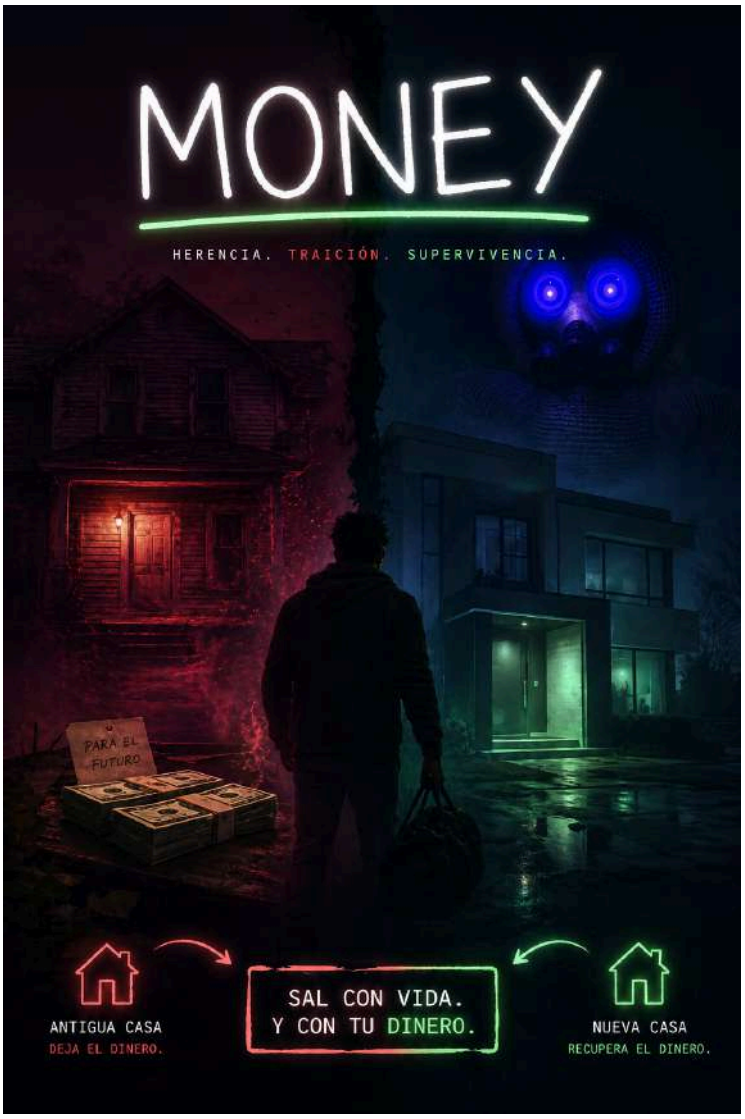


“Money”
Projecte de desenvolupament
CFGS Administració de Sistemes Informàtics i Xarxes

Yasser Belouafi
Marcos Ruiz
SMX2B
2025-2026

Índex

Resum del projecte :

Aquest projecte consisteix en el desenvolupament d'un videojoc de terror en primera persona anomenat *Money*, realitzat amb el motor de joc Godot. La història segueix un jove que, després de mudar-se de casa, s'adona que ha deixat enrere els diners de la fiança a la seva antiga residència. En tornar a buscar-los, es troba amb un escenari pertorbador: hi ha morts a la casa i alguna cosa, o algú, el persegueix. L'objectiu del jugador és recuperar els diners i escapar amb vida.

El joc es basa en mecàniques senzilles d'exploració, recollida d'objectes i fugida, acompanyades d'una ambientació fosca i elements sonors inquietants que generen tensió constant. Ens hem inspirat en un joc indie de la plataforma itch.io de característiques similars, adaptant la idea amb la nostra pròpia narrativa i disseny.

Aquest projecte ha estat desenvolupat en grup com a treball final del cicle formatiu de grau mitjà de Sistemes Microinformàtics i Xarxes (SMX), amb l'objectiu d'aprendre a utilitzar eines de desenvolupament de videojocs reals i aplicar els coneixements adquirits durant el curs. A més a més intentarem aprende a utilizar la ia de manera eficient y correcta ya que ara mateix es de molta ayuda en sector de la programació i marketing

Paraules clau (entre 4 i 8): terror, videojoc, Godot, primera persona, exploració, escapament

Abstract :

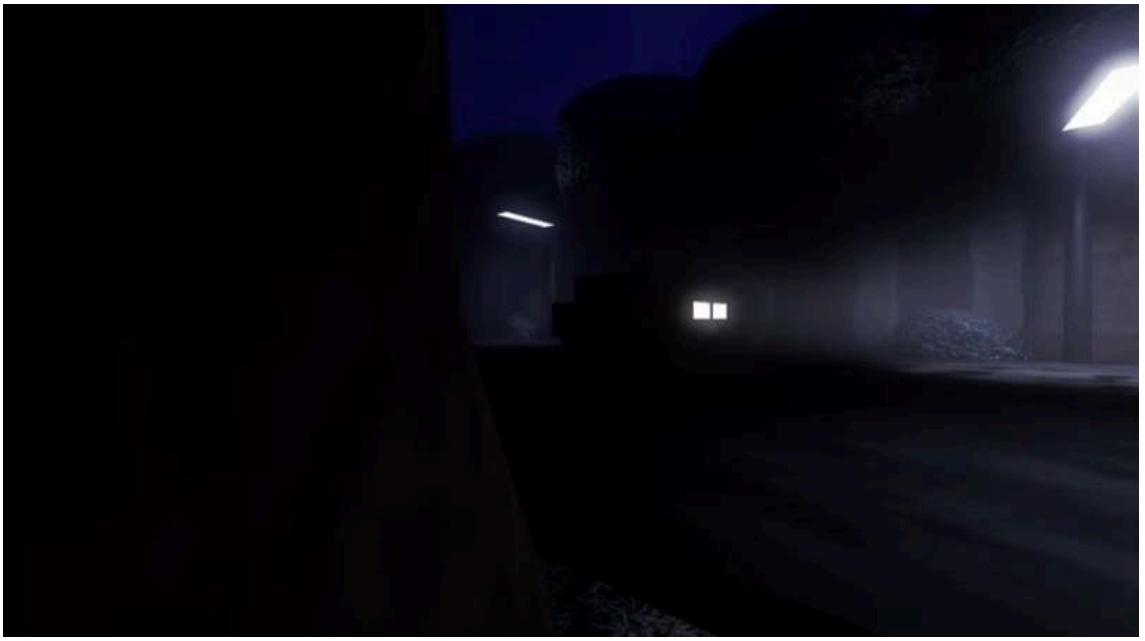
This project consists of the development of a first-person horror video game called Money, built using the Godot game engine. The story follows a young man who, after moving out of his old home, realizes he left his security deposit money behind. When he returns to retrieve it, he finds a disturbing scene: there are dead bodies inside the house and something, or someone, is hunting him. The player must find the money and escape alive.

The game features simple mechanics focused on exploration, item collection, and escape, combined with a dark atmosphere and unsettling sound design that builds constant tension. The project was inspired by an indie game found on the itch.io platform, which we adapted with our own narrative and visual identity.

This project was developed as a group final project for the Microcomputer Systems and Networks (SMX) vocational training programme, with the goal of learning how to use real game development tools and applying the knowledge gained throughout the course.Furthermore, we will aim to learn how to use artificial intelligence efficiently and correctly, as it is currently of great help in the fields of programming and marketing.

Keywords (entre 4 i 8): horror, video game, Godot, first person, exploration, escape

Llista de figures



1. Introducció

En aquest projecte de síntesi hem desenvolupat un videojoc de terror en primera persona anomenat Money, utilitzant el motor de videojocs Godot Engine. El projecte ha estat realitzat per en Yasser i en Marcos, dos alumnes del cicle formatiu de grau mitjà de Sistemes Microinformàtics i Xarxes (SMX), com a treball final de cicle.

La idea del joc sorgeix de la inspiració en un joc indie trobat a la plataforma itch.io, del qual hem agafat la base conceptual per desenvolupar la nostra pròpia versió, afegint-hi idees i elements propis. El joc es basa en una mecànica senzilla però efectiva: el jugador torna a la seva antiga casa a recuperar els diners de la fiança i s'adona que algú el persegueix. Haurà de recollir els diners i escapar amb vida.

L'objectiu principal ha estat crear un joc funcional i jugable, aplicant els coneixements adquirits durant el curs en àrees com la programació, els sistemes informàtics i la resolució de problemes tècnics. A més, el projecte ens ha servit com a oportunitat per aprendre eines professionals de desenvolupament de videojocs que s'utilitzen en l'àmbit real, com ara el llenguatge GDScript, el sistema de nodes de Godot, les tècniques bàsiques de modelatge 3D amb Blender i el disseny de nivells i escenaris.

1.1 Context

El projecte sorgeix de l'interès comú dels dos membres del grup pels videojocs de terror. Tots dos som aficionats a aquest gènere i creiem que és un dels que millor aprofita les possibilitats d'un entorn 3D, ja que permet construir atmosferes opressives i inquietants mitjançant la il·luminació, el disseny dels espais i els elements sonors.

La idea del joc es va inspirar en un joc indie trobat a la plataforma **itch.io**, de mecàniques senzilles, centrat en l'exploració en primera persona i la fugida d'un perseguidor. A partir d'aquesta referència, vam decidir crear la nostra pròpia versió amb una narrativa original i un disseny propi, mantenint la simplicitat però afegint les nostres idees. El resultat és **Money**, un joc en què el protagonista torna a la seva antiga casa a recuperar els diners de la fiança i descobreix que algú el persegueix. Haurà de recollir els diners repartits per la casa i escapar amb vida.

El projecte ha estat desenvolupat íntegrament durant el curs, aprenent de manera autònoma les eines necessàries per tirar-lo endavant, com ara **Godot Engine**, **Blender** i **GDScript**, combinant-les amb els coneixements adquirits al llarg del cicle formatiu.

1.2 Justificació

Aquest projecte permet aplicar de manera pràctica els coneixements adquirits durant el cicle de Sistemes Microinformàtics i Xarxes, especialment en programació, ús d'eines informàtiques i resolució de problemes tècnics.

L'elecció d'un videojoc de terror en primera persona com a projecte de síntesi es justifica per diverses raons. En primer lloc, el desenvolupament de videojocs és un sector en creixement constant que integra àrees com la programació, el disseny i la gestió de projectes, totes elles directament relacionades amb el perfil professional del cicle. En segon lloc, les eines utilitzades, com **Godot Engine**, **Blender** o **GitHub**, són gratuïtes, accessibles i cada vegada més presents en l'àmbit professional, cosa que ens permet adquirir experiència real sense necessitat de llicències de pagament.

A més, un projecte pràctic com aquest ens obliga a enfrontar-nos a problemes reals de desenvolupament, com ara la gestió d'errors, la presa de decisions tècniques i l'organització del treball en equip, aspectes que consoliden els coneixements del curs d'una manera molt més efectiva que la teoria.

1.3 Objectius

Els objectius principals del projecte són els següents:

- Desenvolupar un videojoc de terror en primera persona funcional i jugable, anomenat Money, utilitzant Godot Engine com a motor de desenvolupament.
- Aprendre i aplicar el llenguatge de programació GDScript per implementar les mecàniques del joc, com ara el moviment del jugador, la intel·ligència artificial de l'enemic, la recollida d'objectes i la gestió de les escenes.
- Modelar i dissenyar l'escenari principal del joc mitjançant Blender, creant una casa de dues plantes amb ambientació de terror coherent.
- Integrar elements d'àudio i il·luminació per reforçar l'atmosfera inquietant del joc i augmentar la immersió del jugador.
- Gestionar el projecte de manera col·laborativa utilitzant GitHub com a eina de control de versions, garantint un flux de treball organitzat entre els dos membres de l'equip.
- Documentar tot el procés de desenvolupament de manera clara i estructurada en aquesta memòria, justificant les decisions tècniques preses en cada fase del projecte.

