc s'écrase au sol puis, après un bref délai, il remonte lentement jusqu'à sa position initiale, laissant ainsi le champ libre au joueur. Ce système repose sur un mouvement contrôlé par un simple mouvement rectiligne, avec une descente rapide et une remontée plus lente pour équilibrer la difficulté. Le bloc tombant constitue un piège dynamique qui oblige le joueur à anticiper ses déplacements et à rester mobile.



Enfin, nous avons conçu l'ennemi de base, qui incarne l'archétype classique du mob de mêlée. Cet ennemi patrouille librement dans la carte en suivant un parcours aléatoire généré via un système de Patrol ou de déplacement entre points définis. Il utilise un NavMeshAgent pour naviguer sur la carte, ce qui lui permet d'éviter les obstacles et de circuler naturellement dans l'environnement. Lorsqu'un joueur entre dans sa zone de perception, il cesse immédiatement sa patrouille et entre dans un mode de poursuite. L'ennemi se dirige alors directement vers le joueur en accélérant, et tente de l'atteindre pour lui infliger des dégâts par simple contact. Ce type de comportement oblige le joueur à rester constamment en mouvement, à esquiver, voire à contre-attaquer rapidement. L'ennemi de base peut être éliminé par le joueur après plusieurs coups, ce qui ajoute une dimension stratégique au combat.





En parallèle, une logique de détection visuelle a été implémentée.

Chaque IA possède un champ de vision défini, au sein duquel elle peut percevoir le joueur et engager une réaction adaptée grâce au tag player en effet les enemies sont donc en constante recherche du tag et des qu'il est trouvé il lance leurs fonction d'attaque.

En cas de perte de visuel, les IA retournent à leur point de patrouille initial, générant ainsi un comportement crédible et immersif.

Ce système de détection a par ailleurs nécessité une optimisation rigoureuse du positionnement des colliders et des tags, afin de garantir des déclenchements fiables dans tous les contextes.

Avec l'introduction du mode multijoueur en réseau via Mirror, une refonte complète de l'architecture des IA s'est avérée nécessaire. Les ennemis devaient désormais être visibles et synchronisés sur tous les clients, ce qui a impliqué une mise en réseau de chaque aspect de leur fonctionnement : position, points de vie, animations, déclenchement des attaques et effets visuels. Pour y parvenir, les SyncVar et les Remote Procedure Calls (RPCs) ont été utilisés de manière fine afin de garantir que toutes les interactions liées aux IA soient identiques pour chaque joueur.

Ce travail de synchronisation a été essentiel pour assurer une expérience fluide et sans incohérences, particulièrement lors des phases de combat ou de poursuite.

Un autre défi majeur a été la gestion des effets spéciaux liés aux attaques ennemis. Dans les premières versions, les effets visuels comme les rayons ou les ondes de choc n'étaient pas visibles simultanément pour tous les joueurs, ce qui nuisait fortement à la jouabilité.

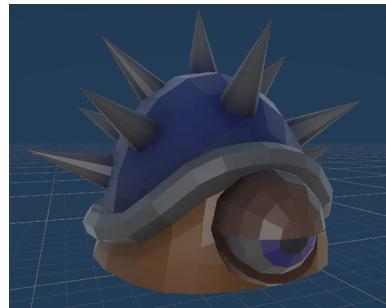
Ce problème a été résolu en plaçant tous les effets concernés sous contrôle serveur, en les déclenchant au bon moment via des hooks réseau, garantissant ainsi leur affichage synchronisé.

Ce travail a permis d'atteindre une stabilité visuelle et une clarté dans les phases de combat multijoueur.



Dans le prolongement de ces systèmes existants, un nouveau type d'ennemi a récemment été introduit afin d'enrichir encore davantage la diversité des situations de combat et d'apporter de nouvelles mécaniques dans le gameplay.

Premièrement, nous avons ajouté un nouvel adversaire à distance, qui reprend en partie le comportement de l'ennemi de base. Comme ce dernier, il se déplace librement sur la carte à l'aide d'un NavMeshAgent, se repérant sur une surface de navigation définie il a 4 pattern différent soit carre cercle ligne ou zigzag. Toutefois, ce nouvel ennemi ne cherche pas à entrer en contact direct avec le joueur pour infliger des dégâts. Dès qu'il détecte la présence du joueur dans son champ de vision basé sur le tag « Player », il se met en mouvement pour s'approcher, mais s'arrête volontairement à une certaine distance de sécurité, évitant ainsi toute confrontation de mêlée. Une fois à portée optimale, il déclenche automatiquement une attaque à distance, en lançant un projectile de type "goutte de café".



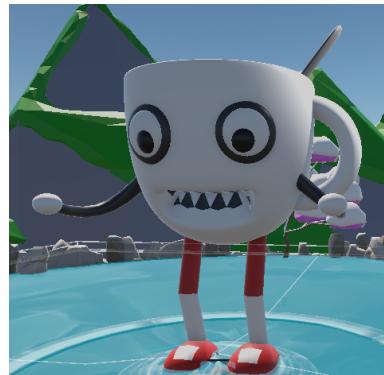
Ce projectile reprend la logique déjà mise en place dans les IA précédentes, notamment celles de l'ennemi volant et de la tourelle. Il est instancié côté serveur, est dirigé vers la position du joueur au moment du tir, et suit une trajectoire rectiligne. Si la pique touche un obstacle du décor, elle est immédiatement détruite via un Network.Destroy. En revanche, si elle atteint le joueur, la fonction TakeDamage est alors appelée, infligeant des dégâts en fonction des paramètres définis. Enfin, pour éviter que le projectile ne reste actif trop longtemps dans la scène, un système de désintégration automatique a été mis en place : il s'autodétruit au-delà d'un certain temps ou d'une certaine distance, s'il n'a touché aucune cible. Ce nouvel ennemi permet de remplacer à la fois l'ennemi volant et la tourelle, dans une volonté de simplification et d'optimisation du gameplay, tout en concentrant plusieurs mécaniques dans une seule entité plus cohérente et flexible.

Deuxièmement, un nouvel ennemi intuable a également été intégré. Contrairement aux autres IA, le joueur ne peut pas l'éliminer. Ce type d'ennemi est conçu avant tout comme un piège mobile. Lorsqu'un joueur entre dans son rayon d'action, il déclenche immédiatement une onde de choc qui se propage autour de lui. Cette onde oblige le joueur à esquiver rapidement pour ne pas subir de dégâts. Son rôle principal est d'augmenter la difficulté de certaines zones et d'exiger du joueur une vigilance permanente, sans toutefois constituer un adversaire classique que l'on peut affronter de front.

En parallèle, un travail conséquent a été réalisé pour introduire un boss central, situé dans une arène dédiée, pensée comme un moment clé et marquant dans le déroulement



du jeu. Contrairement aux autres ennemis, le boss est invincible au départ. Le joueur ne peut pas l'attaquer directement, ce qui rompt avec les schémas classiques de combat. Une fois le combat engagé, le boss entre dans une séquence offensive répétée au cours de laquelle il exécute deux attaques en simultané : une onde de choc circulaire, que le joueur doit impérativement esquiver en sautant ou en se déplaçant rapidement, et un projectile frontal, tiré à intervalles réguliers et dirigé droit vers la position du joueur. Ces attaques combinées visent à tester la réactivité, le positionnement et le sens du timing du joueur, créant une véritable épreuve d'endurance.



Cependant, la clé de la victoire ne repose pas sur l'affrontement direct avec le boss, mais sur une mécanique indirecte. Pour rendre le boss vulnérable, le joueur doit d'abord éliminer plusieurs mini-ennemis situés autour de lui. Ces entités possèdent un comportement de fuite automatique : dès qu'elles détectent le joueur, elles s'éloignent rapidement de lui. Elles s'arrêtent cependant à intervalles réguliers, ce qui constitue le seul moment où le joueur peut les atteindre. Ce système crée une dynamique particulière : le joueur doit gérer à la fois l'évitement des attaques du boss et la poursuite de ces ennemis secondaires. Pour ajouter un degré de complexité supplémentaire, lorsqu'un joueur réussit à toucher l'un de ces ennemis, il subit en retour un effet de ralentissement temporaire, qui réduit sa mobilité et augmente considérablement la difficulté du combat. Cette contrainte stratégique pousse le joueur à évaluer le bon moment pour attaquer ces cibles secondaires, sans se mettre en danger.



