Creació d'un videojoc de terror psicològic a Unity

TREBALL DE RECERCA

Kronos

“En molts casos, per passar d’una bona experiència a una altra realment memorable, la narrativa és la clau que ens permetrà desbloquejar la porta d’entrada al següent nivell.”

Pepe Pedraz

Resum

Aquest treball explica tot el procés de creació i desenvolupament d'un videojoc propi, des del moment en què neix la idea fins a aconseguir un prototip jugable. L'objectiu és mostrar com una idea pot anar evolucionant i convertir-se en un projecte real. Al llarg del treball explico les diferents fases que he seguit, començant per la planificació i continuant amb la programació de les mecàniques bàsiques, la creació visual dels escenaris i els objectes, el disseny del so i la música, i també tota la part d'ambientació i narrativa.

El que es busca és entendre com totes aquestes parts treballen juntes i com cada petita decisió pot influir en el resultat final del joc. També es vol destacar la importància de mantenir una manera de treballar ordenada i progressiva, sobretot quan es tracta d'un projecte fet de manera individual. En aquest cas, tot depèn del propi ritme i de com s'organitza el temps, i per això he preferit avançar per blocs o parts tancades per poder veure resultats concrets a cada etapa i anar millorant a mesura que avançava.

Fer un videojoc no és només un procés tècnic sinó també una experiència creativa i personal. Cada elecció, ja sigui visual, sonora o narrativa, ajuda a transmetre emocions i a construir una atmosfera coherent amb el que es vol explicar. La combinació entre l'art, la programació, el so i la història és el que dona sentit a tot el conjunt. En definitiva, aquest treball és una manera pràctica d'explorar com es pot crear un videojoc des de zero i de veure com la part tècnica i la part artística s'han d'equilibrar per aconseguir un resultat complet i funcional.

Resumen

Este trabajo explica todo el proceso de creación y desarrollo de un videojuego propio, desde el momento en que nace la idea hasta conseguir un prototipo jugable. El objetivo es mostrar cómo una idea puede ir evolucionando y convertirse en un proyecto real. A lo largo del trabajo explico las diferentes fases que he seguido, empezando por la planificación y continuando con la programación de las mecánicas básicas, la creación visual de los escenarios y los objetos, el diseño del sonido y la música, y también toda la parte de ambientación y narrativa.

Lo que se busca es entender cómo todas estas partes trabajan juntas y cómo cada pequeña decisión puede influir en el resultado final del juego. También se quiere destacar la importancia de mantener una forma de trabajo ordenada y progresiva, sobre todo cuando se trata de un proyecto hecho de manera individual. En este caso, todo depende del propio ritmo y de cómo se organiza el tiempo, y por eso he preferido avanzar por bloques o partes cerradas para poder ver resultados concretos en cada etapa e ir mejorando a medida que avanzaba.

Hacer un videojuego no es solo un proceso técnico, sino también una experiencia creativa y personal. Cada elección, ya sea visual, sonora o narrativa, ayuda a transmitir emociones y a construir una atmósfera coherente con lo que se quiere contar. La combinación entre el arte, la programación, el sonido y la historia es lo que da sentido al conjunto. En definitiva, este trabajo es una forma práctica de explorar cómo se puede crear un videojuego desde cero y de ver cómo la parte técnica y la parte artística deben equilibrarse para conseguir un resultado completo y funcional.

Abstract

This project explains the entire process of creating and developing a personal video game, from the moment the idea is born to achieving a playable prototype. The goal is to show how an idea can gradually evolve and become a real project. Throughout the work, I explain the different stages I followed, starting with the planning phase and continuing with the programming of the basic mechanics, the visual creation of environments and objects, the design of sound and music, and also the whole part related to atmosphere and narrative.

The purpose is to understand how all these elements work together and how every small decision can influence the final result of the game. It also aims to highlight the importance of keeping an organized and progressive workflow, especially when it is an individual project. In this case, everything depends on one's own pace and time management, which is why I preferred to move forward in closed blocks or stages, allowing me to see concrete results at each step and improve gradually.

Creating a video game is not only a technical process but also a creative and personal experience. Every choice, whether visual, sound, or narrative, helps convey emotions and build an atmosphere consistent with the story being told. The combination of art, programming, sound, and storytelling is what gives meaning to the entire project. In short, this work is a practical way to explore how a video game can be created from scratch and to see how the technical and artistic sides must be balanced to achieve a complete and functional result.

Índex

Introducció

En aquest treball d'investigació s'explorarà el procés de creació d'un videojoc utilitzant Unity, una de les plataformes més populars per al desenvolupament de videojocs. Es tractaran temes importants com la programació, el disseny de personatges i escenaris, i la creació de música i efectes de so, amb l'objectiu d'entendre com aquests elements es combinen per oferir una experiència interactiva.

He escollit aquest tema perquè em fascina el món dels videojocs i la programació. Sempre m'ha cridat l'atenció com els videojocs poden explicar històries, crear mons immersius i oferir experiències úniques als jugadors. A més, m'agradaria treballar en la indústria del videojoc en el futur, per la qual cosa aquest projecte em permetrà adquirir coneixements fonamentals i una primera experiència en el desenvolupament de jocs.

Aquest treball suposa un gran repte per a mi, ja que és la primera vegada que faig un projecte d'aquest tipus. Hauré d'aprendre des de zero diversos aspectes clau del desenvolupament de videojocs, com la programació a Unity, el disseny i modelatge 3D amb Blender, i la creació de música i efectes de so per al joc.

Per dur a terme aquest videojoc, faré servir diferents eines i recursos. Unity serà el motor principal per a la programació i el disseny de la jugabilitat. També utilitzaré Blender per modelar personatges i escenaris en 3D, i programari d'edició musical per compondre la banda sonora i els efectes de so. Al llarg del projecte, investigaré sobre disseny de personatges, composició musical i creació d'entorns virtuals per entendre millor com aquests elements influeixen en la qualitat i la immersió del videojoc.

Aquest treball no només em permetrà aprendre sobre la part tècnica del desenvolupament de videojocs, sinó també sobre els diferents aspectes creatius que en formen part, acostant-me així a una visió més completa del que implica crear un videojoc des de zero.

Hipòtesi, objectius i metodologia

Amb aquest treball vull descobrir si soc capaç de crear un videojoc 3D que sigui jugable, entretingut i ben fet en un temps determinat. Mai abans he creat un videojoc complet, així que serà un repte en molts aspectes. No només hauré d'aprendre a programar i fer models en 3D, sinó que també hauré d'entendre com fer bones animacions, com crear música que encaixi amb el joc i com posar-ho tot junt a Unity perquè funcioni bé.

La hipòtesi que plantejo és si seré capaç d'assolir aquest repte i si el joc que acabi fent tindrà prou qualitat perquè algú el pugui jugar i divertir-se. També vull veure fins a quin punt puc aprendre totes aquestes coses i si soc capaç d'organitzar-me per anar progressant sense perdre'm en els detalls.

Per arribar a aquest objectiu, hauré d'aprendre diverses coses:

- Programació en C# per poder fer mecàniques de joc, col·lisions, moviments i altres funcionalitats dins de Unity.
- Modelatge 3D amb Blender, per crear personatges, escenaris i objectes. Aquí també he de veure com optimitzar els models perquè el joc no vagi lent.
- Animació 3D, que és una de les parts que més em preocupa perquè ara mateix no tinc clar com es fan ni com es passen a Unity. Hauré d'investigar i fer proves fins que entengui bé el procés.
- Creació de música i efectes de so, perquè vull que el joc tingui un ambient sonor que encaixi bé amb el que es veu en pantalla. Això implica estudiar una mica de teoria musical i provar eines per crear àudio.
- Disseny 2D, per fer coses com la interfície del joc, icones, menús o imatges per als elements de la interfície.

Com que tot això és molta feina, la metodologia que seguiré serà aprendre i aplicar alhora. Primer buscaré informació i tutorials per entendre els conceptes bàsics de cada àrea. Després, faré petites proves per veure com funcionen abans de passar-ho al joc final.

M’organitzaré en diferents fases:

1. **Recerca i aprenentatge:** Mirar tutorials, llegir documentació i entendre com funcionen Unity, Blender i les eines d’àudio.
2. **Proves pràctiques:** Fer mini-projectes per anar entenent cada part per separat (com un model 3D senzill, una animació bàsica o una mecànica petita de joc).
3. **Creació dels elements principals:** Quan ja tingui una mica més de pràctica, començaré a fer els personatges, escenaris i mecàniques que vull que tinguin el joc.
4. **Integració a Unity:** Posar tot el que he creat dins del motor de joc i veure com encaixa. Aquí segurament hauré de fer molts ajustos perquè tot funcioni bé junt.
5. **Polir i millorar:** Ajustar animacions, mecàniques, àudio i qualsevol detall per fer que el joc es vegi i es jugui millor.

Com que aquest projecte és força gran, no sé si aconseguiré tenir un joc complet al final, però el més important per a mi és aprendre el màxim possible i veure si m’agrada aquest procés de desenvolupament. Si aconsegueixo un joc funcional i divertit, serà genial, però si més no, segur que hauré après moltes coses pel camí.

Part teòrica

1. Història dels videojocs

1.1 Origen i evolució dels videojocs

Els videojocs no van néixer de cop com els coneixem avui. Tot va començar a mitjans del segle XX, en un context molt diferent, on les tecnologies encara estaven en una fase molt inicial. Els primers intents de crear jocs digitals venien d'investigadors i científics que, experimentant amb ordinadors, van acabar dissenyant petites aplicacions interactives que es podrien considerar els primers videojocs.

Un altre precedent important va ser OXO (1952), creat per Alexander S. Douglas a la Universitat de Cambridge. Es tractava d'una versió digital del tres en ratlla, jugable en un ordinador Ferranti Mark 1. Tot i que era molt primitiu, és considerat un dels primers videojocs digitals documentats de la història.

Un dels exemples més coneguts és el joc Tennis for Two, creat l'any 1958 per William Higinbotham. Era molt senzill: simulava un partit de tennis sobre un oscil·loscopi, però ja incloïa la base de la interactivitat entre una màquina i un jugador. Anys més tard, el 1962, un grup d'estudiants del MIT va crear Spacewar!, un altre dels primers videojocs de la història. Aquest ja mostrava un cert nivell de complexitat i va tenir una gran influència en els jocs que van venir després.

Imatge 1. Tennis for Two. *Imatge extreta de:* https://en.wikipedia.org/wiki/Tennis_for_Two

Aquests jocs no tenien cap finalitat comercial. Van néixer com a experiments o entreteniment entre científics. No obstant això, van posar les bases del que més tard es convertiria en una de les indústries més potents del món de l'entreteniment.[1]

[1] GM, A. (2024, 10 enero). ¿Cuál fue el primer videojuego? Esta es la historia de un pasatiempo universal. *Historia National Geographic*. https://historia.nationalgeographic.com.es/a/cual-fue-primer-videojuego-esta-es-historia-pasatiempo-universal_20123 (Data de consulta: 16/4/2025)

1.2 Els primers videojocs populars i l'inici de la indústria

Amb l'arribada dels anys setanta, els videojocs van començar a deixar de ser simples curiositats tècniques per convertir-se en una forma d'entreteniment amb molt de potencial. Un dels primers videojocs que va assolir èxit comercial va ser *Pong*, creat per Atari l'any 1972. Aquest joc, inspirat en el tennis de taula, era molt senzill en mecàniques, però va aconseguir captar l'atenció de milers de persones arreu del món. Va ser un punt d'inflexió que va demostrar que els videojocs podien triomfar com a producte comercial.[2]

Imatge 2. Pong. *Imatge extreta de:* https://www.bloglenovo.es/pong-videojuego-historia/

L'èxit de *Pong* va fer que moltes empreses apostessin pel mercat arcade, donant lloc a l'aparició de nous títols que avui dia són considerats llegendaris. Alguns dels més destacats són *Space Invaders* (1978), que va fer esclatar la febre dels recreatius al Japó i als Estats Units, i *Asteroids* (1979), que va portar una nova dimensió amb la seva mecànica de gravetat i moviment lliure per la pantalla.[3]

En paral·lel, van començar a aparèixer les primeres consoles domèstiques. La Magnavox Odyssey, llançada el 1972, va ser la primera consola comercial de la història, però no va tenir gaire èxit. En canvi, l'Atari 2600 (1977) va marcar un abans i un després, ja que introduïa els cartutxos intercanviables, cosa que permetia jugar a diferents jocs amb una sola consola. Aquest model també va establir l'estàndard dels cartutxos intercanviables, que marcaria la indústria durant dècades. Paral·lelament, va començar a créixer la rivalitat entre empreses, especialment amb l'entrada de Sega i la seva Master

Imatge 3. Atari 2600. *Imatge extreta de:* https://acortar.link/kmAwNf

[2] *Historia de los videojuegos: Decada de los 70 - ElOtroLado*. (s. f.). https://www.elotrolado.net/wiki/Historia_de_los_videojuegos:_Decada_de_los_70 (Data de consulta: 16/4/2025)
[3] Colaboradores de Wikipedia. (2025, 1 abril). *Historia de los videojuegos*. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Historia_de_los_videojuegos (Data de consulta: 16/4/2025)

System (1985), que va donar inici a una competició amb Nintendo que marcaria els anys següents.[4]

Cap a finals dels anys setanta i començaments dels vuitanta, es van començar a crear també personatges i universos que serien icònics. Un dels casos més importants va ser *Donkey Kong* (1981), que va introduir el personatge de Mario (encara que en aquell moment es deia "Jumpman"), i que marcaria l'inici d'una de les franquícies més famoses de tots els temps. Això també va coincidir amb l'ascens de Nintendo com a un actor clau dins la indústria.[5]

Imatge 4. Donkey Kong. *Imatge extreta de:* https://acortar.link/kmAwNf

Aquest període va servir de base per al que vindria després: una indústria amb creixement exponencial, una nova cultura del joc i una comunitat de jugadors cada cop més gran i diversa. Sense aquests primers passos, l'univers dels videojocs no seria el que és avui.

1.3 L'expansió dels videojocs durant els anys 80 i 90

Després dels primers èxits als anys setanta, els videojocs van començar una època d'expansió sense precedents durant les dècades dels 80 i 90. Aquesta etapa va estar marcada per una millora clara en la qualitat gràfica, l'accessibilitat de les consoles i l'ambició dels jocs.

L'any 1983, però, la indústria va patir una gran crisi coneguda com el "crash del videojoc". Als Estats Units, una saturació de consoles i jocs de baixa qualitat va fer que el mercat s'enfonsés. Un dels casos més famosos va ser E.T. the Extra-Terrestrial (1982), considerat un dels pitjors jocs de la història i que es diu que va acabar enterrat en milers de còpies al desert de Nou Mèxic. Aquesta crisi gairebé va acabar amb el mercat occidental, però va ser també el moment en què Nintendo va entrar amb força i va salvar el sector.[6]

[4] Colaboradores de Wikipedia. (2025a, marzo 18). *Atari 2600*. Wikipedia, la Enciclopedia Libre. https://es.wikipedia.org/wiki/Atari_2600 (Data de consulta: 16/4/2025)
[5] Wiki, C. T. A. (s. f.). *Donkey Kong*. Atari Wiki. https://theatari.fandom.com/wiki/Donkey_Kong (Data de consulta: 16/4/2025)
[6] *Historia de los videojuegos: Década de los 80 - ElOtroLado*. (s. f.). https://www.elotrolado.net/wiki/Historia_de_los_videojuegos:_D%C3%A9cada_de_los_80 (Data de consulta: 16/4/2025)

Imatge 5. The Legend of Zelda. *Imatge extreta de:* https://goo.su/R8Yigr6

Amb el llançament de la Nintendo Entertainment System (NES) el 1985, Nintendo va recuperar la confiança dels consumidors. Jocs com *Super Mario Bros.* i *The Legend of Zelda* no només van vendre milions de còpies, sinó que van establir els fonaments de moltes mecàniques de joc modernes.[7] A més, la figura del "plumber" Mario es va convertir en un símbol universal dels videojocs. Durant aquesta dècada també van destacar altres jocs arcade com *Pac-Man* (1980) i *Tetris* (1984), que van demostrar la capacitat dels videojocs per enganxar tota mena de públics, tant a Orient com a Occident.[8]

Les sales recreatives es van convertir en punts de trobada social, mentre que personatges com Mario, Sonic o Lara Croft es van transformar en icones reconegudes mundialment. A més, les consoles domèstiques van fer que el videojoc deixés de ser una experiència exclusiva d'arcade i entrés a les llars, establint les bases d'un mercat multimilionari que començava a competir amb altres indústries de l'entreteniment, com el cinema i la música.

