Rapport de soutenance n°1

Mars 2023

"STUCK"

Akomi ZEBILA
Salim CHARIKH
Alexis GALOPIN

Groupe : **AVAC#**

EPITA Promo 2027

Table des matières

1 Introduction

Notre projet du S2 s'intitule *Stuck*. Il s'agit d'un survival horror créé à l'aide d'*Unity*. Dans ce jeu, le joueur incarne un personnage qui se réveille dans un manoir sinistre et lugubre. Sa mission est de capturer des rushs vidéo de monstres qui hantent les lieux tout en essayant de survivre. Le joueur dispose d'une caméra et doit filmer un maximum de séquences des monstres. Mais attention, ils sont imprévisibles et peuvent surgir à tout moment pour attaquer le joueur, d'autant plus que la caméra se décharge au fur-et-à-mesure de la partie.

Le mode solo offre une expérience immersive et terrifiante, mais nous avons également ajouté un mode multijoueurs pour ceux qui préfèrent jouer en coopération avec leurs amis. Dans ce mode, les joueurs doivent travailler ensemble pour survivre aux attaques des monstres et capturer le plus de rushs vidéo possible.

Au début du semestre, Vincent Carret a malheureusement quitté l'établissement et donc le groupe ; il était chargé principalement de la modélisation des personnages, de la réalisation du trailer et du site web, du gameplay ainsi que des animations. Nous avons donc dû revoir la répartition des tâches *(cf. second cahier des charges)*.

2 Conception du projet

Pour cette première soutenance nous avons essayé de nous concentrer sur les tâches respectives de départ en essayant tant bien que mal de compenser le manque de membres. Akomi s'est ainsi attelé à la modélisation du manoir, Salim s'est occupé d'une partie du gameplay mais surtout de l'IA ainsi qu'un peu de modélisation personnage. Quant à Alexis, il a implémenté une bonne partie du gameplay de base ainsi que du réseau, certains menus et le début du site web.

Pour faciliter (ou pas) l'échange de nos progressions respectives, nous avons fait appel à Git et nous nous sommes organisés autour d'un dépôt distant hébergé sur GitHub. Nous avons également créé un serveur Discord afin de mieux organiser les différentes tâches en salons, mais également pour pouvoir inviter des connaissances afin qu'elles puissent tester notre jeu et nous donner des avis extérieurs dessus.

2.1 IA

L'intelligence artificielle correspond à la gestion du comportement des monstres. Elle joue un rôle crucial dans le gameplay de notre jeu. En effet, l'IA des monstres sont programmées pour suivre le joueur, créant une expérience immersive.

De plus, une fonctionnalité a été ajoutée à l'IA, celle de suivre le sifflement du joueur ce qui peut être utile pour faire tomber les monstres dans des pièges et de capturer des moments de rush. Grâce au Navmesh, on peut sélectionner toutes les parties du jeu où notre IA pourra se déplacer, en ajoutant aux monstre un composant Navmesh agent, cela permettra de donner à notre IA une "forme invisible" pour qu'elle puisse se déplacer dans la map et interagir avec les éléments de cette dernière. Il y a aussi le script MonsterConstroller, qui permet de détecter notre joueur et le suivre, ceci dit, il y a plusieurs conditions qui empêchent par exemple l'ennemi de voir à travers les murs, ou bien de suivre le joueur après avoir entendu le bruit d'un sifflement, etc...

L'IA des monstres peut aussi attaquer le joueur, créant ainsi le sentiment de peur et de tension typique des survival horrors. Cependant, le monstre peut être ralenti pa le joueur.

