

E C H O

RAPPORT DE SOUTENANCE



Par Team Nightberry

Version v1.0

13/01/2026

Par la Team Nightberry

TABLE DES MATIÈRES

1. Introduction
2. Recherches bibliographiques et description des PoC réalisés
 - 2.1. Déplacements
 - 2.2. Mécanique de jeu
 - 2.3. Objets
 - 2.4. Level Design
 - 2.5. Graphisme
 - 2.6. Intelligence Artificielle
 - 2.7. Coop en ligne / Réseau
 - 2.8. Site Web
 - 2.9. Boss
 - 2.10. Musique
 - 2.11. Lore
 - 2.12. Menu
 - 2.13. Jouabilité
3. Planning actualisé
4. Conclusion

Rapport De Soutenance Technique

Introduction

Nous sommes la Team Nightberry, un groupe de cinq étudiants en informatique : Amaury Giraud-Laforet, Gaspard Sapin, Florian Croiset, Eric Sahakian et Jules Cohen.

Notre projet se nomme Echo. Il s'agit d'un jeu de plateforme 2D de type Metroidvania, se déroulant dans un univers où le joueur évolue dans un monde principalement obscur. Le concept principal repose sur la révélation de l'environnement : Le joueur peut effectuer des actions qui génèrent une onde qui dévoile les contours du décor. Il faudra donc utiliser les capacités du personnage pour explorer et progresser dans un monde où tout est caché dans l'ombre. Avec Echo, notre objectif est de proposer une expérience immersive, mêlant exploration et réflexion, soutenue par une direction artistique inspirée de jeux comme Terraria et Dead Cells, et enrichie par une ambiance sonore originale.

Le public visé par *Echo* est composé des Joueurs amateurs de jeux d'aventure/plateforme, attachés à la progression et la difficulté maîtrisée ainsi qu'aux fans de Metroidvania et de jeux indépendants cherchant des expériences émotionnelles et immersives.

Depuis la validation du CDST, nous avons fait évoluer notre organisation et nos choix techniques afin de mieux répondre au concept central d'Écho. Les différences Les modifications apportées au CDST montrent surtout une réévaluation de nos priorités et une approche plus réaliste du développement.

Certaines tâches comme les déplacements et le menu sont restés à 100 %, car elles constituaient une base solide dès le début du projet. En revanche, les mécaniques de jeu ont été revues entre les deux versions. Nous avons volontairement réduit son avancement en soutenance 1, car les premières implémentations ne correspondaient pas totalement à notre vision du gameplay basée sur la révélation de l'environnement. Cette refonte nous permettra d'obtenir une mécanique plus cohérente et finalisée en soutenance 2.

Concernant le level design et le site web, ceux-ci ont progressé plus rapidement que prévu, notamment grâce à la stabilisation du gameplay, qui nous a permis de créer une carte plus pertinente et de mieux présenter le projet. L'IA a également demandé plus de travail que prévu, car nous devons l'adapter à un environnement sombre.

Enfin, des éléments comme le boss, la musique ou les cinématiques ont été volontairement repoussés. Nous avons choisi de nous concentrer d'abord sur la jouabilité et le cœur du jeu avant d'ajouter ces contenus plus avancés. Cette approche

progressive va nous permettre d'aboutir à un projet plus stable et mieux maîtrisé pour la soutenance finale.

Actuellement, *Echo* dispose d'un exécutable fonctionnel qui présente les bases du gameplay. Les déplacements principaux et la mécanique d'écho sont opérationnels, mais certaines capacités comme le dash ou le double saut ne sont pas encore implémentées.

La map est conçue, mais elle n'est pas encore intégrée dans le jeu. Les IA sont présentes sous forme de prototypes simples, avec des déplacements basiques. Le réseau en P2P est presque fonctionnel, mais nécessite encore des ajustements.

Le jeu ne possède pas encore de graphismes définitifs, de boss ni de musique, ces éléments ayant été volontairement mis de côté pour prioriser le développement technique. En revanche, le site web du projet est très abouti et constitue un support solide de présentation.

En résumé, le projet repose sur des fondations techniques solides, mais reste en phase d'intégration et de finalisation.

2 Recherches bibliographiques et Description des PoC réalisés

2.1 Déplacements

Responsable : Gaspard / Suppléant : Jules

Dans Echo, les déplacements du joueur constituent une base essentielle du gameplay, puisqu'ils conditionnent directement l'exploration du monde et l'utilisation des autres mécaniques du jeu. Le joueur évolue dans un environnement 2D et peut se déplacer initialement à l'aide des touches Q et D pour les mouvements horizontaux, Espace pour le saut, et S pour certaines actions. Nous avons également prévu un système de reconfiguration des touches afin d'améliorer l'accessibilité et le confort de jeu.

Objectifs initiaux

Lors de la rédaction du CDST, nous avions pour objectif de finaliser entièrement le système de déplacements dès la première soutenance, avec un avancement prévu de 100 %. Les fonctionnalités attendues comprenaient une marche fluide à gauche et à droite, un saut intégrant une gravité cohérente, ainsi qu'une gestion correcte des collisions entre le joueur et la carte. Ces éléments devaient constituer une base technique stable pour le reste du projet.

Réalisations concrètes

À ce jour, le prototype permet au joueur de se déplacer horizontalement de manière fluide grâce à un système de vitesse, ainsi que d'effectuer un saut avec une hauteur contrôlée. La détection des collisions avec le sol et les murs est fonctionnelle et repose sur une tilemap, ce qui garantit un comportement cohérent dans l'environnement. Ces fonctionnalités sont principalement implémentées dans le fichier joueur.py, qui gère les déplacements via les composantes horizontales et verticales, tandis que les constantes globales comme la vitesse du joueur et la force de saut sont définies dans parametres.py.

Choix techniques

Nous avons fait le choix d'un système de déplacement basé sur la vitesse, utilisant des constantes globales telles que la vitesse du joueur et la gravité, sans gestion de friction pour le moment. La gravité est implémentée sous la forme d'une accélération constante de 0,6 pixel par frame, ce qui offre un comportement simple et facilement ajustable. Les collisions sont gérées à l'aide d'une tilemap de 32×32 pixels, un format bien adapté au jeu de plateforme 2D et peu coûteux en termes de performances.

Difficultés rencontrées

Plusieurs difficultés ont été rencontrées lors de l'implémentation. La gestion des pentes n'a pas été intégrée, car elle s'est révélée complexe à mettre en place avec Pygame et n'était pas prioritaire pour le prototype actuel. De plus, les collisions dites « pixel-perfect » ont été envisagées, mais rapidement abandonnées en raison de leur impact négatif sur les performances. Nous avons donc opté pour des collisions rectangulaires, qui offrent un bon compromis entre précision et efficacité.