A finals dels anys vuitanta i principis dels noranta, les consoles com la Super Nintendo (SNES) i la Sega Mega Drive van portar millores en gràfics i so, mentre que apareixien els primers intents de jocs amb CD-ROM, que permetien històries més llargues i bandes sonores de major qualitat. Així es començava a preparar el terreny per a la transició cap al 3D i les consoles de la meitat dels anys noranta.[9]

1.4 Els videojocs des dels anys 90 fins a l'actualitat

Els anys 90 van marcar un canvi profund en la indústria dels videojocs. L'evolució tecnològica va permetre la creació de jocs amb gràfics en 3D, que van donar lloc a un nou tipus d'experiència per als jugadors. La dècada va veure l'aparició de consoles com la Sony PlayStation (1994), que va aconseguir popularitzar els gràfics

[7] Campus, C. (2024, 18 marzo). *La evolución de los videojuegos a lo largo de la historia*. Universidad Europea Creative Campus. https://creativecampus.universidadeuropea.com/blog/evolucion-videojuegos/ (Data de consulta: 16/4/2025)
[8] Moisés. (2022, 28 diciembre). Los videojuegos de los 80 y 90 más influyentes de la historia | MiArcade. *MiArcade*. https://miarcade.com/los-videojuegos-de-los-80-y-90-mas-influyentes-de-la-historia/ (Data de consulta: 16/4/2025)
[9] Merino, L. J. (2024, 12 diciembre). LOS40. *LOS40*. https://los40.com/2024/12/12/la-batalla-por-el-trono-playstation-vs-sega-saturn/ (Data de consulta: 16/4/2025)

tridimensionals en jocs com *Final Fantasy VII* i *Gran Turismo*, establint un nou estàndard en la creació de videojocs. Durant aquesta dècada, van destacar també altres consoles com la Nintendo 64 (1996), amb títols icònics com *Super Mario 64* i *The Legend of Zelda: Ocarina of Time*, que van establir noves fites en disseny i jugabilitat.[10]

Imatge 6. Super Mario 64. *Imatge extreta de:* https://mario.fandom.com/es/wiki/Super_Mario_64

També cal destacar l'aparició del CD-ROM com a format principal, que va substituir els cartutxos i va permetre incloure més capacitat, vídeos i bandes sonores de major qualitat. Amb l'arribada de consoles més potents i l'avanç de la tecnologia, els videojocs van començar a explorar nous gèneres i a consolidar franquícies que captaven milions de jugadors arreu del món. Així, van aparèixer jocs de rol com *Final Fantasy VII* (1997), jocs de lluita com *Street Fighter II* (1991), *survival horror* com *Resident Evil* (1996), i franquícies esportives com *FIFA* i *Pro Evolution Soccer*, que van establir la tradició de llançaments anuals. Jocs com *Doom* (1993) i *Quake* (1996) van obrir el camí als FPS (First Person Shooter) en 3D, creant experiències immersives que encara avui dia són referents del gènere. Aquesta dècada va marcar també la creació i consolidació de grans franquícies que perduren fins avui, com *Pokémon*, *Resident Evil* i *Tomb Raider*. Al mateix temps, les sales recreatives van començar a perdre força a Occident, mentre que al Japó van mantenir un paper cultural important.[11]

Imatge 7. Final Fantasy VII. *Imatge extreta de:* https://ffvii.square-enix-games.com/es

Imatge 8. Halo 2. *Imatge extreta de:* https://halo.fandom.com/es/wiki/Halo_2

Ja als anys 2000, la indústria va viure una expansió amb el naixement de consoles com la Xbox de Microsoft

[10] Tones, J. (2021, 22 enero). *Cuál ha sido el mejor año de la historia de los videojuegos: analizamos tecnología y éxitos para. . .* Xataka. https://www.xataka.com/videojuegos/cual-ha-sido-mejor-ano-historia-videojuegos-analizamos-tecnologia-exitos-para-responder-a-pregunta-definitiva#:~:text=El%20primer%20a%C3%B1o%20que%20sale,videojuegos%20con%20una%20calidad%20acongojante. (Data de consulta: 17/4/2025)

[11] Coslada, R. (2025, 17 marzo). *20 Videojuegos que marcaron la década de los 90 | Por Rodrigo Coslada*. Colossus Gamers. https://colossusgamers.com/20-videojuegos-que-marcaron-la-decada-de-los-9 (Data de consulta: 16/4/2025)

(2001) i l'Xbox 360 (2005), que es van centrar en gràfics de nova generació i un joc en línia massiu, amb títols com *Halo 2* i *Gears of War*. A més, el desenvolupament de jocs en línia i les xarxes socials van portar els videojocs a un nou nivell, amb multijugador massiu en línia (MMO) com *World of Warcraft* (2004). Aquest període també va consolidar les consoles portàtils, amb èxits com la Game Boy Advance (2001) i la Nintendo DS (2004), que van vendre milions d'unitats i van fer dels videojocs un producte encara més accessible.[12]

A mesura que van passar els anys, les consoles de nova generació com la PlayStation 4 i la Xbox One van seguir impulsant el desenvolupament gràfic, amb títols com *The Witcher 3: Wild Hunt* (2015) i *Red Dead Redemption 2* (2018), oferint grans mons oberts plens de detalls i interacció. Els videojocs també van expandir-se a noves plataformes com els mòbils, fent que molts jocs, com *Candy Crush Saga* o *Clash of Clans*, arribessin a audiències massives arreu del món. La popularització de les microtransaccions i dels jocs free-to-play també va transformar la manera com les empreses monetitzen el producte.

Imatge 9. Red Dead Redemption 2.
Imatge extreta de: https://goo.su/H1ZVfj

Finalment, des dels anys 2020 fins avui dia, la indústria continua evolucionant amb noves tecnologies com la realitat virtual (VR), la realitat augmentada (AR) i el desenvolupament de jocs basats en el núvol. Amb streaming de videojocs, com ara Google Stadia i Xbox Cloud Gaming, els jugadors poden accedir als seus jocs des de qualsevol dispositiu, sense necessitat d'una consola potent. A més, els eSports i les plataformes de streaming com Twitch han convertit el videojoc en un espectacle global, seguit per milions d'espectadors en temps real.

La història dels videojocs, des dels anys 90 fins avui, ha anat canviant molt gràcies a les noves tecnologies, les innovacions constants i les franquícies que han enganxat generacions de jugadors. Els jocs s'han tornat molt més variats i immersius, i sembla que encara queda molt per veure en el futur.[13]

---

[12] Reyes, A. (2021, 10 marzo). *Los videojuegos más importantes de la historia (Parte 2) - UAMedia Blog*. UAMedia Blog. https://uamedia.org/blog/los-videojuegos-mas-importantes-de-la-historia-parte-2/ (Data de consulta: 17/4/2025)

[13] Contributors to Wikimedia projects. (2025, 7 marzo). *Història dels videojocs*. Viquipèdia, L'enciclopèdia Lliure. https://ca.wikipedia.org/wiki/Hist%C3%B2ria_dels_videojocs (Data de consulta: 17/4/2025)

2. Eines utilitzades

Per fer aquest videojoc, utilitzaré diferents eines que em serviran per cobrir tot el procés de creació: des de la programació i el disseny visual fins a la música i el so. Cada eina té un paper important i m'ajudarà a donar forma a la idea que tinc al cap. A continuació explico quines són i per a què les faré servir.

2.1 Unity

Unity és una de les eines més potents i utilitzades per crear videojocs de tot tipus, des de jocs en 2D fins a grans mons en 3D. És un motor de desenvolupament multiplataforma, la qual cosa vol dir que un cop tinguis el joc creat, pots exportar-lo a diferents dispositius com ordinadors, consoles, mòbils i fins i tot navegadors web, tot des d'un mateix projecte.

Imatge 10. Unity. *Imatge extreta de:* https://goo.su/AbuxO

Unity et permet desenvolupar gairebé tots els aspectes d'un joc: des de la part visual, gestionant els escenaris, personatges i animacions que importes o crees dins del motor, fins a la programació que controla què passa en cada moment. És molt popular entre creadors independents, però també s'utilitza en grans produccions com *Monument Valley*, *Cuphead*, *Hollow Knight* o *Pokémon Go*. Aquesta versatilitat fa que serveixi tant per a simuladors 3D realistes com per a jocs de plataformes 2D dibuixats a mà.

Imatge 11. Hollow Knight. *Imatge extreta de:* https://goo.su/0ZoA1x

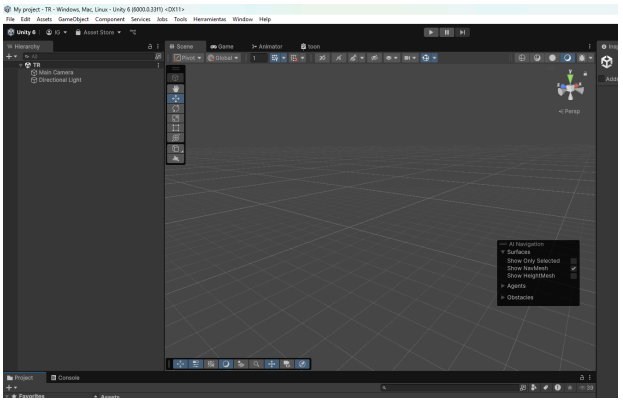
2.1.1 Com funciona Unity?

Unity és un programa amb un entorn visual intuïtiu, on pots dissenyar els escenaris i veure en temps real com quedarà el joc. Les seves funcions principals es poden dividir en diverses parts:

2.1.1.1 Escena i joc

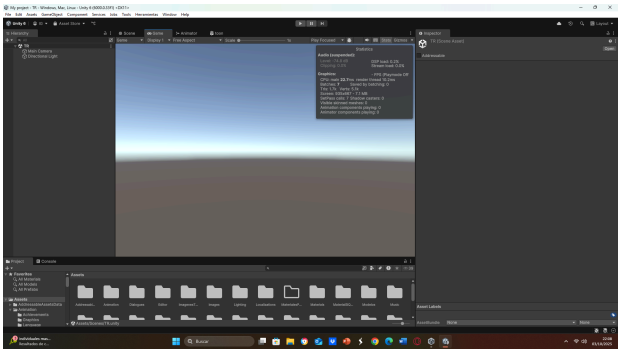
Unity funciona amb el concepte d'escenes, que són com els diferents espais on es desenvolupa el joc. Això pot ser un nivell, un menú principal, una habitació o qualsevol altre lloc que l'usuari pugui veure i explorar. Cada escena inclou tots els objectes, la il·luminació, la càmera i altres elements que formen aquell espai.

- **Scene View**: és la finestra de treball on pots col·locar, moure i organitzar els objectes dins de l'escena. És gairebé com si fossis un director que prepara l'escenografia d'una obra de teatre, decidint on ha d'estar cada element perquè tot tingui sentit i funcioni bé.



Imatge 12. Scene View. *Font pròpia*

- **Game View**: mostra com es veurà el joc quan algú hi jugui. Aquí pots comprovar que tot funciona correctament, que els objectes interactuen com toca i que el jugador tindrà l'experiència que has dissenyat. Els ajustos de posició d'objectes, càmera o il·luminació s'han de fer al Scene View, i després es poden veure els resultats al Game View.



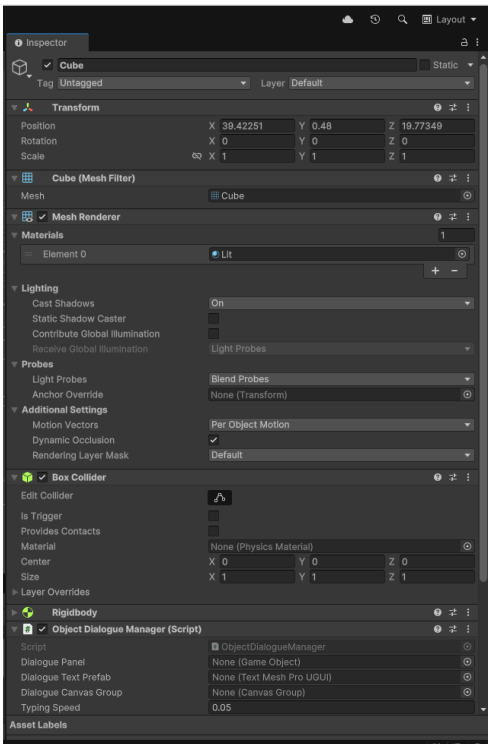
Imatge 13. Game View. *Font pròpia*

2.1.1.2 Objectes i components

A Unity, cada element del joc és un objecte, com un personatge, un arbre, una porta o fins i tot una llum. Però aquests objectes, per si sols, no fan res. Necessiten components, que són com peces que els donen funcions concretes i els permeten interactuar amb el joc.

Alguns dels components més habituals són:

- **Transform** que indica on està l'objecte dins de l'escena, com està girat i la seva mida. Sense aquest component Unity no sabria ni on col·locar-lo.
- **Collider** que serveix per detectar col·lisions, perquè per exemple un personatge no travessi una paret o una bala impacti un enemic.
- **RigidBody** que aplica les lleis de la física com la gravetat o els rebotaments. Això fa que una pilota rodoli de manera realista o que una caixa caigui segons el seu pes.
- **Scripts en C#** que són trossos de codi que defineixen com es comporta un objecte. Per exemple, què fa un enemic quan et veu o què passa quan premem un botó.



Imatge 13. Components principals. *Font pròpia*

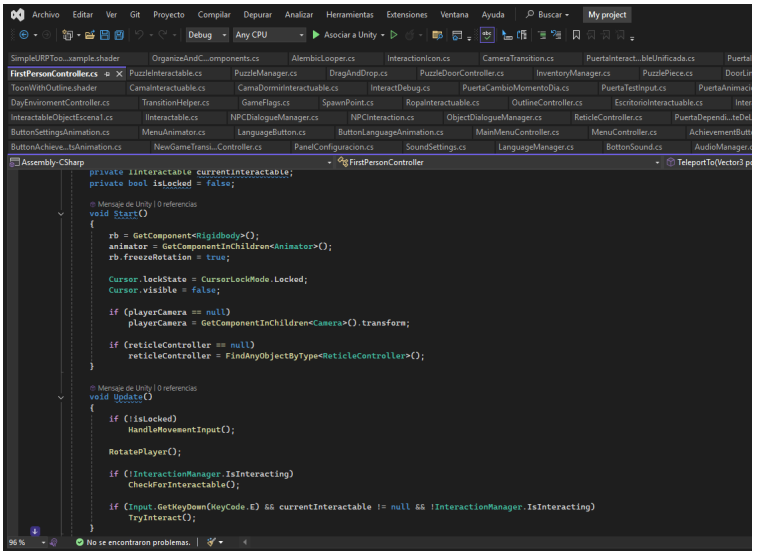
Aquests són només alguns dels components més bàsics. Unity té desenes de components més, cadascun amb característiques úniques que permeten fer coses molt més complexes i creatives dins del joc.

Una de les grans avantatges dels components és que es poden reutilitzar. Per exemple, si crees un enemic amb tots els seus components configurats, no cal tornar-lo a fer cada vegada. Pots convertir-lo en un prefab, que és com una plantilla, i fer-ne còpies amb les mateixes característiques. Això estalvia molt de temps i assegura que tots els enemics es comportin igual.

2.1.1.3 Programació i scripts en C#

Unity utilitza el llenguatge de programació C# per donar vida a tots els elements del joc. Amb aquest llenguatge es defineixen aspectes com el moviment dels personatges, les normes que marquen la partida o la manera com els objectes i l'usuari interactuen entre si. Per exemple, amb C# pots escriure el codi que fa que un personatge camini, corri, salti o utilitzi una arma. També pots establir quan un

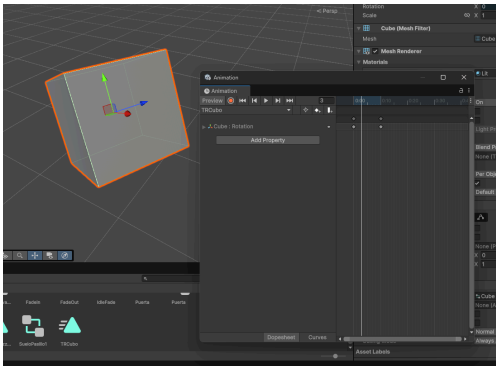
enemic és derrotat o en quin moment el jugador guanya; i fins i tot programar què passa quan es prem un botó, com començar una missió o obrir un menú dins del joc.



Imatge 14. Scripts en C#. *Font pròpia*

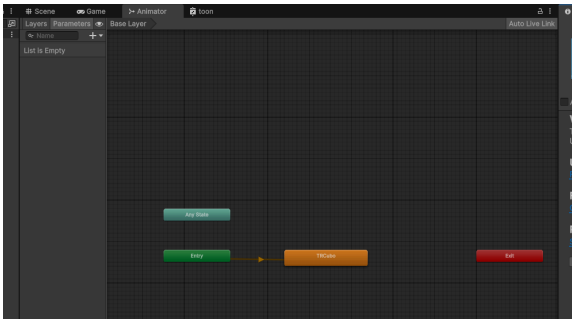
Tot i que programar pot semblar complicat, Unity inclou moltes eines que et faciliten la feina. Per exemple, no cal que inventis un sistema des de zero per organitzar els nivells o pantalles del teu joc, perquè Unity ja ofereix la gestió d'escenes preparada. Tampoc cal escriure tot el codi per crear un objecte nou cada vegada, ja que amb els prefabs pots tenir un model base i generar còpies d'aquest objecte tantes vegades com vulguis de manera automàtica. A més, si vols que els objectes tinguin gravetat, que xoquin entre ells o que es moguin de manera realista, Unity ja incorpora un sistema de física que s'encarrega de tot això.

2.1.1.4 Animació i interacció



Imatge 15. Animació en Unity. *Font pròpia*

Unity té un sistema d'animació molt complet que permet donar moviment i personalitat tant als personatges com als objectes del joc que controlis directament amb animacions. Això vol dir que pots fer que un personatge camini, corri, salti o ataqui amb fluïdesa, i que objectes com portes o altres elements tinguin moviments específics.



Imatge 16. Animator. *Font pròpia*

Tot això es gestiona amb l'eina Animator, que controla quan i com han de canviar les animacions segons les condicions que estableixis, per exemple passar de caminar a córrer o d'estar quiet a atacar.

L'Animator permet definir transicions i regles perquè els canvis siguin naturals i fluidos. A més, Unity permet crear cinemàtiques dins del joc, petites seqüències semblants a escenes de pel·lícula, que serveixen per explicar parts de la història, mostrar diàlegs entre personatges o fer que el jugador se senti més immers en el món del joc.

2.1.1.5 Multiplataforma i exportació

Un dels punts més forts de Unity és que et permet exportar el joc a moltes plataformes diferents:

- **Ordinadors (PC i Mac)**
- **Mòbils (Android i iOS)**
- **Consoles (PlayStation, Xbox i Nintendo Switch)**
- **Web (mitjançant WebGL)**
- **Realitat virtual i augmentada (VR/AR)**

Això no vol dir que el joc funcioni perfecte en tots els dispositius des del primer moment: cal fer ajustos específics perquè cada versió funcioni bé. Però Unity simplifica molt aquest procés, estalviant molta feina als creadors.

2.1.1.6 Física i intel·ligència artificial

Unity incorpora un motor de física que simula les lleis de la natura: la gravetat, els rebotaments o la inèrcia. Per exemple, una pilota pot rodar per una pendent i xocar contra una paret, igual que a la vida real.

També ofereix eines per crear intel·ligència artificial (IA) en els personatges no jugables. Això permet que els enemics es moguin sols, ataquin o reaccionin al que fa el jugador. Pots programar-los perquè segueixin un camí, patrullin una zona o canviïn de comportament segons les accions del jugador, fent el joc més dinàmic i interessant.[14]

[14] *Plataforma de desarrollo en tiempo real de Unity | Motor 3D, 2D, VR y AR*. (s. f.). Unity. https://unity.com/es (Data de consulta: 17/4/2025)

2.2 Blender

Blender és molt més que un programa per fer models en 3D: és un entorn complet de creació visual i animació. Quan desenvolupes un videojoc, Blender et permet construir l'aspecte físic dels personatges, els objectes i els escenaris, i fins i tot controlar com es mouen o com reaccionen a l'entorn. Es podria dir que és com un taller digital on tot el que després es veurà a Unity pren forma i personalitat.

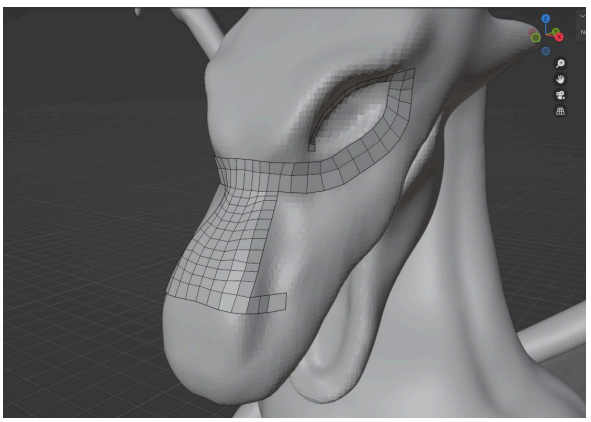
Imatge 17. Blender. *Imatge extreta de:* https://goo.su/TmlJS79

2.2.1 Modelatge 3D

El modelatge és el primer pas per donar forma a qualsevol element del videojoc. A Blender, pots començar amb formes molt simples com cubs, esferes o cilindres, i anar-les transformant fins aconseguir figures més complexes. És com fer una escultura, però en un entorn digital, on pots tallar, estirar, duplicar o suavitzar les superfícies fins aconseguir la forma que vols.