Une fois tous les mini-ennemis éliminés, le boss entre dans une phase de vulnérabilité critique. À ce moment précis, il meurt automatiquement, et déclenche une fonction de fin



de partie. Cette fonction vérifie si le joueur a bien éliminé le boss et récupéré l'ensemble des objets clés du jeu, notamment les cafés répartis sur la carte. Si toutes les conditions sont remplies, la séquence finale est déclenchée, marquant la conclusion de l'expérience de jeu.

Ce système de boss complexe et original permet de proposer une confrontation beaucoup plus riche et scénarisée, mêlant action, tactique, observation et mobilité. Il ne s'agit plus seulement d'un duel de force, mais d'un combat intelligent, où chaque mouvement compte, où chaque ennemi secondaire a une importance stratégique, et où la victoire ne s'obtient qu'en maîtrisant tous les éléments du gameplay.

1.5 Le Multijoueur

Le mode multijoueur représentait l'un des défis techniques les plus ambitieux de notre projet The Coffee Tales. Dès la conception initiale, nous avions identifié la nécessité de permettre à plusieurs joueurs de participer simultanément à l'expérience de jeu, créant ainsi une dimension collaborative essentielle à l'engagement des utilisateurs.

Spécifications fonctionnelles requises :

- Support minimum de 2 joueurs simultanés
- Possibilité de connexion via réseau local (LAN)
- Interface utilisateur intuitive pour l'établissement des connexions
- Synchronisation temps réel des actions et mouvements
- Gestion centralisée des sessions de jeu
- Maintien de la cohérence de l'état du jeu entre tous les participants

Contraintes techniques identifiées :

- Limitations des ressources réseau disponibles
- Nécessité de maintenir des performances optimales malgré la communication réseau
- Compatibilité entre différentes configurations matérielles
- Gestion des erreurs de connexion et des déconnexions inattendues

L'implémentation du multijoueur nécessitait une architecture robuste capable de gérer les communications inter-clients tout en préservant l'expérience de jeu fluide que nous



avions développée en mode solo. Cette fonctionnalité devait s'intégrer harmonieusement avec les systèmes existants sans compromettre la stabilité générale du projet.

Notre objectif était de créer une expérience multijoueur transparente où les joueurs pourraient se connecter facilement depuis leurs propres ordinateurs et profiter d'une session de jeu partagée sans être gênés par des problèmes techniques. L'accent était mis sur la simplicité d'utilisation et la stabilité de la connexion plutôt que sur des fonctionnalités complexes.

Au départ, nous avions une incompréhension fondamentale sur ce qu'était réellement le réseau et comment l'implémenter dans notre jeu. Nous pensions initialement que l'utilisation d'un écran scindé suffirait pour répondre aux exigences du multijoueur, mais nous avons rapidement compris que cela ne correspondait pas aux besoins réels du projet. L'écran scindé permettait certes de jouer à plusieurs sur un même appareil, mais il était limité et peu adapté pour une expérience plus immersive et flexible.

Cette prise de conscience nous a menés à abandonner cette approche et à nous concentrer sur l'implémentation d'un réseau local (LAN), afin que plusieurs joueurs puissent se connecter à partir d'ordinateurs différents. Cette réorientation, bien qu'impliquant un retard dans notre planning, était nécessaire pour créer une expérience multijoueur authentique et moderne.

Pour l'implémentation réseau, nous avons utilisé un asset appelé Mirror sur Unity, qui propose de nombreuses options facilitant la gestion du réseau et la communication entre les différents clients. Cet outil nous a permis d'intégrer un système d'Host/Client où un joueur peut héberger une partie en tant que serveur, tandis qu'un autre peut se connecter en tant que client depuis un autre appareil ayant la même version du jeu.

Le choix de Mirror s'est avéré judicieux pour plusieurs raisons :

- Solution open source
- Documentation extensive et communauté active
- Intégration native avec Unity
- Flexibilité architecturale permettant la personnalisation selon nos besoins

Un des premiers ajouts majeurs a été la mise en place d'un menu de démarrage dédié au mode multijoueur. Désormais, au lancement du jeu, les joueurs ont la possibilité de choisir entre "Héberger une partie" ou "Rejoindre une partie" en entrant l'IP de l'hôte. Cette interface simplifie énormément la connexion entre les utilisateurs en masquant la complexité technique sous-jacente derrière une interface utilisateur intuitive.

Nous avons également implémenté un système de pause synchronisée. Désormais, lorsque l'hôte met le jeu en pause, cela affecte également le client, garantissant ainsi une expérience cohérente pour tous les joueurs connectés. Cependant, le client ne peut pas



mettre le jeu en pause de son côté, ce qui assure un contrôle centralisé de la session de jeu.

Cette approche évite les conflits de contrôle et maintient une hiérarchie claire d'autorité dans la gestion de la session multijoueur.

L'intégration de Mirror a représenté une avancée majeure, car elle a ouvert la voie à un mode multijoueur plus dynamique et extensible. Nous avons commencé à explorer des fonctionnalités telles que la gestion des connexions en temps réel, la synchronisation des mouvements et des interactions, ainsi que l'affichage des avatars de chaque joueur.

Ces explorations nous ont permis de comprendre les mécanismes complexes de la synchronisation réseau et de développer une expertise dans la gestion des états partagés entre plusieurs instances du jeu.

Après de nombreux efforts et ajustements techniques, nous avons finalement réussi à faire fonctionner le réseau entre deux PC distincts. Cette percée majeure a été rendue possible par :

Nous avons ajusté les paramètres de transport, les valeurs de timeout, les ports de communication et les protocoles pour permettre une communication stable entre machines distinctes.

L'implémentation d'un système de saisie manuelle d'adresse IP avec validation des formats et gestion d'erreurs explicites s'est révélée être la solution la plus fiable. Identification et résolution des problèmes liés aux pare-feu et paramètres de sécurité qui bloquaient les communications entre machines.

Avec la communication inter-machines fonctionnelle, nous avons pu nous concentrer sur l'amélioration de la qualité de synchronisation. Les résultats ont été très satisfaisant avec une réduction drastique des bugs de synchronisation. Les problèmes de téléportation, de désynchronisation d'animations et de perte de contrôle qui caractérisaient nos premiers tests ont été pratiquement éliminés.

La synchronisation des objets interactifs a également été grandement améliorée, avec des états cohérents entre tous les clients et une gestion optimisée des interactions simultanées.

Au terme du développement, nous avons atteint notre objectif principal : deux joueurs peuvent désormais se connecter de manière stable depuis des PC différents et profiter d'une expérience de jeu partagée de qualité. Le processus de connexion par IP fonctionne de manière fiable, et l'expérience de jeu multijoueur approche la qualité du mode solo. Bien que nous observions encore de légers délais occasionnels, ces problèmes de latence mineurs n'affectent pas significativement le plaisir de jeu et restent dans des limites acceptables pour une implémentation de réseau local.



1.6 Les Menus

Avant l'intégration des dernières fonctionnalités, deux menus principaux avaient été conçus et intégrés dans le jeu : le menu principal et le menu en jeu. Ces menus jouaient un rôle central dans la navigation et la gestion de l'expérience utilisateur, tant en solo qu'en multijoueur.

1. Menu principal

Le menu principal apparaissait automatiquement au lancement du jeu. Il proposait trois boutons principaux :

- “Start Game”,
- “Settings”,
- “Quit”.

Cependant, une modification majeure avait été apportée à la structure de ce menu. Contrairement à la version initiale où le bouton “Start Game” lançait directement la partie, ce dernier ouvrait désormais un sous-menu, permettant au joueur de choisir entre deux modes de jeu réseau :

- “Host Game” : cette option permettait au joueur de lancer un serveur et de devenir l'hôte de la partie.
- “Join Game” : cette option offrait la possibilité de rejoindre une partie existante en saisissant l'adresse IP de l'hôte.

Les boutons “Settings” et “Quit” étaient restés inchangés :

- “Settings” donnait accès aux réglages liés à l'affichage (résolution, mode plein écran).
- “Quit” permettait de fermer complètement l'application.

2. Menu en jeu Le menu en jeu, tel qu'il avait été conçu initialement, était accessible à tout moment par le joueur, simplement en appuyant sur une touche. Il mettait alors le jeu en pause et affichait un menu contenant plusieurs boutons :

- “Resume”, pour reprendre la partie,
- “Settings”, pour ajuster les paramètres sans quitter la session,
- “Menu”, pour retourner au menu principal,



- et “Quit”, pour fermer l’application.

Cependant, dans un souci de cohérence avec le mode multijoueur synchronisé, ce système avait été entièrement repensé. L’un des changements majeurs concernait la gestion exclusive de la pause, désormais réservée à l’hôte de la partie. Ce dernier seul pouvait interrompre la session en cours, déclenchant alors l’affichage des menus de pause.