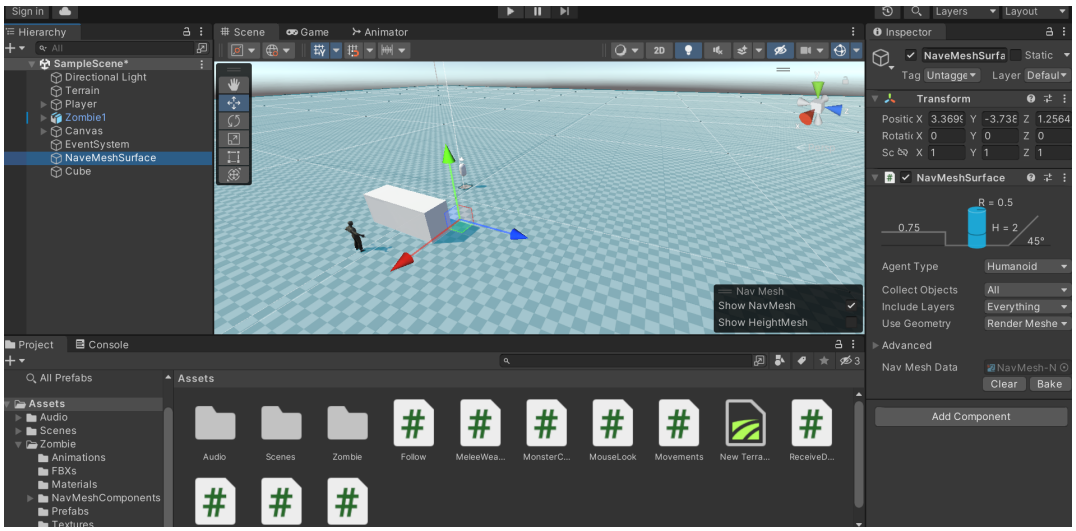

FIGURE 1 - La zone en bleu est la seule où notre IA peut marcher.

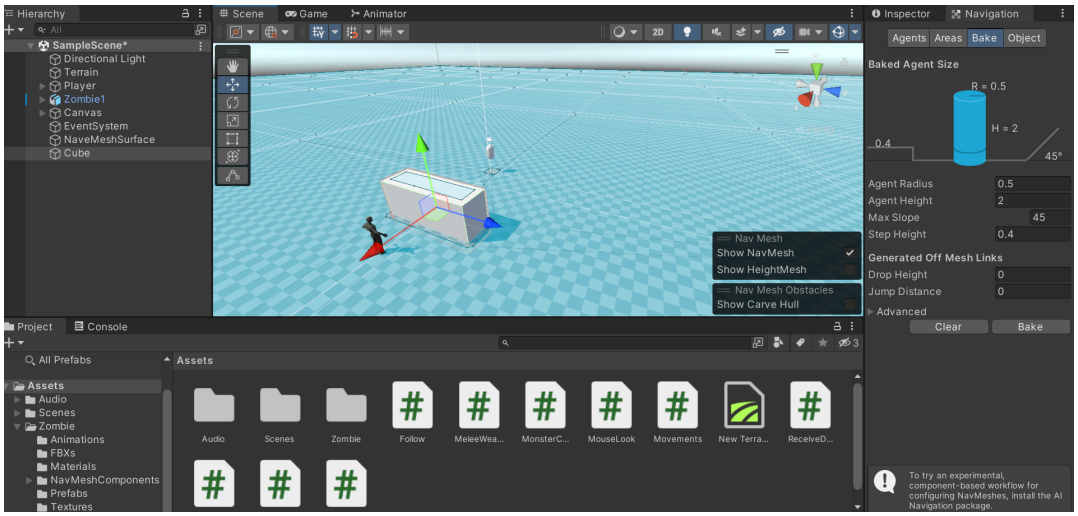

FIGURE 2 - Ici, grâce au Navmesh, on peut accéder à la partie haute du cube.

2.2 Gameplay

Une bonne partie du gameplay “classique” que l’on retrouve dans un FPS a été implémenté ; c’est-à-dire les déplacements du joueur, le déplacement de la vue avec la souris, le saut ainsi que la gravité. Le joueur peut sprinter en appuyant sur la touche shift gauche, et, pour les besoins du jeu, peut marcher plus lentement pour être plus discret en appuyant sur la touche contrôle gauche. Nous avons fait appel à la méthode ’SmoothDamp()’ de la classe ’Vector2’ ainsi que ’Math.Clamp()’ pour respectivement fluidifier les mouvements du joueur ainsi que de la caméra et empêcher le joueur de tourner sa caméra à l’infini sur l’axe vertical.

Le joueur peut également siffler en appuyant sur R. Cette action est soumise à un cooldown de 10 secondes. Pour ce faire, un script nommé Whistle.cs requiert un tableau de Clips audios à assigner dans l’éditeur parmi lequel le script va venir choisir aléatoirement à chaque fois que l’on presse R et que le cooldown est à 0.