1.3.1 Objectiu general

L'objectiu general d'aquest projecte és desenvolupar un videojoc de terror en primera persona anomenat Money utilitzant el motor Godot Engine, aplicant coneixements de programació, modelatge 3D i gestió de projectes adquirits durant el cicle formatiu. A més, volem aprendre a gestionar correctament la documentació tècnica del projecte i a crear materials audiovisuals com un

tràiler i portades, explorant l'ús d'eines d'intel·ligència artificial com a suport durant el desenvolupament.

1.3.2 Objectius específics

- Aprendre a utilitzar **Godot Engine** com a motor de desenvolupament de videojocs 3D i adquirir coneixements de **GDScript** per programar totes les mecàniques del joc, com ara el moviment del jugador, la intel·ligència artificial de l'enemic, la recollida d'objectes i la gestió de les escenes.
- Desenvolupar un joc jugable i complet, amb un escenari de terror en primera persona, un objectiu clar per al jugador i una atmosfera coherent i immersiva.
- Aprendre a utilitzar **Blender** per al modelatge 3D de l'escenari principal del joc, creant una casa de dues plantes amb mobles i objectes decoratius propis.
- Aprendre a utilitzar **GitHub** per al control de versions del projecte i per treballar de forma col·laborativa i organitzada entre els dos membres del grup.
- Familiaritzar-nos amb les eines de **Google** com Google Drive i Google Docs per gestionar i documentar el projecte de manera eficient.
- Explorar l'ús de la **intel·ligència artificial** en l'àmbit del desenvolupament, aprenent com pot ajudar en tasques com la creació del tràiler, les portades o la resolució de problemes tècnics.

1.4 Estratègia i planificació del projecte

L'estratègia triada per desenvolupar aquest projecte ha estat la creació d'un producte nou basat en una idea existent. Ens hem inspirat en un joc indie de la plataforma itch.io per definir el concepte general, però el disseny, la programació, l'escenari i la narrativa són originals i desenvolupats íntegrament per nosaltres. Aquesta estratègia ens ha semblat la més adequada perquè ens permet tenir una referència clara de cap on volem anar sense limitar la nostra creativitat ni copiar res existent.

Pel que fa a la viabilitat del projecte, creiem que és realitzable dins del temps disponible perquè les mecàniques que volem implementar són senzilles: moviment en primera persona, recollida d'objectes i un personatge que persegueix el jugador. No busquem fer un joc complex, sinó un joc ben acabat i amb una bona atmosfera de terror.

Rols del grup

Des del principi vam decidir dividir les tasques segons les habilitats de cadascú. En Yasser, que té més experiència en programació, s'ha encarregat principalment del desenvolupament del joc a Godot, la programació de les mecàniques i tota la part tècnica. En Marcos s'ha encarregat de la gestió de la documentació, l'organització de les tasques i els materials del projecte com el tràiler o les portades. Tot i aquesta divisió, tots dos ens informem constantment

dels canvis que fa l'altre i compartim les decisions importants de manera conjunta.

Eines de comunicació i organització

Per coordinar-nos hem utilitzat una llista de tasques compartida que ens permet saber en tot moment en què està treballant cadascú. Per a decisions més elaborades fem servir Discord, i per a avisos puntuals i canvis ràpids ens coordinem per WhatsApp.

1.5 Metodologia de treball

La metodologia que hem seguit al llarg del projecte ha estat una **metodologia àgil adaptada** a les nostres necessitats. No hem aplicat cap framework estricte com Scrum o Mètrica 3.0, sinó que hem optat per una forma de treballar flexible i iterativa, on s'avança per parts i s'adapta el pla segons els obstacles que van sorgint.

Aquesta decisió va venir motivada per un canvi important a mig trimestre. Inicialment teníem plantejat un projecte diferent, però en adonar-nos que no era viable dins del temps i els recursos disponibles, vam haver de canviar completament la idea i reorientar el projecte cap al que és avui. Aquesta situació ens va ensenyar que en el desenvolupament de programari és essencial ser flexible i saber adaptar-se quan un plantejament inicial no funciona, en lloc d'insistir en quelcom que no és realitzable.

A partir d'aquell moment vam optar per avançar de forma més lliure, cadascú al seu ritme segons el seu rol, establint petites metes a curt termini i comunicant-nos els avenços de manera constant. Quan una cosa no funcionava, la canviàvem sense bloquejar-nos ni perdre temps innecessari.

Pel que fa al seguiment del projecte, no hem utilitzat cap eina de gestió formal com Trello o diagrames de Gantt. La nostra organització s'ha basat en una llista de tasques pròpia, la comunicació per **Discord** per a decisions importants i **WhatsApp** per a canvis puntuals. Tot i ser un sistema senzill, ha resultat suficient i efectiu per a un equip de dues persones.

1.6 Estudi econòmic i pressupostari

Eines i recursos utilitzats

Totes les eines utilitzades en aquest projecte són gratuïtes o proporcionades pel centre, cosa que fa que el cost en llicències i programari sigui zero. Les eines principals són les següents:

- Godot Engine: motor de desenvolupament gratuït i de codi obert.
- GDScript: llenguatge de programació propi de Godot, sense cost addicional.
- GitHub: plataforma de control de versions gratuïta per a projectes públics.
- Google Drive i Google Docs: eines de gestió i documentació gratuïtes.

- Discord i WhatsApp: eines de comunicació gratuïtes.
- Ordinadors del centre durant les hores de classe, i ordinadors propis a casa.

Càlcul d'hores de desenvolupament

Per calcular el cost orientatiu del projecte hem tingut en compte les hores dedicades per cada membre del grup al llarg de tot el trimestre.

Durant la fase inicial del projecte, hem dedicat 3 hores setmanals al centre durant aproximadament 3 mesos, és a dir, unes 12 setmanes. Això representa 36 hores per persona al centre. A casa, hem dedicat aproximadament 1 hora setmanal durant aquest mateix període, és a dir, unes 12 hores per persona.

Durant el tram final del projecte, que durarà aproximadament 4 setmanes, dedicarem 10 hores setmanals al centre i entre 4 i 5 hores setmanals a casa. Això representa unes 40 hores al centre i unes 18 hores a casa per persona en aquest últim tram.

El total estimat per persona és d'aproximadament **106 hores**, i entre els dos membres del grup el total és d'unes **212 hores** de treball.

Cost orientatiu

Per calcular el cost de desenvolupament hem aplicat un preu orientatiu de programador júnior, que se situa al voltant dels 12€ per hora.

Concepte	Hores	Cost per hora	Total
Desenvolupament (Yasser)	106 h	12 €/h	1.272 €
Gestió i documentació (Marcos)	106 h	12 €/h	1.272 €
Programari i llicències	—	—	0 €
Total	**212 h**		**2.544 €**

Costos de manteniment

En tractar-se d'un projecte acadèmic sense servidor ni infraestructura externa, els costos de manteniment un cop finalitzat el projecte són pràcticament nuls.

Si en un futur es volgués publicar el joc en una plataforma com itch.io, el cost de publicació també seria gratuït.

2. Descripció del projecte

Money és un videojoc de terror en primera persona desenvolupat amb el motor **Godot Engine**. El jugador encarna un noi que torna a la seva antiga casa per recuperar els diners de la fiança que s'havia deixat en mudar-se. En arribar, descobreix que quelcom terrible ha passat a l'interior: hi ha cadàvers escampats per la casa i una presència desconeguda i pertorbadora el persegueix des de les ombres.

L'escenari principal és la casa, un entorn tancat, fosc i claustrofòbic que el jugador ha d'explorar per trobar els diners repartits per les diferents habitacions i escapar amb vida. La única font de llum disponible és una **llanterna**, que el jugador pot encendre i apagar, cosa que limita la visibilitat i augmenta la sensació constant de vulnerabilitat i perill. En algunes zones de la casa l'enemic persegueix activament el jugador, mentre que en d'altres la tensió es genera únicament per l'atmosfera, els sons ambients i el disseny de l'entorn.

La casa disposa de **dues plantes**: una planta principal, on es desenvolupa la major part de l'acció, i un **soterrani**, una zona més tancada i opressiva que incrementa la sensació d'angoixa. Ambdues plantes han estat modelades íntegrament amb **Blender** i importades a Godot, on s'hi han afegit els elements interactius, la il·luminació i les col·lisions.

El joc es basa en tres pilars fonamentals. El primer és l'**exploració**, on el jugador ha de recórrer la casa i localitzar tots els diners abans de poder escapar. El segon és la **tensió**, generada per la il·luminació escassa, els sons inquietants i la presència constant de l'enemic. El tercer és la **fugida**, on el jugador ha d'evitar ser atrapat mentre completa el seu objectiu.

La inspiració per al projecte prové d'un joc indie de la plataforma **itch.io** de mecàniques similars, del qual hem agafat la base conceptual per desenvolupar la nostra pròpia versió amb una narrativa i un disseny originals.

2.1 Anàlisi de requisits [projecte de desenvolupament]

2.1.1 Requisits funcionals

Els requisits funcionals descriuen les funcionalitats que el joc ha de tenir per ser jugable i complir els objectius del projecte.

- **Moviment en primera persona:** El jugador ha de poder moure's lliurement per l'interior de la casa en una perspectiva en primera persona, explorant les diferents habitacions de l'escenari.
- **Recollida d'objectes:** El jugador ha de poder recollir els diners de la fiança escampats per la casa. Aquesta és la condició principal per poder escapar i completar el joc. Opcionalment, es podrien afegir notes que aporten context a la història.
- **Sistema d'enemic:** Hi ha un personatge enemic que persegueix el jugador en determinades zones de la casa. En altres zones predomina l'exploració lliure, on la tensió es genera únicament per l'atmosfera i l'entorn, sense que l'enemic estigui present activament.
- **Llanterna:** El jugador disposa d'una llanterna per il·luminar l'entorn fosc de la casa. És l'única font de llum disponible i és essencial per a la jugabilitat, ja que sense ella l'escenari és pràcticament innavegable.
- **Condició de victòria:** Un cop el jugador ha recollit els diners, ha de poder escapar de la casa per completar el joc i accedir a la pantalla de victòria.
- **Condició de derrota:** Si l'enemic atrapa el jugador, apareix una pantalla de mort que permet tornar a intentar-ho des del principi.
- **Menú principal:** El joc ha de disposar d'un menú inicial amb les opcions bàsiques de començar la partida i sortir del joc.

2.1.2 Requisits no funcionals

Els requisits no funcionals descriuen les característiques de qualitat que ha de complir el joc més enllà de les funcionalitats concretes.

. **Rendiment:** El joc ha de funcionar de manera fluida sense caigudes de rendiment significatives. L'objectiu és mantenir una taxa de fotogrames estable que no afecti negativament l'experiència del jugador.
. **Usabilitat:** Els controls han de ser intuïtius i fàcils d'aprendre, sense necessitat de tutorials complexos. El jugador ha de poder entendre com moure's i interactuar amb l'entorn de forma natural des del primer moment.
. **Atmosfera i immersió:** L'entorn visual i sonor ha de ser coherent amb el gènere de terror. La il·luminació fosca, els espais tancats i els elements sonors han de contribuir a generar una experiència immersiva i angoixant.
. **Estabilitat:** El joc no ha de presentar errors crítics durant la partida, com tancaments inesperats o bloquejos que impedeixin al jugador continuar.

- **Compatibilitat:** El joc ha de poder executar-se correctament en els ordinadors disponibles, tant els del centre com els ordinadors personals dels membres del grup.
- **Mantenibilitat:** El codi ha d'estar organitzat i estructurat de manera que sigui fàcil de modificar o ampliar en cas que es vulgui afegir noves funcionalitats en el futur.

2.2.1 Tecnologies

Per al desenvolupament del projecte hem utilitzat eines gratuïtes i accessibles que s'adapten al nostre nivell i als recursos disponibles. Com a motor principal hem triat Godot Engine, que ens permet crear el joc en 3D utilitzant GDScript com a llenguatge de programació. Per a la gestió i documentació del projecte hem fet servir Google Drive i Google Docs, que ens permeten compartir i organitzar tot el material entre els dos membres del grup. Com a eines de comunicació hem utilitzat Discord i WhatsApp per coordinar-nos durant el desenvolupament. Finalment, hem incorporat diverses eines d'intel·ligència artificial com ChatGPT, Claude i Qwen com a suport per a la programació, la documentació i la creació de materials audiovisuals.