Lorsque l’hôte mettait le jeu en pause, deux comportements distincts étaient observés :

- Côté hôte, un menu complet de gestion de partie s’affichait, avec les options suivantes :
 1. “Resume” : reprenait la partie à l’endroit où elle avait été interrompue
 2. “Restart” : relançait une nouvelle partie depuis le début, sans passer par le menu principal.
 3. “Menu” : renvoyait tous les joueurs vers le menu principal et fermait le serveur.
 4. “Quit” : fermait définitivement l’application.
- Côté clients, un menu minimaliste était affiché, ne contenant aucun bouton, uniquement le message “Pause”, indiquant que le jeu était en attente d’une action de l’hôte. Les clients n’avaient plus aucun contrôle sur la pause et devaient attendre que l’hôte reprenne, redémarre ou termine la partie.

Lorsque ce menu de pause était activé, le temps de jeu était complètement figé, garantissant que tous les joueurs reprenaient exactement au même moment, assurant ainsi une parfaite synchronisation du gameplay.

3. Évolutions et nouvelles améliorations des menus

Dans la continuité des efforts d’amélioration de l’expérience utilisateur, plusieurs évolutions significatives ont été apportées aux différents menus du jeu, tant sur le plan fonctionnel qu’esthétique.

Tout d’abord, le menu principal a été repensé, tant au niveau du design que des fonctionnalités. Un nouveau bouton “Load Game” a été ajouté, permettant au joueur de charger une partie précédemment sauvegardée grâce au système de sauvegarde mis en place. Cette fonctionnalité améliore considérablement l’expérience de reprise de partie et offre une plus grande liberté au joueur.

Par ailleurs, le design visuel du menu principal a été retravaillé. Désormais, celui-ci s’affiche sur une scène d’arrière-plan aux couleurs chaleureuses rappelant l’ambiance d’un café, en cohérence avec le thème général du jeu. Des éléments de décor issus de l’univers du jeu y ont été intégrés, renforçant l’immersion et la cohérence visuelle globale.



Toujours dans une optique d'amélioration de l'expérience utilisateur, un écran de chargement a été ajouté. Cet écran s'affiche désormais automatiquement à chaque fois qu'une partie est lancée, rejointe ou chargée, assurant une transition visuelle fluide entre les différentes étapes de navigation. Cela permet non seulement d'informer le joueur du chargement en cours, mais aussi de masquer d'éventuels temps de latence, rendant l'expérience plus professionnelle et immersive.

Ensuite, le menu pause a également été enrichi d'un bouton "Save", permettant au joueur de sauvegarder sa progression en cours de partie. Ce bouton déclenche le système de sauvegarde, enregistrant notamment la position du joueur, sa vie, son score ainsi que les pièces collectées.



Un comportement non souhaité a également été corrigé dans ce menu : auparavant, le joueur pouvait toujours bouger la caméra en jeu, même lorsque le menu pause était actif. Désormais, lors de l'ouverture du menu pause, la caméra est figée, empêchant toute interaction ou déplacement accidentel du point de vue du joueur pendant la pause.

4. Ajout d'un menu de mort

Enfin, pour compléter l'expérience de jeu et mieux encadrer la fin de partie, un menu de mort a été implémenté. Ce dernier s'affiche automatiquement dès que le joueur perd la partie, avec une inscription "Game Over" en rouge bien visible. Il comprend trois boutons :

- "Menu" : retourne à l'écran principal,
- "Restart" : relance la partie depuis le début,
- "Quit" : ferme complètement le jeu.





Dans une logique cohérente avec le fonctionnement en multijoueur, ce menu complet est réservé à l'hôte, car c'est lui qui contrôle la session. Les clients disposent eux aussi d'un menu de mort, mais plus minimaliste, affichant uniquement le message "Game Over" sans les boutons de contrôle. Ce choix s'inscrit dans la volonté de centraliser la gestion des actions entre les mains de l'hôte.

Grâce à l'ensemble de ces évolutions, l'architecture des menus est devenue plus robuste, plus immersive et parfaitement adaptée à une expérience multijoueur fluide et cohérente, tout en garantissant une meilleure maîtrise de la progression du joueur.

1.7 L'Audio

Les éléments sonores d'ambiance sont désormais entièrement terminés. Ils couvrent une grande diversité d'environnements présents dans le jeu, que ce soit des forêts calmes, des villes animées ou des zones mystérieuses. Ces sons contribuent fortement à plonger le joueur dans l'univers du jeu, en rendant chaque lieu vivant et crédible.

De plus, les musiques d'ambiance ont été composées et finalisées. Chaque musique a été pensée pour accompagner et renforcer les émotions liées aux différentes phases du jeu : exploration tranquille, combat intense, menus, ou encore moments clés de l'histoire. Ces compositions musicales aident à créer une atmosphère cohérente et immersive qui renforce l'identité sonore du jeu. Tous ces éléments sonores ont été intégrés dans le moteur du jeu et font actuellement l'objet de tests pour vérifier leur fonctionnement dans toutes les situations possibles.

En parallèle, l'équipe se concentre sur l'optimisation de l'intégration sonore. Il s'agit d'assurer que le jeu puisse gérer toutes les pistes audio efficacement, sans ralentissements ni bugs, et que la qualité sonore reste optimale même dans les situations complexes. Des tests de performance audio sont programmés pour la semaine prochaine, afin de valider cette intégration.

La création des autres sons progresse aussi de manière satisfaisante. L'équipe travaille simultanément sur les bruitages liés aux actions du joueur, tels que les sauts, les coups, les pas, ou encore les interactions avec l'environnement. Par ailleurs, les sons de l'interface utilisateur, comme les clics, notifications ou confirmations — sont développés pour rendre l'expérience plus intuitive et agréable.



Enfin, les premières versions des sons d'armes et d'impacts sont déjà finalisées et intégrées dans la version actuelle du jeu. Les tests internes montrent des retours très positifs sur ces éléments, même si quelques ajustements restent à faire, notamment sur l'équilibrage des volumes pour que rien ne soit trop fort ou trop discret.

1.8 Le Site Web

la conception du site web du projet a représenté un autre chantier d'envergure. Dès les premières étapes, une étude des langages de développement web a été réalisée, avec l'apprentissage des fondamentaux de HTML, CSS et JavaScript. L'objectif était de produire un site à la fois informatif, esthétique et en cohérence avec l'univers visuel du jeu. Plusieurs maquettes ont été réalisées et testées afin de définir une direction graphique claire, inspirée de sites professionnels comme ceux de grandes franchises vidéoludiques tel que electronique art , ubisoft ou encore epic Games.

Le site a été pensé comme une véritable vitrine du jeu, présentant ses mécaniques, ses personnages, ses visuels, et les membres de l'équipe de développement. Un soin particulier a été apporté à l'ergonomie, avec une navigation fluide, des boutons interactifs, et un design cohérent avec l'univers cartoon de The Coffee Tales lorsque vous observez la page du jeu en lui-même. Afin de garantir une accessibilité optimale, une version mobile responsive a été implémentée avec un menu burger et une mise en page adaptative.

Une fois la structure technique du site stabilisée, il a été mis en ligne sur GitHub Pages, facilitant son hébergement et sa maintenance. Cette transition a permis une mise à jour plus rapide des contenus (captures d'écran, vidéos, informations) et une ouverture du projet à une communauté extérieure.

Le site n'est pas simplement décoratif : il constitue un outil de communication et d'engagement, qui sera amené à accueillir le jeu jouable en ligne à terme.

Ce site respecte les droits d'auteur donc nous avons décidé de mettre en avant toute les ressources utilisées afin de créer notre projet dans sa globalité avec les applications utilisées tel que :

- unity : le développement du jeu en lui-même
- Blender : création des personnages et de la map
- Canva : design des jachets des jeux
- Vscode : codage du jeu en C#

Des sites tel que :

- - Epic Games



- - Steam
- - GOG
- - Electronic Arts
- - Nitendo
- - Origin

Le travail accompli sur l'intelligence artificielle et sur le site web a représenté une part essentielle de l'évolution du projet.

Il a permis de donner vie aux ennemis du jeu de manière crédible et fluide, tout en assurant une présentation publique claire et professionnelle de notre production. Ces deux composantes, bien que très différentes techniquement, ont été pensées en synergie afin de soutenir l'expérience utilisateur, tant dans le jeu que dans l'univers qui l'entoure.