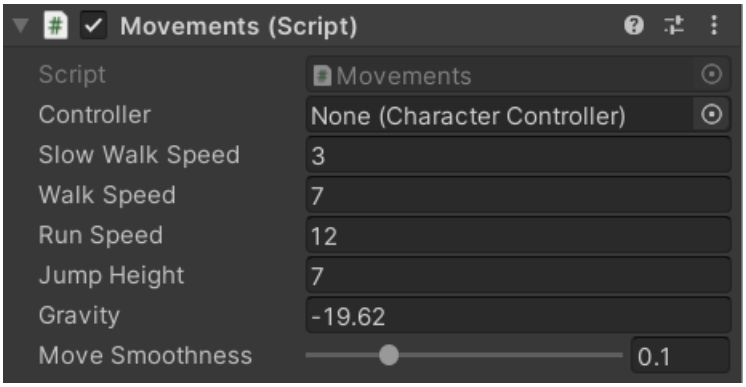

FIGURE 3 - Propriétés du script responsables des déplacements.

2.3 Réseau

Cette partie a été probablement l’une des plus longues jusqu’à maintenant. Nous avons décidé d’utiliser *Mirror* pour nous aider dans cette tâche.

Le joueur à la possibilité de se connecter ou bien comme hôte et client d’une partie, ou bien comme simple client pour rejoindre une partie déjà existante.

Le component 'Network Start Position' ajouté à un GameObject vide permet d'en faire un point de spawn potentiel pour le joueur.

Lorsqu'un joueur rejoint une partie, un script qu'il possède nommé 'NetworkSetup' va s'occuper de vérifier s'il s'agit de notre joueur ('isLocalPlayer'). Si ce n'est pas le cas, on va désactiver les scripts responsables des mouvements ainsi que de components comme la caméra ou l'audio listener. Sinon, on va désactiver la caméra principale pour que la caméra du client prenne le relais et instancier un 'PlayerUI'. De cette manière, il n'y aura pas de conflits (comme un joueur qui fait bouger les autres lorsqu'il se déplace) et chaque joueur peut agir indépendamment des autres.

Enfin, pour pouvoir voir les déplacements des autres clients certains événements à travers le serveur, nous avons utilisé le composant 'Network Transform' fourni par *Mirror*, et qui permet de synchroniser des événements avec le serveur.

Le tout est orchestré par un GameObject disposant du component 'Network Manager'

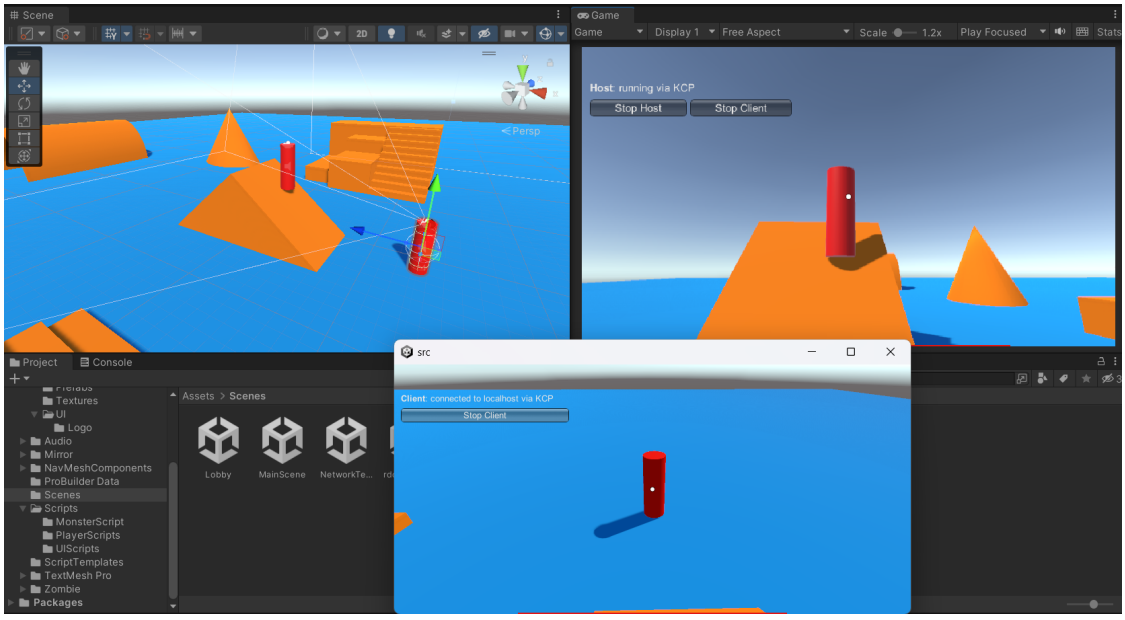

FIGURE 4 - À gauche, le point de vue serveur, à droite l'hôte (aussi client) de la partie et en bas un client