2.2.2 Comparativa de les tecnologies valorades

Motor de videojoc: Roblox Studio vs Godot Engine

Al inici del projecte vam valorar dues opcions per desenvolupar el joc. La primera opció era Roblox Studio, una plataforma de creació de jocs que utilitza el llenguatge de programació Luau. Tot i que és una eina accessible i amb una comunitat gran, vam descartar-la perquè les seves limitacions creatives i el seu entorn tancat no ens permetien desenvolupar el tipus de joc de terror en primera persona que teníem en ment. A més, Roblox està orientat a un públic i un estil molt concret que no encaixava amb la nostra idea.

Finalment vam triar Godot Engine perquè és un motor de codi obert, gratuït i amb suport per a entorns 3D complets. A més, és l'eina que ens proporciona el centre i té una corba d'aprenentatge més adequada per al temps disponible.

	Roblox Studio	Godot Engine
Cost	Gratuït	Gratuït
Llenguatge	Luau	GDScript
Suport 3D	Limitat	Complet
Llibertat creativa	Limitada	Alta

Dificultat	Baixa	Mitjana
Triat	No	Sí

Control de versions: GitHub

Inicialment teníem previst utilitzar GitHub per gestionar les versions del projecte. Finalment hem decidit no fer-lo servir de forma activa durant el desenvolupament, ja que el joc es lliurarà en una versió definitiva sense necessitat de gestionar branques ni historial de canvis. La compartició d'arxius i la coordinació entre els dos membres del grup s'ha gestionat mitjançant Google Drive.

Intel·ligència artificial: ChatGPT vs Claude vs Qwen

Hem decidit no limitar-nos a una sola eina d'intel·ligència artificial, sinó explotar les qualitats de cadascuna per obtenir el millor resultat possible. Cada IA té punts forts diferents i combinar-les ens permet tenir diferents punts de vista davant d'un mateix problema.

	ChatGPT	**Claude**	**Qwen**
Empresa	OpenAI	Anthropic	Alibaba
Punts forts	Versatilitat general	Redacció i raonament	Multilingüe i tècnic
Ús principal	Dubtes de programació	Documentació	Consultes tècniques
Triat	**Sí**	**Sí**	**Sí**

2.2.3 Tecnologies escollides

Godot Engine Motor de desenvolupament escollit per crear el videojoc. És gratuït, de codi obert i ofereix suport complet per a entorns 3D. A més, és l'eina proporcionada pel centre i té una corba d'aprenentatge adequada per al nivell i el temps disponible.

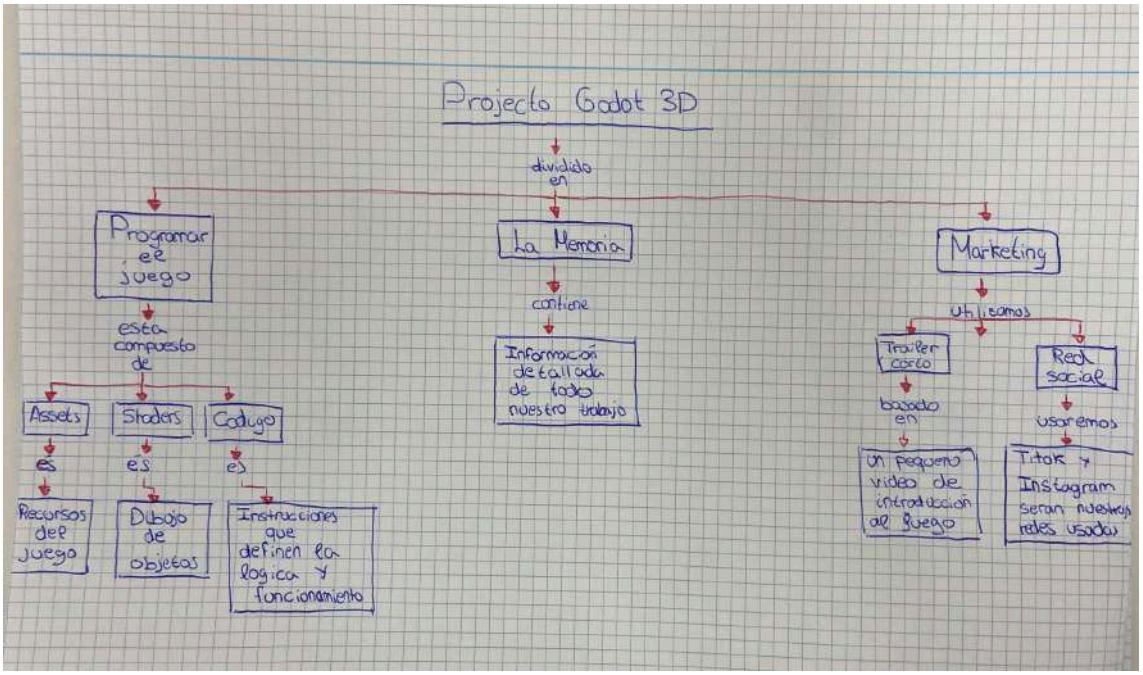
GDScript Llenguatge de programació propi de Godot escollit per programar tota la lògica del joc. La seva sintaxi és similar a Python, cosa que el fa relativament accessible per a programadors amb poca experiència en desenvolupament de videojocs.

Google Drive i Google Docs Eines escollides per a la gestió i emmagatzematge de la documentació del projecte. Permeten compartir arxius entre els dos membres del grup de forma senzilla i tenir tot el material centralitzat i accessible des de qualsevol dispositiu.

ChatGPT, Claude i Qwen Eines d'intel·ligència artificial escollides com a suport durant tot el projecte. S'han utilitzat de forma complementària, aprofitant les qualitats de cadascuna per resoldre dubtes de programació, obtenir idees i redactar la documentació. A més, tenim previst utilitzar alguna eina d'IA per a la creació del tràiler i les portades del joc.

Discord i WhatsApp Eines de comunicació escollides per coordinar-nos entre els dos membres del grup de manera eficient i àgil al llarg de tot el desenvolupament del projecte.

2.3 Estructura del projecte



2.4 Descripció dels components

- **Menú principal** És la primera pantalla que veu el jugador en iniciar el joc. Conté les opcions bàsiques per començar la partida o sortir del joc. És el punt d'entrada a tota l'experiència.
- **Pantalla de mort** Apareix quan l'enemic atrapa el jugador. Informa el jugador que ha perdut i li ofereix la possibilitat de tornar a intentar-ho des del principi.
- **Pantalla de final** Apareix quan el jugador recull els diners i aconsegueix escapar de la casa. És la condició de victòria del joc.
- **El jugador:** És el component central del joc. Està format pels següents elements:
    - **Càmera en primera persona:** Defineix el punt de vista del jugador i li permet explorar l'entorn de forma immersiva.
    - **Llanterna:** Única font de llum disponible. El jugador la porta durant tota la partida i és imprescindible per veure l'entorn fosc de la casa. Genera ombres i limita la visibilitat, cosa que augmenta la tensió.

- **Col·lisions:** Permeten que el jugador interactuï físicament amb l'entorn, evitant que travessi parets, portes o objectes de l'escenari.
- **Sistema de recollida:** Permet al jugador interactuar amb els objectes del joc, principalment els diners que ha de recuperar per completar la partida.

L'enemic És el principal element de perill del joc. No està present a totes les zones de la casa, sinó que apareix i persegueix el jugador en determinats moments i àrees concretes. En la resta de zones la tensió es genera únicament per l'atmosfera i l'entorn. Si l'enemic atrapa el jugador, s'activa la condició de derrota.

L'escenari La casa és l'escenari únic del joc i està dividit en dues zones principals:

- **Planta principal:** Zona més àmplia de la casa, formada per diverses habitacions que el jugador ha d'explorar per trobar els diners. És on predomina l'exploració i on l'enemic pot aparèixer en determinats moments.
- **Desvan:** Zona superior de la casa, més tancada i claustrofòbica. Afegeix una capa addicional d'exploració i pot contenir elements clau per a la progressió del jugador.

Tots dos espais estan dissenyats amb una il·luminació mínima per mantenir l'atmosfera fosca i opressiva característica del gènere de terror.

2.4.1 Component 1 — Menús

Els menús són les pantalles que el jugador veu fora de la partida pròpiament dita. El joc disposa de tres pantalles principals:

El menú principal és la primera pantalla que apareix en iniciar el joc. Conté les opcions bàsiques per començar la partida o sortir del joc. El seu disseny ha de transmetre ja des del primer moment l'atmosfera de terror del joc mitjançant colors foscos i una estètica coherent amb l'entorn.

La pantalla de mort apareix quan l'enemic atrapa el jugador. Informa que la partida ha acabat i ofereix la possibilitat de tornar a intentar-ho des del principi. És un component essencial per mantenir el bucle de joc actiu.

La pantalla de final apareix quan el jugador completa l'objectiu principal, és a dir, quan recull els diners i escapa de la casa. És la condició de victòria i tanca l'experiència del jugador de forma satisfactòria.

2.4.2 Component 2 — El jugador

El jugador és el component més important del joc, ja que és l'element que l'usuari controla directament. Està format per diversos elements que treballen junts per oferir una experiència en primera persona immersiva.

La càmera en primera persona defineix el punt de vista del jugador. Tot el que el jugador veu durant la partida passa a través d'aquesta càmera, cosa que reforça la sensació d'estar dins de la casa i augmenta la immersió.

La llanterna és l'única font de llum disponible durant tota la partida. El jugador la porta amb ell en tot moment i és imprescindible per poder veure l'entorn fosc de la casa. La seva il·luminació limitada genera zones d'ombra i racons foscos que contribueixen directament a la tensió i la por del jugador.

Les col·lisions permeten que el jugador interactuï físicament amb l'entorn de forma realista, evitant que travessi parets, portes, mobles o qualsevol altre element de l'escenari.

El sistema de recollida permet al jugador interactuar amb els objectes del joc. Quan el jugador s'apropa als diners i prem el botó d'interacció, l'objecte es recull i s'acumula al comptador necessari per completar la partida.

2.4.3 Component 3 — L'enemic

L'enemic és el principal element de perill i tensió del joc. A diferència d'altres jocs de terror, l'enemic no està present a totes les zones de la casa de forma constant, sinó que apareix i persegueix activament el jugador en determinades àrees i moments concrets de la partida.

En les zones on l'enemic no està present, la tensió es genera únicament per l'atmosfera, la foscor i el disseny de l'entorn. Aquesta decisió de disseny permet alternar moments d'exploració tranquil·la amb moments de perill real, cosa que fa que el jugador estigui sempre en tensió sense saber quan apareixerà l'amenaça.

Quan l'enemic detecta el jugador dins de la seva zona, comença a perseguir-lo activament. Si aconsegueix atrapar-lo, s'activa la condició de derrota i apareix la pantalla de mort.

2.4.4 Component 4 — L'escenari

L'escenari és la casa on transcorre tota l'acció del joc. Està dissenyat com un entorn tancat i fosc que el jugador ha d'explorar per trobar els diners i escapar. Es divideix en dues zones principals:

La planta principal és la zona més àmplia de la casa. Està formada per diverses habitacions interconnectades que el jugador ha d'explorar. És l'espai principal de joc, on es troben la majoria dels objectius i on l'enemic pot aparèixer en determinats moments.

El desvan és la zona superior de la casa, accessible des de la planta principal. És un espai més petit, tancat i claustrofòbic que afegeix una capa addicional d'exploració. El seu disseny reforça la sensació d'opressió i vulnerabilitat del jugador.

Tots dos espais estan dissenyats amb una il·luminació mínima i una estètica coherent amb el gènere de terror, on la foscor, els racons i els elements de l'entorn contribueixen a generar una atmosfera inquietant i angoixant.

2.5 Definició de les funcionalitats [projecte de desenvolupament]

En aquest apartat es descriu de manera conceptual i detallada totes les funcionalitats que ofereix el videojoc *Money*, explicant com funcionen, quin procés segueixen i quin és el seu estat d'implementació actual.

2.5.1 Funcionalitat 1

Moviment del jugador en primera persona: El jugador pot desplaçar-se per l'escenari en primera persona mitjançant WASD. El sistema de moviment inclou caminar i la possibilitat de córrer durant un temps limitat. Aquesta funcionalitat s'implementa a través d'un personatge 3D amb un script de control d'input i física.