En parallèle du développement du jeu The Coffee Tales, le site web officiel a continué à évoluer afin d'offrir une vitrine de plus en plus fidèle et immersive de notre univers. Conçu dès le départ comme un outil de présentation, d'information et de communication avec les futurs joueurs, le site a bénéficié de plusieurs phases de mise à jour importantes, dont la dernière en date a marqué un tournant majeur dans sa qualité visuelle, sa lisibilité et sa cohérence avec l'esthétique du jeu.



L'une des améliorations les plus significatives apportées récemment a été l'intégration de véritables captures d'écran du jeu, prises directement depuis les phases de gameplay. Contrairement aux images de démonstration précédentes, qui étaient issues de prototypes ou de rendus en développement, ces nouvelles captures sont issues de séquences concrètes, sélectionnées avec soin pour mettre en valeur l'expérience de jeu réelle. Elles ont été choisies pour refléter la richesse visuelle du jeu, la diversité des environnements, la mise en scène des ennemis et les interactions du joueur avec l'univers.

Ces images ont été intégrées de façon stratégique dans les différentes sections du site, notamment dans la page d'accueil, les galeries visuelles et les sections de présentation du gameplay. L'objectif est de permettre au visiteur de se projeter : en voyant les environnements colorés, les effets d'attaque, ou les éléments de décor tels qu'ils apparaissent dans le jeu, l'utilisateur peut désormais se faire une idée précise de ce qu'il vivra manette



en main. Ce changement a considérablement renforcé l'impact du site, en remplaçant les éléments abstraits ou génériques par des visuels concrets, pertinents et directement liés à notre projet.

Parallèlement à cette évolution visuelle, un travail d'allègement du contenu textuel a également été mené. Lors des premières versions, certaines sections du site comportaient des blocs de texte relativement longs et redondants, ce qui pouvait alourdir la navigation ou décourager les lecteurs. Nous avons donc procédé à une refonte rédactionnelle ciblée, visant à conserver l'essentiel des informations tout en les rendant plus digestes. Des paragraphes ont été condensés, des tournures simplifiées, et certaines zones explicatives ont été transformées en éléments visuels ou interactifs (icônes, survols, encadrés). Ce travail a permis d'améliorer considérablement la lisibilité du site, en rendant chaque section plus claire, plus directe et mieux hiérarchisée.

Un autre axe d'amélioration important a concerné la représentation du personnage principal. Dans les premières versions du site, les images du héros étaient issues de versions intermédiaires du modèle 3D, parfois éloignées de l'apparence finale que le joueur incarne réellement dans le jeu. Ce décalage visuel pouvait introduire une confusion entre ce que l'on voyait sur le site et ce que l'on découvrait dans le jeu. Pour corriger cela, nous avons intégré de nouvelles images du personnage, plus fidèles, plus expressives, et mieux mises en scène. Ces images utilisent désormais la version finale du modèle 3D, avec les textures, animations et poses officielles. Elles ont été retravaillées en haute qualité, intégrées dans des arrière-plans issus du jeu, et accompagnées de légendes qui décrivent brièvement les capacités du personnage ou ses particularités. Le résultat est une meilleure cohérence esthétique et narrative, qui permet au visiteur de créer un lien immédiat entre le personnage présenté sur le site et celui qu'il incarnera dans le jeu.

Enfin, toutes ces améliorations ont été réalisées en cohérence avec l'univers graphique de The Coffee Tales. Les nouvelles photos ont été intégrées dans un cadre visuel chaleureux, reprenant les couleurs principales du jeu (bruns, dorés, verts doux) afin de préserver l'unité esthétique. Les sections du site ont été repensées pour favoriser une navigation fluide, y compris sur les versions mobiles grâce à une structure responsive optimisée. Le menu burger, les transitions animées et la hiérarchisation des contenus assurent une expérience agréable sur tous les supports.

2 Les problèmes rencontrés et les solutions apportées

2.1 Les Graphismes

La question de l'uniformité visuelle s'est révélée plus complexe que prévu. Nous avons dû développer un véritable cahier des charges graphique, détaillant non seulement les palettes de couleurs et les proportions, mais également les techniques d'ombrage, les styles de texture, et même les philosophies d'éclairage à respecter.

La standardisation des textures a nécessité la création d'un fichier spécifique contenant tous les matériaux de base du jeu. Cette approche nous a permis de garantir la cohérence tout en facilitant les modifications globales. Par exemple, un changement dans la texture



”bois” se répercutait automatiquement sur tous les éléments utilisant ce matériau à travers l’ensemble du jeu.

L’équilibre entre qualité visuelle et performances techniques a constitué un défi constant, particulièrement compte tenu de notre objectif d’accessibilité sur une large gamme de configurations matérielles. La réduction de résolution des textures s’est accompagnée d’innovations dans les techniques d’optimisation.

La maîtrise de Blender s’est révélée plus ardue que prévu . La complexité de ce logiciel a nécessité la mise en place de formation internes (par le biais de vidéo youtube et de forum. Les fonctionnalités avancées comme la sculpture numérique, les modificateurs de maillage, et les systèmes de particules ont demandé des semaines de pratique avant d’être maîtrisées suffisamment pour la production.

La gestion de la topologie des maillages 3D a représenté un apprentissage technique considérable. Nous avons dû comprendre les implications de chaque choix de modélisation sur les performances finales, apprenant à équilibrer détail visuel et efficacité computationnelle. Cette expertise s’est particulièrement révélée cruciale lors de la création du boss final, dont la complexité géométrique initiale causait des chutes de framerate importantes impactant la RAM.

Les problèmes de mémoire vive liés aux modèles trop détaillés nous ont forcés à repenser notre approche de la modélisation. Nous avons développé des techniques (subdivisions intégrés à Blender) permettant de réduire le nombre de polygones tout en préservant la silhouette générale des objets. Cette atout technique, acquis par nécessité, s’est finalement révélée bénéfique pour l’ensemble du projet.

L’application des couleurs face par face, contrairement à nos attentes d’application globale, a révélé les limitations de notre workflow initial. Cette découverte nous a menés à explorer des techniques alternatives comme le mapping UV et la peinture de texture directement dans Blender, compétences qui enrichiront nos futurs projets. Les incompatibilités entre logiciels lors des phases d’exportation ont nécessité la mise en place d’un protocole strict de validation et de conversion. Nous avons créé des check-lists détaillées pour chaque étape du processus, de la modélisation à l’implémentation finale dans Unity.

2.2 Le Gameplay

Dans les premières étapes du développement, avant l’intégration du mode multijoueur, nous avions déjà rencontré plusieurs problèmes techniques liés au gameplay de base, notamment au niveau du collisionneur et du Character Controller du personnage. Le collisionneur, mal calibré, provoquait des déplacements incontrôlés : le personnage était projeté dans différentes directions, rendant le jeu injouable. Pour corriger cela, nous avions dû ajuster manuellement les dimensions du collisionneur afin de retrouver un contrôle stable.

Le Character Controller, de son côté, était mal positionné verticalement, ce qui donnait l’impression que le personnage flottait au-dessus du sol. Cela engendrait des problèmes de collisions et nuisait à l’immersion. En ajustant sa hauteur pour qu’elle cor-



responde aux dimensions réelles du personnage, nous avions résolu ce souci et obtenu une interaction plus réaliste avec l'environnement.

Un autre problème initial concernait le système d'attaque. À l'origine, chaque pression sur la touche d'attaque déclenchaient immédiatement une animation et l'application des dégâts. Cela posait problème si le joueur appuyait plusieurs fois rapidement : les dégâts s'accumulaient sans respecter la fin de l'animation, créant un déséquilibre. La solution envisagée avait été d'empêcher le lancement d'une nouvelle attaque tant que l'animation en cours n'était pas terminée, assurant ainsi un comportement plus cohérent.

Lorsque nous avons introduit le mode multijoueur, de nouveaux défis sont apparus, complexifiant davantage les problèmes déjà rencontrés en solo. Cette transition nous a obligés à revoir l'ensemble de la logique du gameplay afin d'assurer une synchronisation parfaite entre tous les joueurs.

La première difficulté fut de comprendre le fonctionnement du réseau, ce qui nous a fait perdre du temps au début. La mise en place d'un système multijoueur nécessitait une bonne compréhension des échanges entre le serveur et les clients, notamment pour garantir des déplacements fluides et cohérents. Nous avons donc dû nous familiariser avec les notions de commandes, de variables synchronisées, et d'appels distants, indispensables pour mettre à jour les positions des joueurs sur tous les clients en temps réel.

