



---

RAPPORT DE SOUTENANCE 1  
PROJET DE PROGRAMMATION - SEMESTRE 2

---

# Nyctalopia

---

Créé par :

Iñigo Aldaraborda Hoyos - Chef de projet

Hugo Meleiro

Guilhem Cros

Marin Godefroy

EPITA Toulouse

Vendredi 11 Mars 2022

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Répartition des tâches</b>	<b>2</b>
<b>3</b>	<b>Progrès réalisé</b>	<b>3</b>
3.1	I.A. - Intelligence Artificielle . . . . .	3
3.2	Audio et Effets Sonores . . . . .	3
3.3	Gameplay . . . . .	4
3.4	Communication . . . . .	8
3.5	Graphismes, Modèles et Terrain . . . . .	9
3.6	Multijoueur . . . . .	11
3.7	UI/UX - Interface . . . . .	13
3.8	Site Web . . . . .	16
<b>4</b>	<b>Objectifs pour la deuxième soutenance</b>	<b>19</b>
4.1	Sauvegardes . . . . .	19
4.2	Gameplay . . . . .	20
4.3	Map . . . . .	20
4.4	Communication . . . . .	21
<b>5</b>	<b>Planning</b>	<b>22</b>
<b>6</b>	<b>Conclusion</b>	<b>23</b>
<b>7</b>	<b>Annexe</b>	<b>24</b>

# 1 Introduction

Dans le cadre du projet de S2 à EPITA, nous sommes poussés à mettre en pratique les différentes connaissances acquises en Travaux Pratiques et en cours dans la réalisation d'un projet en groupe. Le projet de notre studio gameHUB a été la création d'un jeu d'horreur : *Nyctalopia*.

A l'arrivée de cette première soutenance, nous avons préféré nous concentrer sur l'aspect technique du jeu, le “backend”, toute la partie que le ou les joueurs ne verront pas, ce qui nous permettra ensuite d'avancer plus rapidement dans notre projet, surtout dans la partie plus visuelle.

C'est ainsi qu'on a réussi à mettre en place le code fondamental du jeu : tout ce qui concerne le déplacement du joueur et de la caméra, le multijoueur et surtout le menu principal du jeu avec ses différentes options.

Nous nous sommes souciés aussi de l'aspect de notre jeu face au public pour cette première soutenance, notamment par la mise en fonctionnement du site web *Nyctalopia* qui est déjà complètement fonctionnel. Ceci et les logos que nous avons créés autant pour le jeu que pour le studio, nous forge déjà une identité d'un point de vue externe.

Nous présentons, dans ce rapport de soutenance, tout le progrès réalisé dans notre projet dans cette première période de travail, comment les tâches ont été réparties, et les objectifs que l'on envisage d'atteindre pour la prochaine soutenance.

## 2 Répartition des tâches

Voici la répartition prévue des tâches entre les étudiants du groupe gameHUB. Celle-ci n'a pas changé depuis le cahier des charges.

Tâches	Iñigo	Hugo	Guilhem	Marin
AI - Intelligence Artificielle			◊	✓
Audio	✓			
Communication & Gameplay			✓	◊
Graphismes & Modèles	✓			
Manuels				✓
Map	◊	✓		
Multijoueur	✓	◊		
Saves			✓	
UI/UX & Réseaux		✓		
VFX		✓		
Website				✓

✓ - Responsable

◊ - Support - Suppléant

### 3 Progrès réalisé

#### 3.1 I.A. - Intelligence Artificielle

Pour l'IA, nous avons décidé de commencer à la programmer pour la deuxième soutenance car on a préféré poser les bases avant de se lancer dans le gameplay. Néanmoins nous avons quand même fait des recherches et choisi notre entité qui sera le monstre de Nyctalopia.



FIGURE 1 – *Entité*

Notre entité utilisera le pathfinding c'est-à-dire que dès lors que le joueur rentrera dans son champs d'actions il se fera attaquer. De plus, le monstre aura des chemins établis à l'avance.

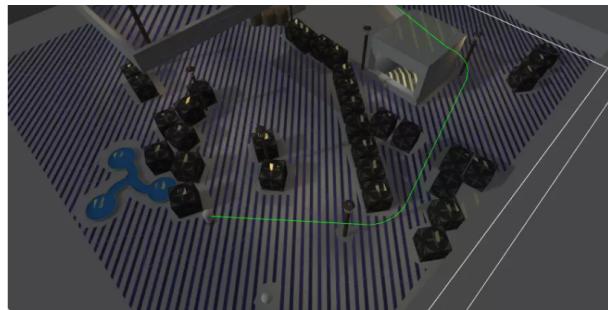


FIGURE 2 – *PathFinding*

#### 3.2 Audio et Effets Sonores

Pour l'audio, la cinématique principale et le menu d'accueil du jeu comptent déjà avec ses propres musiques, libres de droits et utilisables pour usage commercial, qui représentent bien l'aspect mystérieux et effrayant de notre jeu d'horreur.