État d'avancement

Bien que les déplacements de base soient fonctionnels, le système n'est pas encore totalement finalisé. L'avancement réel est estimé à environ 85 %, contre 100 % prévu initialement. Certaines fonctionnalités importantes, comme le dash horizontal et le double saut, ne sont pas encore implémentées et ont été repoussées à la seconde phase du projet.

Prochaines étapes

Pour la suite du développement, nous prévoyons d'ajouter un dash horizontal afin d'enrichir la mobilité du joueur, ainsi qu'un double saut lié à la progression. Nous comptons également améliorer la fluidité des animations et le ressenti général des déplacements, afin de rendre l'expérience de jeu plus agréable et plus dynamique.

2.2 Mécaniques de jeux

Responsable : Gaspard / Suppléant : Éric

Dans *Echo*, les mécaniques de jeu sont au cœur de l'expérience proposée, puisqu'elles définissent la manière dont le joueur interagit avec un environnement volontairement plongé dans l'obscurité. Le concept principal repose sur l'utilisation d'ondes sonores, appelées échos, permettant de révéler temporairement les éléments du décor. Ces mécaniques conditionnent à la fois l'exploration, la compréhension de l'espace et la progression du joueur dans le monde du jeu.

Objectifs initiaux

Lors de la rédaction du CDST, nous avions prévu un avancement d'environ 40 % des mécaniques de jeu pour la première soutenance. L'objectif principal était de poser les bases du système d'échos, avec au minimum une première mécanique fonctionnelle permettant de révéler les murs à proximité du joueur. D'autres variantes d'échos, plus complexes et plus directionnelles, étaient envisagées pour les phases ultérieures du projet.

Réalisations concrètes

À ce stade du développement, l'écho de base est entièrement implémenté et fonctionnel. Cet écho permet de révéler les murs autour du joueur sur une portée de 250 pixels, en émettant 360 rayons répartis uniformément sur 360 degrés depuis la position centrale du joueur. Chaque rayon progresse jusqu'à rencontrer un obstacle, ce qui permet de dévoiler la structure des murs sans traverser les éléments solides.

Cette mécanique est implémentée dans le fichier `carte.py`, via la fonction `reveler_par_echo`, qui se charge de lancer les rayons, de vérifier les collisions avec les tuiles de la carte et de mettre à jour la carte de visibilité. Les paramètres principaux du système (portée, nombre de rayons et temps de recharge) sont définis dans le fichier `parametres.py`, ce qui facilite les ajustements et l'équilibrage.

L'écho est soumis à un cooldown de 6000 ms, empêchant son utilisation en continu et encourageant une exploration réfléchie. Cette mécanique est actuellement stable et pleinement intégrée au prototype.

Choix techniques

Nous avons opté pour un système d'écho basé sur un lancer de rayons circulaire, inspiré du principe du raycasting. Chaque rayon est calculé à partir d'un angle précis et progresse pixel par pixel jusqu'à rencontrer un mur. Ce choix permet une révélation progressive et cohérente de l'environnement, tout en restant compatible avec une carte basée sur une tilemap.

L'écho est déclenché par défaut via la touche E, avec la possibilité de la reconfigurer afin d'améliorer l'accessibilité. Le système reste volontairement simple pour cette première implémentation, avec des constantes globales clairement définies :

- `PORTEE_ECHO = 250`
- `NB_RAYONS_ECHO = 360`

- COOLDOWN_ECHO = 6000

Difficultés rencontrées

Plusieurs difficultés ont été rencontrées lors de l'implémentation. Le système de révélation par rayons est relativement coûteux en ressources, car il se rapproche d'un raytracing simplifié, ce qui a nécessité des concessions pour préserver les performances du processeur. Nous avons notamment dû limiter le nombre de rayons afin d'éviter des chutes de performances.

De plus, la gestion correcte des collisions des rayons avec les murs s'est révélée complexe. Il a fallu ajuster la largeur des murs et affiner les calculs de collision pour éviter que les rayons ne traversent les obstacles, ce qui demandait un calibrage précis entre la taille des tuiles et la progression des rayons.

État d'avancement

L'avancement réel des mécaniques de jeu est actuellement estimé à environ 60 %, ce qui est supérieur aux prévisions initiales du CDST. La mécanique principale d'écho s'est révélée plus simple à implémenter que prévu, ce qui a permis de consolider rapidement une base technique solide pour la suite du projet.

Prochaines étapes

La prochaine étape majeure consistera à implémenter l'écho précis, une variante plus directionnelle offrant une portée plus élevée mais un angle de révélation limité (environ 45 degrés). Cette mécanique viendra enrichir le gameplay en introduisant un choix stratégique entre exploration large et analyse ciblée de l'environnement, et devrait être ajoutée avant la prochaine soutenance.

2.3 Objets

Responsable : Amaury / Suppléant : Jules

Les objets dans Echo représentent l'ensemble des éléments avec lesquels le joueur peut interagir ou qui structurent l'environnement de jeu. Ils jouent un rôle central dans la construction des niveaux, la lisibilité de l'espace et l'évolution du gameplay. Lors de la première phase du projet, nous avons principalement travaillé sur la conception graphique des objets plutôt que sur leur intégration visuelle complète dans le jeu.

Objectifs initiaux

Selon le CDST, l'objectif pour la soutenance 1 était d'atteindre un avancement de 20 % sur la gestion des objets. Nous avions prévu de mettre en place les bases des objets fixes, incluant les plateformes et les éléments de décoration. Ces objets devaient permettre de structurer les niveaux et de servir de support aux déplacements du joueur et aux collisions.

Réalisations concrètes

À l'heure actuelle, une classe “AmePerdu” générique a été créée afin de définir une base pour les objets du jeu. Cette classe permet de gérer l'objet de que le perso lâche quand il meure. Cependant, aucun objet concret n'a encore été entièrement implémenté dans le livrable final.

Des éléments de décor et de plateformes en pixel art ont été réalisés, mais ils ne sont pas encore intégrés dans l'exécutable. Pour le moment, les murs et plateformes sont représentés par des rectangles sans textures, utilisés uniquement pour tester les collisions et les déplacements du joueur. Cette approche nous a permis de valider les aspects techniques avant d'intégrer les visuels définitifs.

Choix techniques

Nous avons fait le choix de concevoir une architecture orientée objet, avec une classe “AmePerdu” servant de base à tous les éléments du décor. A cette classe pourrait être ajouter des classes plus spécifiques, comme “Mur”, “Plateforme” ou “Decoration”. Ce choix permet une meilleure évolution du code et facilite l'ajout de nouveaux types d'objets sans modifier la structure existante.