Une fois les déplacements fonctionnels, nous avons constaté que les animations ne suivaient pas. Initialement, seule la position du joueur était synchronisée, ce qui donnait lieu à des comportements incohérents sur les autres clients : les personnages glissaient au lieu de marcher, ou effectuaient des actions décalées. Pour corriger cela, nous avons synchronisé l'état de l'Animator à l'aide de variables réseau, mises à jour par le serveur, ce qui a permis aux animations de s'exécuter correctement sur chaque client.

Enfin, malgré ces améliorations, un dernier problème de fluidité subsistait. Lorsqu'un joueur se déplaçait, les autres le voyaient parfois bouger par à-coups ou sembler mal ancré au sol. Cela provenait d'un mauvais réglage du Rigidbody, notamment au niveau de l'interpolation et de la gestion des vitesses. En ajustant précisément ces paramètres, nous avons réussi à assurer une transition plus douce des mouvements, et à offrir une expérience multijoueur stable et synchronisée.

Lors des dernières phases de développement, de nouveaux problèmes sont apparus, principalement liés à l'optimisation du gameplay multijoueur et à l'ajout du système de sauvegarde.

Tout d'abord, l'amélioration de la fluidité des déplacements du joueur a demandé beaucoup de temps, malgré sa simplicité apparente. Le souci venait d'une unique condition manquante, mais encore fallait-il avoir l'intuition de la chercher au bon endroit. Ce problème illustre bien la difficulté de certains bugs : ce ne sont pas toujours les plus complexes techniquement qui prennent le plus de temps, mais ceux qui nécessitent de repenser sa logique ou de changer de perspective.

Ensuite, lors de l'implémentation du système de sauvegarde, nous avons rencontré



un problème inattendu au niveau du score. En effet, après avoir chargé une partie sauvegardée, le score s'affichait correctement à l'écran avec la bonne valeur, mais lorsque le joueur ramassait une nouvelle pièce, le score ne s'incrémentait plus. Après investigation, nous avons découvert que la variable utilisée pour afficher le score était bien mise à jour, mais que la vraie variable servant à calculer les points en jeu ne l'était pas. Il a donc fallu établir un lien correct entre les données chargées et les mécanismes internes du score.

Un autre problème est survenu lors du chargement des données de sauvegarde, notamment la position du joueur. Bien que la vie et le score étaient bien restaurés, le personnage n'était pas replacé au bon endroit dans la scène. Ce décalage était dû au fait que les modifications de position s'appliquaient trop tôt, parfois avant même que le serveur ne soit lancé ou que la scène n'ait fini de charger, ce qui empêchait le joueur de se repositionner correctement.

Pour résoudre ce problème, nous avons dû introduire un délai entre le lancement du serveur, le chargement complet de la scène et l'application des données de sauvegarde. Ce temps d'attente garantit que tous les éléments nécessaires sont bien en place avant de restaurer les informations de la partie, évitant ainsi les conflits de synchronisation et assurant un chargement fiable et cohérent.

2.3 Le Game design

La programmation des pièges avancés nous a confrontés aux limites de nos connaissances en programmation. Les calculs de trajectoires pour les projectiles, les équations d'oscillation pour les mécanismes pendulaires, et les algorithmes de détection de collision en temps réel ont nécessité un approfondissement théorique considérable.

Le développement de scripts modulaires constitue notre principale innovation technique. Ce système permet de créer des "composants de piège" réutilisables qui peuvent se combiner pour générer des mécaniques complexes. Par exemple, un module "détection" peut se coupler avec un module "activation temporisée" et un module "mouvement rotatif" pour créer instantanément un nouveau type de piège.



Les conflits de version lors des phases de collaboration ont révélé les faiblesses de notre organisation initiale.

Les sessions de travail coordonnées incluaient désormais des "merges" quotidiens supervisés où chaque membre de l'équipe présente ses modifications avant intégration. Cette approche collaborative a non seulement réduit les conflits techniques, mais également amélioré la communication interne et la cohérence globale du projet.

L'optimisation des performances pour les mécaniques complexes nous a initiés aux techniques avancées de programmation efficace. Le profilage régulier des performances nous a appris à identifier les goulets d'étranglement et à prioriser les optimisations selon leur impact réel. Cette approche nous a évité les optimisations prématuées tout en nous permettant de résoudre efficacement les vrais problèmes de performance.

L'implémentation du comportement de fuite des petites tasses représentait un défi ambitieux. Les premières versions utilisaient des algorithmes simplistes qui généraient des mouvements aléatoires et peu crédibles. L'amélioration progressive nous a menés à développer un système multicouches combinant évitement de collision, recherche de chemin, et comportement de groupe.

La gestion des mouvements incohérents a nécessité l'implémentation de systèmes de lissage de trajectoire et de prédiction de mouvement. Ces algorithmes analysent les intentions de mouvement sur plusieurs frames pour générer des trajectoires fluides et naturelles, évitant les changements de direction brutaux qui brisaient l'illusion d'intelligence.

L'ajustement des vitesses et des algorithmes de fuite s'est fait par itérations successives avec des tests utilisateurs réguliers. Nous avons découvert que la vitesse optimale se situait dans une zone très étroite : trop lent, et le défi disparaissait ; trop rapide, et la frustration dominait. Cette fine-tuning a requis plusieurs heures de test et d'ajustement.

2.4 L'Intelligence artificielle

Le développement des intelligences artificielles dans The Coffee Tales a été l'un des aspects les plus complexes et exigeants du projet. Dès les premières phases, nous avons été confrontés à une série de problèmes techniques qui nous ont contraints à revoir, corriger et optimiser en profondeur l'ensemble de notre système d'ennemis. Ces difficultés ont concerné aussi bien la logique de détection, la gestion des déplacements, que la synchronisation en multijoueur.

L'un des premiers obstacles majeurs rencontrés fut le manque initial de maîtrise du langage C#, indispensable pour manipuler les scripts liés aux comportements des ennemis dans Unity. Ce manque de connaissances a nécessité un temps d'adaptation important, pendant lequel nous avons consulté de nombreux tutoriels, documentations officielles et forums. Ce travail de recherche a été essentiel pour comprendre les bases de la programmation événementielle, de la navigation IA, et de la synchronisation réseau.

Un second problème est apparu au niveau de la détection du joueur par les IA. À



l'origine, les ennemis n'étaient pas capables de repérer la présence du joueur dans leur champ de vision. Après investigation, nous avons découvert que ce dysfonctionnement provenait d'une mauvaise gestion des tags dans la scène. Les objets « Player » et « Bot » n'étaient pas correctement identifiés par le moteur de jeu, ce qui empêchait toute interaction logique. Il a donc fallu revoir entièrement la hiérarchie des objets et appliquer soigneusement les bons tags aux bons éléments. Cette correction a permis aux IA d'activer leurs comportements d'attaque ou de poursuite de manière fiable.

La fusion des différentes parties du projet a également posé de nombreuses difficultés. Pendant une période, chaque membre de l'équipe travaillait indépendamment sur sa propre scène : certains sur les ennemis, d'autres sur le joueur, ou encore sur la carte. Lors de la mise en commun, de nombreux conflits de code et d'incohérences sont apparus, en particulier concernant les noms de variables, les systèmes de collision, et la gestion des états. Certains ennemis ne fonctionnaient plus, les interactions étaient incohérentes, et il a fallu réconcilier les scripts et harmoniser l'organisation de la scène Unity pour permettre une interaction stable entre tous les éléments du jeu.

Un autre problème récurrent a été la mauvaise configuration initiale du système de navigation, pourtant essentiel pour les déplacements intelligents des ennemis. Dans la première version de la carte, aucune NavMesh Surface n'avait été définie, ce qui empêchait les IA utilisant des NavMeshAgent de se déplacer. Elles restaient figées, ou se déplaçaient de manière erratique, parfois même en tombant dans le vide. Pour y remédier, nous avons ajouté une surface de navigation adaptée à l'ensemble du terrain jouable, recalculé les chemins et ajusté les paramètres d'agent pour chaque type d'ennemi.

Par la suite, nous avons remarqué que les ennemis ne réagissaient pas correctement aux obstacles. Ils avançaient tout droit sans jamais adapter leur trajectoire, se retrouvant bloqués contre des murs invisibles ou des éléments du décor. Cela nuisait fortement à la fluidité de leur comportement, brisant l'immersion et nuisant au gameplay. Pour corriger ce défaut, nous avons implémenté une logique de détection d'obstacles : un système capable d'identifier la proximité d'un mur ou d'un objet et de forcer l'ennemi à changer de direction. Cette fonctionnalité a nettement amélioré le réalisme et la réactivité des déplacements des bots.