Les effets sonores du menu d'accueil ont déjà été mis en place et les effets extradiégétiques trouvables dans le jeu ont déjà été sélectionnés (ramassage d'objets, interaction quelconque).

Nous continuons d'effectuer des recherches concernant les effets sonores liés aux différents matériaux du jeu lorsque les joueurs y marchent dessus. Cependant, les effets sonores gratuits seront très accessibles grâce aux bibliothèques d'effets sonores gratuits *Zapsplat* ou *Freesound* auxquelles on a déjà jeté un coup d'œil.

### 3.3 Gameplay

Concernant les contrôles en jeu du personnage, le début a demandé beaucoup de recherches, n'étant pas familiers avec Unity3D. Nous avons d'abord opté pour une gestion des contrôles en utilisant le système des « presskey » et nous avons implanté les déplacements, le saut, la course et s'accroupir, puis d'autres recherches ont été nécessaire pour la gestion de la caméra à la première personne. La caméra a été placée dans la tête du modèle, est gérée avec la souris et est limitée, en partant d'une vue au centre, à quatre-vingt-dix degrés vers le haut et le bas, quand la souris est bougée horizontalement, le corps du modèle tourne.

Il a donc fallu par la suite implémenter le fait qu'en maintenant la touche « avancer » ou « reculer » et tourner la caméra horizontalement, le personnage continue de marcher vers la direction de la tête. Ensuite, nous nous sommes rendu compte qu'il fallait garder le curseur au milieu de l'écran ce qui a été fait assez facilement avec quelques recherches.

Par la suite, nous avions en tête d'implémenter un système permettant de pouvoir changer les touches liées aux actions. Après des recherches sur le sujet nous nous sommes rendu compte que le système précédemment expliqué pour gérer les contrôles du personnage ne permettait pas de faire cela. Nous avons donc opté pour l'utilisation du “New Input System” de Unity3D.

Il nous a donc fallut réimplémenter les contrôles avec ce nouvel outil qui fut assez compliqué à prendre en main, comme les ressources disponibles sur internet sont moins riches que pour les précédents systèmes de gestion d'entrée. Cependant, une fois compris ce système est bien plus facile et efficace, de plus, celui-ci nous permet laisser à l'utilisateur le choix de ses touches et de pouvoir facilement changer les touches utilisées et détectées selon le statut du jeu : en jeu/dans les menus.

Ensuite nous avons choisi de supprimer l'action de course et de saut pour mieux coller à l'ambiance du jeu, le rendant plus oppressant du fait de nos mouvements limités. La gestion de l'action « s'accroupir » a aussi été revu car peu compréhensible dans le code et ne fonctionnant pas bien.

Après quelques tests, nous nous sommes aussi rendu compte que le curseur restait bloqué. Nous avons donc rajouté sur la touche « escape », sur les contrôles en jeu, le fait que le curseur redevenait libre et les contrôles passent du mode « en jeu » au mode « menu ». Concernant la partie pour laisser le joueur choisir ses touches, nous sommes partis d'un code type de la documentation Unity3D expliqué à l'aide d'une vidéo.

Nous nous sommes ensuite rendu compte que pour l'adapter à notre menu, on devait mettre plusieurs paramètres dans une fonction appelée par un bouton. Nous avons donc codé un script pour le bouton à la place d'utilisé l'inspecteur de Unity3D.

Nous avons pu coder ses scripts en nous aidant de la documentation Unity, de vidéos explicatives et de forum.

Voici le code des contrôles en jeu appelés à l'aide du “New Input System” :

```
1 void Awake()
2 {
3     playerCamera = GetComponentInChildren<Camera>();
4 }
5
6 private void Update()
7 {
8     if (controller.isGrounded)
9     {
10         movementSpeed = isCrouching ? crouchSpeed : walkSpeed;
11         finalMovement = transform.TransformDirection(inputMovement) *
12             movementSpeed * Time.deltaTime;
13     }
14
15     finalMovement.y -= gravity * Time.deltaTime;
16     controller.Move(finalMovement);
17 }
```