2.4 Modélisation des personnages et animation

Sur cette partie, nous n'avons pas fait grand chose, néanmoins nous avons commencé à prendre en main le logiciel *Blender* qui nous permettra par la suite de modéliser la forme de notre personnage et de les animer.

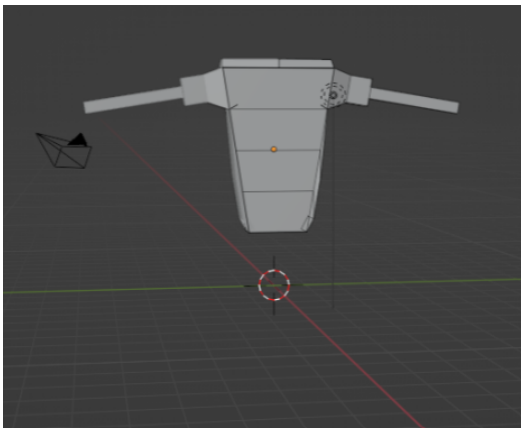

FIGURE 5 - Tout début de modèle 3D des joueurs

2.5 HUD et menus

La création des UI est d'abord passée par la recherche et la création d'assets et d'icônes. A l'aide des sites *corelvector.app* et *Photopea*, nous avons eu pour idée d'afficher le nom du jeu à travers une imitation de l'écran d'affichage d'une caméra, puisqu'il s'agit d'un des éléments majeurs du jeu. Nous avons également voulu utiliser une police dans un style un peu horreur et avons ainsi communément choisi l'actuelle qui se nomme "Restless Soul" et que nous avons trouvé sur le site *1001fonts.com*.

Après quelques bidouillages effectués à gauche à droite pour trouver d'autres idées, nous avons pensé à rétrécir et "emprisonner" le '*u*' de *Stuck* à l'intérieur de l'objectif, rappelant ainsi le fait que le joueur est bloqué. Nous avons ensuite flouté les autres lettres pour donner un effet de focus.

Une fois le logo créé, nous avons importé le tout dans Unity mais également les éléments séparés de celui-ci pour avoir une meilleure

liberté quant au placement des éléments. Nous avons ensuite ajouté un menu de pause ainsi qu'un fond d'écran pour le lobby.

Enfin nous avons ajouté une barre de vie très simple et un point blanc en guise de réticule.

FIGURES 6 ET 7 - À gauche le logo du jeu, à droite le logo adapté pour le serveur Discord

FIGURE 8 - Menu Pause

2.6 Modélisation décors (map)

La map est l'un des éléments clés du jeu : elle définit l'ambiance globale et impacte grandement l'immersion. Nous avons utilisé l'outil *Probuilder*, qui est implémenté dans *Unity*, pour façonner la map. Cet outil ludique a été facile à prendre en main grâce à divers tutoriels sur Youtube.

Le jeu se déroulant dans un manoir, il est important de créer et arranger des pièces en suivant une certaine logique. Par exemple, nous avons placé les chambres à l'étage, l'escalier principal au centre, des couloirs séparant les pièces, et enfin la salle de buffet proche des cuisines. Pour cette première soutenance, nous avons réalisé le rez-de-chaussée, le premier étage, mais également une grande partie du sous-sol, qui sera un labyrinthe.