Enfin, l'arrivée du multijoueur via Mirror a introduit un ensemble de nouveaux défis. Il a été nécessaire de réécrire entièrement le système de spawn des ennemis, afin qu'ils apparaissent correctement sur tous les clients et soient synchronisés depuis le serveur. Pour ce faire, nous avons créé des points de spawn définis, contrôlés exclusivement par le serveur. Nous avons ensuite mis en réseau toutes les données critiques des ennemis : leur position, leurs points de vie, leur état (vivant, en attaque, ou mort), ainsi que leurs animations et leurs effets visuels. Cette synchronisation fine, réalisée via SyncVar et RPC, a été fondamentale pour garantir que tous les joueurs voient exactement la même chose au même moment.

La gestion des effets visuels, comme les projectiles ou les ondes de choc, a également nécessité une adaptation. Au départ, certains effets n'étaient visibles que pour l'hôte, ou bien leur déclenchement était désynchronisé entre les clients. Pour résoudre cela, tous les effets ont été placés sous le contrôle du serveur, et leurs déclenchements ont été gérés via



des appels réseau garantissant leur diffusion simultanée sur toutes les machines. Cette démarche a permis d'assurer une cohérence totale entre les différentes instances du jeu, et d'éviter les décalages visuels qui auraient pu nuire à l'expérience multijoueur.

Dans cette même logique, un cas particulier a posé un problème critique : celui de l'ennemi bleu, conçu pour lancer des projectiles en direction du joueur. Lors des premiers tests multijoueurs, nous avons constaté un bug majeur : les projectiles n'étaient visibles que par l'hôte, tandis que le joueur client ne voyait rien. Pourtant, ce dernier subissait bel et bien les dégâts des projectiles invisibles, ce qui rendait l'expérience injuste et frustrante. L'hôte avait l'illusion que tout fonctionnait normalement, voyant les tirs toucher la cible, mais côté client, il n'y avait aucun retour visuel. Ce décalage était causé par une instanciation incorrecte du projectile, qui n'était ni spawné sur le réseau ni propagé aux clients. Pour résoudre cela, nous avons modifié l'instanciation via des appels ServerRpc et NetworkSpawn, garantissant ainsi une apparition contrôlée par le serveur et visible par tous les joueurs. Cette correction a assuré la synchronisation parfaite entre l'effet visuel, la trajectoire du projectile, et l'application des dégâts.

Le second problème majeur concernait le boss, figure centrale du jeu. À son implémentation initiale, le boss pouvait être tué en un seul coup, ce qui allait complètement à l'encontre de sa conception. Ce dysfonctionnement venait du fait qu'il héritait du script Damage.cs utilisé pour tous les ennemis standards. Ainsi, dès qu'un projectile touchait le boss, il subissait la logique de mort classique sans tenir compte de sa spécificité. Pour corriger cela, nous avons attribué un tag spécifique au boss et développé un système indépendant de vérification de condition de mort. Désormais, le boss ne devient vulnérable que lorsque trois mini-ennemis autour de lui ont été détruits, ce qui respecte la logique prévue et évite toute destruction prématurée.

Enfin, ces trois mini-ennemis liés au boss ont, eux aussi, présenté des problèmes de synchronisation. Leur état de vie n'était pas correctement communiqué à tous les clients, et leurs déplacements apparaissaient parfois erratiques, surtout pour les joueurs non-hôtes. Ce défaut de cohérence a été corrigé en mettant en réseau leurs états et positions via SyncVar, et en réécrivant leur logique de déplacement pour garantir que leurs actions (fuite, pause, mort) soient visibles par tous les joueurs. Grâce à ces corrections, le boss peut maintenant détecter de manière fiable l'état des ennemis protecteurs, et enclencher sa propre séquence de fin lorsque les conditions sont remplies. Ces nombreuses corrections, bien que chronophages, ont été essentielles pour poser des bases solides. Elles ont surtout permis de renforcer nos compétences en débogage, en architecture logicielle, et en programmation réseau. Grâce à ces apprentissages, le système d'intelligence artificielle du jeu est aujourd'hui stable, réactif, et pleinement intégré dans l'écosystème multijoueur du projet. Les joueurs bénéficient ainsi d'une expérience cohérente, équilibrée et techniquement fiable, quel que soit leur rôle dans la session.

2.5 Le Multijoueur

L'un des principaux défis rencontrés a été la synchronisation des éléments du jeu. L'implémentation du réseau a entraîné de nombreux problèmes techniques qui ont nécessité une refonte quasi totale du projet. En intégrant Mirror, nous avons constaté que tous les



systèmes précédemment mis en place, y compris les menus, les bots et la map, ne fonctionnaient plus correctement. Le réseau a modifié la logique interne du jeu, causant des erreurs inattendues et des conflits entre les différents modules.

Les menus existants, conçus pour un environnement mono-utilisateur, ne fonctionnaient plus correctement. Les transitions entre écrans causaient des erreurs de synchronisation, et l'état des interfaces utilisateur n'était pas cohérent entre les différents clients. Mirror proposait déjà ses propres menus spécifiques pour la gestion des connexions, ce qui a rendu difficile leur modification afin de les adapter à notre interface et à nos besoins.

Les algorithmes de comportement des personnages non-joueurs entraient en conflit avec le système de réseau. Les bots apparaissaient à des positions différentes selon les joueurs, leurs décisions n'étaient pas synchronisées, créant des incohérences importantes dans le gameplay.

L'initialisation et la gestion des éléments environnementaux posaient des problèmes majeurs de synchronisation. Certains objets de la map n'apparaissaient que pour certains joueurs, d'autres se dupliquaient de manière inattendue.

Mirror utilise un système d'identifiants réseau pour les objets synchronisés, ce qui entraînait en conflit avec notre système de références directes développé pour le mode solo.

Les objets créés dynamiquement ne suivaient plus le même cycle de vie entre l'hôte et les clients, causant des erreurs de référence nulle fréquentes.

Certaines de nos structures de données personnalisées n'étaient pas compatibles avec les mécanismes de sérialisation réseau de Mirror.

La plus grande difficulté a été d'assurer une synchronisation correcte des différents objets et interactions entre les joueurs. Par exemple, certains objets apparaissaient à des positions différentes selon les joueurs, les animations étaient parfois décalées, et certains joueurs perdaient aléatoirement le contrôle de leur personnage.

Dans certaines situations, il arrivait que les joueurs se téléportent brusquement d'un endroit à un autre ou que la synchronisation entre les deux clients ne soit pas optimale. Ces téléportations inattendues créaient une expérience de jeu frustrante et brisaient l'immersion.

Les animations de marche ne correspondaient pas aux mouvements réels, créant un effet de glissement particulièrement visible et gênant. Les personnages semblaient parfois flotter ou se déplacer sans que leurs animations de pas correspondent à leur vitesse réelle.

Les objets avec lesquels les joueurs pouvaient interagir présentaient des comportements erratiques majeurs :

- Apparition à des positions complètement différentes selon les clients



- États totalement incohérents (ouvert/fermé, activé/désactivé)
- Réactions dupliquées ou manquées aux interactions des joueurs
- Objets qui disparaissaient pour certains joueurs sans raison apparente

Les événements déclenchés par les actions des joueurs ne se propageaient pas correctement, causant des situations où :

- Les actions effectuées par un joueur restaient invisibles pour les autres
- Le même événement se déclenchaît plusieurs fois de manière incontrôlée
- L'ordre d'exécution des événements était complètement incohérent entre les clients

Un autre problème majeur concernait la gestion des connexions. Lors des tests, il arrivait souvent qu'un joueur ne puisse pas rejoindre une partie en raison d'erreurs de synchronisation ou de conflits dans la configuration du réseau.

Une fois les connexions établies, nous observions une instabilité récurrente et problématique :

- Déconnexions brutales et inattendues au milieu des parties
- Impossibilité de reconnecter après une déconnexion, nécessitant un redémarrage complet

Le problème le plus frustrant et bloquant était notre incapacité initiale à faire fonctionner le réseau sur deux PC distincts. Nos tests se limitaient à des connexions sur un même appareil avec deux instances du jeu, et nous devions identifier les ajustements nécessaires pour permettre une communication stable entre plusieurs machines.

Tous nos tests se limitaient à lancer deux instances du jeu sur la même machine, ce qui ne validait que très partiellement notre implémentation. Cette limitation majeure nous empêchait de tester les véritables conditions d'utilisation du multijoueur et de valider la robustesse de notre solution.