Listing 1 – Fonctions C# Awake / Update

La fonction “Awake” permet de récupérer la caméra du joueur lorsque l'on initialise le script. La fonction “Update” est appelée à chaque image et va être responsable du mouvement du joueur. “movementSpeed” donne la vitesse du joueur et change si celui-ci est accroupi ou non. “finalMovement” donne le mouvement final seulement au sol en fonction de la fonction “Move” qui sera traitée ci-dessous. Enfin, dans tout les cas, on va appliquer un mouvement négatif dans l'axe Y qui simule la gravité. “controller.Move()” applique les mouvements précédemment définis au joueur.

```

1 public void Move(InputAction.CallbackContext ctx)
2 {
3     var inputValue = ctx.ReadValue<Vector2>();
4     inputMovement = new Vector3(inputValue.x, 0f, inputValue.y);
5 }

```

Listing 2 – Fonction C# Move

La fonction “Move”, va lire en entrée les touches responsable du déplacement, de base “ZQSD”, et va les assigner à la variable “inputvalue” qui est utilisée dans la fonction “Update” vue précédemment.

```

1 public void Crouch(InputAction.CallbackContext ctx)
2 {
3     if (!ctx.performed) {return; }
4     if (isCrouching)
5     {
6         if (isCrouching && Physics.Raycast(playerCamera.transform.position,
7             Vector3.up, standingHeight - crouchingHeight))
8             // Permet de savoir si un objet se trouve au dessus du joueur quand
9             celui-ci est accroupi
10            {
11                //Debug.Log("Error");
12                return;
13            }
14            else
15            {
16                //Debug.Log("Standing");
17                isCrouching = false;
18                controller.height = standingHeight;
19            }
20        }
21        else
22        {
23            //Debug.Log("Crouching");
24            isCrouching = true;
25            controller.height = crouchingHeight;
26        }
27    }

```

Listing 3 – Fonction C# Crouch

La fonction “Crouch” lit l’entrée de la touche responsable de l’action s’accroupir, par défaut “Shift”. Elle va baisser le joueur et son masque de collision jusqu’à une valeur donnée correspondant à l’animation du modèle et va changer la vitesse. Lorsque le joueur ré-appuie sur la touche la fonction va vérifier si un objet se trouve au dessus de lui, s’il n’y en a pas, la hauteur est remise à la normale.

```

1 public void Escape(InputAction.CallbackContext ctx)
2 {
3     if (!ctx.performed) {return; }
4     Cursor.lockState = CursorLockMode.None;
5     PlayerInput.actions.FindActionMap("Gameplay").Disable();
6     PlayerInput.actions.FindActionMap("Menu").Enable();
7     CameraInput.actions.FindActionMap("Gameplay").Disable();
8     CameraInput.actions.FindActionMap("Menu").Enable();
9 }
```

Listing 4 – Fonction C# Escape

La fonction “Escape” permet de libérer le curseur et change les contrôles de “en jeu” à “menu”. Voici le script permettant de diriger la caméra :

```

1 void Start()
2 {
3     Cursor.lockState = CursorLockMode.Locked; //Garde le curseur au
        centre de l'écran
4 }
5
6 void Update()
7 {
8
9     float mouseX = looking.x * mouseSensitivity * Time.deltaTime;
10    float mouseY = looking.y * mouseSensitivity * Time.deltaTime;
11
12    xRotation -= mouseY;
13    xRotation = Math.Clamp(xRotation, -90f, 90f); //Limite la rotation
        de la caméra
14
15    transform.localRotation = Quaternion.Euler(xRotation, 0, 0); //
        Tourne la caméra
16    playerBody.Rotate(Vector3.up * mouseX); //Tourne le corps
17 }
18
19 public void Look(InputAction.CallbackContext ctx)
20 {
21     looking = ctx.ReadValue<Vector2>();
22 }
```

Listing 5 – Fonction C# Caméra

La fonction “Start” permet de bloquer le curseur au milieu de l’écran avant d’enregistrer les entrée souris. La fonction “Update” va récupérer les entrées lues par la fonction “Look” et appliquer ces changement à la caméra qui est bloquée comme évoqué précédemment. “playerBody.Rotate” permet de pouvoir maintenir la touche “avancer” et de suivre la direction pointée par la caméra.

### 3.4 Communication

Pour le logo du studio, nous avons opté pour un logo s’inspirant du logo de la plateforme GitHub, en le stylisant pour le faire correspondre au thème de notre jeu. En effet, on peut y voir la silhouette d’une chouette devant une pleine lune. Nous avons fait des tests avec plusieurs silhouettes d’animaux notamment des corbeaux et des chouettes, qui représente le thème de la nuit.

Finalement nous avons opté pour la chouette qui rendait le mieux car occupe un assez grand espace dans le cercle, est assez ronde et est entièrement visible. Concernant le logo du jeu, nous avons opté pour un style simple en stylisant la lettre « N », première lettre du nom du jeu.