Pour l'instant, les objets sont représentés par des collisions rectangulaires simples, indépendantes des textures. Cette séparation entre la logique (collisions, interactions) et l'affichage (sprites) a été volontairement choisie afin de pouvoir modifier ou améliorer les graphismes sans impacter le gameplay. Elle est également adaptée à un développement itératif basé sur des prototypes.

Difficultés rencontrées

La principale difficulté rencontrée concerne l'intégration simultanée des objets avec la tilemap, les collisions et le système de révélation par l'écho. Il a été nécessaire de réfléchir à la manière dont les objets décoratifs devaient interagir, ou non, avec le joueur, tout en restant cohérents visuellement dans un monde majoritairement obscur.

De plus, le manque de graphismes définitifs a compliqué l'intégration des objets dans le livrable, car il était difficile de valider leur lisibilité et leur utilité sans direction artistique finalisée. Nous avons donc préféré repousser leur implémentation complète afin d'éviter des refontes inutiles.

État d'avancement

L'avancement réel de cette partie est estimé entre 15, contre 20 % prévu initialement. Les bases techniques sont présentes avec la création de la classe "AmePerdu", mais les objets fixes tels que les plateformes, les murs et les éléments de décoration ne sont pas encore intégrés de manière concrète dans le jeu.

Prochaines étapes

Pour la seconde phase du projet, nous prévoyons d'implémenter les objets fixes (murs, plateformes et décor) avec leurs textures définitives. Nous souhaitons également introduire des objets non fixes, comme des clés ou des armes, qui permettront d'enrichir le gameplay.

Enfin, nous commencerons la création d'objets à états, tels que des sources de lumière pouvant être allumées ou éteintes, ainsi que des portes ouvertes ou fermées. Ces objets joueront un rôle important dans la progression du joueur et dans l'exploitation de la mécanique d'écho.

2.4 Level Design

Responsable : Eric / Suppléant : Gaspard

Le level design d'Echo a pour objectif de structurer l'exploration et la progression du joueur dans un environnement cohérent avec le genre metroidvania. Il vise à encourager l'exploration non linéaire, le retour sur ses pas et la découverte progressive de nouvelles zones à travers l'acquisition de capacités et d'objets spécifiques. Cette partie est essentielle, car elle conditionne directement le rythme du jeu, la sensation de découverte et la compréhension implicite des objectifs par le joueur.

Objectifs initiaux

D'après le CDST, l'objectif pour la première soutenance était d'atteindre environ 60 % d'avancement sur le level design. Il ne s'agissait pas d'obtenir une map totalement jouable et intégrée dans le moteur, mais de définir une structure globale cohérente, incluant l'organisation des zones, la logique de progression et les principaux points de blocage typiques d'un metroidvania.

La map devait permettre une exploration relativement libre au début, tout en intégrant des zones volontairement inaccessibles, destinées à être débloquées plus tard grâce à des capacités ou des objets spécifiques.

Réalisations concrètes

La conception de la map a été réalisée en plusieurs phases. Une première version a été dessinée sur papier afin de définir la structure générale et les grandes zones. Une seconde itération, également sur papier, a permis d'affiner cette structure, d'identifier les zones inaccessibles et de réfléchir aux objets, capacités ou événements nécessaires pour débloquer.

Ce n'est qu'après ces étapes que la map a été reproduite dans l'outil Tiled, en y intégrant directement le design visuel.

La map est conçue comme un ensemble de salles interconnectées, formant une grotte aux embranchements multiples. La structure est volontairement non linéaire, avec de nombreuses bifurcations et zones secondaires. Une partie importante de la carte est accessible sans capacité particulière, mais certaines zones, notamment en hauteur, nécessiteront l'obtention du double saut. D'autres zones seront bloquées par des portes verrouillées, nécessitant des objets spécifiques, comme une torche. Certains boss sont également pensés comme des éléments clés de la progression, permettant au joueur d'acquérir de nouvelles capacités indispensables pour accéder à de nouvelles zones.

L'exploration est conçue de manière à encourager le retour sur ses pas. Le joueur est régulièrement amené à atteindre des zones fermées ou à se retrouver dans des espaces sans issue immédiate, ce qui l'incite à revenir ultérieurement avec de nouvelles capacités. Le level design cherche également à guider le joueur de manière visuelle, tout en le désorientant volontairement à certains moments afin de renforcer l'atmosphère propre au genre metroidvania.

Choix techniques

L'utilisation de Tiled pour la conception de la map permet de travailler avec une tilemap adaptée au pixel art, tout en facilitant la gestion de grandes structures complexes. Les calques ont été pensés à l'avance afin de séparer les éléments visuels, les collisions et les éléments interactifs, même si leur exploitation dans Pygame n'a pas encore été mise en place.

Le choix de ne pas intégrer immédiatement la map dans Pygame est volontaire. La priorité a été donnée à la stabilisation du gameplay et des mécaniques principales, afin d'éviter de complexifier le débogage avec une map encore non testée. Cette approche permet de réduire les risques techniques avant la soutenance 2.

Difficultés anticipées

La principale difficulté réside dans le temps de conception d'une map de grande taille respectant les codes du metroidvania. Trouver un équilibre entre liberté d'exploration, guidage implicite et progression contrôlée s'est avéré particulièrement chronophage. Une autre difficulté importante concerne l'implémentation future de la map dans Pygame. L'absence de tests en conditions réelles rend encore incertaine la manière dont les collisions et les interactions seront gérées techniquement, malgré une anticipation au niveau des calques dans Tiled.

État d'avancement

L'avancement du level design est actuellement estimé à environ 70 %, contre 60 % prévu dans le CDST. La structure globale de la map et la logique de progression sont bien définies, mais l'absence d'intégration dans le moteur et de tests en jeu empêche encore toute validation définitive du flow et de la difficulté.

Prochaines étapes

Les prochaines étapes consisteront à intégrer la map dans Pygame, à implémenter les portes et zones bloquées, puis à effectuer des phases de test afin d'ajuster la progression et l'équilibrage. L'ajout de nouveaux biomes n'est envisagé qu'en cas de temps supplémentaire en fin de projet.

2.5 Graphisme

Responsable : Eric / Suppléant : Amaury

Le graphisme d'Echo a pour objectif de construire une identité visuelle cohérente avec le lore du jeu et de renforcer l'immersion du joueur. Il repose sur un style pixel art, adapté à un jeu 2D développé avec Pygame, tout en prenant en compte les contraintes techniques et de performance liées à ce moteur.

Objectifs initiaux

D'après le CDST, l'objectif pour la première soutenance était d'atteindre environ 20 % d'avancement sur le graphisme. Cette phase devait principalement permettre de définir la direction artistique, les palettes de couleurs et les premiers designs des entités importantes du jeu, sans viser une intégration complète ou des animations finalisées.