Nous avons essayé de nombreuses approches pour résoudre ces problèmes critiques :

- Tests avec différentes versions de l'asset et différentes configurations
- Recherches approfondies dans la documentation et les forums communautaires

Malgré tous ces efforts considérables, nous n'arrivions pas à identifier la cause racine des problèmes de communication inter-machines, ce qui constituait un blocage majeur



pour le projet.

La percée est venue d'une analyse systématique et approfondie de la configuration Mirror et des paramètres réseau. Nous avons découvert que plusieurs éléments critiques bloquaient la communication inter-machines.

Nous avons donc dû revoir notre manière de gérer l'hébergement et la connexion des clients. Les ajustements dans la gestion de la synchronisation des positions ont pratiquement éliminé les téléportations inattendues. La synchronisation des animations de personnages et la stabilité du contrôle des personnages ont été grandement améliorées grâce à une meilleure gestion des états d'autorité entre l'hôte et les clients.

Au terme de ce long processus de développement, nous pouvons affirmer avec satisfaction que nous avons pleinement atteint notre objectif principal : créer un mode multijoueur fonctionnel permettant à deux joueurs de jouer ensemble depuis des PC distincts. Cette réussite représente l'aboutissement de mois de travail intensif, de recherche approfondie et de résolution méthodique de problèmes techniques complexes.

Le système multijoueur de The Coffee Tales fonctionne désormais de manière stable et fiable entre deux ordinateurs différents connectés au même réseau local. Les joueurs peuvent établir une connexion simplement en utilisant l'adresse IP, ce qui constitue une solution robuste et transparente pour notre contexte d'utilisation.



L'expérience de jeu en multijoueur atteint désormais un niveau de qualité qui approche celle du mode solo. Les joueurs peuvent se déplacer naturellement, interagir de manière intuitive avec l'environnement et accomplir les objectifs du jeu sans être significativement gênés par des problèmes techniques.

La synchronisation des mouvements et des interactions est devenue suffisamment fluide pour créer une véritable sensation de jeu partagé, où les actions de chaque joueur sont immédiatement visibles et cohérentes pour tous les participants.

Comparé à nos premières implémentations chaotiques, le nombre de bugs a été drastiquement réduit. Les problèmes critiques de téléportation, de désynchronisation d'animations et de perte de contrôle qui caractérisaient nos premiers tests ont été pratiquement éliminés.



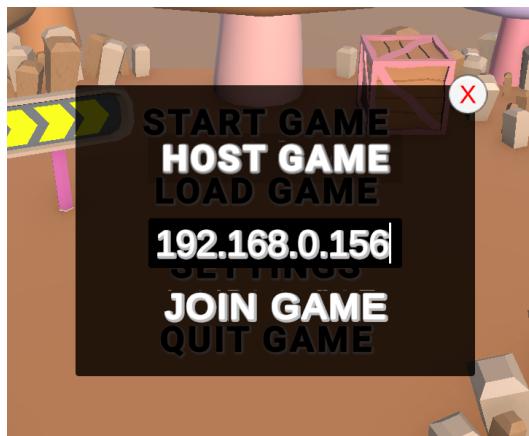
Cette amélioration spectaculaire de la stabilité transforme complètement l'expérience utilisateur, passant d'un prototype technique instable à un système de jeu véritablement utilisable et agréable.

Bien que nous observions encore quelques légers délais occasionnels, ces problèmes de latence mineurs restent dans des limites tout à fait acceptables et n'affectent pas significativement le plaisir de jeu. La plupart des joueurs ne remarquent même pas ces micro-délais lors d'une session normale de jeu.

Les performances générales du système restent stables même lors de sessions prolongées, témoignant de la robustesse de notre implémentation finale.

La méthode de connexion par adresse IP s'est révélée être la solution la plus fiable et efficace pour notre cas d'usage. Le processus de connexion est maintenant standardisé et prévisible :

- Le joueur hôte lance une partie et communique son adresse IP locale
- Le second joueur saisit cette adresse dans l'interface de connexion intuitive
- La connexion s'établit automatiquement avec validation et feedback utilisateur clair
- Les deux joueurs peuvent commencer à jouer immédiatement dans un environnement synchronisé



Le menu de connexion multijoueur a été considérablement raffiné pour offrir une expérience utilisateur professionnelle.

Notre décision de nous concentrer sur une expérience optimisée à deux joueurs s'est révélée être un choix de conception judicieux. Cette limitation volontaire nous a permis de :



- Atteindre un niveau de qualité et de stabilité élevé
- Optimiser spécifiquement l'expérience pour le cas d'usage le plus courant
- Concentrer nos ressources de développement sur la résolution complète des problèmes techniques fondamentaux
- Créer une base technique solide qui pourrait être étendue si nécessaire

L'architecture client-serveur basée sur Mirror s'est révélée être un choix technique excellent. Elle offre la flexibilité nécessaire pour notre projet tout en fournissant les outils robustes indispensables pour gérer la complexité de la synchronisation réseau.

L'implémentation réussie du mode multijoueur transforme fondamentalement The Coffee Tales d'une expérience solo en un jeu véritablement social. Les joueurs peuvent maintenant partager l'aventure, collaborer pour résoudre les défis du jeu et créer ensemble des moments de jeu mémorables.

Cette dimension sociale ajoutée enrichit considérablement la proposition de valeur du jeu et ouvre de nouvelles possibilités d'engagement pour les utilisateurs, validant pleinement l'investissement technique considérable que nous avons consacré à cette fonctionnalité.

2.6 Les Menus

Dans les premières étapes du développement, avant d'ajouter le mode multijoueur, la conception du menu en jeu avait déjà soulevé certains problèmes de gestion de la pause. Lorsque le joueur ouvrait le menu pause, le temps de jeu était correctement figé (Time.timeScale = 0) et les actions du personnage suspendues. Toutefois, un comportement inattendu survanait lorsque le joueur maintenait une touche de déplacement pendant l'ouverture du menu. En reprenant la partie, le personnage relançait automatiquement son animation de marche, bien qu'aucune touche ne soit pressée. Cette incohérence dans le gameplay nuisait à l'expérience utilisateur.

Pour corriger ce problème, nous avons désactivé manuellement l'Animator du personnage pendant la pause, puis réactivé uniquement lors de la reprise effective du jeu. Cette solution garantissait que les animations ne redémarraient que si le joueur appuyait à nouveau sur une touche, assurant ainsi une parfaite cohérence entre l'action du joueur et l'état visuel du personnage.

Avec l'introduction du mode multijoueur, la gestion du menu pause est devenue beaucoup plus complexe, notamment en termes de synchronisation entre les joueurs. Initialement, lorsque l'hôte mettait le jeu en pause, seul son jeu était suspendu, tandis que les autres clients continuaient à jouer normalement, ce qui entraînait une désynchronisation critique du gameplay.



Nous avons d'abord tenté de propager la pause à tous les clients en même temps, mais cela a généré un nouveau bug : bien que le jeu se figeait correctement sur tous les postes, le menu de pause ne s'affichait plus pour l'hôte, le privant ainsi de tout contrôle. Ce comportement bloquait la partie, car aucune option (reprise, redémarrage, retour au menu) n'était accessible.

Pour résoudre cela, nous avons totalement restructuré le système de pause. Nous avons introduit une variable synchronisée, couplée à un hook, qui permettait de notifier tous les clients d'un changement d'état de pause. Lorsqu'un joueur (en l'occurrence, l'hôte uniquement) activait la pause, une commande envoyée au serveur mettait à jour cette variable, et un RPC était ensuite déclenché pour informer tous les clients.

Une fois la synchronisation fonctionnelle, nous avons différencié l'affichage du menu en fonction du rôle du joueur :-

- L'hôte voyait un menu complet (Resume, Restart, Menu, Quit).
- Les clients, eux, n'avaient accès à aucun bouton, mais seulement à une indication visuelle "Pause", les informant que la partie était suspendue.

Enfin, pour garantir une pause totale du jeu, nous avons désactivé toutes les animations, mis le Time.timeScale à 0, et veillé à bloquer les inputs côté client jusqu'à la reprise décidée par l'hôte. Grâce à cette approche, nous avons pu garantir une pause synchronisée, contrôlée exclusivement par l'hôte, et éviter toute interaction ou animation parasite chez les clients. Cela a renforcé la cohérence du gameplay et la stabilité de l'expérience multijoueur.

Lors des dernières phases de développement, de nouveaux problèmes sont apparus, principalement liés au système d'écran de chargement et à l'ajout du menu de mort.