FIGURE 3 – Logo *Nyctalopia*

### 3.5 Graphismes, Modèles et Terrain

Nous avons décidé de générer un terrain de manière procédurale à l'aide d'une bibliothèque de génération de terrain et ensuite le modifier selon notre convenance pour obtenir ce côté réaliste d'une forêt en montagne.

Une grande variété de modèles 3D d'arbres, arbustes et herbe ont déjà été sélectionnés et utilisés dans le jeu pour couvrir ce terrain et créer une forêt riche et variée. La plupart de ces modèles de végétation proviennent de la bibliothèque de modèles 3D *Quixel's Megascans* et du site officiel de modèles Unity *Unity Asset Store*. Plusieurs modifications et conversions ont été nécessaires pour les rendre compatibles avec notre projet qui est basé sur le système de rendu HDRP (*High-Definition Render Pipeline*) qui offre des résultats graphiques plus attractifs.

L'histoire du jeu se basera sous forme de chapitres, pour l'instant, nous n'avons réalisé qu'une cinématique : Chapitre 0 : Dark Night.

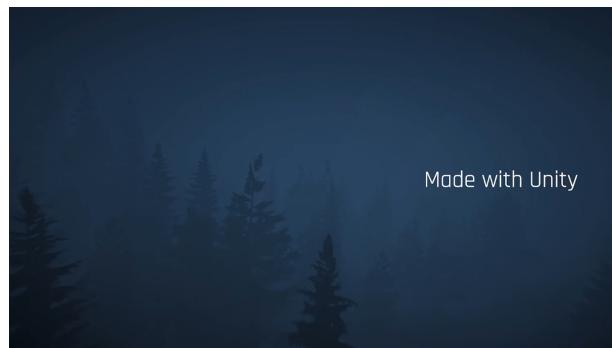


FIGURE 4 – *Chapitre 0 : Dark Night*

Nous aurons ensuite une partie qui se déroulera dans la forêt (Chapitres 1 et 2), puis un troisième dans des souterrains.



FIGURE 5 – *Représentation du Chapitre 3*



FIGURE 6 – *Représentation du Chapitre 3*

Les modèles plus spécifiques telles que la route, les différentes voitures, le bâtiment et le lampadaire du lobby multijoueur proviennent des mêmes bibliothèques.



FIGURE 7 – Voiture



FIGURE 8 – Scène de l'accident

Concernant les personnages jouables, nous comptons actuellement avec deux modèles de personnages complètement animés, un homme et une femme, provenant de la bibliothèque d'Adobe *Mixamo*.



FIGURE 9 – Personnage masculin



FIGURE 10 – Personnage féminin

Pour l'entité qui suivra le joueur plus tard dans le jeu, un modèle a déjà été choisi.  
(c.f. I.A. - Intelligence Artificielle)

## 3.6 Multijoueur

Pour le multijoueur nous avons décidé d'utiliser le SDK Steamworks fourni par Steam Inc. N'étant pas nativement compatible avec C#, nous avons dû utiliser une bibliothèque tierce nommée *Steamworks.NET*. Ce SDK permettra de créer des lobbys et d'intégrer une liste d'amis, qui facilitera la communication entre les deux joueurs étant donné que Steam est la plateforme de vente de jeux vidéos la plus populaire au monde avec 100+ millions de connexions mensuelles. Également, nous avons besoin :

- D'un HLAPI, *High Level Application Programming Interface* ou, interface permettant au jeu de communiquer avec les serveurs Steam. Celle-ci sera Mirror, une bibliothèque Unity open-source censé remplacer officieusement UNet.
- D'un LLAPI, *Low Level Application Programming Interface* ou interface permettant au LLAPI de communiquer avec le HLAPI. Celui-ci sera FizzySteamworks, également Open Source et disponible sur GitHub.

Avec tout ces outils en place, il nous a suffi de lire la documentation officielle de Steam, et de mettre en place un script consistant à créer et à joindre des lobbys grâce aux boutons présents au sein de l'interface (c.f. UI/UX).

```
1 void Start()
2 {
3     // START HOST thanks to the UIButton StartHost
4     StartHost?.onClick.AddListener(() =>
5     {
6         CreateNewLobby(ELobbyType.k_ELobbyTypeFriendsOnly);
7     });
8
9     // START CLIENT thanks to the UIButton StartClient
10    StartClient?.onClick.AddListener(() =>
11    {
12        JoinLobby(new CSteamID(System.Convert.ToInt64(
13            LobbyCSteamIDInput.text)));
14    });
}
```