Réalisations concrètes

Plusieurs éléments graphiques majeurs ont été réalisés. Les designs du personnage principal, du boss final *Vex* et des ennemis secondaires ont été créés en pixel art. Ces éléments permettent de fixer l'identité visuelle du jeu et serviront de base pour les futures animations.

Le style graphique s'inspire notamment de jeux comme *Dead Cells* ou *Terraria*, tout en cherchant à développer une identité propre. La map présente un niveau de détail relativement élevé, tandis que les personnages sont volontairement plus pixelisés afin de faciliter la création d'animations ultérieures. Les sprites ont été réalisés à la main à l'aide du logiciel Aseprite. Pour certains personnages, une image de base libre de droit a été utilisée comme point de départ, puis largement retravaillée afin d'obtenir un rendu original et cohérent avec l'univers du jeu.

À ce stade, aucune animation (déplacements, sauts, combats) n'a encore été réalisée. Le HUD n'a pas non plus été pris en charge directement ; un HUD temporaire et fonctionnel a été implémenté par un autre membre de l'équipe.

Choix techniques

Le choix du pixel art s'explique par sa compatibilité avec Pygame et par son coût en performance relativement faible. Pygame n'étant pas un moteur de jeu complet comme Unity ou Unreal, le niveau de détail graphique a dû être limité afin de garantir de bonnes performances.

Le graphisme a été pensé dès le départ pour faciliter les animations futures. La réduction volontaire du nombre de pixels par sprite permettra une animation plus rapide et plus lisible. La direction artistique, les couleurs et l'ambiance ont été définies après l'écriture du lore, ce qui a permis d'assurer une cohérence globale entre narration et visuel.

Difficultés anticipées

La création de graphismes en pixel art, et en particulier l'animation de personnages, s'est révélée très chronophage. Cette contrainte de temps a conduit à prioriser la création des designs statiques au détriment des animations. Les limitations techniques de Pygame ont également influencé le niveau de détail possible, imposant des compromis entre qualité visuelle et performance.

État d'avancement

L'avancement du graphisme est actuellement estimé à environ 35 %, contre 20 % prévu initialement. La direction artistique et les principaux designs sont définis, mais l'absence d'animations et de HUD final limite encore l'intégration graphique dans le jeu.

Prochaines étapes

Les prochaines étapes incluront la réalisation des animations du personnage principal, des ennemis et du boss final, ainsi qu'une éventuelle amélioration du HUD afin de le rendre plus cohérent avec l'identité visuelle du jeu. Ces évolutions dépendront du temps restant après la stabilisation du gameplay.

2.6 Intelligence Artificielle

Responsable : Amaury / Suppléant : Eric

L'intelligence artificielle dans Echo a pour objectif de donner vie aux ennemis et de renforcer l'aspect immersif et stratégique du jeu. Elle doit permettre aux entités non-joueurs de réagir à la présence du joueur et de s'adapter à la situation. Cependant, au cours de la première phase du projet, cette partie a volontairement été placée en priorité basse afin de concentrer nos efforts sur les autres parties du gameplay.

Objectifs initiaux

D'après le CDST, l'IA devait atteindre un avancement d'environ 10 % lors de la soutenance 1. L'objectif n'était pas d'obtenir une IA complète, mais de poser les bases des comportements attendus. Trois types de comportements avaient été définis : un comportement d'errance lorsque l'ennemi n'est pas en interaction avec le joueur, un comportement d'attaque lorsque le joueur est détecté, et un comportement de fuite dans certaines conditions, notamment lorsque les points de vie de l'ennemi deviennent faibles.

Réalisations concrètes

À ce stade du projet, aucun code d'intelligence artificielle n'a encore été implémenté dans l'exécutable. En revanche, un travail de réflexion important a été mené en amont. Nous avons effectué des recherches théoriques sur les différentes approches possibles, notamment sur les systèmes d'états et le pathfinding notamment avec des vidéos sur youtube, qui sont détaillées dans la section correspondante du rapport. Une architecture générale de l'IA a également été définie sur papier, ce qui permettra une implémentation plus structurée par la suite.

Le retard observé s'explique principalement par la priorisation des déplacements du joueur et de la mécanique d'onde, qui constituent le cœur du gameplay d'Echo. De plus, la complexité du pathfinding sur une tilemap a été initialement sous-estimée, ce qui a conduit à repousser l'implémentation de l'IA.

Choix techniques

Nous avons prévu d'utiliser une machine à états finis (FSM) pour gérer les comportements des ennemis. Chaque ennemi disposera de plusieurs états, comme l'errance, l'attaque ou la fuite, avec des transitions conditionnelles en fonction de la position du joueur ou de l'état de l'ennemi. Par exemple, un ennemi en état d'errance passera en état d'attaque lorsqu'il détectera le joueur, et pourrait basculer en état de fuite si ses points de vie passent sous un certain seuil.

Pour les déplacements intelligents, nous envisageons l'utilisation de l'algorithme A* afin de permettre aux ennemis de naviguer efficacement sur la tilemap. La détection du joueur serait basée sur un rayon de vision circulaire, permettant de déterminer si le joueur se trouve à portée de perception.

Difficultés anticipées

Plusieurs difficultés ont été identifiées avant même l'implémentation. L'intégration de l'algorithme A* avec des obstacles potentiellement dynamiques représente un défi technique important. De plus, des problématiques d'optimisation sont à prévoir, notamment lorsque plusieurs ennemis seront actifs simultanément à l'écran. Il sera donc nécessaire de limiter les calculs ou de mettre en place des solutions adaptées pour garantir de bonnes performances.

État d'avancement

L'avancement réel de l'intelligence artificielle est actuellement de 0 %, contre 10 % prévu initialement. Cet écart reste toutefois acceptable, car l'IA était considérée comme une fonctionnalité de priorité basse lors de la première phase du projet. Les bases théoriques posées permettront néanmoins une meilleure avancée du projet dans le futur.

Prochaines étapes

Pour la seconde phase du projet, nous prévoyons d'implémenter un premier ennemi basique doté d'un comportement simple, reposant sur un déplacement aléatoire. Une détection du joueur par distance sera ajoutée, accompagnée d'une machine à états limitée à deux états dans un premier temps : errance et attaque. Cette approche progressive nous permettra de valider l'architecture choisie avant d'ajouter des comportements plus complexes.