La difficulté survenue lors de l'ajout de notre système de chargement de scène avec écran de transition, utilisé lorsqu'un joueur héberge, rejoint ou charge une partie. Initialement, l'écran de chargement était intégré au Canvas du menu principal. Cela posait problème, car au moment du changement de scène, le Canvas du menu étant détruit, l'écran de chargement l'était également. Le joueur se retrouvait alors sans indication visuelle pendant toute la durée du chargement, ce qui créait un effet de rupture dans l'interface et pouvait laisser penser que le jeu s'était figé.

Pour corriger cela, nous avons conçu un Canvas dédié exclusivement à l'écran de chargement, activé uniquement lorsqu'un bouton de transition est cliqué. Ce Canvas est désormais marqué en DontDestroyOnLoad, ce qui lui permet de persister entre les scènes. Ainsi, l'écran de chargement reste visible même après la destruction du menu principal, et ne disparaît qu'une fois que le joueur a effectivement spawn dans la nouvelle scène.

Cette solution a permis de maintenir une expérience utilisateur fluide et lisible, tout en garantissant que l'écran de chargement couvre bien toute la durée des opérations techniques en arrière-plan (lancement du serveur, chargement de la scène, application des données de sauvegarde).



Enfin lors de l'intégration du menu de mort, celui-ci se déclenait correctement au moment du décès d'un joueur, affichant l'interface prévue et bloquant le jeu comme attendu. Cependant, un problème est apparu lorsque, dans cet état, un joueur appuyait sur le bouton pause : cela ouvrait le menu pause classique, ce qui n'était pas souhaité.

Pour corriger ce comportement, nous avons ajouté une condition spécifique dans la gestion de la pause : la fonction pause ne peut désormais être activée que lorsque le bouton pause est pressé par l'hôte et qu'aucun joueur n'est mort. Ainsi, lorsque le menu de mort est affiché, la pause est automatiquement désactivée et inaccessible, empêchant la superposition des menus et évitant toute confusion ou blocage dans l'interface.

2.7 L'Audio

Dans l'ensemble, le projet avance bien et nous ne rencontrons pas de gros problèmes techniques majeurs. Cependant, quelques petits soucis ont nécessité notre attention afin de garantir la qualité finale du jeu.

Le premier problème concerne l'enregistrement des fichiers audio. Normalement, tous les sons doivent être automatiquement sauvegardés dans un format universel appelé MP3, reconnu pour sa bonne qualité et sa compatibilité avec la plupart des appareils. Or, au début du projet, nous avons remarqué que les fichiers n'étaient pas systématiquement enregistrés dans ce format. Cela aurait pu poser des problèmes lors de l'intégration des sons dans le jeu ou sur certains supports. Nous avons donc dû modifier la procédure d'enregistrement pour nous assurer que chaque fichier respecte ce format précis, ce qui assure une meilleure uniformité et compatibilité.

Ensuite, nous avons constaté un léger décalage entre les actions du joueur et le moment où les sons correspondants se déclenchent dans le jeu. Ce décalage, appelé latence audio, est un problème fréquent dans les systèmes où l'audio doit être synchronisé avec des événements visuels. Ce phénomène perturbe l'immersion du joueur, car il entend le son un peu trop tard par rapport à ce qu'il voit ou fait. Pour corriger cela, nous avons ajusté la manière dont le son est préparé avant d'être envoyé au moteur du jeu, un processus appelé gestion du buffer audio. En optimisant cette étape, nous réduisons le délai entre l'action et le son, rendant l'expérience beaucoup plus fluide et réaliste.

Un autre défi important concernait la gestion du mélange des différents sons dans le jeu, notamment entre la musique d'ambiance et les effets sonores. Parfois, ces pistes audio se superposaient de façon maladroite, créant une cacophonie qui brouillait la compréhension et la sensation d'immersion. Pour résoudre ce problème, nous avons repensé la façon dont les pistes sonores sont organisées et diffusées. En définissant clairement des priorités et des canaux distincts pour chaque type de son, nous obtenons un rendu final plus équilibré, où chaque son trouve sa place sans masquer les autres.

Enfin, nous avons rencontré des soucis spécifiques sur certaines configurations matérielles, comme des ordinateurs ou des consoles moins courantes. Certains fichiers audio présentaient des artefacts, c'est-à-dire de petits bruits parasites ou une qualité dégradée à cause d'une compression mal adaptée. La compression permet de réduire la taille des fichiers audio



pour qu'ils occupent moins d'espace, mais si elle est mal réglée, elle altère la qualité du son. Nous avons donc ajusté ces paramètres de compression pour que les fichiers restent légers tout en conservant une excellente qualité sonore, quel que soit le matériel utilisé par le joueur.

2.8 Le Site Web

Le développement du site web de The Coffee Tales a représenté un défi à part entière, notamment en raison de notre manque initial de connaissances en développement web. Aucun membre de l'équipe ne maîtrisait à l'avance les langages nécessaires à la création d'un site fonctionnel, fluide et responsive. Il nous a donc fallu apprendre en autonomie, en nous formant progressivement sur les langages fondamentaux que sont HTML, CSS et JavaScript, chacun ayant un rôle bien spécifique dans la conception d'une page web.

Le premier défi rencontré a été la compréhension du rôle de chaque langage. En tant que débutants, nous avons dans un premier temps assimilé à tort HTML à un langage de mise en forme, alors qu'il est en réalité destiné à structurer le contenu d'une page : titres, paragraphes, images, zones de texte, etc. Grâce à l'étude de plusieurs tutoriels, notamment sur YouTube, nous avons compris que c'est CSS qui gère l'apparence visuelle (couleurs, tailles, polices, positionnement), tandis que JavaScript permet de rendre le site plus interactif, en ajoutant des animations, des effets de transition, ou encore des événements liés aux actions de l'utilisateur (comme cliquer sur un bouton ou faire défiler une galerie d'images).

Par la suite, nous avons été confrontés à un problème important d'organisation du code, notamment au niveau de l'architecture des fichiers HTML et CSS. Nous avions mal compris le fonctionnement des classes et des sections, ce qui a conduit à un désordre général : des éléments mal positionnés, du code redondant, et une structure difficile à maintenir. Ce manque de cohérence rendait les modifications longues et les résultats imprévisibles à chaque ajustement. Après plusieurs heures de tests et de débogage, nous avons fini par adopter une organisation plus logique et modulaire, avec des identifiants clairs, des classes bien nommées et un découpage cohérent entre structure (HTML), style (CSS) et comportement (JavaScript). Cette réorganisation a été une étape essentielle pour stabiliser l'ensemble du site.

Enfin, l'un des derniers problèmes majeurs que nous avons rencontrés concerne l'adaptation du site aux supports mobiles. Comme beaucoup de débutants, nous avons d'abord conçu le site pour une résolution de bureau (PC), sans anticiper les contraintes liées à l'affichage sur smartphone ou tablette. Cette approche a engendré de nombreux problèmes d'affichage sur petits écrans, avec des boutons soit trop petits, soit trop grands, des textes tronqués, et des éléments superposés. Après plusieurs recherches et tests, nous avons compris qu'il est en réalité plus simple d'adapter un site pensé dès le départ pour mobile à un écran large, que l'inverse. Nous avons donc commencé à intégrer des fonctionnalités de responsive design, notamment un menu burger et des règles CSS spécifiques pour les écrans réduits.

Cependant, faute de temps et de ressources, l'optimisation mobile n'a pour l'instant



été appliquée que sur la page principale dédiée au jeu lui-même. Cette page, qui présente les captures d'écran, les informations de gameplay et le personnage principal, bénéficie désormais d'un affichage adapté aux smartphones, avec un contenu lisible et une navigation fluide. Les autres sections du site restent encore partiellement adaptées, et leur mise à niveau est prévue pour les phases suivantes du projet.

3 Conclusion

Au terme de ce projet, nous pouvons affirmer avec satisfaction que la grande majorité de nos objectifs ont été atteints. Malgré les nombreux défis techniques et organisationnels rencontrés, nous avons su rester rigoureux, créatifs et persévérandts.

Le système réseau fonctionne désormais de manière fiable, permettant une expérience multijoueur stable et synchronisée. Les graphismes correspondent pleinement à l'univers visuel que nous souhaitions créer : colorés, attrayants et cohérents avec le ton général du jeu. Le gameplay et le game design ont été pensés et ajustés pour offrir une expérience fluide, intuitive et amusante, fidèle à notre vision initiale. Enfin, l'ambiance sonore contribue efficacement à l'immersion, avec des effets et musiques adaptés à chaque situation de jeu.

Ce projet nous a permis de développer de nombreuses compétences techniques et humaines, et nous sommes fiers du résultat final. Il marque une étape importante dans notre parcours, et pose des bases solides pour nos futurs travaux.