Hi ha dues maneres principals de treballar:

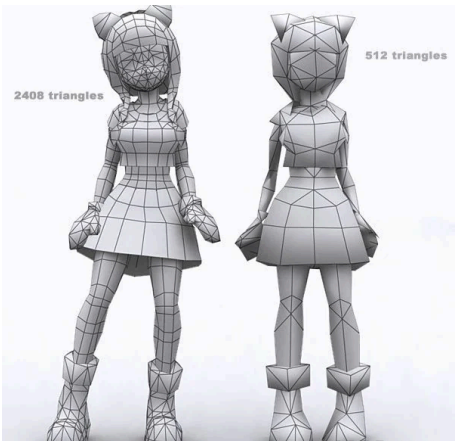
- **Modelatge High Poly + Retopologia**: en aquest mètode es comença creant un model molt detallat, amb milions de vèrtexs, gairebé com si fossis fent una escultura digital molt realista. Però aquests models són massa pesats per utilitzar-los directament en un joc, perquè farien que el joc anés lent. Per això es fa la retopologia, que consisteix a crear un nou model amb menys vèrtexs sobre la forma original. D'aquesta manera es manté la mateixa aparença i els detalls generals del model, però amb una versió més lleugera i eficient per al joc. Els detalls més fins, com arrugues o ornaments, es poden conservar després mitjançant



Imatge 18. Retopolització d'un model high poly. *Imatge extreta de:* https://goo.su/WoR2

textures especials com els normal maps, que simulen profunditat sense augmentar el nombre de vèrtexs.

- **Modelatge Low Poly directe**: en aquest mètode es creen models senzills i lleugers des del principi, pensant ja en el rendiment i la jugabilitat. És més ràpid i pràctic, ideal per a objectes simples o de decoració que no necessiten un alt nivell de detall però que continuen tenint bona aparença dins del joc.



Imatge 19. Modelatge low poly. *Imatge extreta de:* https://goo.su/ysFKT

Amb aquests dos mètodes, els creadors de jocs poden decidir si volen més realisme i detall (High Poly amb retopologia) o més rapidesa i eficiència (Low Poly), segons les necessitats del projecte.

Aquest procés requereix tenir una bona topologia: els polígons (que són les peces que formen el model) han d'estar ben ordenats i connectats, perquè si no, quan es vulgui animar o exportar, el resultat pot fallar i deformar-se. Per això, encara que sembli que només es tracta de "donar forma", en realitat s'ha de pensar molt bé en com es construeix cada model.

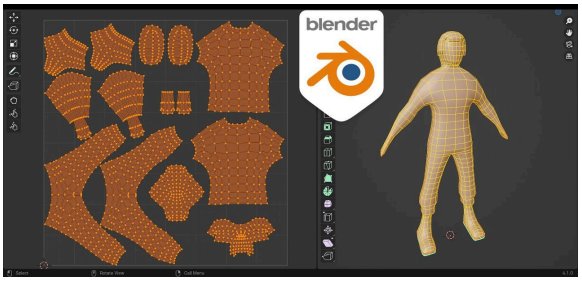
A més a més, en el món dels videojocs és essencial treballar amb models optimitzats. Això significa aconseguir una bona aparença amb pocs polígons (el que s'anomena *low poly*), deixant que els detalls més fins no estiguin en la geometria sinó en les textures. També cal ajustar mides reals i un punt d'origen correcte (*pivot point*), perquè quan es col·loquin a Unity es comportin de manera natural i no quedin desplaçats o girats.[15]

2.2.2 Texturització i materials

Quan el model està creat, el següent pas és definir la seva aparència visual. Aquest procés transforma un model 3D senzill en un element amb color, detalls i aspecte realista, preparat per ser integrat dins d'un joc.

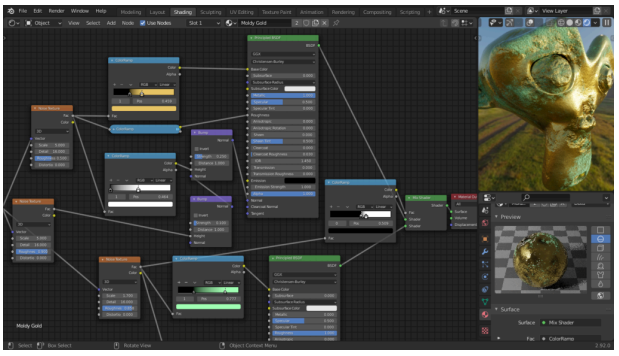
[15] Moyano, M. (2024, 29 noviembre). ¿Qué es blender? | The Core School. *The Core*. https://www.thecoreschool.com/blog/que-es-blender-y-como-puedes-sacarle-partido-para-la-animacion-3d/ (Data de consulta: 17/4/2025)

- **UV Mapping**: consisteix a “desenrotllar” la superfície del model 3D en un pla 2D, com si obrissis una caixa i la despleguessis perquè puguis pintar-la sense problemes. Això permet aplicar textures de manera precisa. Per fer-ho correctament, cal crear talls estratègics a la malla, anomenats seams, que indiquen per on s’ha de desplegar el model. Un UV ben organitzat evita que les textures s’estirin o es deformin i assegura que els detalls es col·loquin al lloc correcte. Sense un bon UV, la pintura o les textures poden aparèixer torçades o amb errors visuals.



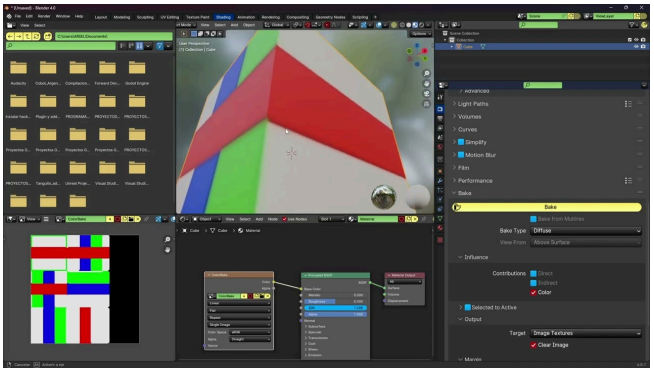
Imatge 20. UV Mapping. *Imatge extreta de:* https://goo.su/04YI

- **Texturització i materials amb nodes**: un cop s’ha definit l’UV, es poden aplicar textures. Aquestes textures poden ser pintades directament dins del programa o importades d’altres eines. Amb els materials i nodes es poden controlar aspectes com el color, la brillantor, la transparència o com la llum interactua amb la superfície. Això permet crear textures úniques per a cada objecte i donar-li un estil concret o un aspecte més realista, com metall polit, fusta rugosa o teixits amb detalls complexos.



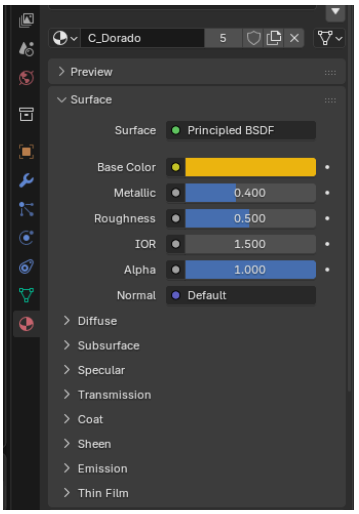
Imatge 21. Texturització amb nodes. *Imatge extreta de:* https://goo.su/E1tFwk5

- **Bake de materials**: els motors de joc poden interpretar els materials de manera diferent a com es veuen dins del programa de modelatge. Per això és necessari bakejar els materials, que vol dir convertir tots els efectes i propietats definits amb nodes en una imatge plana que el motor pugui llegir i aplicar correctament. Això assegura que els models mantinguin l’aspecte desitjat dins del joc, amb tots els detalls, colors i efectes de llum, sense perdre res del que s’ha creat amb nodes.



Imatge 22. Bake dels materials. *Imatge extreta de:* https://goo.su/VXKb3U

- **Materials**: defineixen com es veu i es comporta la superfície del model. Permeten controlar si és mat, brillant, translúcid o reflectiu, i es poden combinar amb les textures creades amb nodes per aconseguir un resultat final més complex i coherent. Els materials ben definits són essencials per donar uniformitat visual al joc i assegurar que els models es mostrin correctament segons la llum i l'entorn.



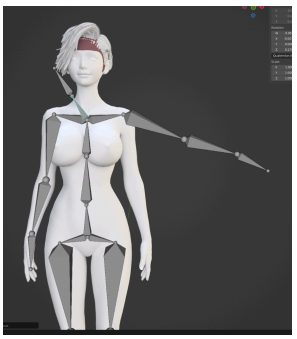
Imatge 23. Materials a Blender. *Font pròpia*

Amb els nodes pots fer textures molt detallades per a cada objecte. Quan portes els models a Unity, cal "bakejar" els materials perquè tots els detalls, els colors i els efectes de llum es mantinguin dins del joc. Si no tens ben definides les textures, els materials i les UVs, els models poden sortir deformats, amb colors estranys o sense alguns detalls, i això afecta com es veu el joc i fa que el jugador perdi immersió.[16]

2.2.3 Rigging

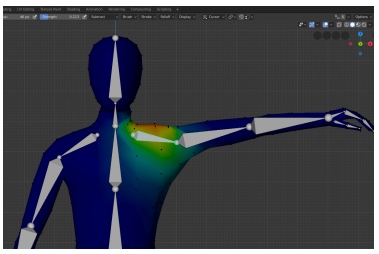
El rigging és el procés de crear un esquelet digital dins d'un model 3D per poder animar-lo després. Sense rigging, els models són simplement figures fixes que no es poden moure de manera natural.

- **Armature**: és l'esquelet del model i està format per diversos ossos (bones), cadascun dels quals controla una part específica del model, com un braç, una cama o el cap. Són aquests ossos els que després s'animaran per generar moviment.



Imatge 24. Armature. *Imatge extreta de: https://goo.su/e3Ncus*

- **Weight painting**: un cop col·locats els ossos, cal assignar cada part del model a un os. Això es fa pintant les zones que depenen de cada os, indicant quina proporció del moviment de l'os afectarà cada part de la malla. Per exemple, si pintem correctament el genoll, quan el dobleguem

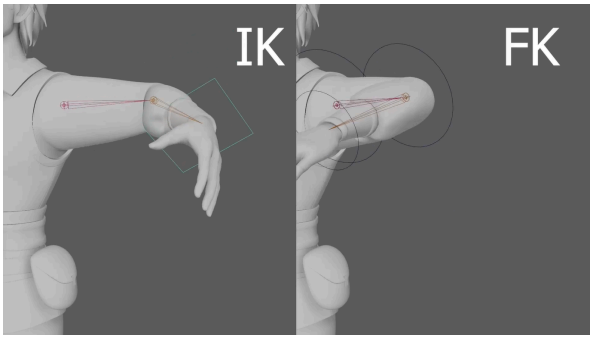


Imatge 25. Weight painting. *Imatge extreta de: https://goo.su/bLmweU1*

[16] De Marino, M. E. (2024, 5 julio). *Texturizar en blender*. Euroinnova International Online Education. https://www.euroinnova.com/arquitectura-y-diseno/articulos/texturizar-en-blender (Data de consulta: 17/4/2025)

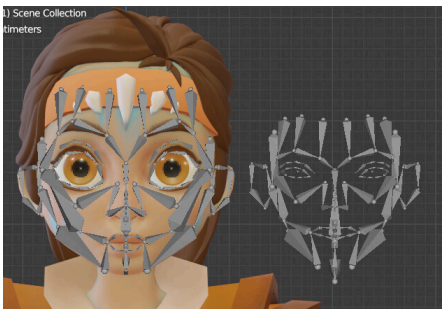
només es mouran les parts adequades de la cama, evitant deformacions estranyes.

- **Importància del rigging correcte**: si el rigging no es fa bé, el model es deformarà quan es mogui. Colzes, genolls o altres articulacions poden torçar-se de manera irreal, i les animacions perdran qualitat i naturalitat. Un rig correcte és essencial perquè totes les animacions posteriors funcionin bé i el model es mogui de manera creïble dins del joc o projecte 3D.
- **Rigs avançats**: a més del rig bàsic, es poden crear rigs més complexos per facilitar animacions més realistes i detallades:
    - **FK (Forward Kinematics)**: és el mètode més directe. Cada os es mou de manera independent i els ossos "fills" segueixen el moviment dels ossos "pares". Per exemple, si vols moure un braç, has de girar l'espatlla, després el colze i després el canell. És com manipular un braç articulat de joguina, ossos un per un. Aquest sistema dóna molt control, però pot ser més lent per fer moviments complexos.



Imatge 26. FK i IK. *Imatge extreta de: https://goo.su/kjlyxvL*

    - **IK (Inverse Kinematics)**: és més automàtic. Definint la posició final d'una part del cos, el sistema calcula automàticament com s'han de moure els ossos superiors perquè arribin a aquesta posició. Per exemple, si col·loques la mà sobre una taula, l'IK fa que el braç i el colze s'ajustin automàticament perquè la mà quedi al lloc correcte, sense haver de girar cada os un per un. És com moure la punta d'un braç articulat i que la resta del braç s'adapti sola.
    - **Constraints i ossos especials**: es poden afegir ossos per controlar expressions facials, dits, cadenes cinemàtiques o altres parts complexes del cos. Això permet animacions molt més detallades i naturals, millorant la qualitat final del model.

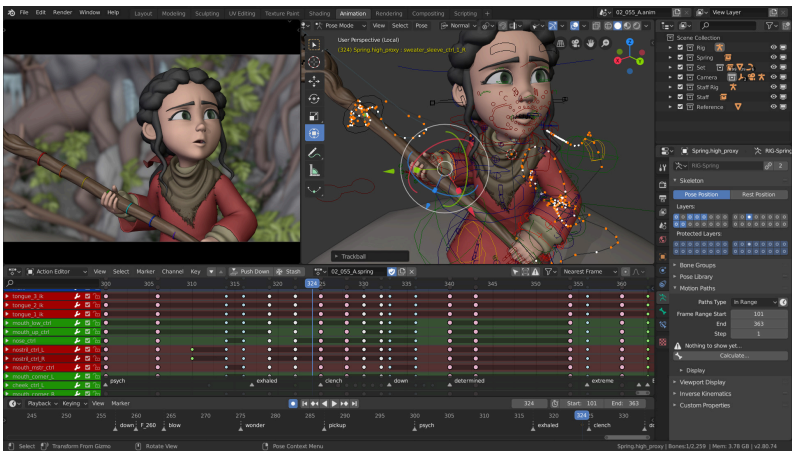


Imatge 27. Rig facial. *Imatge extreta de: https://goo.su/O4o4qRh*

El rigging és un pas clau per poder animar els models. Si no tens un esquelet ben fet i els pesos ben assignats, després serà molt difícil fer que les animacions surtin bé i els models es poden deformar quan es moguin dins del joc o projecte 3D. Amb un rig correcte i usant FK, IK i ossos especials de manera adequada, pots aconseguir que els personatges i objectes es moguin de manera natural i realista.[17]

2.2.4 Animació

Un cop el rig està fet, podem començar a animar els models. L'animació és el procés que permet que els personatges i objectes es moguin de manera natural dins del joc. A Blender, això es fa fotograma a fotograma, definint la posició dels ossos en diferents moments del temps i deixant que el programa interpol·li els moviments entre aquests punts, creant un moviment fluid i coherent.



Imatge 28. Animació a Blender. *Imatge extreta de:* https://goo.su/G8D7

Cal tenir en compte que l'animació pot portar molt de temps i esforç, fins i tot per accions aparentment senzilles. Per exemple, una animació de caminar, obrir una porta o aixecar un braç pot trigar hores a completar-se, ja que cal ajustar cada os, la fluïdesa del moviment i assegurar que tot sembli natural. Aprendre a fer animacions de qualitat requereix pràctica i paciència.

Algunes eines clau per animar a Blender són:

- **Pose Mode**: permet definir les postures dels ossos en moments específics. Per exemple, posar el braç aixecat, la cama doblegada o el cap girat. Cada postura constitueix un fotograma clau de l'animació, que serveix de punt de referència perquè Blender interpol·li els moviments entre postures.

[17] Instructables. (2018, 5 octubre). *Blender: Basic rigging process*. Instructables. https://www.instructables.com/Blender-Basic-Rigging-Process/ (Data de consulta: 17/4/2025)

- **Timeline i Dope Sheet**: serveixen per organitzar els fotogrames i controlar quan passa cada moviment dins de l'animació. Així podem ajustar la velocitat del moviment o sincronitzar diferents parts del cos, com braços i cames, perquè tot sembli coordinat.
- **Action Editor**: permet crear animacions individuals que després es poden reutilitzar en diferents situacions. Per exemple, podem fer una animació de "córrer" i utilitzar-la cada cop que el personatge es desplaci, sense haver de repetir tot el procés. Això facilita molt la gestió de múltiples animacions dins del joc.
- **NLA Editor (Non Linear Animation)**: aquest editor permet barrejar animacions. Per exemple, podem fer que un personatge estigui caminant mentre mou els braços o utilitza una eina. Això permet crear comportaments més naturals i complexos sense haver de fer una animació nova per a cada combinació possible.

Les animacions creades a Blender es poden exportar juntament amb el model i després reproduir-les dins de Unity utilitzant el Animator Controller, que gestiona quina animació es mostra en cada moment segons les accions del jugador o les regles del joc. Això permet que els moviments siguin coherents i reactius, com caminar, córrer, atacar o interactuar amb objectes dins del món del joc.[18]

2.2.5 Optimització

A diferència del cinema, on els models poden tenir milions de polígons sense problema, un joc ha de funcionar en temps real. Això significa que tots els objectes, animacions i efectes s'han de calcular mentre el jugador juga, sense retards ni caigudes de rendiment. Per això és fonamental optimitzar els models i les animacions, assegurant que el joc corri de manera fluida i mantingui una bona qualitat visual.