Estat: implementada totalment.

2.5.2 Funcionalitat 2

Exploració de l'escenari: El jugador pot recórrer lliurement les diferents habitacions de la casa. L'escenari està dissenyat per guiar l'exploració de manera indirecta, mitjançant la il·luminació, els sons ambients i la disposició dels elements.

Estat: implementada totalment.

2.5.3 Funcionalitat 3

Recollida d'objectius (diners): El jugador ha de localitzar i recollir el dinero de la fiança repartit per diverses zones de la casa. Quan el jugador s'acosta a un objecte recollible i prem la tecla d'interacció, l'objecte desapareix de l'escena i es registra al comptador intern del joc. Quan s'han recollit tots els objectius, s'activa la condició d'escapada.

Estat: implementada totalment.

2.5.4 Funcionalitat 4

Sistema de persecució per part de l'enemic: Un personatge antagonista patrulla la casa i persegueix el jugador quan entra al seu camp de visió o genera soroll. L'enemic utilitza el sistema de navegació de Godot (*NavigationAgent3D*) per moure's per l'escenari i calcular la ruta cap al jugador. Si l'enemic atrapa el jugador, s'activa la pantalla de fi de partida.

Estat: implementada totalment.

2.5.5 Funcionalitat 5

Sistema de so i tensió ambiental: El joc utilitza efectes de so ambientals (crits, sorolls de passos, música de tensió) per generar por i desorientació en el jugador. Els sons varien en funció de la proximitat de l'enemic i de les accions del jugador. S'implementa mitjançant nodes per posicionar els sons en l'espai 3D.

Estat: implementada totalment.

3. Altres capítols

En aquest capítol expliquem de manera detallada com hem desenvolupat el videojoc Money, descrivint les eines que hem utilitzat, les decisions tècniques que hem pres i els motius que ens han portat a prendre-les.

El capítol està dividit en dues grans parts. La primera està dedicada a **Godot**, on descriurem les escenes que formen el projecte, els nodes que les componen i el codi que les fa funcionar. La segona està dedicada a **Blender**, on explicarem com hem modelat i dissenyat l'escenari principal del joc.

En cada apartat hem intentat reflectir no només el resultat final, sinó també el procés seguit: les alternatives que vam considerar, la decisió que vam prendre i per quin motiu, demostrant que cada elecció s'ha fet seguint criteris tècnics i raonats.

3.1 Godot

En aquest apartat explicarem quines escenes formen el nostre joc i les descriurem detalladament, indicant quins nodes utilitzen i què fa el codi que conté cada una, per tal d'entendre com funcionen conjuntament per formar el joc complet.

Com funciona Godot amb escenes

Godot treballa amb escenes, de les quals hi ha diversos tipus. Els principals són:

- **Escena 3D** → per als elements del joc que existeixen dins l'entorn tridimensional.
- **Interfície d'usuari** → per a tots els elements visuals que es mostren per sobre del joc (menús, pantalles, textos, etc.).

En el nostre cas, com que el joc és en 3D, utilitzem **Escenes 3D** per formar el món del joc i **Interfície d'usuari** per a tots els elements de la interfície.

Les escenes que conté el nostre projecte són:

- Casa (Escena 3D) → l'escenari principal del joc, modelat amb Blender.

- Jugador (Escena 3D) → el personatge que controla el jugador en primera persona.
- Enemic (Escena 3D) → el personatge antagonista que persegueix el jugador.
- Introducció (Interfície d'usuari) → la pantalla que presenta el personatge i la història del joc.
- Presentació de l'enemic (Interfície d'usuari) → la pantalla que introdueix l'antagonista abans que aparegui.
- Menú principal (Interfície d'usuari) → la pantalla inicial des d'on es pot iniciar la partida.
- Pantalla de victòria i derrota (Interfície d'usuari) → les pantalles que es mostren quan el jugador guanya o perd.

Cada escena conté diferents nodes, cadascun amb una funció específica. Alguns d'aquests nodes necessiten un script en **GDScript** per funcionar de la manera desitjada. Ho explicarem tot detalladament en els punts següents.

3.1.1 Godot - Moviment del personatge

El moviment del jugador s'ha implementat mitjançant un node **CharacterBody3D**, que és el tipus de node que Godot ofereix específicament per a personatges que es mouen per un escenari amb col·lisions i física.

El script s'executa a través de la funció **_physics_process**, que es crida automàticament en cada fotograma del joc, garantint que el moviment sigui fluid i consistent independentment del rendiment de l'ordinador.

Constants definides

```
const SPEED = 5.0
const JUMP_VELOCITY = 4.5
```

Al principi del codi es defineixen dues constants: **SPEED = 5.0**, que determina la velocitat de desplaçament del jugador, i **JUMP_VELOCITY = 4.5**, que defineix la força amb la qual el jugador salta cap amunt.

Gravetat

```
if not is_on_floor():
    velocity += get_gravity() * delta
```

En cada fotograma es comprova si el jugador està a terra. Si no ho està, s'aplica la gravetat de forma progressiva multiplicada per **delta**, fent que la caiguda sigui natural i consistent.

Salt

```
if Input.is_action_just_pressed("saltar") and is_on_floor():
    velocity.y = JUMP_VELOCITY
```

Si el jugador prem la tecla de salt i en aquell moment es troba a terra, s'aplica una velocitat vertical positiva igual a **JUMP_VELOCITY**, fent que el personatge salti. La condició **is_on_floor()** evita que el jugador pugui saltar en l'aire.

Lectura de les tecles de moviment

```
var input_dir := Input.get_vector("izquierda", "derecha", "adelante", "atras")
```

El codi llegeix en temps real quines tecles de direcció està prement el jugador i les converteix en un vector de dues dimensions que representa la direcció de moviment desitjada.

Càlcul de la direcció real

```
var direction := (transform.basis * Vector3(input_dir.x, 0, input_dir.y)).normalized()
```

La direcció obtinguda de les tecles es combina amb l'orientació actual del personatge mitjançant **transform.basis**, de manera que el jugador sempre es mou cap on està mirant. La funció **normalized()** garanteix que la velocitat sigui la mateixa en totes les direccions.

Aplicació de la velocitat

```
if direction:
    velocity.x = direction.x * SPEED
    velocity.z = direction.z * SPEED
else:
    velocity.x = move_toward(velocity.x, 0, SPEED)
    velocity.z = move_toward(velocity.z, 0, SPEED)
```

Si hi ha una direcció activa, el jugador es desplaça a la velocitat definida per **SPEED**. Si el jugador deixa de prémer les tecles, la funció **move_toward** redueix la velocitat progressivament fins a zero, evitant parades brusques i poc naturals.

Aplicació final del moviment

```
move_and_slide()
```

Aquesta funció aplica tot el moviment calculat i gestiona automàticament les col·lisions amb les parets, el terra i els obstacles de l'escenari, sense necessitat de programar-les manualment.

3.1.2 Godot - Càmera en primera persona i llanterna

Sensibilitat del ratolí

```
1   extends Node3D
2
3   var sensitivity = 0.2
4
```

Es defineix una variable que controla la sensibilitat del ratolí. Com més alt sigui el valor, més ràpid girarà la càmera en moure el ratolí.

Captura del ratolí

```
4
5  func _ready() -> void :
6      Input.set_mouse_mode(Input.MOUSE_MODE_CAPTURED)
7
```

Quan s'inicia l'escena, el ratolí queda capturat dins de la finestra del joc i s'amaga el cursor. Això és imprescindible en un joc en primera persona, ja que el ratolí s'utilitza per mirar al voltant i no ha de sortir de la pantalla.

Activació de la llanterna

```
8  func _process(_delta: float) -> void :
9      if Input.is_action_just_pressed("flashlight"):
10         get_tree().current_scene.get_node("flashlight/sound").play()
11         get_tree().current_scene.get_node("flashlight").visible = !get_tree().current_scene.get_node("flashlight").visible
12
```

Quan el jugador prem la tecla assignada a la llanterna, succeeixen dues coses: primer es reprodueix un so de clic, i després la llanterna canvia el seu estat entre visible i amagada. Això es fa invertint el valor de **visible**, de manera que si estava encesa s'apaga, i si estava apagada s'encén.

Rotació vertical de la càmera

```
12
13 func _input(event: InputEvent) -> void :
14     if event is InputEventMouseMotion:
15         get_parent().rotate_y(deg_to_rad( - event.relative.x * sensitivity))
16         rotate_x(deg_to_rad( - event.relative.y * sensitivity))
17         rotation.x = clamp(rotation.x, deg_to_rad(-90), deg_to_rad(90))
18
```

Quan el jugador mou el ratolí verticalment, la càmara gira sobre l'eix X, simulant que el jugador mira cap amunt o cap avall. La funció **clamp** limita aquest gir entre -90 i 90 graus, evitant que la càmera pugui donar la volta completament i generar un moviment antinatural.

3.1.3 Godot - Enemic

Detecció del jugador

```
83  func chase_player(chasecast: RayCast3D):
84      if chasecast.is_colliding():
85          var hit = chasecast.get_collider()
86          if hit.name == "player":
87              if !chasing:
88                  $monster_enemy / AnimationPlayer.play("walk")
89                  chasing = true
90                  destination = player
```

La funció **chase_player** rep un **RayCast3D**, que és un raig invisible que l'enemic llança des de la seva posició per detectar si hi ha alguna cosa al davant. Si el raig col·lisiona amb un objecte i aquest objecte és el jugador, l'enemic activa l'estat de persecució, reprodueix l'animació de caminar i estableix el jugador com a destinació.

Bucle principal de l'enemic

```
92  func _physics_process(_delta: float) -> void :
93      if !killed:
94          if chasing:
95              kill_player()
96          chase_player($chasecast)
97          chase_player($chasecast2)
98          chase_player($chasecast3)
99          chase_player($chasecast4)
100         chase_player($chasecast5)
```

En cada fotograma, si l'enemic no ha estat eliminat, comprova si està en mode persecució. En cas afirmatiu, primer intenta matar el jugador i després crida la funció **chase_player** cinc vegades amb cinc raigs diferents. Utilitzar múltiples raigs permet a l'enemic detectar el jugador en un angle més ampli, simulant un camp de visió més realista.

Càlcul i aplicació del moviment

```
101    if destination != null:
102        var current_location = global_transform.origin
103        var next_location = $NavigationAgent3D.get_next_path_position()
104        var new_velocity = (next_location - current_location).normalized() * speed
105        $NavigationAgent3D.set_velocity(new_velocity)
```

Si l'enemic té una destinació assignada, calcula la seva posició actual i demana al **NavigationAgent3D** quin és el següent punt del camí òptim fins al jugador. Amb això calcula la velocitat necessària per arribar-hi i l'aplica, fent que l'enemic es mogui de manera fluida evitant obstacles.

3.1.4 Godot - Menú de pausa

Variables inicials

```
1  extends CanvasLayer
2
3  var player_head
4  var environment: Environment
5  var can_play_sound = false
6
```

Es defineixen tres variables: **player_head** guarda una referència a la càmera del jugador, **environment** guarda una referència a l'entorn visual de l'escena, i **can_play_sound** controla si el joc pot reproduir sons en aquell moment, inicialment desactivat.

Inicialització de l'escena

```
7  func _ready() -> void :
8      if get_tree().current_scene.name == "level":
9          player_head = get_tree().current_scene.get_node("player/head")
10         environment = get_tree().current_scene.get_node("WorldEnvironment").environment
```

Quan s'inicia el menú d'ajustos, el codi comprova si l'escena activa és el nivell de joc. Si és així, obté les referències al cap del jugador i a l'entorn mundial, que necessitarà per aplicar alguns dels ajustos gràfics.