2.7 Coop en ligne / Réseau

Responsable : tout le groupe

La fonctionnalité de coop en ligne occupe une place importante dans la vision initiale de Echo. Le jeu a été pensé dès le départ comme une expérience pouvant être vécue aussi bien en solo qu'en coopération en ligne, cette dernière restant optionnelle. L'objectif était de permettre à plusieurs joueurs d'évoluer simultanément dans le même monde, afin de renforcer l'aspect exploration et la dimension collaborative du jeu.

Objectifs initiaux

Dans le CDST, la coop en ligne était considérée comme une fonctionnalité majeure du projet. Il était prévu de proposer un mode multijoueur en ligne pouvant accueillir jusqu'à trois joueurs simultanément, avec un partage de la progression entre les participants.

Bien que les joueurs n'aient pas d'interactions directes entre eux (comme des compétences combinées), leur présence devait influencer l'exploration et la progression globale du jeu, notamment à travers la coopération dans les niveaux et la gestion des menaces.

Choix techniques

Pour l'implémentation du réseau, nous avons envisagé l'utilisation de Pygame couplé à un système de sockets, avec une architecture client / serveur. Un modèle peer-to-peer a également été étudié afin de comparer les différentes approches et leurs implications techniques.

Ces choix visaient à proposer une solution flexible, capable de gérer la synchronisation des joueurs et le partage de la progression, tout en restant compatible avec les contraintes du moteur et du langage utilisés.

Réalisations concrètes

Le développement de la partie réseau a été amorcé et certaines bases techniques ont été mises en place. La communication entre plusieurs instances du jeu est partiellement fonctionnelle (pour l'instant la fonctionnalité coop fonctionne en locale malgré une nécessaire modification dans le part-feu de l'hôte) permettant de valider la faisabilité du multijoueur en ligne dans Echo.

Cependant, la synchronisation complète des états de jeu, ainsi que l'intégration fluide de la coop dans l'expérience globale, ne sont pas encore finalisées. L'avancement actuel de cette fonctionnalité est estimé à environ 30 %.

Difficultés rencontrées

La principale difficulté rencontrée concerne la complexité technique du réseau, notamment la nécessité de configurer le pare-feu Windows pour autoriser les communications entre les joueurs. Cette contrainte a ralenti les phases de test et de débogage.

En revanche, aucune incompatibilité majeure n'a été rencontrée avec Pygame, et le développement du mode solo n'a pas été priorisé au détriment du multijoueur.

État d'avancement et décisions

Face aux contraintes techniques et au temps disponible, la mise en place complète de la coop en ligne a été reportée, sans pour autant être abandonnée ni réduite dans son ambition. Le mode coop reste un objectif central du projet, mais son intégration complète interviendra dans une phase ultérieure du développement.

Prochaines étapes

Les prochaines étapes consisteront à poursuivre l'implémentation du système réseau, en exploitant la fonctionnalité en ligne (allant au-delà du local) sans même devoir apporter des modifications dans le pare-feu de l'hôte). L'objectif est de parvenir à une coop en ligne pleinement fonctionnelle, respectant la vision initiale du projet et s'intégrant de manière cohérente à l'expérience de jeu proposée par Echo.

2.8 Site web

Responsable : Florian Croiset | Suppléant : Amaury Giraud-Laforet

Objectifs initiaux

Avancement prévu S1 : 30%

Fonctionnalités attendues :

- Page d'accueil présentant le projet
- Structure de base du site (navigation, footer)
- Début d'intégration de la charte graphique

Réalisations concrètes

Le site web d'Echo est actuellement pleinement fonctionnel (excepté quelques bugs mineurs). Voici les principales réalisations :

Interface et présentation

Le site propose une page d'accueil animée avec une vidéo et un style néon. La navigation est fluide grâce à un menu adaptatif. Un compte à rebours annonce l'arrivée de la bêta, l'équipe Team Nightberry est présentée, et des pages expliquent le gameplay, l'histoire et l'installation du jeu.

Fonctionnalités

Un système de téléchargement relié à Supabase permet de gérer les versions du jeu. Un dashboard administrateur facilite la gestion du projet. Une page Design regroupe les éléments visuels, l'historique des versions est affiché clairement, et une roadmap interactive montre l'avancement du développement.

Optimisation et accessibilité

Le site est entièrement compatible mobile. La musique de fond peut être activée ou désactivée, un bouton de partage avec QR Code est disponible, des notifications informent des mises à jour, et une page de feedback permet de recueillir les avis des utilisateurs.

Fichiers concernés :

index.html, admin.html, design.html, doc.html, feedback.html, links.html, roadmap.html, versions.html, test.html

Dossiers css/ (styles, responsive, mobile) et js/ (interactions, API Supabase, musique, animations)

Repository GitHub : <https://github.com/florian-croiset/jeusite/>

Choix techniques

Nous avons choisi une stack web simple et efficace basée sur HTML5, CSS3 et JavaScript, hébergée sur GitHub Pages. Ce choix permet un site facilement accessible, rapide à charger et simple à maintenir, tout en nous laissant un contrôle total sur le code. Il nous a également permis de consolider les bases du développement web sans utiliser de framework.

Pour les données dynamiques comme les versions et les téléchargements, nous avons utilisé Supabase. Cette solution offre une gestion simple des données et une intégration fluide avec le site, notamment pour le panneau d'administration. L'hébergement sur GitHub Pages permet enfin un déploiement automatique et sécurisé à chaque mise à jour du projet.

Difficultés rencontrées

Les principales difficultés ont concerné l'intégration des données dynamiques, en particulier la connexion entre le site et Supabase pour gérer les téléchargements et le

compte à rebours. La compatibilité entre navigateurs a également demandé des ajustements afin d'assurer un fonctionnement correct sur tous les supports.

Le responsive design a été un point important, notamment pour adapter l'interface aux écrans mobiles et aux interactions tactiles. Enfin, certaines fonctionnalités JavaScript ont nécessité un travail supplémentaire pour gérer les actions asynchrones, optimiser les performances et corriger des bugs liés aux animations et aux interactions.

État d'avancement

Avancement réel : 70% (prévu 30%)

Écart : +40% d'avance sur le planning initial

Prochaines étapes

Pour atteindre les 100% à la soutenance 2, plusieurs axes d'amélioration sont prévus :

Optimisation globale du site : amélioration des performances (compression des images, minification du code, lazy loading), optimisation du temps de chargement, amélioration du SEO pour un meilleur référencement

Amélioration de la version mobile : peaufiner l'ergonomie sur petits écrans, tester sur différents appareils

Ajout de nouvelles fonctionnalités : finalisation de la roadmap interactive, amélioration du bot Discord, ajout de nouvelles pages si nécessaire

Correction de bugs résiduels : tests approfondis sur différents navigateurs, correction des bugs remontés par les utilisateurs

Documentation technique : rédiger une documentation complète pour faciliter la maintenance future du site

2.9 Boss

Responsable : Gaspard / Suppléant : Jules

Dans *Echo*, les boss ont vocation à représenter des moments forts du jeu, à la fois sur le plan du gameplay et de la narration. Ils constituent des épreuves majeures venant conclure une phase importante de progression et mettre à l'épreuve la maîtrise des mécaniques principales, en particulier l'utilisation des échos dans un environnement plongé dans l'obscurité.