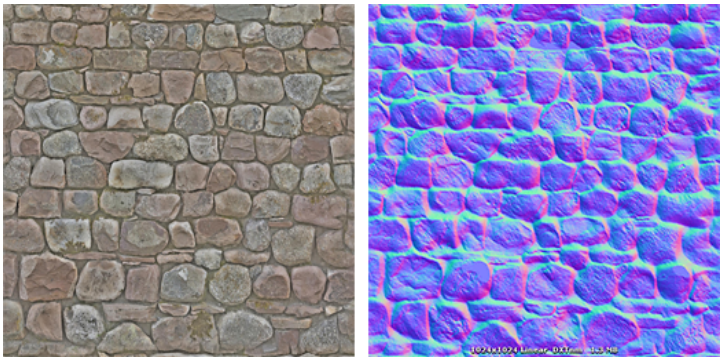
Els punts clau de l'optimització són:

- **Nombre de polígons**: si un model té massa polígons, el joc es torna lent, especialment si hi ha molts objectes a la pantalla. Per això cal buscar un

[18] González, A. C. (2022, 20 febrero). *Blender, qué es y para qué se utiliza*. Profesional Review. https://www.profesionalreview.com/2022/02/20/blender-que-es-y-para-que-se-utiliza/ (Data de consulta: 17/4/2025)

equilibri entre detall i rendiment, utilitzant malles simples (low poly) i aplicant textures i normal maps que simulin volum i detalls sense necessitat d'augmentar la geometria del model.

- **Bones per model**: el nombre d'ossos també afecta el rendiment, ja que cada os necessita càlculs per a les animacions. Moltes plataformes, especialment mòbils, limiten la quantitat d'ossos que es poden utilitzar sense perdre rendiment. Per això cal planificar bé el rig i evitar ossos innecessaris.
- **Textures optimitzades**: les textures han de tenir la mida adequada i no ser massa pesades. Textures molt grans poden provocar retards en carregar-se dins del joc o fer que el joc vagi lent. També és recomanable utilitzar formats que compressin bé la informació sense perdre qualitat visual.
- **Normal maps i bake**: aquests recursos permeten simular detalls i volum sense augmentar la geometria del model. Això significa que un objecte pot semblar molt detallat, amb relleu i textures complexes, però continuar sent lleuger per al motor de joc, ajudant a mantenir el rendiment sense perdre qualitat visual.



Imatge 29. Normal Maps. *Imatge extreta de:* https://goo.su/aVj5Duy

Blender té eines que ajuden a optimitzar els models abans d'exportar-los. Per exemple, et permet veure el nombre de polígons, vèrtexs i ossos d'un model, cosa que ajuda a anticipar si podria afectar el rendiment del joc. També pots aplicar modificadors que redueixen detalls quan no són necessaris, com en objectes que estan lluny del jugador o que no cal que siguin molt detallats. A més, Blender permet exportar versions optimitzades dels models en formats com FBX o glTF, preparades per Unity o altres motors de joc, perquè el joc funcioni fluidament sense perdre l'aspecte ni els detalls importants dels objectes.[19]

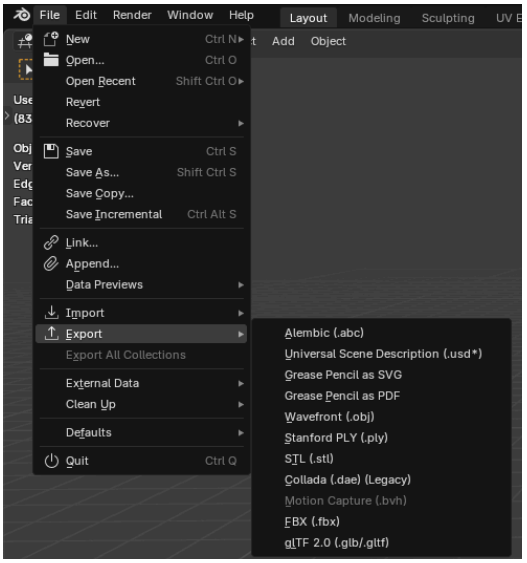
[19] *Foro3D*. (s. f.). https://www.foro3d.com/showthread.php?t=149525 (Data de consulta: 17/4/2025)

2.2.6 Exportació a Unity

Quan el model està complet i llest per ser utilitzat dins del joc, cal exportar-lo de Blender a Unity. Aquest pas és molt important perquè Unity necessiti llegir correctament els models, les textures i les animacions. Si hi ha algun problema en l'exportació, els objectes podrien aparèixer deformats, sense textures o sense animacions.

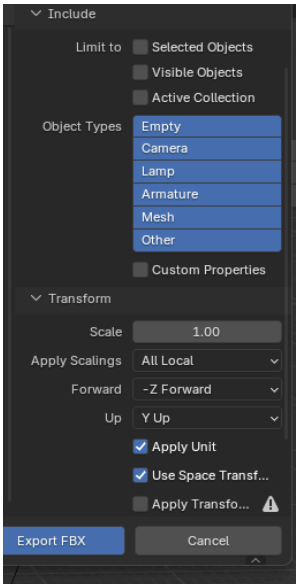
Els formats més habituals són:

- **.FBX**: és el més recomanat, ja que pot incloure la geometria del model, materials, textures i animacions. Funciona molt bé amb gairebé totes les funcions que necessites per portar un model de Blender a Unity sense perdre informació important.
- **.OBJ**: un format més senzill que només conté geometria i textures. S'aconsella només per a models estàtics que no necessiten animació.
- **.GLTF / .GLB**: formats moderns i lleugers, ideals per exportacions ràpides o per a jocs web. Poden incloure materials i animacions, però depèn de com Unity interpreti els nodes i textures del model.



Imatge 30. Exportació en Blender. *Font pròpia*

Abans d'exportar és fonamental comprovar alguns aspectes perquè tot es mantingui correcte dins de Unity. Primer cal revisar les escales ja que les dimensions del model han de coincidir amb les unitats de Unity perquè un model massa gran o petit podria afectar la jugabilitat. També cal tenir en compte l'orientació ja que Unity utilitza l'eix Y com a vertical mentre que Blender utilitza l'eix Z i sovint cal girar el model perquè quedi orientat correctament.



Imatge 31. Opcions d'exportació. *Font pròpia*

Pel que fa als materials i textures, cal tenir en compte que alguns tipus complexos creats amb nodes a Blender no es poden portar directament a Unity. Com ja vaig explicar abans, en aquests casos cal bakejar les textures perquè es mantinguin els colors i detalls dins del joc.

Un cop exportat el fitxer, només cal arrossegar-lo dins del projecte de Unity. Allà es convertirà automàticament en un prefab editable, és a dir, un objecte que es pot duplicar, modificar i reutilitzar fàcilment dins de diferents escenes.

A partir d'aquest punt, dins de Unity es poden afegir diferents elements perquè el model funcioni correctament dins del joc. Per exemple, es poden afegir col·lisions per definir com interactua amb altres objectes o personatges, assignar scripts per controlar el comportament del model (moviment, interacció amb el jugador o altres mecanismes del joc), i assignar animacions amb l'Animator Controller, que permet decidir quina animació es reproduirà en cada situació, com caminar, córrer, atacar o saltar.

Seguint aquests passos, els models creats a Blender funcionaran correctament dins de Unity, mantenint tant l'aspecte visual com el comportament dins del joc, i facilitant la creació d'escenes riques i interactives.[20]

[20] Blender Foundation. (s. f.). *blender.org - Home of the Blender project - Free and Open 3D Creation Software*. blender.org. https://www.blender.org/ (Data de consulta: 17/4/2025)

2.3 LMMS (Linux MultiMedia Studio)

LMMS és una eina gratuïta i de codi obert per fer música digital, el que s'anomena una estació de treball d'àudio digital (DAW). Dit d'una altra manera, és un programa que et permet crear, editar i organitzar música directament a l'ordinador, sense necessitat de tenir instruments físics. Pots compondre, fer mescles, experimentar amb sons i fins i tot produir peces musicals completes.

Imatge 32. LMMS. *Imatge extreta de:* https://goo.su/0AxoGm

Quan parlem de videojocs, LMMS és especialment útil perquè et permet crear música d'ambient, com per exemple una melodia tranquil·la per a un bosc o una música més intensa per a una batalla. També pots fer sons de menús, efectes curts com un *clic* quan selecciones opcions, o loops musicals, que són trossos de música que es repeteixen una i altra vegada sense que es noti el tall. Tot això ajuda a donar identitat i emoció al joc. El millor és que, encara que mai hagis utilitzat programes musicals, la interfície de LMMS és bastant intuïtiva i visual, així que pots començar a crear de manera organitzada i sense perdre't.[21]

2.3.1 Per a què utilitzaré LMMS en el meu videojoc?

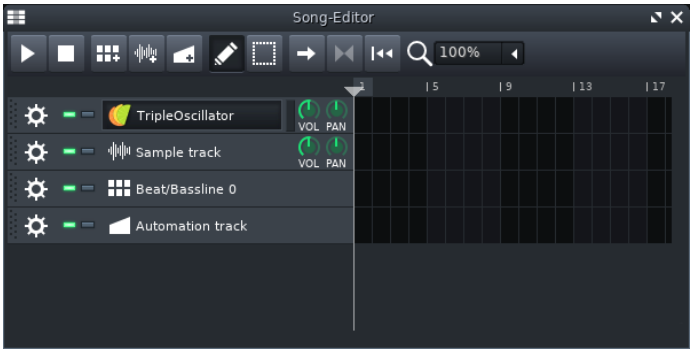
Amb LMMS es poden fer diferents tipus de música i sons dins del joc:

- **Música de fons (background):** és la música que sona durant la partida i crea una atmosfera (aventura, tensió, calma, suspens…).
- **Música de menú i interfícies:** melodies curtes i agradables que fan que moure's pels menús sigui més immersiu.
- **Sons o música de situació**: per exemple, un tema especial quan guanyes, quan perds o quan entres en pausa.
- **Loops:** fragments musicals que es repeteixen de forma contínua sense notar-se el tall, com un bucle de pocs compassos que sona mentre el jugador està explorant una zona.

[21] Deepin en Español. (2021, 17 febrero). *Linux Multimedia Studio (LMMS) - Wiki de Deepin en español*. Deepin En Español. https://xn--deepinenespaol-1nb.org/wiki/linux-multimedia-studio-lmms/ (Data de consulta: 18/4/2025)

2.3.2 Funcions principals de LMMS

- **Song Editor (Editor de cançons):** És com el centre de comandament del programa. Aquí pots organitzar totes les pistes de la teva cançó (melodia, baix, bateria, efectes…) en una línia de temps. Et permet veure-ho tot de manera visual i estructurada, gairebé com si estiguessis muntant un trencaclosques musical.



Imatge 33. Song Editor. *Imatge extreta de: https://goo.su/mN7tBJe*

- **Beat + Bassline Editor:** Serveix per crear ritmes repetitius de bateria i línies de baix. És molt útil perquè pots fer patrons (per exemple, un ritme de bateria bàsic) i reutilitzar-los en diferents parts del joc sense haver-los de tornar a fer cada cop.
- **Piano Roll:** Aquí pots dibuixar les notes musicals de manera visual, amb quadradets que corresponen a cada nota i al seu temps. És molt pràctic perquè no cal saber teoria musical per fer melodies o harmonies, només cal col·locar les notes i escoltar com sonen.

Imatge 34. Piano Roll. *Imatge extreta de: https://goo.su/8GaPA*

- **Instruments virtuals i samples:** LMMS porta diversos sintetitzadors incorporats (com ZynAddSubFX o TripleOscillator) que et permeten crear sons nous i únics. A més, pots importar arxius de so externs (WAV, OGG, MP3) o utilitzar instruments VST per ampliar encara més el catàleg. Gràcies a això, pots simular tota mena de sons dins del joc: des de passos o cops d'espasa fins a explosions o efectes màgics.
- **Efectes i mescla:** És la part on pots donar-li personalitat i qualitat final al so. Alguns efectes típics són:

- **Reverb**: Simula l'eco d'un espai, com si estiguessis dins d'una cova o en una sala gran.
- **Delay**: Fa que un so es repeteixi diverses vegades creant una sensació d'eco.
- **EQ (equalitzador)**: Serveix per retocar freqüències i millorar la qualitat del so, com treure greus que molesten o remarcar aguts.
- **Filtres i distorsions**: Canvien el caràcter d'un so, fent-lo més robòtic, brut o especial segons el que vulguis aconseguir.

2.3.3 Exportació d'àudio per videojocs

Quan la música ja està acabada, cal exportar-la a un format que entengui Unity o qualsevol altre motor de joc. LMMS permet exportar en formats com .ogg, .wav o .MP3, i aquí és important pensar en el pes i la qualitat de l'arxiu. Per exemple, si és una música de fons que ha de sonar molt de temps, potser convé exportar-la en .ogg amb qualitat mitjana per reduir la memòria que consumeix el joc i assegurar que funcioni bé fins i tot en dispositius menys potents.

2.3.4 Altres característiques útils per videojocs

LMMS també té funcions que ajuden molt quan es tracta de música per videojocs. Una és l'automatització, que et permet fer que certs paràmetres canviïn sols al llarg del temps, com el volum o els filtres, i així la música pot anar creixent en intensitat a mesura que la partida es complica. També hi ha els loop points, que serveixen per marcar punts exactes perquè la música es repeteixi de manera fluida, sense tallar de cop i trencar la immersió.

A més, LMMS porta una llibreria de sons predefinits i et dona la possibilitat d'afegir els teus propis sons o descarregar-ne de gratuïts. Això fa que puguis crear efectes i música variada més ràpidament, sense haver de començar cada cop des de zero.[22]

[22] *Welcome to LMMS | User manual*. (s. f.). User Manual. https://docs.lmms.io/user-manual

3. Concepte de disseny de videojocs

3.1 Què és el Game Design?

El *Game Design* és el procés de concebre, planificar i donar forma a l'experiència de joc. No es tracta només de fer bons gràfics o escriure codi, sinó de transformar una idea en un sistema que sigui jugable, que tingui sentit i que faci sentir alguna cosa al jugador.

Quan juguem, tot el que veiem, escoltem i fem està pensat perquè ens desperti una reacció concreta. Res no és casual. Darrere de cada acció, moviment o so hi ha una decisió de disseny feta amb una intenció clara.

Un bon *game design* parteix de preguntes molt bàsiques però essencials:

- Com es juga? Quines accions pot fer el jugador i com interactua amb els objectes o l'entorn?
- Què motiva el jugador a continuar? Hi ha recompenses, descobriments o objectius clars que el fan voler seguir jugant?
- Com s'organitza el ritme i la dificultat? Quan el joc ha de ser més intens, quan cal donar respir o quan és millor sorprendre el jugador?

El disseny és, en realitat, com construir un món amb les seves pròpies regles i equilibri. Cada element —les mecàniques, els visuals, la música, els sons i la narrativa— ha d'estar connectat per reforçar la mateixa experiència.

Imatge 35. Game Design. *Imatge extreta de: https://goo.su/dYtD0*

Un joc ben dissenyat és aquell que no només funciona tècnicament, sinó que aconsegueix emocionar. Pot ser una aventura plena d'acció, una història emotiva o un viatge estrany i misteriós, però el més important és que deixi empremta. Un bon *game design* fa que el jugador no només jugui, sinó que recordi allò que ha viscut i en vulgui parlar després.[23]

[23] MasterD. (2023, 27 abril). *¿Qué es el Game Design? ¿Cómo llegar a ser Game Designer?* MasterD. https://www.masterd.es/blog/game-design-que-es (Data de consulta: 18/4/2025)

3.1.1 Conceptualització i ambientació

Tot videojoc comença amb una idea que, al principi, pot ser molt abstracta: una sensació, una imatge mental o una emoció que vols transmetre. Pot ser tan simple com voler crear un joc que faci sentir tensió, misteri o curiositat, o aconseguir que el jugador tingui la sensació d'estar explorant un lloc diferent, amb les seves pròpies regles i sorpreses. Aquest primer punt serveix com a guia per construir tot l'univers del joc i decidir quines emocions o experiències vols provocar mentre es juga.

A partir d'aquí, es van definint alguns punts bàsics del projecte:

- **To emocional:** quin tipus d'emoció vols transmetre? Que sigui tens, misteriós, tranquil però amb un toc d'inquietud, o potser una barreja de tot segons el moment?
- **Univers:** funciona com el nostre món o té regles diferents? Potser el temps o la gravetat no actuen de la mateixa manera.
- **Espais:** on es mou el jugador? Són llocs reals, inventats o una barreja? Canvien depenent de les decisions del jugador o del progrés dins del joc?
- **Objectiu emocional:** què vols que senti el jugador? Curiositat, desconcert, fascinació, calma...?

L'ambientació no és només l'escenari on passen les coses. És tot allò que li dona coherència i personalitat al joc: els colors, la música, els sons, els detalls visuals i la manera com s'explica la història. Tot això, sumat, fa que el món sembli viu i interessant.

Hi ha alguns elements que ajuden molt a crear aquesta sensació:

- **Colors i il·luminació:** poden transmetre emocions o guiar el jugador sense que se n'adoni, per exemple marcant zones importants o creant una atmosfera concreta.
- **Arquitectura i espais:** la forma dels llocs o com estan col·locats els objectes pot influir en com el jugador es mou o com percep el món, fent-lo més interessant de recórrer.
- **Estil visual:** pot ser realista, estilitzat o més abstracte, segons el que vulguis transmetre.

- **So i música:** són claus per crear ambient. Un simple canvi en la música pot fer que el jugador senti calma o tensió sense que passi res visualment.
- **Narrativa ambiental:** detalls com objectes, escrits o elements amagats poden explicar històries sense diàlegs ni textos directes.

L'ambientació no ha de ser del tot lògica, però sí coherent dins del seu propi món. El més important és que desperti interès i emoció, i que faci que el jugador vulgui explorar, entendre i viure el món que has creat.[24]

3.1.2 Mecàniques del joc

Les mecàniques del joc són la base de com es juga. Són les regles i sistemes que marquen què pot fer el jugador i com respon el món a aquestes accions. Dit d'una altra manera: són el que transforma una idea en una experiència realment jugable.

Aquestes mecàniques poden ser molt variades. Poden anar des de recollir i combinar objectes, resoldre trencaclosques o endevinalles, prendre decisions que afectin la història, amagar-se d'enemics o escapar d'ells, fins a utilitzar eines com una llanterna, una càmera o qualsevol altre recurs que ajudi a explorar o avançar dins del joc. També poden ser tan simples com moure's lliurement per un espai o seguir rutes predefinides, depenent del tipus de joc i de l'experiència que es vol crear.

Imatge 36. Mecàniques d'un videojoc. *Imatge extreta de: https://goo.su/D8a8NY*

És important que totes aquestes mecàniques estiguin connectades entre elles i tinguin sentit dins del món del joc. Per exemple, potser el jugador troba una clau que serveix per resoldre un trencaclosques; això li permet accedir a una nova zona, descobrir informació important i, al final, veure l'espai d'una manera diferent. Quan les accions i les conseqüències estan ben unides, el joc se sent coherent i interessant.