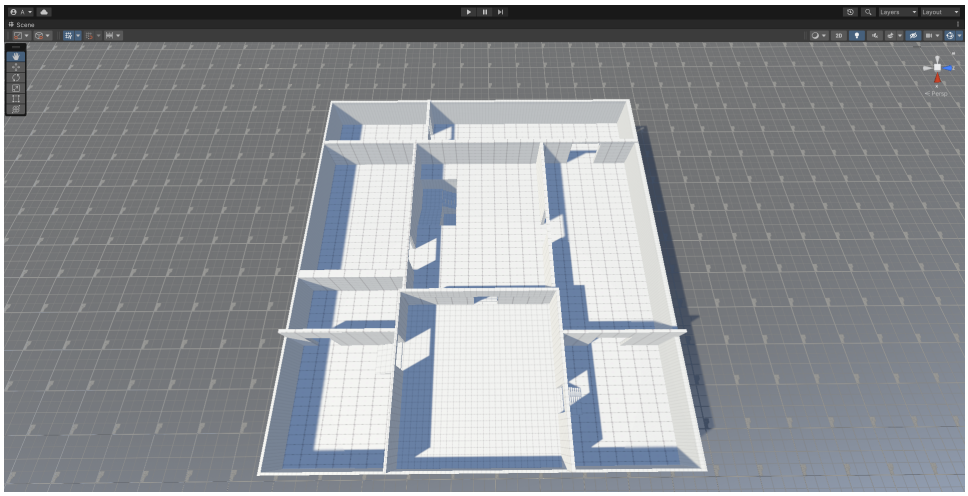

FIGURE 9 - 1er étage

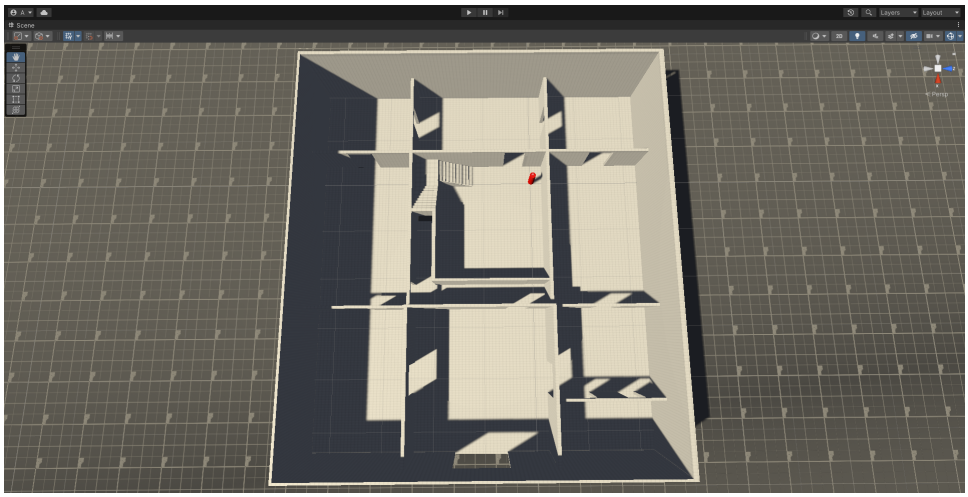

FIGURE 10 - Rez-de-chaussé

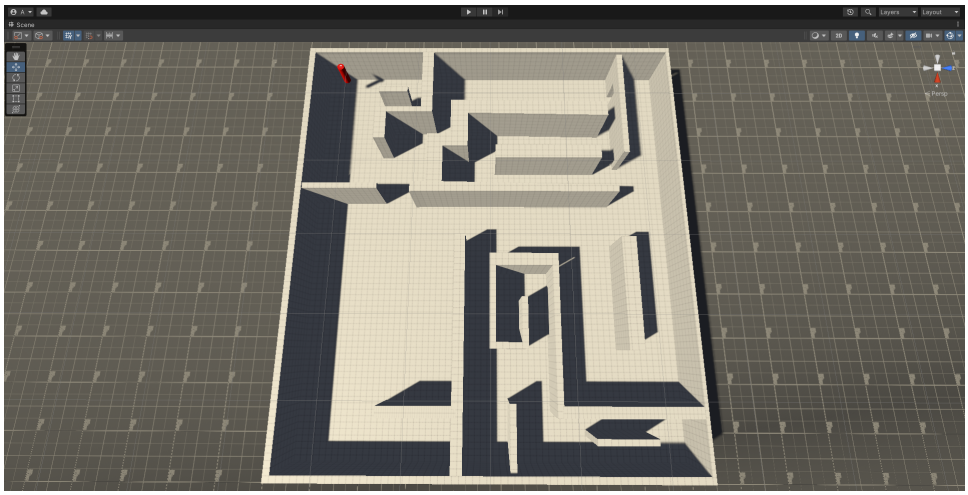

FIGURE 11 - Sous-sol (non fini)

2.7 Site Web

Pour ce qui est du site web, nous voulons faire quelque chose de simple et assez épuré, nous avons donc décidé de simplement garder le combo html/css/js. Alexis utilise également le préprocesseur *SASS* pour faciliter l'écriture du css. Pour l'instant, le design global du site à été trouvé et la barre du menu mise en place.