Inicialització dels botons i opcions

```
11    set_button_setting("volumetricos", $volumetric_button)
12    set_button_setting("brillo", $glow_button)
13    set_button_setting("iluminacion", $ssil_button)
14    set_button_setting("vsync", $vsync_button)
15    set_option_setting("fps", $fps_button)
16    set_option_setting("antialiasing", $aa_button)
17    set_option_setting("sombras", $shadow_button)
18    set_option_setting("modo_ventana", $window_mode)
19    set_slider_setting("escala_3d", $scaling_slider)
20    set_slider_setting("volumen_general", $volume_slider)
21    set_slider_setting("volumen_efectos", $sfx_slider)
22    set_slider_setting("volumen_musica", $music_slider)
23    set_slider_setting("sensibilidad", $look_slider)
24    set_volumetrics($volumetric_button.button_pressed)
25    set_glow($glow_button.button_pressed)
26    set_aa($aa_button.selected)
```

Es carreguen tots els ajustos guardats i s'apliquen als elements visuals del menú. Hi ha tres tipus d'elements:

- **Botons** → per a opcions que s'activen o desactiven (volumètrics, brillantor, il·luminació, vsync).
- **Opcions desplegables** → per a opcions amb múltiples valors (fps, antialiasing, ombres, mode de finestra).
- **Sliders** → per a valors numèrics ajustables (escala 3D, volum general, volum d'efectes, volum de música i sensibilitat del ratolí).

3.6 Godot - Interactuar con las puertas

Detecció d'interacció amb la porta

```
1  extends RayCast3D
2
3  func _physics_process(delta: float) -> void:
4      if is_colliding():
5          var interactuar = get_collider()
6          if interactuar.name == "puerta":
7              if Input.is_action_just_pressed("interactuar"):
8                  interactuar.get_parent().get_parent().get_parent().toggle_door()
```

Aquest script s'assigna a un RayCast3D del jugador. En cada fotograma comprova si el raig que emet el jugador cap endavant està tocant una porta i, si el jugador prem la tecla d'interacció, crida la funció toggle_door() per obrir-la o tancar-la.

Animació de la porta

```gdscript
extends Node3D

var abierto = false

func toggle_door():
    if $AnimationPlayer.current_animation != "abierto" and $AnimationPlayer.current_animation != "cerrar":
        abierto = !abierto
        if !abierto:
            $AnimationPlayer.play("cerrar")
        if abierto:
            $AnimationPlayer.play("abrir")
```

Aquest script s'assigna al node de la porta. La variable **abierto** guarda si la porta està oberta o tancada, començant tancada per defecte. Quan es crida **toggle_door()**, comprova que no hi hagi cap animació en curs per evitar errors, inverteix l'estat de la porta i reprodueix l'animació corresponent: **"abrir"** o **"cerrar"**.

3.2 Blender

Blender és un programa de modelatge 3D gratuït i de codi obert àmpliament utilitzat en el desenvolupament de videojocs, cinema i disseny. En el nostre projecte l'hem utilitzat exclusivament per al modelatge dels elements 3D que formen l'escenari del joc, com ara la casa, els mobles i els objectes decoratius.

Un cop finalitzat el modelatge de cada element, l'hem exportat en format .glb, que és el format estàndard recomanat per Godot per importar models 3D. Aquest format conserva la geometria, els materials i les textures del model en un únic fitxer, facilitant la seva integració al projecte.

Dins de Godot, cada model importat s'ha convertit en una escena independent a la qual s'hi han afegit els nodes necessaris per al seu funcionament dins del joc, com ara col·lisions, materials i punts d'ancoratge.

3.2.1 Blender - Modelatge d'escenaris

Per al disseny i modelatge de l'escenari principal del joc hem utilitzat **Blender**, un programa de modelatge 3D gratuït i de codi obert. Aquesta eina ens ha permès crear la casa on transcorre tota l'acció del joc de manera totalment personalitzada, adaptant-la a les necessitats i l'ambientació del projecte.

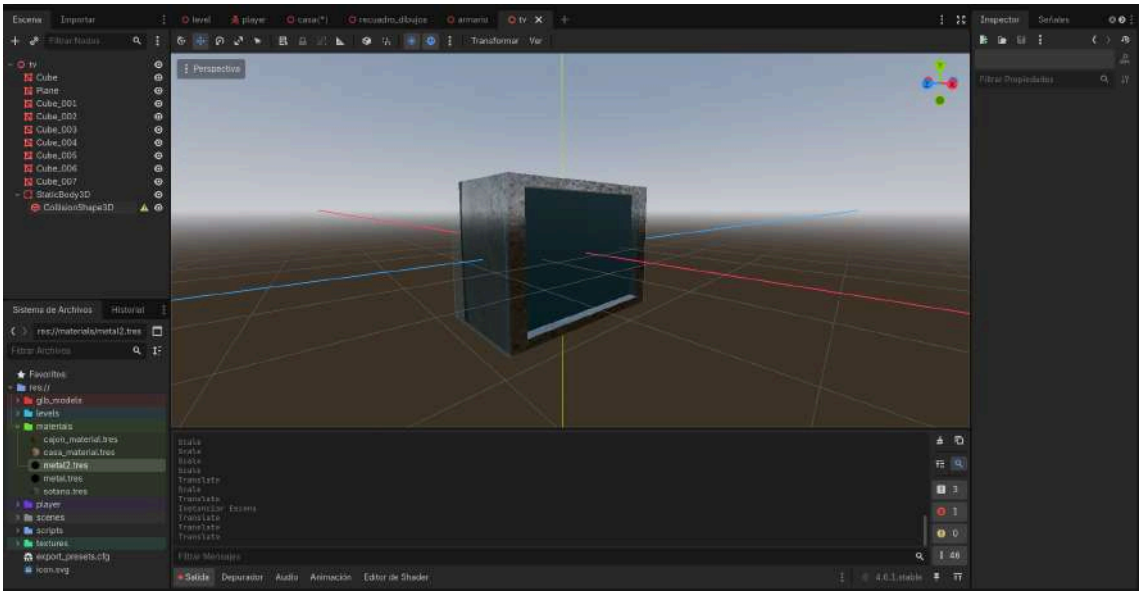
En els punts següents explicarem detalladament com hem construït l'escenari, quines decisions hem pres durant el procés de modelatge, com hem aplicat les textures i finalment com hem integrat tot el resultat dins de Godot per formar l'escenari jugable.

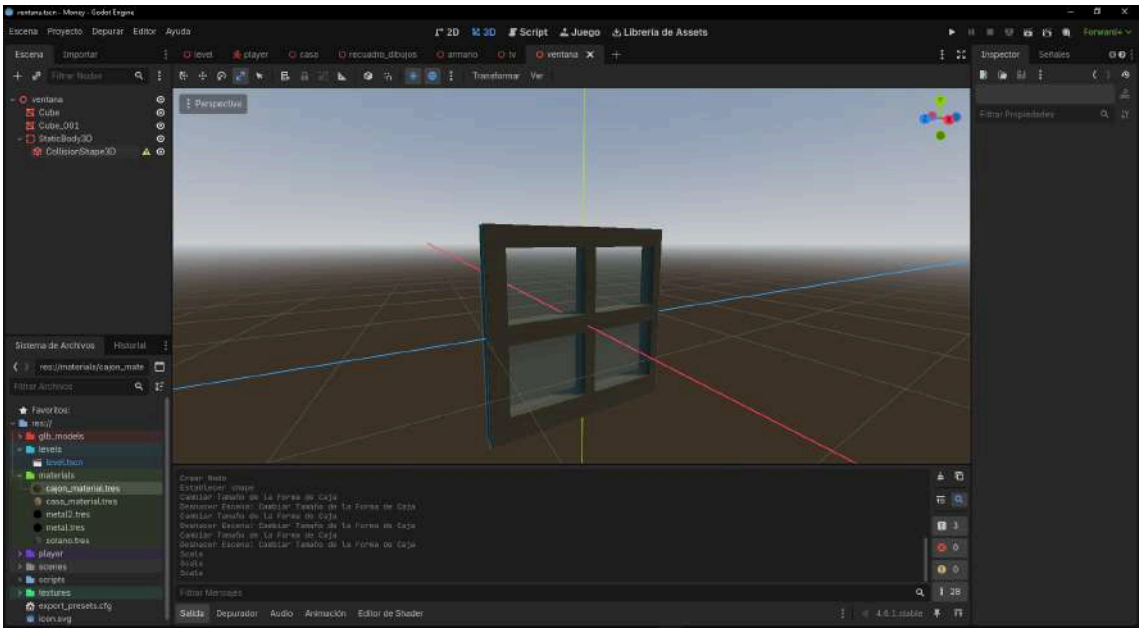
3.2.2 Blender - Modelatge d’objecte

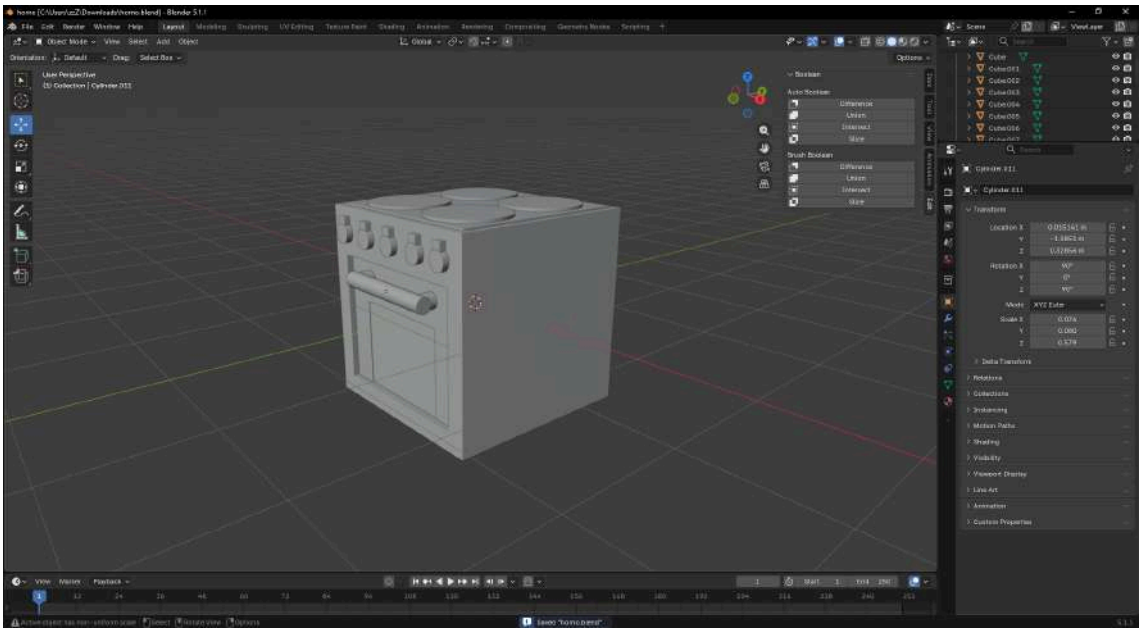
A més de la casa, també hem modelat els objectes i mobles que formen part de l'escenari del joc, com ara finestres, televisors, sofàs i llànternes, entre d'altres. Tots aquests elements s'han creat a Blender utilitzant formes bàsiques (**Cube**, **Plane**) que s'han combinat i modificat fins a obtenir la forma desitjada.

Les textures aplicades als objectes són senzilles, utilitzant materials metàl·lics, de fusta i foscos, coherents amb l'ambientació de terror del joc. Com que l'escenari és majoritàriament fosc, no ha estat necessari invertir temps en textures molt detallades.

Un cop modelats, els objectes s'han exportat en format **.glb** i importat a Godot com a escenes independents. Dins de Godot, a cada objecte se li ha afegit un node **StaticBody3D** amb un **CollisionShape3D** per tal que el jugador no pugui travessar-los i interactuïn correctament amb la física del joc.







3.2.3 Blender - Modelatge d'enemic

Per al personatge antagonista del joc no hem creat un model propi, sinó que hem descarregat un model ja existent de la plataforma **Sketchfab**, que ofereix models 3D gratuïts creats per la comunitat. Aquesta decisió ens ha permès estalviar temps en el modelatge i centrar-nos en la programació del comportament de l'enemic.

El model escollit és una figura humanoide fosca amb ulls blaus lluminosos i màscara de gas, que encaixa perfectament amb l'ambientació de terror del joc i genera una sensació d'inquietud en el jugador des del primer moment que l'avista.