[24] Ramírez, L. (2023, 23 marzo). *Game Design: ¿Qué es y cómo convertirte en un diseñador de videojuegos?* Thinking For Innovation. https://www.iebschool.com/hub/game-design-que-es-disenador-de-videojuegos-innovacion/ (Data de consulta: 18/4/2025)

També cal pensar en com el jugador progressa. Quins obstacles troba? Com desbloqueja noves àrees o coneixements? Les recompenses no sempre han de ser objectes materials; poden ser informació, emocions, noves experiències o canvis subtils en el món del joc. El que importa és que cada acció tingui sentit i aporti alguna cosa nova.

Les mecàniques han d'estar equilibrades: cap hauria de tornar-se repetitiva ni perdre interès, i totes haurien de servir per mantenir la curiositat i la sensació de progrés. La dificultat hauria d'augmentar de manera gradual, començant per accions senzilles i introduint poc a poc elements més complexos, perquè el jugador tingui temps d'aprendre i adaptar-se.

Imatge 37. Com fer bones mecàniques.
Imatge extreta de: https://goo.su/Dx6rzw

Una bona mecànica ha de ser fàcil d'entendre, però amb prou profunditat perquè, si el jugador vol explorar-la més a fons, pugui fer-ho. També és important preveure possibles problemes: mecàniques que es puguin "trencar", que el jugador pugui abusar-ne o que acabin avorrint. Tot això pot afectar molt l'experiència general del joc, així que cal trobar un bon equilibri entre simplicitat, coherència i diversió.[25]

3.1.3 Disseny de nivells i estructura del joc

Quan ja tenim les mecàniques definides, toca decidir on passen totes aquestes accions: els nivells del joc. Encara que el joc només tingui una gran zona oberta, és important pensar bé com s'organitza, com es guia el jugador i com evoluciona tot a mesura que avança.

Dissenyar nivells no és només col·locar parets o objectes. És pensar en el ritme i en la manera com el jugador viu l'experiència: quan hi ha tensió, quan pot explorar amb

[25] Admin, & Admin. (2019, 13 julio). Consejos de Game Design ¿Que son las mecánicas y cómo explotarlas? - Pixel Perfect Studio. *Pixel Perfect Studio - Dream it, build it!* https://www.pixelperfectstudio.mx/2019/07/12/consejos-de-game-design-que-son-las-mecanicas-y-como-explotarlas/ (Data de consulta: 18/4/2025)

calma o quan descobreix alguna cosa nova. També entra en joc la narrativa ambiental, és a dir, com els espais poden explicar històries sense necessitat de diàlegs o textos, només a través del que es veu o se sent.

Els nivells han de ser variats, reconeixibles i tenir moments que el jugador recordi. Això es pot aconseguir amb zones que es bifurquen, camins bloquejats que més endavant es poden obrir, detalls visuals que criden l'atenció o punts d'interès que guien sense forçar. Per exemple, una llum que destaca dins la foscor, un soroll que desperta curiositat o un objecte col·locat estratègicament poden fer que el jugador s'hi acosti sense adonar-se'n.

Imatge 38. Disseny de nivells. *Imatge extreta de: https://goo.su/MqhLU*

En jocs de misteri o exploració, sovint el primer nivell és petit i senzill per ensenyar les bases, i a poc a poc els espais es van obrint, apareixen nous reptes i el món es torna més complex. Aquesta progressió és important perquè el jugador senti que aprèn i descobreix.

Un bon disseny de nivell aconsegueix que el jugador sàpiga on anar sense necessitat d'indicacions constants. En canvi, un nivell mal dissenyat pot confondre, frustrar o fins i tot fer perdre l'interès. Per això, els nivells se solen provar moltes vegades, fent petits canvis fins que tot flueix de manera natural, mantenint l'equilibri entre llibertat, claredat i diversió.[26]

3.1.4 Psicologia del jugador i emocions

Una part molt important del disseny és entendre com pensa el jugador i què volem que senti mentre juga. No es tracta només de posar reptes o escenes boniques, sinó de provocar emocions concretes: curiositat, tensió, calma o fins i tot una mica d'inquietud.

[26] Danielparente. (2022, 18 diciembre). El proceso de diseño de niveles: una guía para principiantes - Daniel Parente Blog. *Daniel Parente Blog*. https://www.danielparente.net/es/2022/12/18/el-proceso-de-diseno-de-niveles-una-guia-para-principiantes/ (Data de consulta: 18/4/2025)

Un bon dissenyador té en compte què espera el jugador, què el pot sorprendre i què li resulta satisfactori. Per exemple, es pot jugar amb la curiositat deixant zones bloquejades, sons estranys o objectes que semblen importants però que encara no pots utilitzar. També és clau la sensació de recompensa, com quan aconsegueixes resoldre un trencaclosques o trobes una sortida després d'estar buscant durant una bona estona.

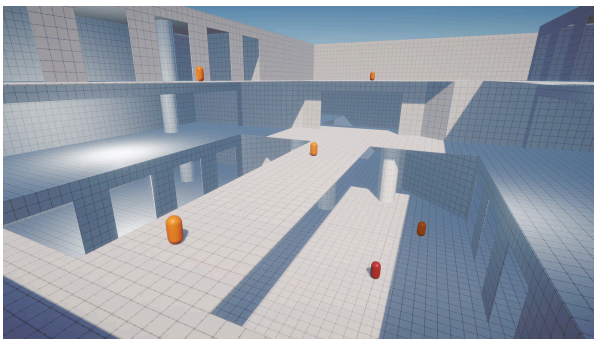
La tensió també és molt útil. No sempre cal fer un ensurt directe, a vegades n'hi ha prou amb un ambient que et fa sentir que pot passar alguna cosa en qualsevol moment. Aquella sensació d'incomoditat o de "no estar del tot segur" és molt més efectiva si està ben treballada.

Imatge 39. Psicologia del jugador i emocions. *Imatge extreta de: https://goo.su/XuOTz*

El so, la llum i la música juguen un paper molt important. Un soroll lleu on no t'ho esperes, una llum que s'apaga de cop o un passadís massa llarg poden transmetre més sensació que molts efectes visuals.

Al final, el disseny emocional no va només de fer que el joc funcioni, sinó de fer que deixi empremta. Que el jugador recordi com s'ha sentit mentre jugava, no només el que ha fet.[27]

3.1.5 Iteració i testejat



Imatge 40. Prototip d'un videojoc. *Imatge extreta de: https://goo.su/sovcbji*

El procés d'iteració i testejat és imprescindible en qualsevol videojoc. Cap joc surt perfecte a la primera, i gairebé sempre comença amb prototips senzills. Aquests prototips no tenen gràfics acabats ni música definitiva, però serveixen per veure si la idea funciona quan algú hi juga de debò.

[27] *A continuación, te explicamos cómo puedes entender la psicología del jugador en el diseño de juegos.* (2024, 21 junio). www.linkedin.com. https://es.linkedin.com/advice/0/heres-how-you-can-grasp-player-psychology-game-design-8qzbc?lang=es (Data de consulta: 18/4/2025)

Durant aquesta fase es fan preguntes constants: el jugador sap què ha de fer? Es perd o s'avorreix? Les mecàniques són clares i aporten alguna cosa? Hi ha moments que criden l'atenció o tot sembla pla? El feedback dels primers jugadors és fonamental perquè ajuda a detectar problemes que, des de dins del projecte, poden passar desapercebuts, com trencaclosques massa difícils, recorreguts poc intuïtius o detalls que no provoquen les emocions que esperàvem.

Iterar significa millorar contínuament i, de vegades, també descartar idees que semblaven bones al principi però que no funcionen tan bé en la pràctica. Aquest procés és especialment important en jocs amb to misteriós o psicològic, on cal ajustar bé el ritme, la tensió i la claredat per evitar frustracions sense perdre l'ambient i el sentit del misteri.[28]

[28] Ramírez, L. (2023b, marzo 23). *Game Design: ¿Qué es y cómo convertirte en un diseñador de videojuegos?* Thinking For Innovation. https://www.iebschool.com/hub/game-design-que-es-disenador-de-videojuegos-innovacion/ (Data de consulta: 18/4/2025)

3.2 Eines i procés de producció individual

Com que aquest projecte el desenvolupament completament jo, m'organitzo per fases i tasques utilitzant les eines que ja he triat abans segons el que fa cada una. Tot el procés combina art, so, narrativa i mecàniques, i tot ho faig de manera personal i artesanal. Encara que sembli un projecte gran, he estructurat la feina per blocs clars i treballo de manera modular i iterativa. Primer em fixo que tot funcioni a nivell bàsic i després vaig refinant cada element fins a arribar al resultat final.

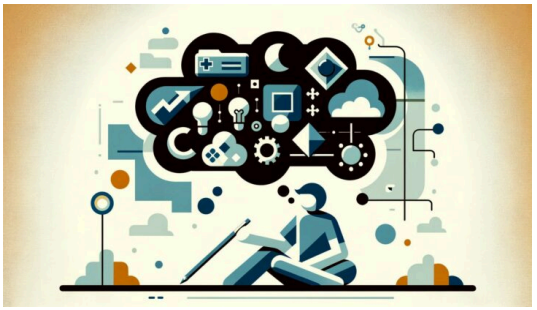
3.2.1 Planificació i estructuració del projecte

Treballant sol, tenir-ho tot clar és fonamental. Normalment començo amb un prototip inicial, una versió molt simple del joc que serveixi per veure si la idea funciona i detectar problemes abans d'invertir temps en detalls visuals o sonors. Després vaig dividint el projecte en escenes i seccions concretes, com una entrada, un passadís o un espai especial, i treballo cada part una a una. Això ajuda a tenir control i veure el progrés.

També mantinc documents de suport amb esquemes del món, les sensacions que vull transmetre i les mecàniques principals, així encara que passi temps sense tocar el projecte, no perdo el fil. Normalment segueixo un ordre de treball: primer programació bàsica i flux de joc, després models i escenaris, després música i àudio, i finalment detalls visuals i poliment.

A més, anoto idees creatives en qualsevol moment, encara que no les implementi immediatament. Això m'ajuda a no perdre inspiració i detectar possibles millores més endavant.

Tot i que no vegi factible acabar la idea principal del videojoc en el temps que tinc, l'important és continuar avançant fins on pugui i aprenent tot el possible al llarg del camí. Cada petit pas serveix per millorar, entendre millor les eines i consolidar les mecàniques i el món que estic creant.



Imatge 41. Estructuració del projecte. *Imatge extreta de: https://goo.su/45hzfNU*

És normal que, sent principiant, calcular el temps necessari sigui complicat. Algunes coses poden portar més del previst i, fins i tot en estudis professionals, hi ha projectes que s'allarguen més del que s'esperava. L'important és no perdre el ritme i seguir avançant, encara que sigui pas a pas.

3.2.2 Procés d'integració de continguts

En lloc de treballar en tot el joc alhora, és molt millor fer-ho per blocs tancats. Cada escena, trencaclosques o objecte important es tracta per separat, així pots centrar-te en els detalls i anar veient què funciona sense perdre el fil del projecte.

Una manera pràctica d'organitzar-ho és seguir un cicle que es repeteix per cada bloc. Primer, penso què passa en aquesta part del joc i quines sensacions ha de generar. Després faig els models i l'espai necessari per a l'escena, assegurant que tot encaixa dins de l'univers del joc. Un cop tinc els elements visuals, implemento les interaccions i els efectes dins del motor, definint com reaccionen els objectes i què pot fer el jugador en aquesta secció.

A continuació, afegeixo so, música o silencis segons el to i la sensació que vull transmetre. No és només posar música de fons; el so ajuda a reforçar l'ambient, guiar el jugador i donar pistes subtils sobre què passa o què pot passar.

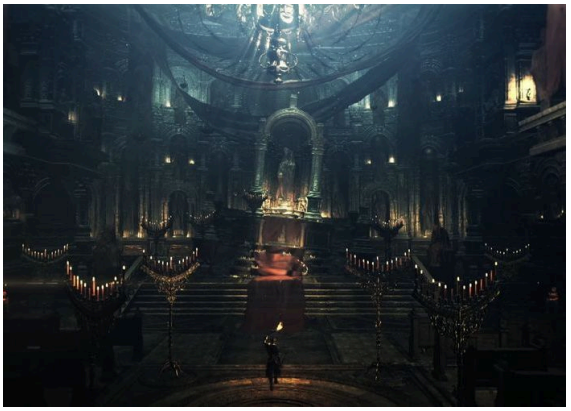
Finalment, toca provar i polir cada bloc, ajustant detalls fins que tot rutlla com toca. Durant aquesta fase pots detectar errors de jugabilitat, problemes de ritme o elements que no aconsegueixen l'efecte que vols. Sovint cal repetir aquest cicle diverses vegades, perquè cada iteració millora el bloc i el consolida dins del joc global.

Treballar bloc a bloc també facilita afegir més tard noves idees o fer canvis sense haver de desmuntar tot el projecte. A més, és una manera de seguir aprenent mentre vas resolent problemes i afinant models, textures, animacions i mecàniques de joc.

3.2.3 Producció artística i narrativa

Els elements visuals i els espais no es posen només "per omplir"; cada model, objecte i escenari té un motiu clar dins del món del joc. La manera com es

col·loquen, les formes, els colors o la il·luminació poden canviar completament com el jugador percep l'espai i com entén què està passant. No és només decoració: cada cosa que veus ha de tenir sentit dins de la història i ajudar a guiar el jugador sense que ni se n'adoni.



Imatge 41. Producció artística (Dark Souls 3).
Imatge extreta de: https://goo.su/gs8Ud

La narrativa no sempre s'explica de manera directa. El jugador ha d'explorar, fixar-se en detalls, interactuar amb objectes o observar l'entorn per descobrir informació. Poden ser petites pistes, objectes fora de lloc o detalls visuals que expliquen alguna cosa sense necessitat de textos o diàlegs. D'aquesta manera, els visuals, els sons i les interaccions es combinen per crear una experiència coherent i immersiva, on tot té un propòsit i reforça les sensacions que vols provocar.

3.2.4 Creació sonora i ambientació

Els sons i la música són una part clau de qualsevol videojoc. No és només posar música de fons o efectes sonors perquè sí: cada so, cada melodia o cada silenci té un propòsit i pot canviar completament com el jugador viu l'escena.

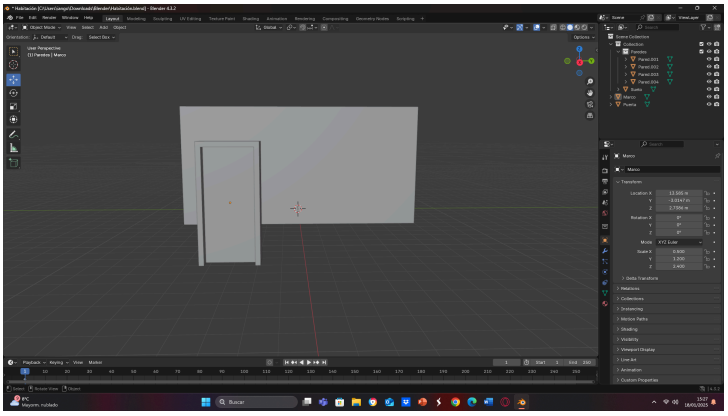
És important pensar en què vols que senti el jugador en cada moment. Hi ha escenes on pot ser millor no posar música i deixar només sons subtils, com passos, portes que cruixen o un vent llunyà, per augmentar la tensió. Altres cops, melodies repetitives amb petites variacions poden fer que alguna cosa sembli fora de lloc, que el jugador s'hi fixi sense que sigui obvi. I alguns efectes sonors especials només apareixen en moments clau per marcar-los i donar impacte.

El més important és planificar tot això pensant en com el so guia el jugador dins de l'espai i reforça la història i l'atmosfera. No es tracta de fer que el joc soni "bé", sinó que cada detall sonor ajudi a que l'experiència sigui coherent i immersiva.

3.2.5 Organització personal i ritme de treball

Quan treballes tot sol en un projecte, encara que hi hagi unes dates de lliurament generals, dins del procés real no tens horaris estrictes ni obligacions fixes. Per això és essencial organitzar-se de manera flexible. Pots dedicar dies a programar, altres a models i art, o a so i música segons com et vingui la inspiració. L'important és avançar, encara que sigui a poc a poc, i anar aprenent en cada pas.

No cal buscar la perfecció absoluta. És millor tenir un joc més petit però complet i coherent, que funcioni i transmeti les idees i emocions que vols, abans que intentar fer un projecte enorme que mai no acabes. Treballar bloc a bloc ajuda molt: et permet veure resultats concrets, ajustar detalls, provar coses noves i aprendre mentre ho fas. Així, encara que la idea final del joc principal no arribi a estar totalment acabada, pots seguir avançant, experimentar i agafar coneixements que després podràs aplicar a altres projectes.



Imatge 42. Organització personal. *Font pròpia*

També és important acceptar que algunes coses poden portar més temps del previst, i que està bé fer pauses o canviar de tasca quan cal. Organitzar-se d'aquesta manera et dona control, evita frustracions i permet que tot el projecte vagi creixent de forma natural, mantenint la coherència i la qualitat de l'experiència que vols oferir.

4. Concepte inicial del videojoc

Imatge 43. Fears to Fathom (terror psicològic). *Imatge extreta de: https://goo.su/jgQjp*

Des de sempre m'han agradat els videojocs amb històries profundes, sobretot aquells que tenen un punt psicològic o misteriós. Sempre m'ha cridat l'atenció com poden transmetre emocions reals sense necessitat de mostrar monstres o acció constant. El que m'interessa d'aquest tipus de jocs és com aconsegueixen generar tensió només amb l'ambient, la música o els petits detalls, fent que el jugador se senti incòmode sense saber ben bé per què. D'alguna manera, és una manera més personal d'entendre la por i de connectar amb el que sent el personatge.

4.1 Gènere i estil del joc

El joc combina elements d'aventura narrativa, exploració i terror psicològic. No hi ha combats ni enemics, perquè l'objectiu no és generar por directa, sinó una sensació constant d'inquietud. El jugador ha d'explorar i intentar entendre què està passant, tot observant els canvis subtils de l'entorn. Visualment, el joc té un estil "toon", amb formes senzilles i colors plans, però utilitzats d'una manera que pot resultar una mica estranya. Aquest contrast entre un estil visual aparentment simple i un ambient tèrbol és el que li dona personalitat i ajuda a crear aquesta sensació d'inestabilitat o desconcert.

4.2 Resum de la premissa

El jugador controla un estudiant que viu sol i que porta una rutina repetitiva: cada dia fa exactament el mateix, sense res nou ni ningú amb qui parlar. Tot sembla normal, fins que comencen a passar coses que no encaixen. A poc a poc, el que semblava un entorn normal es torna més estrany i desconcertant. No hi ha explicacions clares ni res que guiï directament el jugador; tot passa de manera subtil, fent que ell mateix hagi d'interpretar què està passant i què és real. El joc juga molt amb la percepció i

amb el fet de sentir que alguna cosa no va bé, tot i que no se sàpiga exactament què.