Objectifs initiaux

Conformément au CDST, l'objectif pour la première soutenance était un avancement de 0 % concernant les boss. Ce choix était pleinement assumé dans le planning du projet, les priorités ayant été données aux mécaniques fondamentales (déplacements, échos, exploration) avant d'aborder des contenus plus complexes et ponctuels comme les combats de boss.

Concept et intentions de gameplay

À ce stade, un seul boss est envisagé : le Capitaine Vex, qui occupe un rôle central dans le lore du jeu. Il est pensé comme le boss final du projet, compte tenu du temps de développement disponible.

Le combat contre ce boss sera principalement basé sur deux éléments clés :

- L'utilisation des échos,
- La compréhension et l'anticipation de patterns d'attaque.

Le Capitaine Vex pourra devenir invisible, rendant toute attaque directe impossible tant que sa position n'est pas révélée. Le joueur devra alors utiliser un écho pour dévoiler temporairement sa position pendant environ 6 secondes, créant une fenêtre d'opportunité pour attaquer. Le combat ne reposera pas sur des pièges environnementaux, mais uniquement sur la lecture du comportement du boss et la bonne utilisation des mécaniques existantes.

Choix techniques envisagés

Le boss sera une entité unique, qui ne réapparaîtra pas une fois vaincue. Son comportement reposera sur une IA simple, structurée autour de patterns d'attaque prédéfinis. Dans sa version actuelle prévue, le boss ne disposera pas de compétences spéciales complexes. Toutefois, si le temps le permet, l'ajout d'une capacité supplémentaire et un travail d'équilibrage plus poussé restent envisageables.

D'un point de vue technique, l'implémentation du boss s'appuiera sur les systèmes déjà existants du jeu, notamment la gestion des collisions, des déplacements et des échos, afin de limiter la complexité et d'assurer une bonne cohérence globale.

Difficultés anticipées

Les principales difficultés attendues concernent les performances, en raison de l'interaction entre l'IA du boss et le système d'échos, ainsi que la conception de patterns d'attaque lisibles et équilibrés dans un environnement sombre. La création d'une IA à la fois simple, efficace et intéressante pour le joueur constitue également un défi important pour cette partie du projet.

État d'avancement

L'état d'avancement actuel est estimé entre 0 % et 5 %. Aucun développement concret n'a encore été réalisé, mais une réflexion conceptuelle a déjà été menée sur le rôle du boss, ses mécaniques principales et son intégration dans le jeu. Cet avancement est cohérent avec les objectifs fixés pour la soutenance 1.

Prochaines étapes

Les prochaines étapes consisteront à :

- Implémenter le boss Capitaine Vex,
- Définir et coder ses patterns d'attaque,
- Mettre en place son IA,
- Puis travailler sur son design visuel.

Ce boss étant prévu comme le boss final du projet, son développement marquera une étape majeure dans l'aboutissement du jeu *Echo*.

2.10 Musique

Responsable : Florian Croiset | Suppléant : Amaury Giraud-Laforet

Objectifs initiaux

Aucune fonctionnalité musicale n'était prévue pour la soutenance 1. Le travail sur la musique devait débuter entre le semestre 1 et le semestre 2.

Réalisations concrètes

Même si la musique n'était pas planifiée à ce stade, des recherches préliminaires ont été menées. Une première ébauche de morceau a été créée afin de tester l'ambiance sonore du jeu. Des analyses de bandes sonores de jeux du même genre ont également été réalisées pour mieux comprendre les codes musicaux d'un Metroidvania. La composition musicale pourra être réalisée en interne, ce qui permet de créer des musiques originales adaptées à l'univers du jeu.

Choix techniques

Les choix techniques ne sont pas encore définitivement arrêtés. Un logiciel de composition sera sélectionné en fonction des besoins du projet. Le format audio envisagé est principalement le MP3, avec la possibilité d'utiliser d'autres formats si nécessaire. L'intégration dans le jeu se fera via les outils audio fournis par Pygame, permettant de gérer la lecture, le volume et les boucles musicales.

Difficultés rencontrées

Aucune difficulté majeure n'a été rencontrée pour le moment, la production musicale n'ayant pas encore réellement commencé.

État d'avancement

L'avancement est estimé à environ 5 %, ce qui représente une légère avance par rapport au planning initial grâce aux recherches et à la première ébauche réalisée.

Prochaines étapes

Pour la soutenance 2, l'objectif est de composer un premier morceau principal et quelques ambiances sonores. Les choix techniques seront finalisés et les premières musiques seront intégrées au jeu afin de tester leur impact sur l'ambiance et les performances.

2.11 Lore

Responsable : Jules / Suppléant : Gaspard

Dans Echo, le lore joue un rôle central dans l'identité du jeu, en donnant du sens aux actions du joueur et en renforçant l'immersion dans un univers de science-fiction sombre et oppressant. L'histoire se déroule dans un monde où la lumière naturelle a disparu, remplacée par une lumière artificielle contrôlée par une mégacorporation. Le joueur incarne Kaelen, un ancien Nettoyeur militaire, plongé dans une quête à la fois personnelle et universelle, mêlant survie, révélation et espoir.

L'objectif narratif est de proposer un récit fort sans surcharger le joueur d'informations, en intégrant l'histoire directement dans la progression du jeu et dans les éléments qu'il découvre.

Objectifs initiaux

Dans le CDST, le lore devait être présent dès la première soutenance, avec une base narrative solide de science-fiction. L'objectif était de créer un univers cohérent, suffisamment développé pour donner une direction claire au jeu, tout en laissant une certaine liberté d'interprétation au joueur. Le lore devait expliquer le contexte du monde, le rôle du protagoniste et les enjeux principaux, sans pour autant devenir envahissant ou purement narratif.

Il était également prévu que l'histoire soutienne le gameplay et les mécaniques, notamment autour de la lumière, de l'obscurité et de l'exploration.

Réalisations concrètes

À ce stade du projet, le lore de Echo est entièrement écrit et structuré autour d'une narration en plusieurs actes. L'histoire suit Kaelen, un ancien membre d'une milice privée, confronté à la mort de son fils Jonas, leader d'un groupe rebelle. Cette perte agit comme l'élément déclencheur du jeu et donne immédiatement un objectif clair au joueur.