Un cop descarregat, el model s'ha importat a Godot conservant les seves animacions originals, com ara caminar i atacar. Aquestes animacions s'activen mitjançant codi segons el comportament de l'enemic en cada moment, tal com s'ha explicat en l'apartat del script d'intel·ligència artificial.

Decisió tècnica

Es va decidir utilitzar un model extern en lloc de crear-ne un de propi ja que el modelatge de personatges amb animacions és un procés complex que requeriria molt de temps. Utilitzar un recurs ja existent i adaptar-lo al nostre joc és una pràctica habitual en el desenvolupament de videojocs, especialment en projectes amb temps limitat.

4 Conclusions

4.1 Conclusions generals del projecte

Aquest projecte ha estat un gran repte per als dos. Des del principi volíem fer un joc molt gran i amb moltes funcionalitats, però aviat ens vam adonar que era massa per al temps que teníem. Vam haver de canviar la idea i fer-la més senzilla i realista. Això ens va ensenyar que en el desenvolupament de programari no sempre es pot fer el que es vol, sinó el que es pot, és adaptar-se i fer-lo bé amb els recursos que tens.

Un altre dels reptes grans va ser aprendre a programar amb Godot partint de zero. Entendre com funcionen els nodes, les variables i els scripts, i com tot això s'uneix per formar un joc va costar molt. Hi va haver molts errors i moltes hores intentant fer funcionar les coses. Però al final tot va anar sortint.

A nivell acadèmic hem après a usar eines reals com Godot i GDScript, a gestionar la documentació d'un projecte i a fer servir la intel·ligència artificial com a ajuda. A nivell personal hem après que treballar en grup no és tan fàcil com sembla, que cal repartir bé les tasques i que darrere de qualsevol joc o programa hi ha molta feina que no es veu.

Estem molt contents amb el resultat. *Money* és el primer joc jugable que hem creat nosaltres dos des de zero, i això és una cosa de la qual ens sentim molt orgullosos.

4.2 Consecució dels objectius

Aprendre a utilitzar Godot i adquirir coneixements bàsics de GDScript. Aquest objectiu s'ha assolit. Al principi cap dels dos teníem experiència amb Godot ni amb GDScript, però al llarg del projecte hem après a crear escenes, usar nodes, programar scripts i entendre com funciona el motor per dins. Ha estat el repte més gran del projecte, però també el que més hem après.

Desenvolupar un joc jugable i complet. Aquest objectiu s'ha assolit. *Money* és un joc jugable de principi a fi, amb un escenari, un objectiu clar per al jugador, un enemic i les pantalles de menú, mort i victòria. Estem contents amb el resultat tenint en compte que és el primer joc que hem fet mai.

Explorar l'ús de la intel·ligència artificial en la programació i el desenvolupament. Aquest objectiu s'ha assolit. Hem fet servir ChatGPT, Claude i Qwen com a eines de suport durant tot el projecte, tant per resoldre dubtes de programació com per ajudar en la redacció de la documentació. Cada eina té punts forts diferents i combinar-les ha estat molt útil.

4.3 Valoració de la metodologia i planificació

La metodologia que vam seguir va ser molt flexible, sense seguir cap framework estricte. Vam anar treballant al nostre ritme, adaptant-nos als canvis que anaven sorgint i sense pressionar-nos massa. En general creiem que aquesta forma de treballar ha estat positiva, ja que ens ha permès avançar sense arribar a cansar-nos del projecte ni fer-lo pesat. Quan alguna cosa no funcionava, la canviàvem sense bloquejar-nos, i això ens ha ajudat a mantenir les ganes de continuar durant tot el trimestre.

Tot i això, si haguéssim de valorar-ho de forma crítica, hi ha aspectes que hauríem fet diferent. El canvi d'idea a mig trimestre va suposar perdre temps que no teníem previst perdre, i això va fer que al final del projecte tinguéssim menys marge per polir els detalls del joc. Amb una planificació una mica més estricta des del principi, especialment en els terminis de cada fase, hauríem pogut arribar al resultat final amb més temps per millorar i acabar de treballar els aspectes que han quedat menys desenvolupats.

En resum, la metodologia flexible ha funcionat bé per mantenir la motivació i adaptar-nos als imprevistos, però en un projecte futur intentaríem combinar-la amb uns terminis mínims per a cada tasca, per assegurar-nos de tenir prou temps per polir el resultat final.

4.4 Visió de futur

En general, hem pogut implementar tot el que teníem previst per a aquest projecte. Algunes coses han quedat menys polides del que ens hagués agradat, ja que el temps no ha donat per acabar-ho tot amb el nivell de detall que volíem, però les funcionalitats principals estan totes presents i el joc és jugable de principi a fi.

Pel que fa a la continuïtat del projecte, hem decidit no seguir desenvolupant *Money* en un futur. No és un projecte amb el qual vulguem obtenir cap benefici econòmic ni professional, sinó que el veiem com el que és: el nostre primer joc, fet des de zero, amb tot el que hem après durant el cicle. Per a nosaltres té un valor especial precisament per això, i preferim deixar-lo tal com està, com el primer trofeu del nostre camí en el desenvolupament de programari.

Si en un futur decidíssim fer un nou projecte, ho faríem amb una idea nova, aplicant tot el que hem après en aquest, i amb més experiència i coneixements per poder arribar a un resultat més treballat i complet.

5. Glossari

Godot Engine: Motor de desenvolupament de videojocs gratuït i de codi obert. És l'eina principal que hem utilitzat per crear el joc *Money*, tant per construir l'escenari 3D com per programar totes les mecàniques.

GDScript: Llenguatge de programació propi de Godot, amb una sintaxi similar a Python. S'ha utilitzat per programar tota la lògica del joc, com el moviment del jugador, el comportament de l'enemic i les condicions de victòria i derrota.

Node: Element bàsic de Godot. Cada escena del joc està formada per nodes, i cadascun té una funció concreta dins del projecte, com gestionar col·lisions, reproduir animacions o controlar la càmera.

Escena: Conjunt de nodes relacionats entre si que formen una part del joc. En el nostre projecte, cada element principal com el jugador, l'enemic o els menús és una escena independent.

Script: Arxiu de codi escrit en GDScript que s'associa a un node per definir el seu comportament. Els scripts controlen tot el que passa dins del joc de forma dinàmica.

Primera persona: Perspectiva de joc en la qual el jugador veu l'escenari des dels ulls del personatge que controla. És la perspectiva principal del joc *Money*.

Col·lisió: Mecanisme que impedeix que els elements del joc es travessin entre ells. S'utilitza perquè el jugador no pugui atravessar les parets, el terra o els objectes de l'escenari.

Llanterna: Única font de llum disponible per al jugador durant la partida. Sense ella, l'escenari és completament fosc i és impossible navegar per la casa.

Enemic: Personatge controlat pel joc que persegueix el jugador en determinades zones de la casa. Si atrapa el jugador, s'activa la condició de derrota.

Game over: Pantalla que apareix quan el jugador perd la partida, és a dir, quan l'enemic l'atrapa. Ofereix la possibilitat de tornar a intentar-ho des del principi.

IA (Intel·ligència Artificial): En el context d'aquest projecte, fa referència a les eines d'intel·ligència artificial com ChatGPT, Claude i Qwen, utilitzades com a suport per a la programació i la documentació.

GitHub: Plataforma de control de versions que permet emmagatzemar el codi d'un projecte i gestionar els canvis realitzats al llarg del temps.

SMX: Acrònim de Sistemes Microinformàtics i Xarxes. És el cicle formatiu de grau mitjà en el qual s'emmarca aquest projecte de síntesi.

Itch.io: Plataforma en línia de distribució de videojocs independents. És on vam trobar el joc indie que ens va servir d'inspiració per desenvolupar *Money*.

Sprite: Imatge o animació 2D que representa un personatge o element dins d'un videojoc. En el nostre cas, com el joc és en 3D, no fem servir sprites sinó models tridimensionals.

Motor de videojocs: Programari que proporciona les eines necessàries per desenvolupar un videojoc, com la gestió de físiques, la renderització de gràfics o el control de l'àudio.

Metodologia àgil: Forma de treballar en el desenvolupament de programari basada en la flexibilitat, l'adaptació als canvis i l'avanç per parts petites en lloc de seguir un pla rígid des del principi.

6. Bibliografia

Informació de Godot

Godot Engine Documentation – Introducció a la programació 3D en Godot. https://docs.godotengine.org/en/stable/tutorials/3d/index.html —

Godot Engine Documentation – Bases d'àudio i AudioStreamPlayer. https://docs.godotengine.org/en/stable/tutorials/audio/index.html —

Godot Engine Documentation – Física i col·lisions en Godot. https://docs.godotengine.org/en/stable/tutorials/physics/index.html —

Godot Engine Documentation – Sistema d'animacions i AnimationPlayer. https://docs.godotengine.org/en/stable/tutorials/animation/index.html —

Tutorials en vídeo (YouTube)

Tutorial de Blender per a modelatge 3D bàsic. YouTube. https://www.youtube.com/watch?v=O-tV7uBf5LI —

Tutorial de jugador en primera persona amb Godot 4. YouTube. https://www.youtube.com/watch?v=3xYlwEQWLps&list=PLvn1rwek2APGvw2w1xNA0gzBGsXdum714 —

Tutorial d'enemics amb sistema de persecució en Godot 4. YouTube. https://www.youtube.com/watch?v=pGoK4nhpSsk&list=PLpZrYr_gSkThEmjuOyrp2lwrzQFoEz5BO&index=3 —

Tutorial de menú principal en Godot 4. YouTube. https://www.youtube.com/watch?v=hyTu5ZTR5Yk —

Tutorial de como hacer una casa en Blender. https://www.youtube.com/watch?v=dRutCkQiq9c —

Intel·ligència artificial

Grok AI – Eina d'intel·ligència artificial utilitzada com a suport durant el desenvolupament. https://x.ai —

Claude AI – Eina d'intel·ligència artificial utilitzada com a suport durant el desenvolupament. https://claude.ai —

Kling AI – Eina de generació de vídeo amb IA. https://kling.ai —

Pollo AI – Eina de generació de vídeo amb IA. https://pollo.ai —

7. Annexos

7.1 Codi del joc

El nostre codi es basa en una sèrie de nodes, és a dir, cada element del joc té el seu propi script independent que gestiona una funció concreta. En aquest annex incloem tot el codi programat en format de captures, organitzat per scripts, perquè quedi constància de tot el treball de programació realitzat durant el desenvolupament del projecte, no té una correlació unicament es inforamció adicional.