4.3 Mecàniques bàsiques

El joc es basa sobretot en l'exploració i la interacció amb l'entorn.
Les mecàniques principals són:

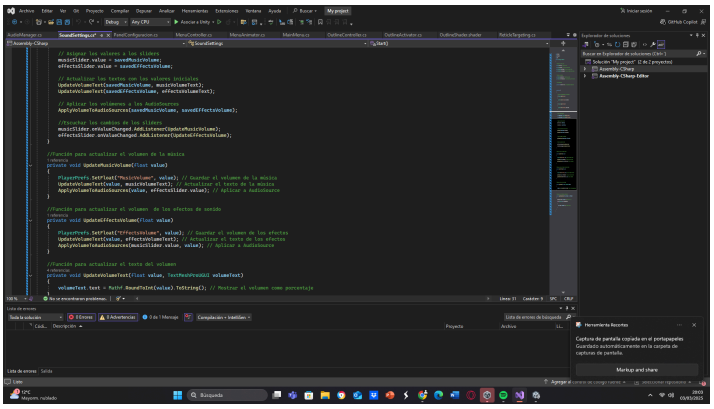
- Moure's lliurement per l'espai i observar els detalls.
- Interactuar amb objectes per descobrir informació o activar petits canvis.
- Anar interpretant tot el que passa per entendre una mica més la situació.

Com he dit abans, no hi ha acció ni enemics, sinó una sensació constant d'anar descobrint coses i intentant entendre què està passant. La gràcia és que cada detall pot tenir un significat o ser una pista per avançar.

Part pràctica

En aquesta part pràctica explico tot el procés de desenvolupament del meu propi videojoc, que he creat completament sol i des de zero. És la primera vegada que afronto un projecte d'aquest tipus, i això m'ha fet aprendre moltes coses alhora: des de fer servir el motor de joc fins a treballar amb eines de modelatge, so i narrativa. Tot plegat ha estat un repte gran, però també una experiència molt interessant i útil, que m'ha ajudat a entendre millor com es construeix un videojoc des de dins.

Des del principi tenia clar que volia fer un joc narratiu amb elements de terror psicològic. Sempre m'han agradat aquest tipus de jocs perquè et permeten viure l'experiència des d'una perspectiva més personal, com si t'endinsessis dins la ment del personatge. No calen monstres ni acció constant per crear tensió; el que realment m'interessa és aconseguir que el jugador senti curiositat, incomoditat o incertesa a través de l'ambient i la història. Aquesta idea connecta molt amb els meus interessos i m'ha ajudat a mantenir-me constant durant tot el procés.



Imatge 44. Programació del videojoc. *Font pròpia*

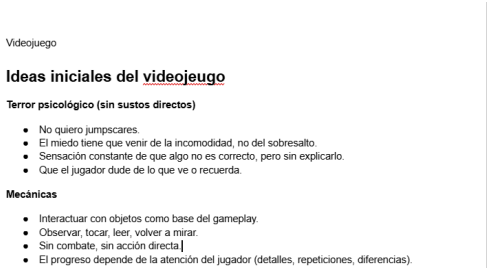
Al llarg del desenvolupament he anat treballant diferents parts del joc: la programació de les mecàniques bàsiques, la creació d'escenaris i objectes, la part sonora i ambiental, i sobretot la construcció d'una narrativa que doni sentit a tot plegat. He preferit treballar de manera modular, avançant pas a pas, ajustant coses i aprenent sobre la marxa. Al principi havia pensat en un projecte més llarg, però amb el temps he anat adaptant-lo per poder arribar a un resultat acabat, coherent i que funcioni bé.

L'objectiu d'aquesta part pràctica no és fer un joc perfecte ni competitiu a nivell professional, sinó aconseguir un prototip jugable que pugui transmetre les emocions i la idea principal que volia expressar. Sobretot, és el resultat d'un procés d'aprenentatge, dedicació i molta curiositat per entendre tot el que implica crear un videojoc des de zero

5.1 Planificació i estructuració del projecte

Quan vaig començar el projecte, em vaig adonar que era essencial tenir una mínima planificació, encara que fos flexible, per poder organitzar tot el procés. La idea inicial era molt més ambiciosa del que realment podia dur a terme amb el temps i els coneixements que tenia, així que el primer repte va ser reduir-la a una versió més senzilla, però que alhora mantingués l'essència del que volia transmetre.

Un dels primers passos va ser elaborar un GDD (Game Design Document), un document on vaig recollir les idees principals del videojoc: la història, les mecàniques bàsiques, l'estètica i l'ambientació que volia aconseguir. El GDD em va servir com a punt de partida per ordenar tot el que tenia al cap i donar-li una primera estructura. Tot i això, amb el pas del temps, aquest document va anar canviant. A mesura que avançava en el desenvolupament, em trobava amb limitacions tècniques o amb noves idees que m'obligaven a adaptar-lo. Això em va fer entendre que un GDD no és un pla rígid, sinó una guia flexible que acompanya tot el procés i es transforma amb ell.


Videojuego

Ideas iniciales del videojeugo

Terror psicológico (sin sustos directos)

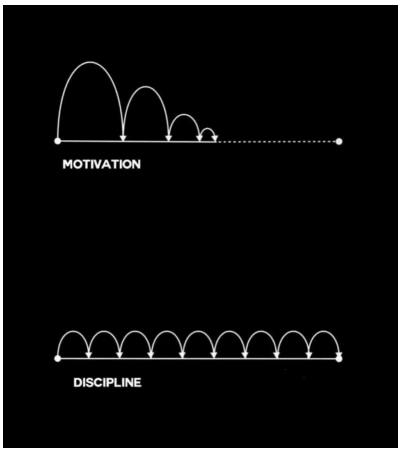
- No quiero jumpscares.
- El miedo tiene que venir de la incomodidad, no del sobresalto.
- Sensación constante de que algo no es correcto, pero sin explicarlo.
- Que el jugador dude de lo que ve o recuerda.

Mecánicas

- Interactuar con objetos como base del gameplay.
- Observar, tocar, leer, volver a mirar.
- Sin combate, sin acción directa.
- El progreso depende de la atención del jugador (detalles, repeticiones, diferencias).

Imatge 45. GDD *Font pròpia*

Un dels problemes principals va ser trobar l'equilibri entre la motivació i la realitat del treball. Hi va haver moments en què em costava avançar, especialment quan em trobava encallat en una sola tasca durant molt de temps. Per exemple, en el disseny del personatge principal vaig estar pràcticament un mes dedicant-hi hores cada dia, i tot i l'esforç, la sensació era que no avançava prou. Aquesta falta de resultats visibles em feia perdre motivació, però alhora em va ensenyar la importància de la constància: encara que fos lent, cada petit progrés acabava sumant.



Imatge 46. Motivació vs Disciplina. *Imatge extreta de: https://goo.su/MLtV6re*

Un altre factor que em va afectar va ser la comparació amb altres videojocs. Sovint mirava projectes independents que havien trigat anys a desenvolupar-se, o fins i tot

grans produccions AAA amb equips enormes al darrere. Aquesta comparació em feia sentir que el meu ritme era insuficient, i em desmotivava veure que el resultat del meu joc era molt més limitat. Tot i això, amb el temps vaig entendre que el meu objectiu no era competir amb ells, sinó aprendre i construir alguna cosa pròpia dins de les meves possibilitats.

També vaig comprovar que moltes vegades subestimava el temps necessari per acabar certes parts del projecte. Tasques que en teoria semblaven senzilles acabaven requerint molt més esforç del previst, sobretot perquè era la primera vegada que feia un videojoc d'aquestes característiques. Tot i així, vaig mantenir una rutina més o menys constant: encara que alguns dies estigués menys inspirat, intentava dedicar unes hores al desenvolupament, ja fos en programació, disseny visual o prova de mecàniques.

Amb tot això, vaig entendre que la planificació no és un esquema fix que s'ha de seguir al peu de la lletra, sinó un procés que s'ha d'anar adaptant a les circumstàncies. En el meu cas, això significava anar reduint o simplificant algunes parts, aprendre a prioritzar el que era essencial i deixar de banda elements que no podia assumir. Finalment, vaig estructurar el joc en escenes més petites i concretes, cosa que em permetia treballar de manera més ordenada i veure progressos tangibles, encara que fossin petits.

5.2 Disseny i creació dels elements del joc

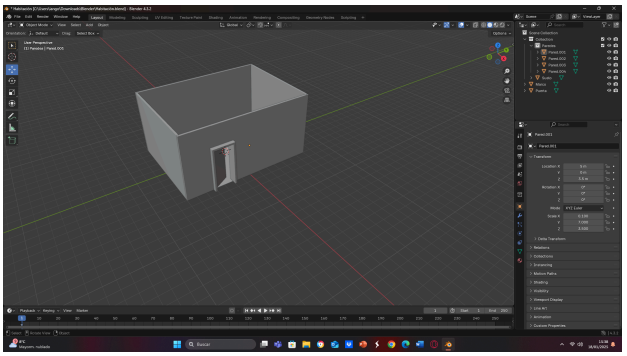
En aquesta part del treball explico tot el procés creatiu i tècnic relacionat amb els elements visuals del videojoc. El disseny i la creació d'aquests components són una de les fases més importants, ja que donen forma al món on el jugador s'endinsa i ajuden a transmetre les emocions i els temes que vull expressar.

Per aconseguir-ho he treballat sobretot amb Blender, un programa de modelatge i creació 3D. Amb ell he pogut desenvolupar tant els escenaris com el personatge principal, així com els objectes que els envolten. Tot aquest procés no només ha estat llarg i complex, sinó també molt enriquidor, perquè m'ha permès aprendre

eines noves, experimentar amb diferents tècniques i entendre millor com l'art i la narrativa poden anar de la mà dins d'un videojoc.

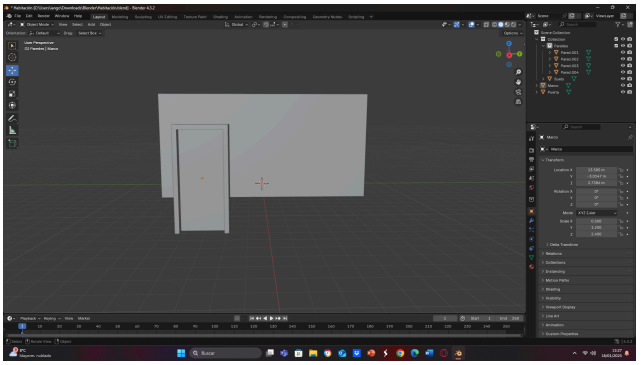
5.2.1 Modelatge i art en Blender

5.2.1.1 Modelatge de l'escenari

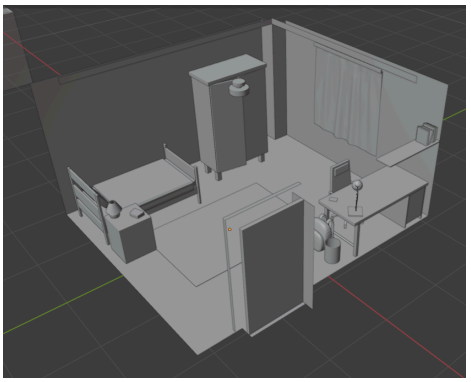


Imatge 47. Primers models d'escenari en Blender. *Font pròpia*

Una de les parts més importants i també de les que més temps m'ha portat del projecte ha estat el modelatge i la creació artística dels escenaris i del personatge. Tot aquest procés el vaig fer amb Blender, un programa que al principi em va semblar molt complicat. Té una quantitat enorme d'eines i funcions, com el modelatge poligonal, l'esculpit, els modificadors, el shading, els nodes, les animacions o les simulacions físiques. Per a algú que no l'havia fet servir mai, la primera impressió va ser bastant confusa i fins i tot una mica aclaparadora. Tot i això, quan vaig començar a practicar i a provar coses, em vaig adonar que en realitat és bastant intuïtiu i molt potent. Si li dediques temps i paciència, pots aconseguir resultats molt bons.



Imatge 48. Primers models d'escenari en Blender. *Font pròpia*

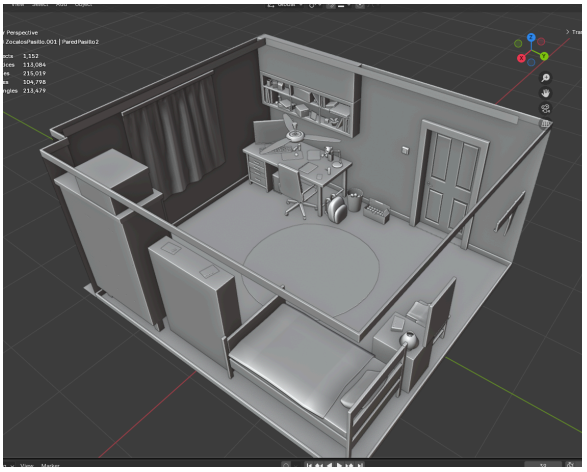


Imatge 49. Petites millores a l'escenari. *Font pròpia*

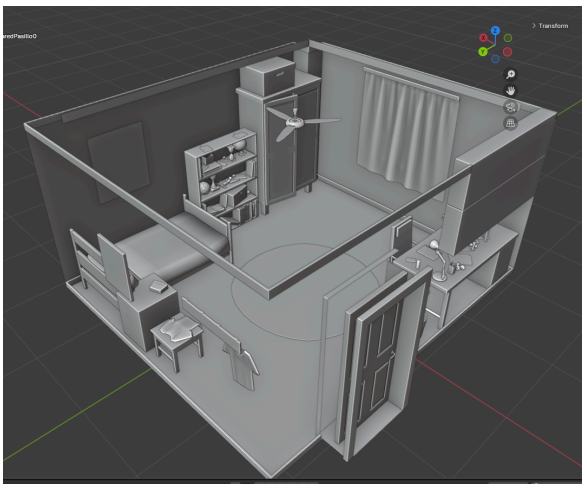
Els meus primers models van ser força simples. Vaig fer proves amb cubs, esferes i cilindres per entendre com es deformaven les malles, com fer extrusions o subdivisions i com funcionaven les eines bàsiques de transformació. No eren objectes amb cap propòsit artístic, però em van servir per practicar i entendre millor com funciona el programa. Aquests primers models van acabar formant part de

l'habitació del protagonista, tot i que amb el temps i a mesura que vaig anar aprenent més sobre Blender, vaig decidir tornar-los a fer des de zero per millorar-los i adaptar-los millor al resultat que volia aconseguir.

Quan ja tenia una mica més de confiança, vaig començar amb el que seria el meu primer escenari complet: l'habitació del protagonista. Per a mi era important que aquesta habitació reflectís la personalitat i l'estat emocional del personatge, perquè és l'espai on passa la major part del temps i també el punt d'inici de la història. Vaig començar amb les estructures bàsiques, com les parets, el terra i el sostre, i després vaig anar afegint mobles i petits detalls.



Imatge 50. Habitació detallada del jugador. *Font pròpia*

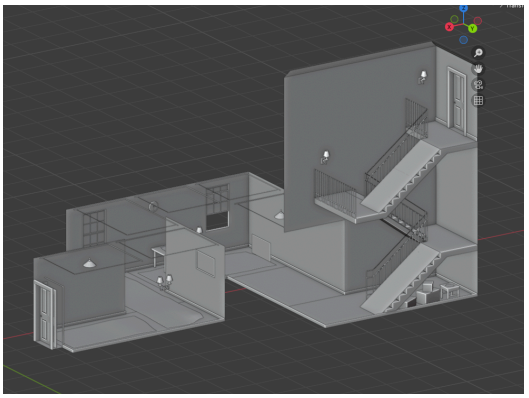


Imatge 51. Habitació detallada del jugador. *Font pròpia*

El repte era aconseguir que cada objecte tingués un sentit dins de la història. No volia que estiguessin allà només per omplir. Per exemple, l'escriptori ple de papers i llibres desordenats transmet rutina i desgana. La motxilla tirada en una cantonada pot representar cansament o falta d'interès, i la paperera plena de papers arrugats mostra frustració o pèrdua de temps. D'aquesta manera, els objectes no només decoren l'espai, sinó que expliquen aspectes de la vida del protagonista sense necessitat de dir res.

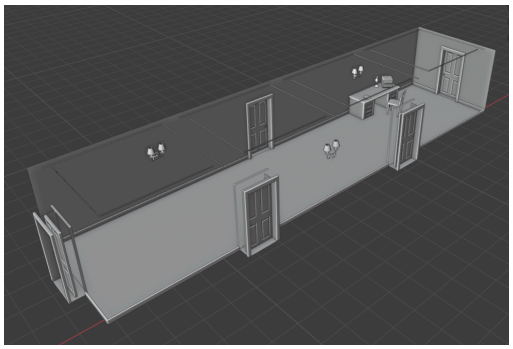
En aquesta part també vaig provar un munt de modificadors de Blender perquè em feien la vida molt més fàcil i acceleraven el treball. Per exemple, l'Array em servia per repetir objectes com llibres o làmpades sense haver-los de fer un a un, el Subdivision Surface per suavitzar formes quan calia, el Solidify quan necessitava donar gruix a algun objecte pla, el Mirror per treballar objectes simètrics, el Boolean per unir o retallar malles, el Decimate quan algun model tenia massa vèrtexs i calia

reduir-los… N'hi vaig utilitzar molts més depenent de la situació, cadascun per coses diferents, i m'ajudaven molt a avançar sense perdre temps. També vaig fer servir simulacions físiques per donar realisme a detalls com els plecs de les cortines o els papers dins la paperera. Tot això em va permetre treballar més ràpid i amb més fluïdesa, i alhora feia que els escenaris tinguessin més credibilitat visual.



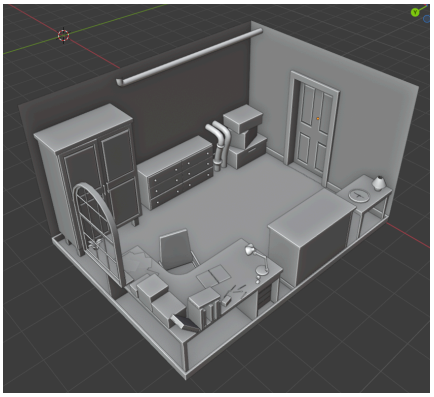
Imatge 52. Model del primer passadís. *Font pròpia*

Un cop acabada l'habitació, vaig continuar amb els passadissos principals, que tenien un paper narratiu important perquè servien de connexió entre el món del protagonista i un altre diferent. Abans de posar-me a modelar, vaig pensar una mica en quina sensació volia transmetre amb aquests espais: desorientació, tensió i estranyesa. Amb això en ment, vaig començar creant l'estructura bàsica i després vaig anar afegint elements decoratius com quadres, làmpades, finestres o mobles.