Le monde est dominé par SOLARIS, une mégacorporation responsable de la mise en place du Voile Noir, un réseau de nanites bloquant artificiellement la lumière du soleil afin de maintenir un contrôle total sur la population. Les créatures appelées Nocturnes, attirées par la chaleur corporelle, renforcent la dimension de survie et de danger permanent.

Le lore est transmis principalement par des textes intégrés au jeu et par des séquences narratives, sans passer par des dialogues directs. Certains environnements, notamment les cavernes et zones obscures, participent également à la narration par leur mise en scène.

Choix narratifs et techniques

Nous avons fait le choix d'un univers de science-fiction sombre et cohérent, avec un protagoniste nommé et une histoire linéaire clairement définie. Contrairement à certains jeux misant sur un lore volontairement flou, Echo propose une trame narrative précise, centrée sur la vérité derrière la disparition du soleil et sur les conséquences du contrôle technologique.

La narration repose sur un équilibre entre texte et mise en situation, afin de ne pas interrompre excessivement le gameplay. Les éléments de lore sont pensés pour accompagner la progression du joueur, tout en renforçant l'ambiance et les thèmes principaux : exploration, oppression, sacrifice et renaissance.

Difficultés rencontrées

La principale difficulté a été de rendre le lore cohérent avec l'ensemble des mécaniques de jeu et du level design. Il a fallu s'assurer que chaque élément narratif ait un sens dans l'univers et qu'il justifie les environnements traversés et les actions du joueur.

Le manque initial d'idées a également conduit l'équipe à accepter et développer toute proposition de lore cohérente, afin de construire progressivement un univers solide et crédible.

État d'avancement

Bien que le lore soit entièrement écrit, son intégration complète dans le jeu n'est pas encore finalisée. L'avancement global est estimé à environ 50 %, car tous les éléments narratifs ne sont pas encore implémentés ou mis en valeur dans le gameplay et les environnements.

Prochaines étapes

Pour la suite du projet le travail portera principalement sur l'ajout de détails narratifs et d'éléments de cohérence afin d'améliorer la compréhension globale de l'univers. L'objectif est de renforcer l'impact émotionnel de l'histoire et de s'assurer que chaque composante du jeu contribue à raconter l'histoire de Echo de manière fluide et immersive.

2.12 MENU

Responsable : Florian Croiset | Suppléant : Gaspard Sapin

Objectifs initiaux

L'objectif était de réaliser un menu de jeu complet et fonctionnel dès le semestre 1. Celui-ci devait inclure un menu principal avec les actions essentielles (nouvelle partie, continuer, paramètres, quitter), un menu pause accessible en jeu et un menu paramètres proposant les options de base.

Réalisations concrètes

Un système de menu complet a été développé et intégré directement dans le jeu avec Pygame. L'interface est fluide et intuitive, avec une navigation au clavier et à la souris.

Le menu principal permet de lancer une nouvelle partie, reprendre une sauvegarde, accéder aux paramètres ou quitter le jeu. Des effets visuels simples améliorent la lisibilité et le retour utilisateur.

Un système de sauvegarde avec trois emplacements indépendants a été mis en place. Chaque slot affiche ses informations principales et une confirmation est demandée avant l'écrasement d'une sauvegarde. Les données sont enregistrées au format JSON.

Un menu pause est accessible en jeu via la touche Échap. Il permet de reprendre la partie, accéder aux paramètres ou quitter la session, tout en conservant le jeu visible en arrière-plan.

Le menu paramètres regroupe les options essentielles : changement de langue, mode plein écran et personnalisation des touches. Les modifications peuvent être appliquées ou annulées, et les textes du jeu se mettent à jour automatiquement lors d'un changement de langue.

Choix techniques

Le développement du menu repose sur Pygame, la bibliothèque imposée pour le projet, qui permet une intégration directe avec le gameplay et un contrôle total sur l'affichage.

Les sauvegardes et paramètres sont stockés en fichiers JSON, un format simple, léger et adapté à Python. Le menu est organisé autour d'un système d'états (menu principal, paramètres, jeu, pause) afin de gérer proprement les transitions.

Difficultés rencontrées

Le développement s'est globalement bien déroulé. Les principales difficultés ont concerné la gestion de la redéfinition des touches, la mise à jour immédiate des textes lors du changement de langue et l'organisation du menu paramètres pour rester lisible, notamment avec le système de défilement.

État d'avancement

L'avancement actuel est estimé à environ 95 %. Le menu est pleinement fonctionnel et les éléments restants correspondent à de futurs ajustements ou ajouts liés à l'évolution du jeu.

Prochaines étapes

Les prochaines améliorations porteront sur l'ajout éventuel de nouvelles options, l'amélioration visuelle du menu, l'optimisation des transitions et des tests utilisateurs afin d'améliorer l'expérience de navigation.

2.13 Jouabilité

Responsable : Jules Suppléants : Gaspard Sapin, Florian Croiset

La jouabilité de Echo a été pensée pour soutenir une expérience de jeu orientée vers l'exploration et la réflexion tactique. Elle vise à offrir une prise en main accessible tout en laissant au joueur la liberté d'aborder les situations à son rythme. Les actions proposées sont volontairement limitées afin de conserver une lecture claire du gameplay et de renforcer l'immersion dans l'univers du jeu.

Le joueur alterne principalement entre des phases d'exploration, de combat et d'évitement des ennemis, dans des environnements souvent plongés dans l'obscurité.

Objectifs initiaux

Dans le CDST, la jouabilité devait reposer sur un ensemble de mécaniques simples mais complémentaires : déplacements horizontaux, saut, attaque, dash, double saut et une capacité spécifique nommée Echo, permettant d'éclairer temporairement l'environnement.

Contrairement à certaines autres fonctionnalités, la jouabilité n'était pas destinée à être finalisée dès la première soutenance. L'objectif était plutôt de poser des bases solides, puis d'enrichir progressivement l'expérience au fil du développement.

Réalisations concrètes

À l'heure actuelle, plusieurs éléments fondamentaux de la jouabilité sont déjà fonctionnels. Le joueur peut se déplacer horizontalement, sauter, attaquer les ennemis et utiliser la capacité Echo, qui éclaire brièvement la zone autour du personnage et facilite l'exploration dans les environnements sombres.

Ces mécaniques permettent déjà une première approche du gameplay, basée sur la découverte des niveaux et la gestion des dangers. En revanche, certaines fonctionnalités prévues, comme le dash et le double saut, ne sont pas encore implémentées, ce qui limite pour le moment la mobilité et les possibilités tactiques du joueur.

Choix de game design

Nous avons fait le choix d'une jouabilité accessible, reposant sur l'apprentissage par l'expérimentation plutôt que sur des tutoriels explicites. Le joueur est encouragé à découvrir les mécaniques par lui-même, en observant les réactions de l'environnement et des ennemis.