checkpoint_trigger.gd

```
1   extends Area3D
2
3   @export var checkpoint_id := 0
4   var triggered := false
5
6   func _ready() -> void:
7       body_entered.connect(_on_body_entered)
8
9   func _on_body_entered(body: Node) -> void:
10      print("ENTRÓ EN CHECKPOINT:", checkpoint_id)
11      if triggered:
12          return
13
14      if body.name == "player":
15          triggered = true
16          print("Checkpoint activado:", checkpoint_id)
```

code_paper.gd

```gdscript
extends RigidBody3D

@export var positions: Array[Node3D]
@export var door: Node3D

@onready var rng := RandomNumberGenerator.new()
@onready var pickup_sound := get_tree().current_scene.get_node_or_null("key_pickup")

var pos_obj: Node3D = null
var picked := false


func _ready() -> void:
    if positions.size() == 0:
        print("ERROR: no hay posiciones")
        return

    var chance := rng.randi_range(0, positions.size() - 1)
    global_transform.origin = positions[chance].global_transform.origin

    print("Key spawn:", chance)

    # SI el nombre no coincide, ocúltalo (debug antiguo)
    if name != "key":
        visible = false


func hit_obj(body: Node3D) -> void:
    pos_obj = body
    freeze = true


func pickup_key() -> void:
    if picked:
        return

    picked = true

    print("KEY PICKED")
```

```
41          # sonido seguro
42          if pickup_sound:
43              pickup_sound.play()
44
45          # desbloquear puerta de forma segura
46          if is_instance_valid(door):
47              if "locked" in door:
48                  door.locked = false
49              elif door.has_method("unlock"):
50                  door.unlock()
51          else:
52              print("WARNING: door no asignada")
53
54          queue_free()
55
56
57      func _physics_process(_delta: float) -> void:
58          if pos_obj:
59              global_transform.origin = pos_obj.global_transform.origin
```

death.gd

```
1   extends Control
2
3
4
5   func _ready() -> void :
6       $AnimationPlayer.play("death")
7       await get_tree().create_timer(5.5, true).timeout
8       get_tree().change_scene_to_file("res://levels/level.tscn")
```

destination.gd

```
1   extends Node3D
2
3   @onready var rng = RandomNumberGenerator.new()
4
5   func enter_trigger(body):
6       if body.name == "enemy" and body.destination == self and !body.chasing:
7           body.stop_enemy()
8           await get_tree().create_timer(rng.randf_range(1.0, 10.0), false).timeout
9           if !body.chasing:
10              body.pick_destination(body.destination_value)
```

door.gd

```
1   extends Node3D
2
3   var opened = false
4   @export var locked = false
5   @export var trapdoor: bool
6
7   func ai_enable_door(body):
8       if body.name == "enemy" and !locked and $AnimationPlayer.current_animation != "open" and !opened:
9           opened = true
10          if has_node("open"):
11              $open.play()
12          if has_node("hinge/open"):
13              $hinge / open.play()
14          $AnimationPlayer.play("open")
15
16  func ai_disable_door(body):
17      if body.name == "enemy" and !locked and $AnimationPlayer.current_animation != "open" and opened:
18          opened = false
19          if has_node("close"):
20              $close.play()
21          if has_node("hinge/close"):
22              $hinge / close.play()
23          $AnimationPlayer.play_backwards("open")
24
25  func toggle_door():
26      if $AnimationPlayer.current_animation != "open" and !locked:
27          opened = !opened
28          if !opened:
29              if has_node("close"):
30                  $close.play()
31              if has_node("hinge/close"):
32                  $hinge / close.play()
33              $AnimationPlayer.play_backwards("open")
34          if opened:
35              if trapdoor:
36                  locked = true
37              if has_node("open"):
38                  $open.play()
39              if has_node("hinge/open"):
40                  $hinge / open.play()
41              $AnimationPlayer.play("open")
```

door_bell.gd

```gdscript
extends Node3D

var times_rung = 0
#@export var door: Node3D
@onready var ui = get_tree().current_scene.get_node("player/player_ui")

# func _ready() -> void :
    # var save = FileAccess.open("user://checkpoint.skibidi", FileAccess.READ)
    # if !save:
        # door.locked = true
    # elif save:
        # door.locked = false
        # times_rung = 2
        # save.close()

func ring_bell():
    if $AnimationPlayer.current_animation != "press" and times_rung < 2:
        $button.play()
        times_rung += 1
        $AnimationPlayer.play("press")
        await get_tree().create_timer(0.5, false).timeout
        $door_bell2.play()
        if times_rung >= 2:
            await get_tree().create_timer(4.0, false).timeout
            ui.set_task("Entra en la casa")
            # door.locked = false
```

ending.gd

```gdscript
extends Area3D

var triggered := false

func _ready() -> void:
    body_entered.connect(_on_body_entered)

func _on_body_entered(_body: Node) -> void:
    if triggered:
        return

    if _body.name == "player":
        triggered = true
        call_deferred("_ending")


func _ending() -> void:
    var tree = get_tree()
    tree.paused = true
    Input.set_mouse_mode(Input.MOUSE_MODE_VISIBLE)

    print("Conseguiste escapar... Por ahora...")

    await tree.create_timer(3.0).timeout
    tree.quit()
```

enemy.gd

```
extends CharacterBody3D

@export var patrol_destinations: Array[Node3D]
@onready var player = get_tree().current_scene.get_node("player")
var speed = 3.0
@onready var rng = RandomNumberGenerator.new()
var destination
var chasing = false
var destination_value
var killed = false
var chase_timer = 0.0
var idle = false
@export var footstep_sounds: Array[AudioStream]
@onready var ambient_music = get_tree().current_scene.get_node("music")
@onready var chase_music = get_tree().current_scene.get_node("chase_music")

func footsteps():
    if !chasing and $feet.pitch_scale != 1.0:
        $feet.pitch_scale = 1.0
    elif chasing and $feet.pitch_scale != 1.5:
        $feet.pitch_scale = 1.5
    if !$feet.playing:
        $feet.stream = footstep_sounds[rng.randi_range(0, footstep_sounds.size() - 1)]
        $feet.play()

func _ready() -> void :
    $monster_enemy / AnimationPlayer.play("idle")
    pick_destination()

func stop_enemy():
    $monster_enemy / AnimationPlayer.play("idle")
    idle = true
    speed = 0

```

```
35  func _process(delta: float) -> void :
36      if !killed:
37          if speed > 0:
38              footsteps()
39          if !chasing:
40              if chase_music.playing:
41                  chase_music.stop()
42                  ambient_music.play()
43              if $monster_enemy / AnimationPlayer.speed_scale != 1:
44                  $monster_enemy / AnimationPlayer.speed_scale = 1
45              if speed != 1.0 and !idle:
46                  speed = 1.0
47          if chasing:
48              if !chase_music.playing:
49                  chase_music.play()
50                  ambient_music.stop()
51              if $monster_enemy / AnimationPlayer.speed_scale != 2:
52                  $monster_enemy / AnimationPlayer.speed_scale = 2
53              if speed != 2.5:
54                  speed = 2.5
55              if chase_timer < 15.0:
56                  chase_timer += 1 * delta
57              else:
58                  chase_timer = 0
59                  if $killcast / killcast.enabled:
60                      $killcast / killcast.enabled = false
61                  chasing = false
62                  pick_destination()
63          if destination != null and !idle:
64              var look_dir = lerp_angle(deg_to_rad(global_rotation_degrees.y), atan2( - velocity.x, - velocity.z), 0.5)
65              global_rotation_degrees.y = rad_to_deg(look_dir)
66              update_target_location()
67
```

```
68      func kill_player():
69          if !$killcast / killcast.enabled:
70              $killcast / killcast.enabled = true
71          $killcast.look_at(player.global_transform.origin)
72          if $killcast / killcast.is_colliding():
73              var hit = $killcast / killcast.get_collider()
74              if hit.name == "player" and !killed:
75                  killed = true
76                  $jumpscare_cam.current = true
77                  $monster_enemy / AnimationPlayer.play("jumpscare")
78                  $monster_enemy / AnimationPlayer.speed_scale = 2
79                  player.process_mode = Node.PROCESS_MODE_DISABLED
80                  await get_tree().create_timer(2.0, false).timeout
81                  get_tree().change_scene_to_file("res://escenas/muerte.tscn")
82
83      func chase_player(chasecast: RayCast3D):
84          if chasecast.is_colliding():
85              var hit = chasecast.get_collider()
86              if hit.name == "player":
87                  if !chasing:
88                      $monster_enemy / AnimationPlayer.play("walk")
89                      chasing = true
90                      destination = player
91
92      func _physics_process(_delta: float) -> void :
93          if !killed:
94              if chasing:
95                  kill_player()
96              chase_player($chasecast)
97              chase_player($chasecast2)
98              chase_player($chasecast3)
99              chase_player($chasecast4)
100             chase_player($chasecast5)
101             if destination != null:
102                 var current_location = global_transform.origin
103                 var next_location = $NavigationAgent3D.get_next_path_position()
104                 var new_velocity = (next_location - current_location).normalized() * speed
105                 $NavigationAgent3D.set_velocity(new_velocity)
106
107
```

```
109     func pick_destination(dont_choose = null):
110         if !chasing and !killed:
111             $monster_enemy / AnimationPlayer.play("walk")
112             speed = 2.0
113             idle = false
114             var num = rng.randi_range(0, patrol_destinations.size() - 1)
115             destination_value = num
116             destination = patrol_destinations[num]
117             if destination != null and dont_choose != null and destination == patrol_destinations[dont_choose]:
118                 if dont_choose < 1:
119                     destination = patrol_destinations[dont_choose + 1]
120                 if dont_choose > 0 and dont_choose <= patrol_destinations.size() - 1:
121                     destination = patrol_destinations[dont_choose - 1]
122 
123     func update_target_location():
124         if process_mode == Node.PROCESS_MODE_INHERIT and !killed:
125             $NavigationAgent3D.target_position = destination.global_transform.origin
126 
127     func compute_velocity(safe_velocity: Vector3) -> void :
128         if process_mode == Node.PROCESS_MODE_INHERIT and destination != null and !killed:
129             velocity = velocity.move_toward(safe_velocity, 0.25)
130             move_and_slide()
```

flashlight.gd

```
1     extends SpotLight3D
2 
3     @onready var player = get_tree().current_scene.get_node("player/head")
4 
5 
6     func _process(_delta: float) -> void :
7         global_transform.origin = player.global_transform.origin
8 
9     func _physics_process(_delta: float) -> void :
10        var rot_y = lerp_angle(global_rotation.y, player.global_rotation.y, 0.2)
11        var rot_x = lerp_angle(global_rotation.x, player.global_rotation.x, 0.2)
12        var rot_z = lerp_angle(global_rotation.z, player.global_rotation.z, 0.2)
13        global_rotation.y = rot_y
14        global_rotation.x = rot_x
15        global_rotation.z = rot_z
```

lamp.gd

```gdscript
extends Node3D

@export var on := false
@export var on_mat: StandardMaterial3D
@export var off_mat: StandardMaterial3D
@export var light_color: Color

@onready var reflection_probe: ReflectionProbe = get_node_or_null("ReflectionProbe")
@onready var omni_light: OmniLight3D = $OmniLight3D
@onready var lamp_head: MeshInstance3D = $lamp_head
@onready var sound: AudioStreamPlayer3D = $sound


func _ready() -> void:
    omni_light.light_color = light_color

    _apply_state()


func _apply_state() -> void:
    lamp_head.material_override = on_mat if on else off_mat
    omni_light.visible = on


func toggle_light() -> void:
    on = !on

    # sonido
    sound.pitch_scale = 1.0 if on else 0.8
    sound.play()

    _apply_state()

    # ❌ NO CRASH SAFE (solo si existe)
    if reflection_probe:
        # aquí NO tocamos origin_offset porque no es fiable en Godot 4
        reflection_probe.update_mode = ReflectionProbe.UPDATE_ONCE
```

light_switch.gd

```
1   extends Node3D
2
3   @export var on = false
4   @export var on_mat: StandardMaterial3D
5   @export var off_mat: StandardMaterial3D
6   @export var light_bulb: Node3D
7   @export var reflection_probe: ReflectionProbe
8
9   func _ready():
10      if !on:
11          light_bulb.get_node("light").material_override = off_mat
12      if on:
13          light_bulb.get_node("light").material_override = on_mat
14      light_bulb.get_node("OmniLight3D").visible = on
15
16  func toggle_light():
17      on = !on
18      if on:
19          $sound.pitch_scale = 1.0
20          $on.visible = true
21          $off.visible = false
22          light_bulb.get_node("light").material_override = on_mat
23      elif !on:
24          $sound.pitch_scale = 0.8
25          $on.visible = false
26          $off.visible = true
27          light_bulb.get_node("light").material_override = off_mat
28      $sound.play()
29      light_bulb.get_node("OmniLight3D").visible = on
30      await get_tree().create_timer(0.05, false).timeout
31      if reflection_probe:
32          if !on:
33              reflection_probe.origin_offset.x += 0.01
34          else:
35              reflection_probe.origin_offset.x -= 0.01
```