Listing 6 – Fonction C# UILinker permettant de créer et rejoindre un lobby



FIGURE 11 – Structure d'un HAPI

### 3.7 UI/UX - Interface

L'interface utilisateur possède trois parties principales, le menu principal, le menu “Play” et le menu “Paramètres”. Ce dernier permet à l'utilisateur de pouvoir régler la résolution, la taille de la fenêtre (fenêtré, borderless ou plein écran), mais également le son et le choix des touches (AZERTY, QWERTY ou n'importe quelle combinaison de touches). Le menu “Play” possède trois boutons, un placé à gauche, permettant au joueur 1 de créer un lobby et de le rejoindre, et à droite deux boutons permettent au joueur 2 de soit, joindre le lobby avec un *CSteamID* (code de lobby délivré par Steam) ou bien en utilisant sa liste d'ami Steam. Enfin, le menu principal possède deux grands boutons appelant le joueur à soit débuter une nouvelle campagne ou bien de continuer là où il avait sauvegardé pour la dernière fois (c.f. Sauvegardes)

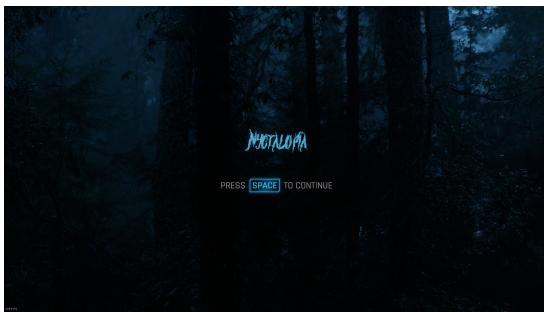


FIGURE 12 – Écran d'accueil

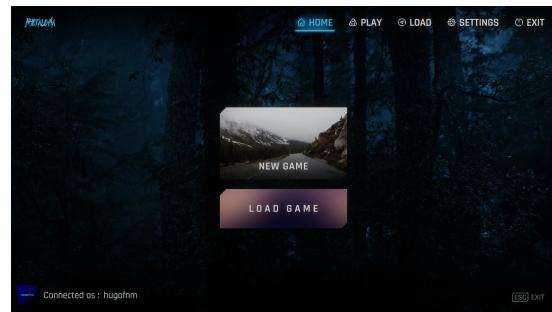


FIGURE 13 – Menu principal

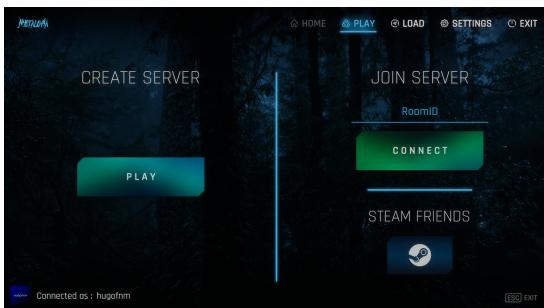


FIGURE 14 – Menu “Play”

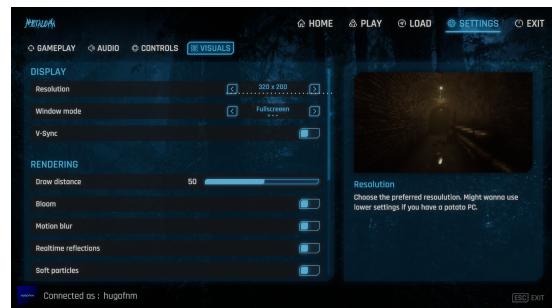


FIGURE 15 – Menu “Paramètres”

Nous avons également mis en place un installateur graphique “.exe” qui est accessible à tout utilisateur possédant une machine Windows 10 ou 11 64 Bits sur le site *nyctalopia.games* ou bien grâce à un lien direct avec une redirection HTTP 301 vers des artifacts CI/CD : *get.nyctalopia.games*. Nous comptons également à mettre en place une version Linux, comme le stipule le sujet, qui sera disponible sur *linux.get.nyctalopia.games* qui renvoie aussi sur un téléchargement direct depuis le CI/CD GitLab Nyctalopia *gitlab.nyctalopia.games* (c.f Site Web).