La capacité Echo est au cœur de cette approche, car elle lie directement la jouabilité à l'exploration, en donnant au joueur un outil pour comprendre et anticiper les dangers dans l'obscurité, sans pour autant supprimer totalement la tension.

Difficultés rencontrées

L'une des principales difficultés rencontrées concerne le ressenti général des déplacements, notamment à cause de problèmes liés à la gestion de la gravité. Ces ajustements ont nécessité plusieurs phases de test afin d'obtenir un comportement satisfaisant et cohérent avec le reste du gameplay.

Aucune difficulté majeure n'a été rencontrée concernant le lien entre la jouabilité et le lore, l'accent étant mis avant tout sur la solidité des mécaniques de base.

État d'avancement

La jouabilité est encore en cours de développement et son avancement est actuellement estimé à environ 30 %. Les fondations sont en place, mais plusieurs mécaniques importantes prévues dans le CDST ne sont pas encore disponibles, ce qui limite la richesse de l'expérience de jeu à ce stade.

Prochaines étapes

Les prochaines étapes consisteront principalement à implémenter le dash et le double saut, afin d'enrichir la mobilité du joueur et d'ouvrir de nouvelles possibilités tactiques et de level design. Ces ajouts permettront également d'introduire une véritable montée en puissance du personnage au fil de la progression.

Une fois ces fonctionnalités intégrées, le travail se concentrera sur l'amélioration du ressenti global et la cohérence de l'ensemble des mécaniques, afin d'offrir une jouabilité plus fluide et plus engageante.

Planning actualisé

[DB] Suivi des tâches										
Restez organisé avec vos tâches, à votre façon.										
★ Toutes les tâches		Par état	Mes tâches	Liste de contrôle						
Aa	Nom de la tâche	Pers...	Etat	[DB] Objectifs	Date d'...	Priorité	Nive...	Type de tâche	Description	
	Changement pour le saut		Terminé	Déplacements	11/27/2025 →	Faible	Moyenne	Code		
	Faire les plateformes		Terminé	Objets	11/27/2025 →	Moyenne	Moyenne	Code	Du bo pixel art	
	Regarder l'implémentation du réseau du jeu	F Florian Crc	Terminé	Réseau	11/24/2025 →	Élevée	Élevé	Recherche		
	Réorganiser le code		Terminé	Réseau IA Mécanique de jeux	11/24/2025 →	Élevée	Moyenne			
	Estimer les différentes tâches de son domaine	F Florian Crc	Pas co...	Graphisme Musique Réseau	12/03/2025 →	Élevée				
	Finir le rapport de soutenance technique de 20 pages	G Gaspard Si	En cours	Mécanique de jeux Objets Déplace	01/11/2026 →	Élevée	Élevé	Ecriture		
	ajouter le dash		Pas co...	Mécanique de jeux	01/11/2026 →	Moyenne	Moyenne	Code		
	implémenter le réseaux		En cours	Réseau	01/11/2026 →	Élevée	Élevé			
	Ajouter les IA		Pas co...	IA	02/15/2026 →	Élevée	Moyenne			
	Ajouter la map		En cours	Level design	01/11/2026 →	Élevée	Faible			
	Configurer la camera		Pas co...	Déplacements	01/25/2026 →	Moyenne	Moyenne	Code		
	Modifier le ratio et l'adatation a differents ecran		Pas co...	Menu Graphisme	02/08/2026 →	Faible	Moyenne	Code		
	Optimiser le site web	F Florian Crc	En cours	Site web	11/01/2025 →	Faible	Moyenne	Code		
	Créer la map	E Eric Sahaki	Terminé	Level design	12/01/2025 →	Moyenne	Moyenne	Recherche		
	Ajouter les ennemis	G Gaspard Si	Terminé	Boss		Élevée	Moyenne	Code		
	Créer un premier exécutable	F Florian Crc	Terminé	Site web		Élevée	Faible	Code		
	Ajout des textures		En cours	Graphisme Level design	02/22/2026 →	Moyenne	Moyenne	Code		
	Ajout de l'objet de mort		En cours	Objets Graphisme	12/14/2025 →	Moyenne	Faible	Code		
	Ajout d'une barre de vie		En cours	Graphisme Mécanique de jeux	11/09/2025 →	Moyenne	Faible	Code		
	Creation de la première page du menu		Terminé	Menu	11/01/2025 →	Élevée	Moyenne	Code		
	Ajout des parametres visuel		Terminé	Menu	11/02/2025 →	Moyenne	Moyenne	Code	Plein ecran	
	Creation du disigne du premier boss		En cours	Boss Level design	12/14/2025 →	Moyenne	Faible	Recherche		
	Creation des attaque du premier boss		En cours	Mécanique de jeux Boss	01/04/2026 →	Faible	Moyenne	Code		
	Creation d'un 2eme echo		Pas co...	Mécanique de jeux	02/01/2026 →	Faible	Faible	Code	Echo plus restreint mais sur plus longue di	

Conclusion – Viabilité du projet

Pour conclure, le projet Echo repose actuellement sur des bases techniques et solides. Les éléments principaux du gameplay, notamment les déplacements et la mécanique d'écho, sont fonctionnels et reflètent bien l'idée de départ du projet. Même si plusieurs fonctionnalités importantes ne sont pas encore intégrées, comme l'IA complète, les boss ou les graphismes définitifs, les choix réalisés montrent une bonne gestion des priorités en se concentrant d'abord sur le cœur du jeu.

Les écarts par rapport au planning initial ont été réfléchis. Certains éléments ont pris du retard, mais cela a permis de renforcer la stabilité du projet et de poser une structure de code évolutive. À l'inverse, d'autres parties comme le level design, le site web ou la direction artistique ont avancé plus vite que prévu, ce qui montre une bonne organisation globale du groupe.

En conclusion, Echo est un projet viable et cohérent, qui dispose de fondations suffisantes pour aboutir à une version finale complète. Les objectifs pour la suite sont clairs, et la deuxième phase du développement sera principalement consacrée à l'intégration des fonctionnalités manquantes et à l'amélioration globale de l'expérience de jeu.

RAPPORT DE SOUTENANCE

Version v1.0

13/01/2026

AUTEUR PRINCIPAL

Team Nightberry

CONTRIBUTEURS

- Amaury Giraud--Laforêt
 - Eric Sahakian
 - Florian Croiset
 - Gaspard Sapin
 - Jules Cohen

DESCRIPTION

Rapport de soutenance technique

*Rapport de soutenance technique du groupe LYN_L2.1_G2
pour la soutenance du 13/01.*