FIGURE 12 - Un aperçu du site

3 Avancement du projet

3.1 Comparaison avec le cahier des charges

Soutenance 1	Prévisions	Avancements
Gameplay	50%	40%
Interfaces Graphiques	33%	30%
Modélisation personnages/objets	30%	5%
Animations	30%	0%
Modélisation décors (map)	35%	33%
Réseau	40%	40%
IA	40 %	40%
Site Web	33%	30%

Légende :
% Rouge -> retard important
% Sans couleur -> pas ou peu de retard

3.2 IA

Pour ce qui est de l'IA, nous avons fait face à plusieurs problèmes, pour tout ce qui est de l'attaque et de l'activation des animations, mais aussi de l'implémentation de cette dernière dans la map. Cependant, nous avons bien l'intention de résoudre ces problèmes d'ici la prochaine présentation.

3.3 Gameplay

Nous avons commencé par les déplacements du joueur, ce qui nous semblait primordial. Mais un mécanisme très important n'a pas encore été ajouté : celui de la caméra. Or, il est l'un des fondements sur lequel repose le jeu et la victoire du joueur. Nous comptons donc attaquer ce mécanisme en priorité pour la suite du développement.

3.4 Réseau

La partie réseau est un peu particulière puisque nous ne voulons absolument pas le faire après le reste, de peur d'avoir à refaire une bonne partie du jeu. Elle évolue donc au fur et à mesure que de nouvelles fonctionnalités arrivent et pour l'instant nous ne sommes ni vraiment en retard ni en avance par rapport aux autres tâches.

3.5 Modélisation personnages et animation

Cette partie est sans doute la moins avancée et celle qui a le plus de retard. Il reste encore (si le temps ne nous manque pas) à modéliser les personnages et le monstre, ainsi que toutes les animations de ceux-ci.

3.6 HUD et menus

Ayant opté pour des interfaces graphiques assez simplistes, nous avons bien entamé cette tâche. Il y a cependant quelques coquilles avec le positionnement et le responsive des éléments d'UI, de même qu'avec l'implémentation du réseau (impossibilité pour les autres clients de se connecter lorsque l'hôte est dans le menu pause). Il faudra aussi faire un menu titre, un écran de mort, un inventaire ainsi qu'un UI pour la caméra en parallèle à la partie gameplay.

3.7 Modélisation décors (map)

Il nous reste à créer le grenier, terminer le sous-sol, et bien-entendu ajouter les textures ainsi que les éléments de décor, comme des tableaux ou des meubles. Nous avons également pensé à créer des passages étroits que seuls les joueurs pourront emprunter afin de semer les monstres.

3.8 Site Web

Le site web n'est qu'un peu commencé, mais il ne s'agit pas d'une tâche principale, bien que nécessaire. Pas de gros retard à déplorer.

4 Prévisions pour la deuxième soutenance

Voici le nouveau tableau d'avancement mis à jour en fonction du départ de Vincent et de notre avancée sur la soutenance 1.

Soutenance	2	3
Gameplay	70%	100%
Interfaces Graphiques	66%	100%
Modélisation personnages/objets	50%	100%
Animations	40%	100%
Modélisation décors (map)	60%	100%
Réseau	60%	100%
IA	60%	100%
Site Web	50%	100%

5 Conclusion

L'absence d'un des membre nous a assez ralenti et causé du retard. Nous sommes encore en cours d'adaptation mais nous avons tout de même essayé de palier cela du mieux possible. Nous tâchons de fournir les efforts nécessaires pour faire avancer ce projet qui nous tient tous à coeur. Maintenant que nous commençons à mieux savoir utiliser *Unity* et les autres outils, nous espérons pouvoir rattraper le retard accumulé, et pouvoir implémenter toutes les fonctionnalités désirées, et bien sûr voir le jeu naître d'ici la fin du semestre sans avoir dû faire de gros sacrifices.

Merci pour votre lecture,
L'équipe d'AVAC#