FIGURE 16 – *Installateur Graphique .exe*

Enfin, un script permettant d'informer ses amis que l'on joue à Nyctalopia sur la plateforme Discord a été créé.

```
1 private void Start()
2 {
3     discord = new Discord.Discord(938129729133899879, (UInt64)Discord.
4 CreateFlags.Default);
5     activityManager = discord.GetActivityManager();
6     activityManager.RegisterSteam(480);
7     UpdatePresence();
}
```

Listing 7 – Fonction C# permettant de créer une activité Discord

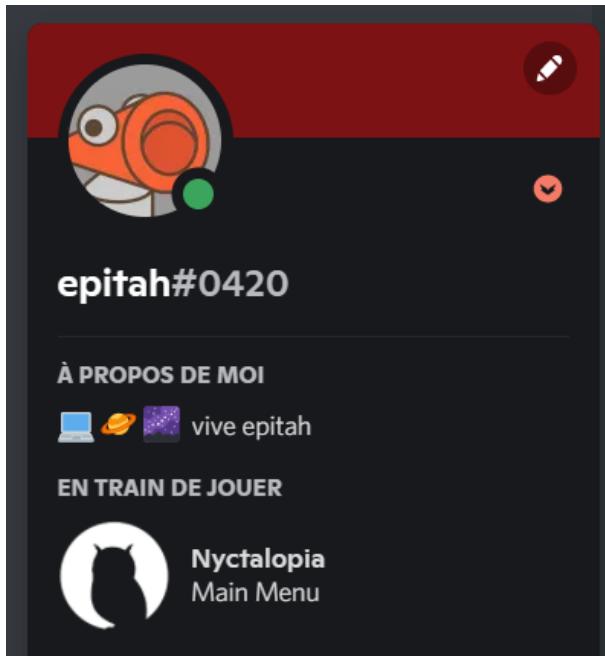


FIGURE 17 – *Discord Rich Presence*

## 3.8 Site Web

Notre objectif pour cette première soutenance était d'avoir la structure du site web prête. Nous avons essayé de rendre le site attractif. On peut le voir avec les effets Parallax ou encore avec le curseur qui suit la souris. Pour ces effets on a fait le choix d'utiliser des scripts JS fournis par Internet pour se rapprocher le plus possible d'un vrai site professionnel. Le code HTML est très bien organisé pour nous permettre d'améliorer ou de modifier le site à tout moment.

Nous avons choisi un fond sombre pour tout de suite mettre le joueur dans l'ambiance du jeu où il sera dans l'obscurité. Enfin le site est adapté pour tout type d'écrans ce qui permet au joueur de nous consulter n'importe quand. A travers ce site nous voulons mettre en confiance l'utilisateur et lui donner envie de jouer en montrant des informations intéressantes sur Nyctalopia et en montrant notre professionnalisme.

En terme de contenu sur le site on peut retrouver :

- Une page d'accueil : forêt animée



FIGURE 18 – Accueil du site

- Image du jeu : tels que la foret ou encore le lobby mais également un scène d'introduction

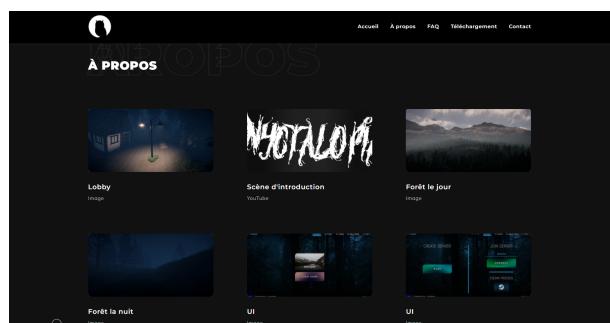


FIGURE 19 – Image du jeu

- Documents : Cahier des charges, rapport de soutenance
- Liens de téléchargements et Configurations : minimale et recommandée
- Contact : Formulaire + e-mail du studio

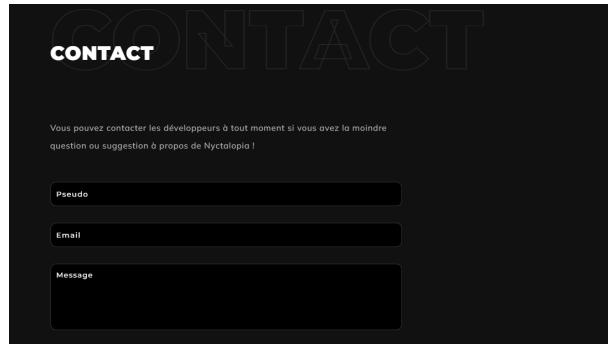


FIGURE 20 – *Contact*

- Équipe : rôles de chaque membre

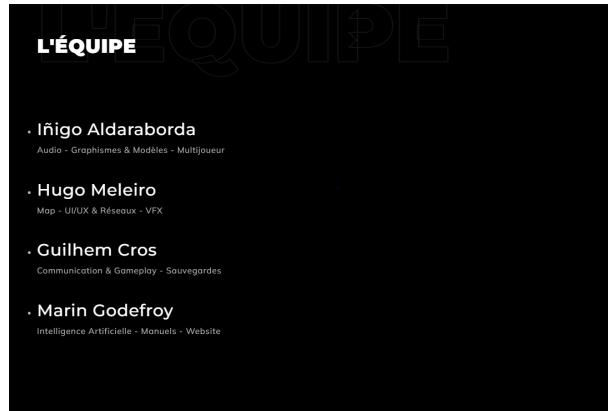


FIGURE 21 – *Membres du studio GameHub*

- FAQ : quelques réponses à des questions courantes.