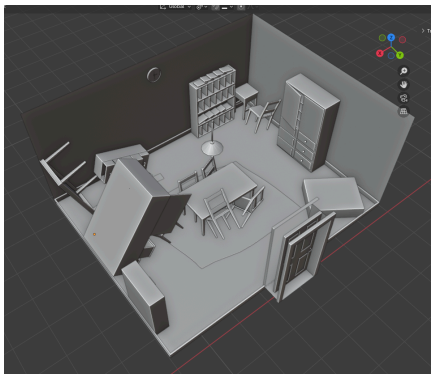


Imatge 53. Model del segon passadís. *Font pròpia*

Després d'això, vaig treballar en tres habitacions addicionals connectades al segon passadís. Cada una té un significat dins del joc, encara que no necessàriament estan relacionades entre elles. Les vaig desenvolupar amb el mateix mètode que l'habitació principal: primer la base (parets, sostre i terra) i després anar afegint progressivament objectes i detalls per donar-los vida. Sempre vaig intentar que cada element tingués un sentit dins del joc i no estigués només "per omplir".



Imatge 54. Model de la primera habitació. *Font pròpia*



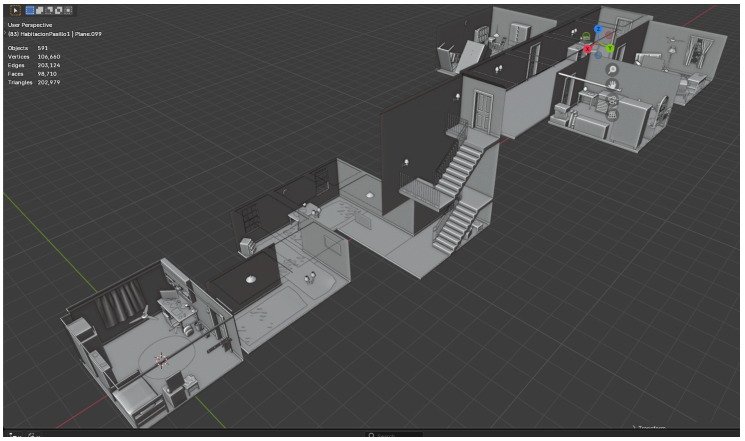
Imatge 55. Model de la segona habitació. *Font pròpia*

Un aspecte clau va ser optimitzar el temps i els recursos. Com que treballava tot sol, vaig haver de reutilitzar models, però col·locant-los de manera diferent per no donar sensació de repetició. Per exemple, llibres, làmpades o cadires apareixen en diversos escenaris, però canviant la seva posició, escala o rotació. Això em va permetre anar més ràpid i mantenir coherència visual.

Imatge 56. Model de la tercera habitació. *Font pròpia*

Tot i això, el procés de modelatge va ser llarg i sovint frustrant. Al principi, amb poques experiències i buscant la perfecció, qualsevol objecte simple em costava molt. Moltes vegades feia un objecte, no m'agradava, el borrava i tornava a començar, així una i altra vegada fins millorar. Amb el temps, aprendre les dreceres i agafar confiança em va permetre fer objectes complexos, que podrien trigar hores, de manera més entretinguda i fluida. Durant tot el procés, vaig intentar que els models tinguessin pocs polígons per assegurar un bon rendiment.

En total vaig acabar creant més de 550 objectes, comptant estructures bàsiques, mobles i petits detalls. A més de col·locar-los bé, havia de tenir en compte un detall tècnic molt important, la posició del pivot de cada objecte. El pivot és el punt des d'on l'objecte gira o es mou, i si està malament posat, dins del joc els moviments no quedarien bé. Per exemple, una porta no pot girar sobre el seu centre perquè llavors rotaria sobre si mateixa en lloc d'obrir-se com toca. Per això el pivot l'havia de posar a la zona de les frontisses. El mateix passava amb les agulles d'un rellotge, que necessiten tenir el pivot exactament al centre per poder girar bé. Això era molt important perquè les animacions simples que després



Imatge 57. Model l'escenari complet. *Font pròpia*

feia a Unity, com obrir portes o moure objectes, funcionessin correctament i no hagués de tornar a ajustar-les cada cop.

No havia d'ajustar el pivot de tots els objectes, només dels que volia animar més endavant. Els elements estàtics, com parets o mobles que no es mouen, no necessitaven cap configuració especial. Però en objectes com portes, calaixos o rellotges, sí que era important col·locar bé el pivot perquè després les animacions sortissin correctament.

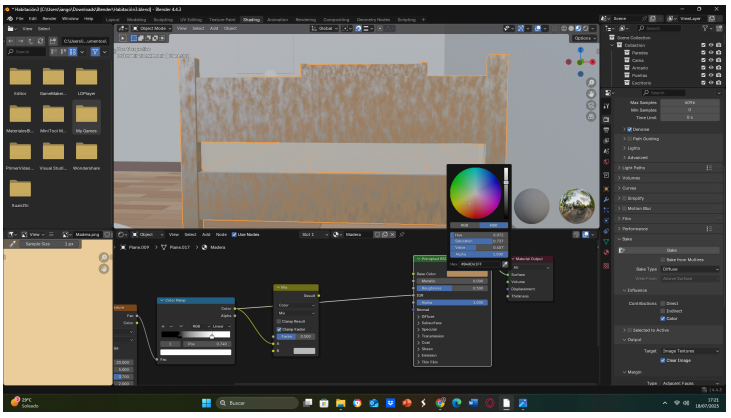
Imatge 58. Pivot d'una porta. *Font pròpia*

Tot això em servia per preparar millor els models abans de portar-los a Unity. Preferia fer les animacions directament allà, ja que eren molt senzilles i així podia tenir-ho tot més organitzat dins del motor. D'aquesta manera, el flux de treball era més net i em resultava més fàcil controlar i ajustar els moviments un cop dins del joc.

Quan vaig acabar el modelatge, quedava encara una part molt important: aplicar textures i materials. Aquesta fase fa que els objectes i els escenaris semblin més creïbles i atractius. Tot i ser un pas posterior al modelatge, no es pot separar, ja que és el que acaba definint el to i l'ambient del món del joc i dona vida a tot el que he creat.

5.2.1.2 Texturització i materials de l'escenari

Un cop finalitzada la fase de modelatge 3D, calia donar color i personalitat a tots els objectes que havia creat. Aquí entra en joc la texturització, que és el procés de definir com es veuran visualment les superfícies dels models: colors, materials, detalls com la fusta d'un moble o l'aspecte del terra.



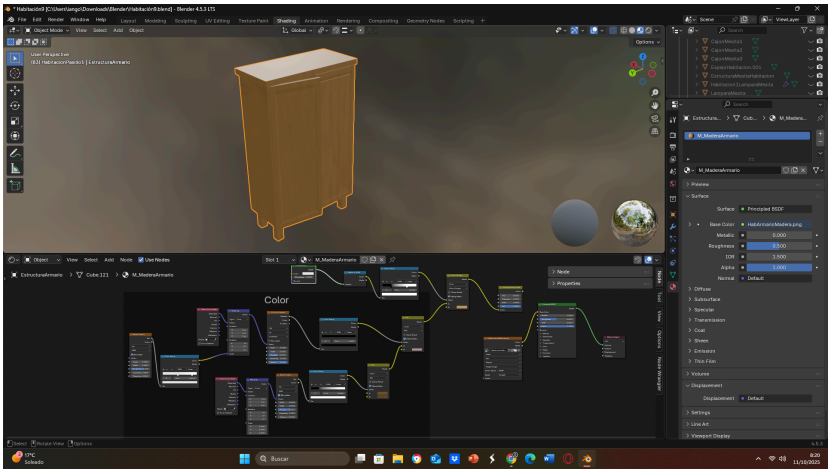
Imatge 59. Texturització amb nodes. *Font pròpia*

Com que el meu videojoc té un estil visual de tipus *toon*, molts dels objectes no necessiten textures realistes, sinó que funcionen simplement amb colors plans. Això vol dir que, per exemple, una cadira pot tenir un material vermell per al respatller i un altre de negre per a les potes. Per aconseguir-ho, havia de crear un material, donar-li un color concret, seleccionar les cares del model on volia aplicar-lo i assignar-lo. Si un objecte requeria més d'un color, repetia el procés fins a tenir tots els materials necessaris. Aquest procediment, encara que sembli senzill, es torna repetitiu quan s'ha d'aplicar a centenars d'objectes diferents.

Imatge 60. Texturització d'una estanteria de l'habitació del jugador. *Font pròpia*

Encara que aquests materials siguin colors plans, dins del joc no es veuran així tal qual. A Unity hi aplicaré un *toon shader* que farà que tot tingui un estil més atractiu i coherent amb el to general del joc. Aquest efecte ajuda a remarcar les formes i a donar un acabat més artístic, fent que els objectes es vegin més vius i agradables visualment.

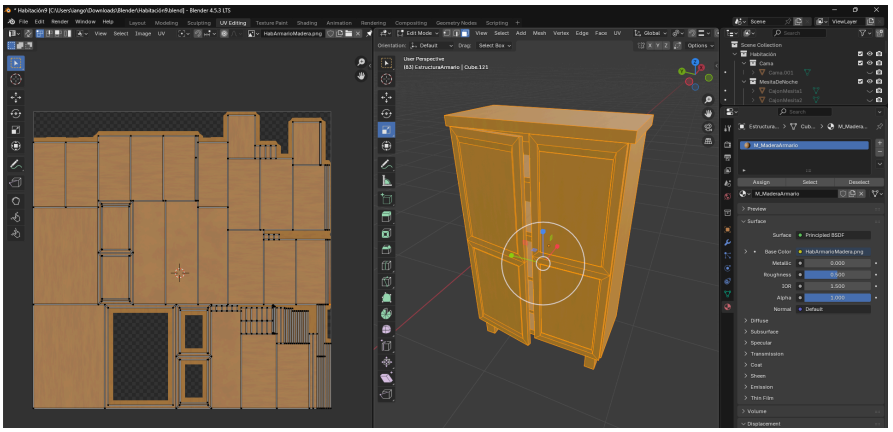
Tot i això, no tots els objectes podien resoldre's amb colors plans. Per elements com els mobles de fusta, vaig haver d'aprendre a utilitzar el sistema de nodes de Blender, que permet generar textures més complexes. A través de combinacions de nodes podia crear patrons que simulessin la beta de la fusta i, modificant paràmetres com l'escala, l'orientació o el color, aconseguia variants adaptades a cada objecte. Així, podia duplicar un mateix material base i fer petites modificacions per aconseguir resultats diferents, sense haver de repetir tota la feina des de zero. Aquest mateix mètode el vaig aplicar per a altres textures, com les dels quadres del passadís o els terres d'algunes habitacions.



Imatge 61. Texturització d'un armari de fusta amb nodes. *Font pròpia*

Ara bé, aquí apareixia un problema: Unity, el motor on s'executa el videojoc, no reconeix els nodes de Blender. Això passa perquè cadascun utilitza un sistema diferent per descriure materials i no són compatibles entre si. Per això, abans d'exportar els models, vaig haver de passar per un procés anomenat bakejat de textures.

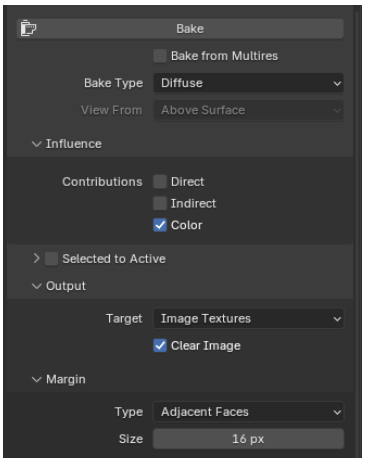
El bakejat consisteix a "cuinar" la informació d'una textura generada amb nodes i convertir-la en una imatge 2D (com un .png) que Unity pugui entendre. Abans, però, cal fer l'unwrap del model, és a dir, desplegar les cares de l'objecte sobre un pla 2D, una mica com si es tallés una caixa de cartró i s'estengués sobre una taula. Això permet que el programa sàpiga com projectar-hi la textura.



Imatge 62. Unwrap de l'armari i bakejat del material. *Font pròpia*

En el meu cas, algunes textures les creava jo directament amb nodes dins de Blender, i altres eren simplement objectes amb colors plans. Mai no vaig utilitzar textures externes, tot ho feia jo mateix. Per això, no calia que l'unwrap fos extremadament precís, ja que les textures es podien adaptar al model, però igualment era necessari per poder bakejar i exportar correctament els materials a Unity.

A l'hora de fer el bakejat, vaig triar el tipus *Diffuse*, que és bàsicament el que guarda el color de l'objecte. Dins de les opcions de contribució hi ha diverses caselles, com *Direct* o *Indirect*, que serveixen per afegir la llum directa i les ombres rebotades del render al resultat final. En el meu cas, no volia que la textura tingués res d'això, només volia el color tal qual, així que vaig desactivar-ho tot i només vaig deixar el *Color*. Tot el comportament de llum, ombres i efectes visuals el faria més



Imatge 63. Opcions de bakejat. *Font pròpia*

endavant directament a Unity, utilitzant un *shader toon* per aconseguir un estil més artístic i vistós.

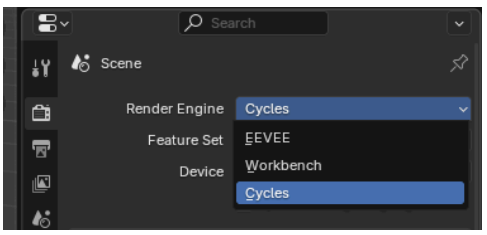
Imatge 64. Imatge obtinguda gràcies al bakejat. *Font pròpia*

En aquest procés vaig haver de decidir també la resolució de les textures. Generalment vaig treballar amb 1024x1024 píxels, que és una mida equilibrada entre qualitat i rendiment. En alguns casos concrets, com objectes grans o on el detall era molt visible, vaig utilitzar resolucions de 2048x2048. Cal tenir en compte que com més gran és la textura, més memòria i temps de càrrega necessita el joc, i per tant pot afectar el rendiment.

Imatge 65. Material amb propietats complexes. *Font pròpia*

Pel que fa a materials especials, com el vidre o les cortines, vaig haver d'acabar configurant-los directament a Unity. Això és perquè Blender només exporta el color base, però no propietats més complexes com la transparència o l'IOR (índex de refracció), que defineix com es doblega la llum en passar pel material (per exemple, l'aigua té un IOR diferent del vidre).

També vaig descobrir que dins de Blender hi ha tres motors de render: *Workbench*, *Eevee* i *Cycles*. *Workbench* és el més simple i està pensat sobretot per treballar de manera ràpida dins de la vista 3D, sense centrar-se en l'aspecte final dels materials. *Eevee* és molt més ràpid i s'utilitza principalment per previsualitzar com quedarà la il·luminació i els materials dins de Blender, però no permet bakejar textures. En canvi, *Cycles*, que és més lent però molt més precís, és l'únic que deixa fer el bakejat correctament. Per això sempre que necessitava treure les textures per exportar-les a Unity havia de treballar amb *Cycles*.

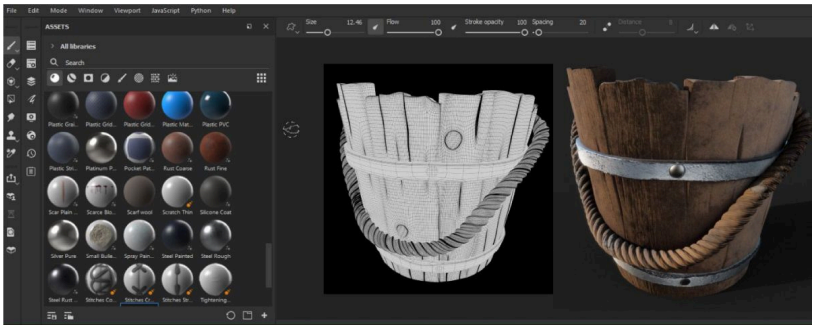


Imatge 66. Motors de renderitzat. *Font pròpia*

Aquest procés, encara que sembli tècnic, va ser llarg i repetitiu. Havia de bakejar cada objecte o material per separat i guardar les imatges corresponents, cosa que requeria temps i paciència. A més, la velocitat del bakejat depenia molt de la potència de l'ordinador, de manera que en alguns moments el procés resultava lent i

fins i tot desmotivador. Tot i això, em va permetre entendre molt millor com funciona el pas de Blender a Unity i la importància de preparar bé els actius visuals perquè el joc funcioni correctament.

Encara que vaig arribar a mirar opcions més professionals com Substance Painter, que és un programa especialitzat en la texturització i molt utilitzat dins de la indústria del videojoc, vaig descartar utilitzar-lo per una qüestió econòmica. Aquest programa requereix una subscripció d'uns 24,5 € mensuals o bé la compra d'una llicència d'uns 194,99 €, cosa que en el meu cas no em resultava viable ni justificable, ja que el meu projecte és acadèmic i no professional. Tot i així, és una eina molt recomanable per a persones que es volen dedicar al sector, ja que facilita molt el procés i ofereix resultats més realistes i detallats.



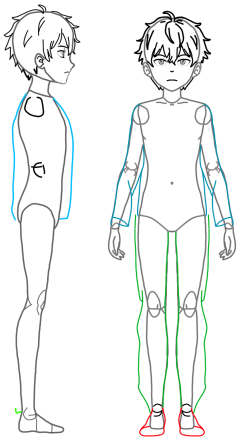
Imatge 67. Substance Painter. *Imatge extreta de: https://goo.su/YZxT*

Per aquest motiu, vaig preferir fer tota la feina de texturització directament dins de Blender. Tot i que era un procés més limitat i lent, em permetia tenir-ho tot al mateix lloc i entendre molt millor com funcionava el programa. Amb el temps, vaig aprendre a crear materials més complexos fent servir nodes, combinant diferents valors i textures per aconseguir resultats interessants sense necessitat de recórrer a programes externs.

Fer-ho tot a mà dins de Blender em va donar molt més control sobre el procés i em va ajudar a comprendre millor com cada material afectava l'aspecte final del joc. Encara que fos més laboriós, també va ser una manera de practicar i millorar, perquè cada textura o material que creava era una oportunitat per aprendre alguna cosa nova i aconseguir que el resultat final fos més propi i personal.

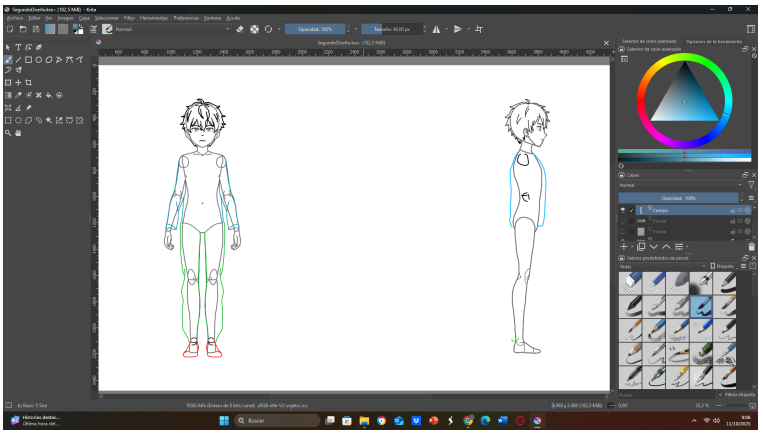
5.2.1.3 Modelatge del personatge

Abans de començar a modelar el personatge en 3D, el primer pas va ser establir unes bones referències visuals. Per fer-ho vaig utilitzar Krita, on vaig dibuixar el personatge tant de cara com de perfil. Aquests dibuixos, coneguts com a *model sheets*, em van servir de guia per mantenir unes proporcions correctes i assegurar que les formes fossin coherents des de totes les vistes.