main_menu.gd

```gdscript
extends Control

func _ready() -> void :
    Input.set_mouse_mode(Input.MOUSE_MODE_VISIBLE)
# var save = FileAccess.open("user://checkpoint.skibidi", FileAccess.READ)
    # if save:
        # $main / Button2.disabled = false
        # save.close()

func continue_game():
    $interact.play()
    await get_tree().create_timer(0.5, true).timeout
    get_tree().change_scene_to_file("res://levels/level.tscn")

func new_game():
    $interact.play()
    $main.visible = false
    $are_you_sure.visible = true

func play_hover():
    $hover.play()

func confirm_yes():
    $interact.play()
    # DirAccess.remove_absolute("user://checkpoint.skibidi")
    # await get_tree().create_timer(0.5, true).timeout
    # get_tree().change_scene_to_file("res://ui/story.tscn")
```

```
28
29      func confirm_no():
30          $interact.play()
31          $main.visible = true
32          $are_you_sure.visible = false
33          $settings.visible = false
34          $controls.visible = false
35          $credits.visible = false
36
37      func quit_game():
38          $interact.play()
39          await get_tree().create_timer(0.5, true).timeout
40          get_tree().quit()
41
42      func open_credits():
43          $interact.play()
44          $main.visible = false
45          $credits.visible = true
46
47      func open_settings():
48          $interact.play()
49          $settings.visible = true
50          $main.visible = false
51
52      func open_controls():
53          $interact.play()
54          $main.visible = false
55          $controls.visible = true
```

money.gd

```
1   extends Node3D
2
3   @onready var ui = get_tree().current_scene.get_node("player/player_ui")
4   var interacted = false
5
6   func interact():
7       if !interacted:
8           interacted = true
9           visible = false
10          $pickup.play()
11          ui.set_task("Escapa de la casa")
12          get_tree().current_scene.get_node("scene_change_trigger/CollisionShape3D").disabled = false
13          await get_tree().create_timer(0.52, false).timeout
14          queue_free()
```

monster_enemy.gd

```
1   extends Node3D
2
3   func stab_sound():
4       $body / upper_r_arm / lower_arm / knife / stab_sound.play()
```

paintings.gd

```
1   extends Node3D
2
3   @export var painting: StandardMaterial3D
4
5
6   func _ready() -> void :
7       $Plane.material_override = painting
```

settings.gd

```
1   extends CanvasLayer
2
3   var player_head
4   var environment: Environment
5   var can_play_sound = false
6
7   func _ready() -> void :
8       if get_tree().current_scene.name == "level":
9           player_head = get_tree().current_scene.get_node("player/head")
10          environment = get_tree().current_scene.get_node("WorldEnvironment").environment
11      set_button_setting("volumetricos", $volumetric_button)
12      set_button_setting("brillo", $glow_button)
13      set_button_setting("iluminacion", $ssil_button)
14      set_button_setting("vsync", $vsync_button)
15      set_option_setting("fps", $fps_button)
16      set_option_setting("antialiasing", $aa_button)
17      set_option_setting("sombras", $shadow_button)
18      set_option_setting("modo_ventana", $window_mode)
19      set_slider_setting("escala_3d", $scaling_slider)
20      set_slider_setting("volumen_general", $volume_slider)
21      set_slider_setting("volumen_efectos", $sfx_slider)
22      set_slider_setting("volumen_musica", $music_slider)
23      set_slider_setting("sensibilidad", $look_slider)
24      set_volumetrics($volumetric_button.button_pressed)
25      set_glow($glow_button.button_pressed)
26      set_aa($aa_button.selected)
27      set_master_volume($volume_slider.value)
28      set_sfx_volume($sfx_slider.value)
29      set_music_volume($music_slider.value)
30      set_look_speed($look_slider.value)
31      set_shadows($shadow_button.selected)
32      set_window_mode($window_mode.selected)
33      set_fps_cap($fps_button.selected)
34      set_vsync($vsync_button.button_pressed)
35      set_ssil($ssil_button.button_pressed)
36      scale_3d($scaling_slider.value)
37      can_play_sound = true
38
39  func set_button_setting(save_name, button):
40      var loadd = FileAccess.open("user://" + save_name + ".ajustes", FileAccess.READ)
41      if loadd:
42          if loadd.get_as_text() == "false":
43              button.button_pressed = false
44          elif loadd.get_as_text() == "true":
45              button.button_pressed = true
46          loadd.close()
47
```

```
48      func set_option_setting(save_name, button):
49          var loadd = FileAccess.open("user://" + save_name + ".ajustes", FileAccess.READ)
50          if loadd:
51              button.selected = int(loadd.get_as_text())
52              loadd.close()
53
54      func set_slider_setting(save_name, slider):
55          var loadd = FileAccess.open("user://" + save_name + ".ajustes", FileAccess.READ)
56          if loadd:
57              slider.value = float(loadd.get_as_text())
58              loadd.close()
59
60      func save_setting(save_name: String, value):
61          var save = FileAccess.open("user://" + save_name + ".skibiditoilet", FileAccess.WRITE)
62          save.store_string(str(value))
63          save.close()
64
65      func set_volumetrics(toggled):
66          save_setting("volumetricos", toggled)
67          if can_play_sound:
68              get_parent().get_node("interact").play()
69          if environment != null:
70              environment.volumetric_fog_enabled = toggled
71
72      func set_glow(toggled):
73          save_setting("brillo", toggled)
74          if can_play_sound:
75              get_parent().get_node("interact").play()
76          if environment != null:
77              environment.glow_enabled = toggled
78
79      func set_ssil(toggled):
80          save_setting("iluminacion", toggled)
81          if can_play_sound:
82              get_parent().get_node("interact").play()
83          if environment != null:
84              environment.ssil_enabled = toggled
85
86      func set_vsync(toggled):
87          save_setting("vsync", toggled)
88          if can_play_sound:
89              get_parent().get_node("interact").play()
90          if !toggled:
91              DisplayServer.window_set_vsync_mode(DisplayServer.VSYNC_DISABLED)
92          elif toggled:
93              DisplayServer.window_set_vsync_mode(DisplayServer.VSYNC_ENABLED)
94
```

```
95  func set_fps_cap(index):
96      save_setting("fps", index)
97      if can_play_sound:
98          get_parent().get_node("interact").play()
99      if index == 0:
100         Engine.max_fps = 30
101     elif index == 1:
102         Engine.max_fps = 60
103     elif index == 2:
104         Engine.max_fps = 0
105
106 func set_aa(index):
107     save_setting("antialiasing", index)
108     if can_play_sound:
109         get_parent().get_node("interact").play()
110     if index == 0:
111         get_viewport().msaa_3d = Viewport.MSAA_DISABLED
112         get_viewport().screen_space_aa = Viewport.SCREEN_SPACE_AA_DISABLED
113         get_viewport().use_taa = false
114     elif index == 1:
115         get_viewport().msaa_3d = Viewport.MSAA_DISABLED
116         get_viewport().screen_space_aa = Viewport.SCREEN_SPACE_AA_FXAA
117         get_viewport().use_taa = false
118     elif index == 2:
119         get_viewport().msaa_3d = Viewport.MSAA_DISABLED
120         get_viewport().screen_space_aa = Viewport.SCREEN_SPACE_AA_DISABLED
121         get_viewport().use_taa = true
122     elif index == 3:
123         get_viewport().msaa_3d = Viewport.MSAA_2X
124         get_viewport().screen_space_aa = Viewport.SCREEN_SPACE_AA_DISABLED
125         get_viewport().use_taa = false
126     elif index == 4:
127         get_viewport().msaa_3d = Viewport.MSAA_4X
128         get_viewport().screen_space_aa = Viewport.SCREEN_SPACE_AA_DISABLED
129         get_viewport().use_taa = false
130     elif index == 5:
131         get_viewport().msaa_3d = Viewport.MSAA_8X
132         get_viewport().screen_space_aa = Viewport.SCREEN_SPACE_AA_DISABLED
133         get_viewport().use_taa = false
134
```

```
135     func scale_3d(value):
136         save_setting("escala_3d", value)
137         if can_play_sound:
138             get_parent().get_node("interact").play()
139         get_viewport().scaling_3d_scale = value
140 
141     func set_master_volume(value):
142         save_setting("volumen_general", value)
143         if can_play_sound:
144             get_parent().get_node("interact").play()
145         AudioServer.set_bus_volume_db(0, linear_to_db(value))
146 
147     func set_sfx_volume(value):
148         save_setting("volumen_efectos", value)
149         if can_play_sound:
150             get_parent().get_node("interact").play()
151         AudioServer.set_bus_volume_db(1, linear_to_db(value))
152 
153     func set_music_volume(value):
154         save_setting("volumen_musica", value)
155         if can_play_sound:
156             get_parent().get_node("interact").play()
157         AudioServer.set_bus_volume_db(2, linear_to_db(value))
158 
159     func set_look_speed(value):
160         save_setting("sensibilidad", value)
161         if can_play_sound:
162             get_parent().get_node("interact").play()
163         if player_head != null:
164             player_head.sensitivity = value
165 
166     func set_window_mode(index):
167         save_setting("modo_ventana", index)
168         get_parent().get_node("interact").play()
169         if index == 0:
170             DisplayServer.window_set_mode(DisplayServer.WINDOW_MODE_WINDOWED)
171         elif index == 1:
172             DisplayServer.window_set_mode(DisplayServer.WINDOW_MODE_FULLSCREEN)
173         elif index == 2:
174             DisplayServer.window_set_mode(DisplayServer.WINDOW_MODE_EXCLUSIVE_FULLSCREEN)
175 
```

```
176 func set_shadows(index):
177     save_setting("sombras", index)
178     get_parent().get_node("interact").play()
179     if index == 0:
180         RenderingServer.directional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_HARD)
181         RenderingServer.positional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_HARD)
182     elif index == 1:
183         RenderingServer.directional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_VERY_LOW)
184         RenderingServer.positional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_VERY_LOW)
185     elif index == 2:
186         RenderingServer.directional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_LOW)
187         RenderingServer.positional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_LOW)
188     elif index == 3:
189         RenderingServer.directional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_MEDIUM)
190         RenderingServer.positional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_MEDIUM)
191     elif index == 4:
192         RenderingServer.directional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_HIGH)
193         RenderingServer.positional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_HIGH)
194     elif index == 5:
195         RenderingServer.directional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_ULTRA)
196         RenderingServer.positional_soft_shadow_filter_set_quality(RenderingServer.SHADOW_QUALITY_SOFT_ULTRA)
197
198
199 func _on_button_pressed() -> void:
200     pass # Replace with function body.
201
202
203 func _on_button_2_pressed() -> void:
204     pass # Replace with function body.
205
206
207 func _on_button_3_pressed() -> void:
208     pass # Replace with function body.
209
210
211 func _on_button_4_pressed() -> void:
212     pass # Replace with function body.
213
214
215 func _on_button_5_pressed() -> void:
216     pass # Replace with function body.
217
218
219 func _on_button_6_pressed() -> void:
220     pass # Replace with function body.
```

task_trigger.gd

```
1   extends Area3D
2
3   @onready var ui = get_tree().current_scene.get_node("player/player_ui")
4   @export var task_text: String
5   var triggered = false
6   @export var enable_code: bool
7
8   func enter_trigger(body):
9       if body.name == "player" and !triggered:
10          triggered = true
11          ui.set_task(task_text)
12          if enable_code:
13              get_tree().current_scene.get_node("music").play()
14              get_tree().current_scene.get_node("music" ).pitch_scale = 0.5
15              get_tree().current_scene.get_node("enemy").process_mode = Node.PROCESS_MODE_INHERIT
16              get_tree().current_scene.get_node("enemy").visible = true
17              get_tree().current_scene.get_node("code_paper").visible = true
```

url_button.gd

```
1   extends Button
2
3   @export var link: String
4
5   func pressed_button():
6       OS.shell_open(link)
```

warning.gd

```
1   extends Control
2
3   @export var wait_time: float = 1.0
4   @export var scene_path: String
5
6   func _ready() -> void:
7       if has_node("CanvasLayer/AnimationPlayer"):
8           $CanvasLayer/AnimationPlayer.play("fade")
9
10      await get_tree().create_timer(wait_time).timeout
11
12      if scene_path != "" and ResourceLoader.exists(scene_path):
13          get_tree().change_scene_to_file(scene_path)
14      else:
15          print("ERROR: scene_path no válido -> ", scene_path)
16
17
18  func _process(_delta: float) -> void:
19      if Input.is_action_just_pressed("pause"):
20          if scene_path != "":
21              get_tree().change_scene_to_file(scene_path)
```

7.2 Carpeta google drive

Aquesta és la carpeta que hem utilitzat per a les captures de pantalla i els scripts més importants. La deixem aquí perquè es puguin veure les fotografies de Blender que no hem afegit al projecte principal però que tenim realitzades.

Google drive projecte