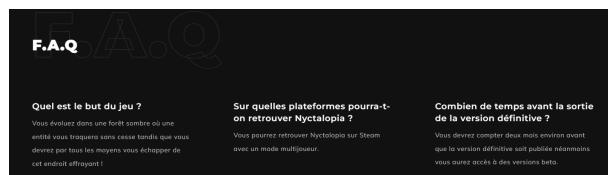


FIGURE 22 – *F.A.Q*



FIGURE 23 – Site sécurisé SSL

Le site web est hébergé sur CloudFlare Pages, un service gratuit de l'entreprise de gestion DNS CloudFlare, permettant de déployer rapidement à partir d'un repo Git, un site web statique rapide grâce aux serveurs CDN disposés partout dans le monde et sécurisé par un certificat SSL et une protection anti-DDOS. Le nom de domaine choisi est *nyctalopia.games*. Il est simple et facile à retenir pour l'utilisateur.

Gestion DNS pour nyctalopia.games						
Type	Nom	Contenu	État du proxy	Durée TTL	Actions	
A	gitlab	[REDACTED]	DNS uniquement	Automatique	<a href="#">Modifier</a>	
CNAME	get	gitlab.nyctalopia.games	Proxied	Automatique	<a href="#">Modifier</a>	
CNAME	linux.get	gitlab.nyctalopia.games	Proxied	Automatique	<a href="#">Modifier</a>	
❶ CNAME	nyctalopia.games	nyctalopia.pages.dev	Proxied	Automatique	<a href="#">Modifier</a>	

FIGURE 24 – Interface CloudFlare

Nous avons également décidé de mettre en place un GitLab (*gitlab.nyctalopia.games*) privé sur un serveur nous appartenant, ce qui nous a permis d'utiliser la puissance brute de Git LFS et des pipelines CI/CD, améliorant notre productivité sans perdre de temps.

## 4 Objectifs pour la deuxième soutenance

Voici la répartition prévue des tâches entre les étudiants du groupe gameHUB pour la deuxième soutenance.

### 4.1 Sauvegardes

Concernant les sauvegardes, nous avons décidé de placer des bornes de sauvegarde, représentées par une ancienne radio, qui seront placées à des endroits stratégiques du jeu. Cette mécanique permet de simplifier le système de sauvegarde car ne demande pas de connaître l'emplacement du joueur et rendra le jeu plus stressant, car empêche le joueur de sauvegarder sa progression à tout instant.



FIGURE 25 – Borne de sauvegarde

Concernant les contrôles en jeu, nous devrons aussi implémenter une touche d'action pour interagir avec les objets du jeu, tel que des portes, des bornes de sauvegarde ou des objets en notre possession nécessaire à l'avancée du jeu. Ceci nous demandera aussi de définir la portée d'action du personnage. L'ajout de l'apparition du menu lors de l'appui sur la touche "escape" doit aussi être implémenté.

Le script, servant à redéfinir les touches de contrôles, n'est pas encore terminé, certains cas ne sont pas pris en compte, notamment le fait d'empêcher le joueur d'assigner la même touche à plusieurs actions. Nous allons aussi devoir, par la suite, adapter ce script pour l'ajouter au menu en jeu. Le script peut déjà être adapté pour la touche « utiliser », qui sera implémentée plus tard. Une sauvegarde des touches redéfinies doit aussi être implémentée.

## 4.2 Gameplay

Concernant le gameplay, nous avons déjà commencé à réfléchir à certaines énigmes telles que la récupération de fusibles pour ouvrir des portes débloquant ainsi la suite de l'histoire.



FIGURE 26 – Boîte à fusibles

## 4.3 Map

Nous envisageons aussi l'introduction d'une nouvelle zone de jeu : les égouts. Cette zone sera réalisée à l'aide de bibliothèques 3D dédiées à la conception de chambres et couloirs modulaires.



FIGURE 27 – *Aperçu des égouts*

#### 4.4 Communication

Nous avons prévu de changer la police d'écriture temporaire du logo du jeu Nyctalopia (actuellement : “Venom”) vers la police “October Crow”.

NYCTALOPIA

FIGURE 28 – *Police d'écriture choisie pour Nyctalopia*

## 5 Planning

Voici le planning (modifié par rapport au cahier de charges) d'avancement des tâches par période (temps séparant deux soutenances). Les deux prochaines périodes sont réparties ainsi :

- - Soutenance 2 : du 25 au 29 avril 2022
- - Soutenance 3 : du 6 au 17 juin 2022

Tâches	Soutenance 1	Soutenance 2	Soutenance 3
AI - Intelligence Artificielle	10%	50%	90%
Audio	50%	80%	100%
Communication & Gameplay	30%	50%	100%
Graphismes & Modèles	50%	70%	90%
Manuels d'instruction	0%	75%	100%
Map	15%	50%	90%
Multijoueur	80%	90%	100%
Saves	5%	50%	100%
UI/UX	80%	90%	100%
VFX	10%	50%	90%
Website	90%	100%	100%

Code couleur :

- **Vert** : En avance
- **Rouge** : En retard

## 6 Conclusion

Le développement de ce projet représente pour nous une grande première. Certains d'entre nous ont déjà un antécédent de programmation, mais cela semble dérisoire face à tous les besoins de ce projet. Beaucoup de temps et de travail a été et sera nécessaire, mais l'objectif est important. En effet, il est inévitable que nous sortirons tous beaucoup plus expérimentés que nous ne le sommes maintenant, ce qui est le but recherché. Nous espérons donc mettre à bien toutes les idées proposées dans ce document et que le résultat de ce projet soit de bonne qualité. Pour nous, avoir un résultat intéressant sera un accomplissement personnel car le développement sera une épreuve mais donnera un jeu amusant auquel nous serons, en tant que joueurs, satisfaits en y jouant.

Enfin, l'esprit de groupe sera également un point important de l'apprentissage c'est à dire savoir s'entraider, communiquer, s'organiser et planifier nos objectifs. Ce sont des points que l'on attend d'un ingénieur et qui nous seront demandés durant nos carrières professionnelles. Il est donc évident que la qualité de ce projet sera un reflet de nos capacités à évoluer au sein d'un groupe pour aller vers un même but.

## 7 Annexe

**Nyctalopia** : Traduction anglaise du mot héméralopie, qui est la cécité nocturne, ou l'incapacité à bien voir dans un éclairage sombre.

**CI/CD** : *Continuous Integration & Continuous Deployment*. En génie logiciel, CI/CD est la combinaison des pratiques d'intégration continue et de livraison continue ou de déploiement continu. Le CI/CD comble le fossé entre les activités et les équipes de développement et d'exploitation en imposant l'automatisation de la création, des tests et du déploiement des applications

**Survival horror** : Le survival horror est un genre de jeu vidéo, sous-genre du jeu d'action-aventure, inspiré des fictions d'horreur. Bien que des aspects de combats puissent être présents dans ce type de jeu, le gameplay fait généralement en sorte que le joueur ne se sente pas aussi puissant qu'il ne le serait typiquement dans un jeu d'action, et ce en limitant par exemple la quantité de munitions, d'énergie ou de vitesse. Le joueur doit parfois chercher certains objets pour avoir accès à un passage vers une nouvelle zone, et résoudre des énigmes à certains moments. Les jeux utilisent des thèmes d'horreur, et le joueur est souvent confronté à des environnements obscurs et à des ennemis qui peuvent surgir de nulle part.

**Jump scare** : Un jump scare est un principe qui recourt à un changement brutal intégré dans une image, une vidéo ou une application pour effrayer brutalement le spectateur ou utilisateur. Ce principe s'est développé dans les années 1990 notamment au cinéma.

**UI - Interface Utilisateur** : L'interface utilisateur est un dispositif matériel ou logiciel qui permet à un usager d'interagir avec un produit informatique. C'est une interface informatique qui coordonne les interactions homme-machine, en permettant à l'usager humain de contrôler le produit et d'échanger des informations avec le produit.

**IP** : Une adresse IP est un numéro d'identification qui est attribué de façon permanente ou provisoire à chaque périphérique relié à un réseau informatique qui utilise l'Internet Protocol. L'adresse IP est à la base du système d'acheminement des paquets de données sur Internet. Il en existe deux versions : IPv4 et IPv6.

**DNS** : Le Domain Name System, généralement abrégé DNS, qu'on peut traduire en « système de noms de domaine », est le service informatique distribué utilisé pour traduire les noms de domaine Internet en adresse IP ou autres enregistrements.

Source : Wikipédia



Nyctalopia - gameHUB Studios

2022

EPITA Toulouse