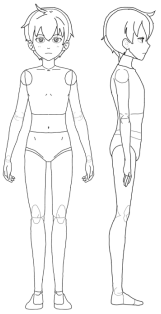
Imatge 68. Dibuixos de vista frontal i lateral del personatge fet en Krita. *Font pròpia*

Tot i així, aquest procés no va ser gens fàcil. Jo no havia dibuixat mai en digital i tampoc sabia utilitzar Krita, així que al principi em vaig haver d'adaptar al programa i aprendre a fer coses molt bàsiques. Encara que el dibuix fos senzill, em va portar temps entendre com funcionava tot i aconseguir un resultat que em servís de referència.



Imatge 69. Dibuixos de vista frontal i lateral del personatge fet en Krita. *Font pròpia*

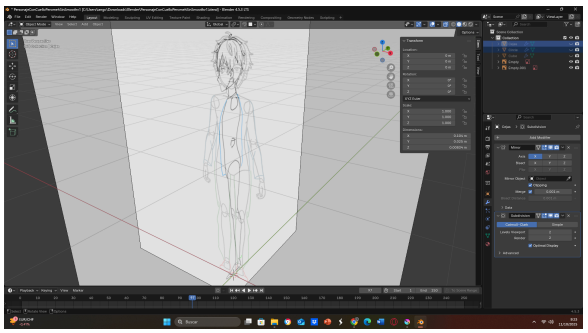
A més, vaig haver de repetir els dibuixos diverses vegades, perquè sovint no quedaven ben alineats. Per exemple, una vista podia quedar una mica més baixa que l'altra, i llavors el model 3D no tenia sentit. També vaig posar els braços massa junts en algun intent, cosa que després feia molt més difícil el modelatge. Va ser un procés de prova i error constant, on anava ajustant i corregint fins que tot encaixava bé.



Imatge 70. Dissenys inicials del personatge. *Font pròpia*

Aquests dibuixos corresponen als dissenys finals (o gairebé finals), però abans d'arribar-hi vaig fer altres models de personatges una mica diferents. Quan vaig començar, encara no tenia del tot clara la idea ni l'estil que volia aconseguir, així que vaig anar explorant i redibuixant fins que vaig trobar un disseny que

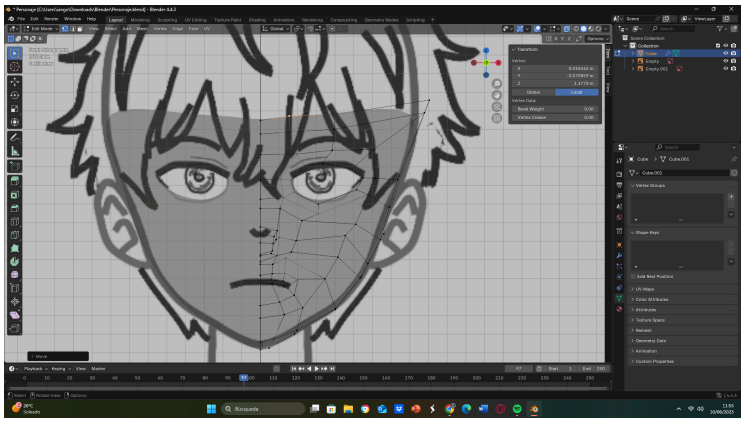
realment em convencia. Tot aquest procés em va ajudar molt a entendre millor el personatge i a preparar-me per la fase de modelatge en 3D.



Imatge 71. Importació dels dibuixos del personatge a Blender. *Font pròpia*

Quan vaig acabar els dibuixos, els vaig importar a Blender i els vaig col·locar dins de l'escena com a imatges de referència. D'aquesta manera podia comparar constantment el model 3D amb el disseny original i assegurar-me que tot quedava fidel. Per a mi aquest pas va ser essencial, ja que em va ajudar a mantenir una coherència visual i em va estalviar molts errors que probablement hauria comès si hagués començat directament en 3D sense punts de partida clars.

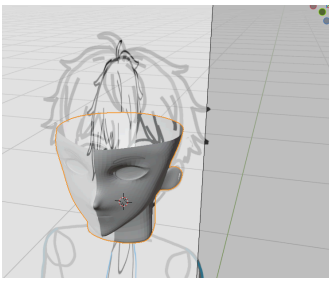
Un cop vaig tenir les referències preparades, vaig començar a construir la base del personatge en 3D. Vaig decidir treballar amb modelatge poligonal i no amb escultura perquè volia aprendre bé la topologia i tenir més control sobre la geometria. La topologia és molt important, ja que d'ella depèn que després el personatge es pugui animar correctament. En zones com els ulls, el nas i la boca vaig haver de ser especialment curós, creant bucles de geometria nets que permetessin deformacions naturals.



Imatge 72. Creació del personatge amb modelatge poligonal. *Font pròpia*

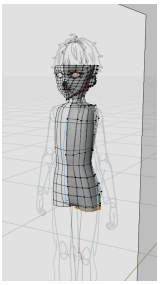
La veritat és que el cap em va portar molta feina. Sovint no m'agradava la forma i l'havia de refer diverses vegades fins a aconseguir un resultat que em convencés. A més, en aquell moment encara no estava gaire familiaritzat amb Blender i em feia respecte tocar segons quins vèrtexs o aplicar modificadors, perquè no sabia si després podria tornar enrere o si el resultat es faria malbé.

Vaig començar definint la boca, després els ulls, el nas i la mandíbula, i a partir d'aquí vaig acabar de donar forma a tot el cap. També vaig modelar el coll i la part posterior, deixant-ho preparat per poder connectar amb el cos. Els ulls, pestanyes i celles els vaig fer per separat, pensant ja en possibles animacions futures. En el cas dels ulls, com que volia aconseguir un estil proper a l'anime, vaig utilitzar formes simples, amb un iris circular i pocs detalls, ja que no serien necessaris amb el tipus de sombrejat que volia aplicar.

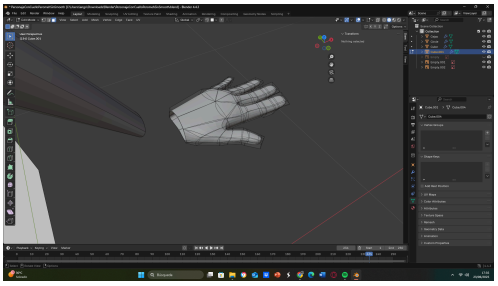


Imatge 73. Creació del rostre del personatge. *Font pròpia*

Quan vaig acabar el cap, vaig continuar amb el cos, començant pel tors i després les cames i el maluc. Primer vaig fer el cos sense roba perquè així la base quedava molt més neta i després era més fàcil afegir-hi la roba o altres detalls. Vaig seguir les imatges de referència i anava extruint i ajustant la forma fins que tot quedava bé. En zones com els colzes, els genolls o les espatlles, vaig haver de posar alguns triangles perquè, quan s'animin aquestes parts en Blender, la malla es deformi bé. Si només utilitzés quadrats, al moure's es deformaria d'una manera molt estranya i no quedaria natural.



Imatge 74. Creació del cos del personatge. *Font pròpia*



Imatge 75. Creació de les mans. *Font pròpia*

Les mans les vaig fer a part, amb una imatge de referència per guiar-me. Les vaig modelar separades del braç i després les vaig unir quan ja tenien la forma correcta. Havia d'anar ajustant perquè encaixessin bé i perquè després, quan animés el personatge, es moguessin de manera fluida. Les mans van ser una de les parts més complicades, sobretot per donar-los una forma que quedés bé des de tots els angles.

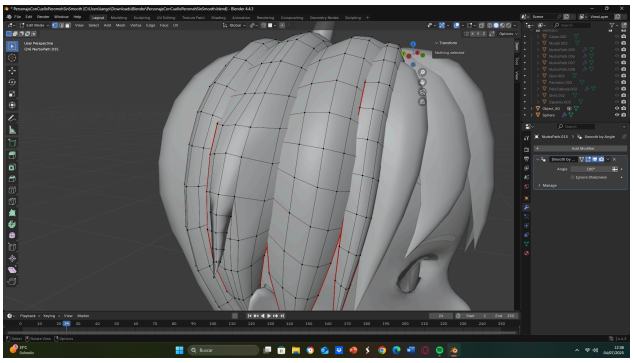
Imatge 76. Cos del personatge. *Font pròpia*

Les cames les vaig fer de la mateixa manera, extruint des del maluc i anant modelant la forma fins als peus. També vaig posar triangles als genolls per evitar deformacions rares a l'hora d'animar. Els peus no tenen dits separats perquè no

calia, però sí que els vaig modelar amb la forma general per mantenir l'estructura del cos coherent.

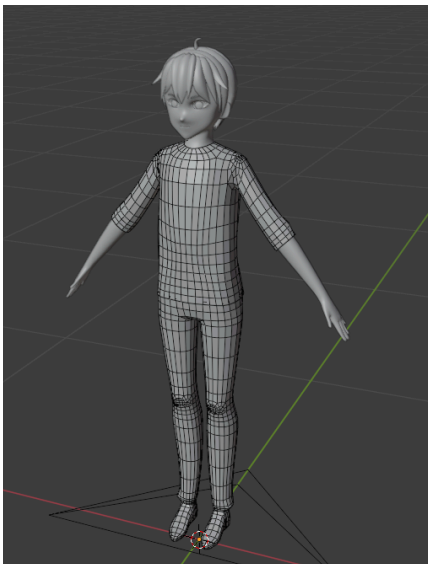
Imatge 77. Primer intent de modelatge dels cabells. *Font pròpia*

Un cop vaig tenir el cos bàsic, vaig passar a afegir els detalls característics del personatge. El cabell va ser, sens dubte, la part més complicada. Vaig provar moltes tècniques diferents, però cap m'acabava de convèncer, sobretot perquè hi ha molt pocs tutorials de cabells masculins d'aquest estil a Blender. Finalment vaig decidir fer-lo amb curves (paths), creant metxes triangulars que després convertia en malles. A partir d'això, simplement anava movent els vèrtexs fins que la forma quedés com volia. Va ser molt assaig i error, provant formes i ajustos fins que quedés bé. Al final vaig trigar uns sis o set dies només amb el cabell fins que vaig aconseguir un resultat que m'agradés.

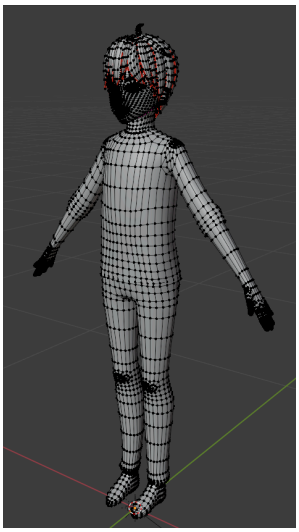


Imatge 78. Disseny gairebé definitiu dels cabells. *Font pròpia*

Per a la roba vaig optar per un disseny senzill, amb una camiseta, uns pantalons i unes sabates bàsiques. El procés va ser duplicar les parts del cos i adaptar-les: la samarreta la vaig fer a partir del tors, els pantalons duplicant i modificant les cames, i les sabates a partir dels peus, donant-los una mica de volum. Tot plegat amb un estil simple, sense afegir detalls massa complexos.



Imatge 79. Disseny gairebé definitiu dels cabells. *Font pròpia*

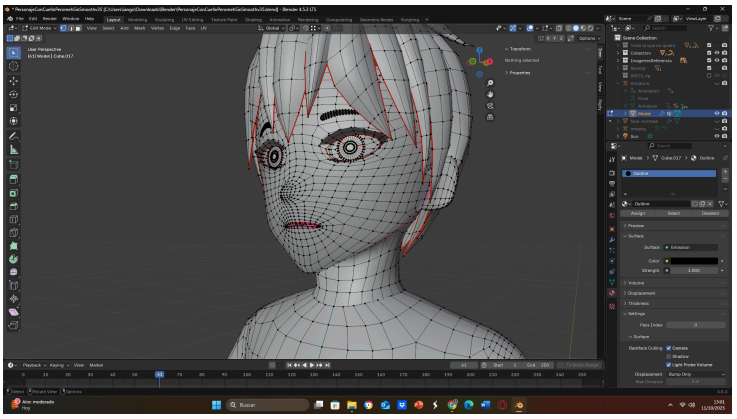


Imatge 80. Unificació de tots el objectes. *Font pròpia*

Quan ja tenia el personatge complet vaig eliminar totes les parts del cos que quedaven cobertes per la roba i vaig unir totes les peces visibles en un

sol objecte. Això em va ajudar a optimitzar el model, a millorar el rendiment i a preparar-lo per al rigging i les animacions.

A l'hora de preparar el personatge per al següent pas, també vaig haver de crear seams (les línies vermelles), que són com unes costures virtuals que serveixen per "obrir" la malla i desplegar-la en un pla 2D. Gràcies a aquestes divisions, és possible fer l'UV unwrap de manera més ordenada i controlar quina part del model correspon a cada zona de la textura. Per exemple, podia separar el cap, el tors o les cames i desplegar-los de manera independent, cosa que em facilitava molt el procés de texturització posterior i evitava deformacions estranyes.



Imatge 81. Creació de seams per poder-ho desplegar posteriorment. *Font pròpia*

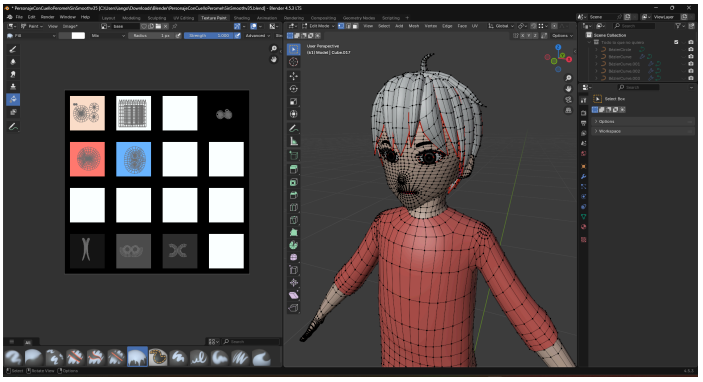
En total, el procés de creació del personatge em va portar 22 dies, durant els quals vaig treballar-hi cada dia. Una de les coses que més vaig aprendre va ser a ser pacient i a repetir tantes vegades com calgués fins a obtenir un resultat satisfactori. Malgrat les dificultats, especialment amb el cap i el cabell, aquest projecte em va servir per conèixer molt millor Blender i entendre la importància de la topologia, de l'optimització i de treballar sempre amb bones referències.

5.2.1.4 Texturització, shading i outline del personatge

Un cop vaig acabar el modelatge del personatge, el següent pas va ser començar a donar-li color i definir-ne l'aparença visual. En gràfics 3D, aquest procés s'anomena texturització, i consisteix a aplicar imatges en 2D (les textures) sobre la superfície del model. Per fer-ho cal utilitzar un procediment anomenat UV mapping, ja explicat anteriorment.

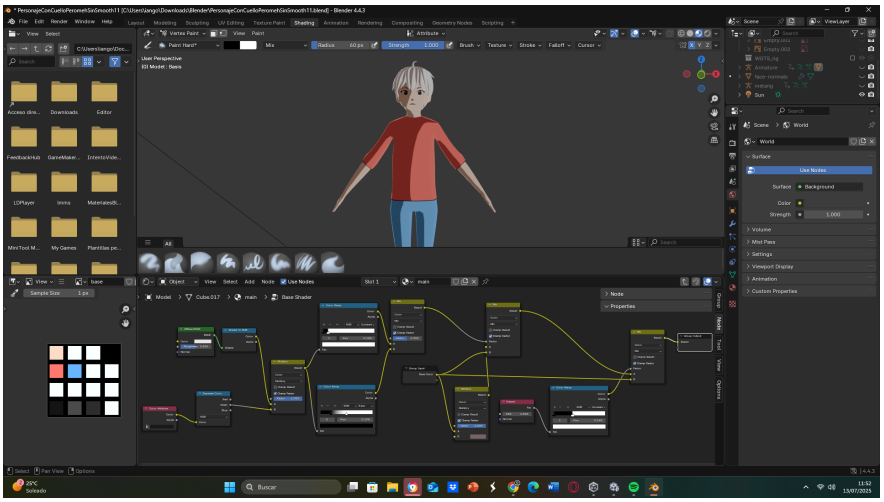
En projectes professionals, el més habitual és fer mapes UV detallats quan el model ja està completament tancat, ja que qualsevol canvi de geometria després obliga a

repetir-los. Jo, en canvi, vaig optar per una solució més senzilla i ràpida: crear una textura petita amb quadrats de colors bàsics i després assignar cada part del model al quadrat del color corresponent. Aquest sistema em permetia treballar de manera iterativa, canviar colors amb facilitat i, sobretot, reduir molt la mida de les textures (uns 16x16 píxels). A més, aquest mètode no només era pràctic, sinó que també encaixava perfectament amb l'estil toon que volia aconseguir, basat en colors plans i superfícies simples.



Imatge 82. Texturització del personatge. *Font pròpia*

Per completar l'aparença, no n'hi ha prou amb les textures: també cal definir com reacciona el model a la llum. Aquí entren en joc els shaders, que són seqüències que especifiquen com una superfície ha d'interactuar amb la llum i les ombres. En Blender, els shaders es creen amb un sistema de nodes que es connecten entre si fins a obtenir el resultat desitjat. Jo vaig voler experimentar amb un toon shader, que genera ombres dures i definides, donant al personatge un estil més proper a l'animació 2D. Em vaig inspirar en estils de videojocs com *The Legend of Zelda: The Wind Waker*, on aquest acabat visual és molt característic.



Imatge 83. Creació del shader a través de nodes. *Font pròpia*

Un element clau per aconseguir aquest estil va ser treballar amb normals personalitzades. Les normals són vectors que indiquen la direcció de cada cara respecte a la llum, i Blender les calcula per defecte. Tot i això, en un estil cartoon,

deixar les normals per defecte pot provocar ombres estranyes, sobretot a la cara. Per resoldre-ho vaig utilitzar el modificador *Data Transfer*, que permet copiar normals d'un altre objecte (com una esfera molt suau) i transferir-les al cap del personatge. D'aquesta manera, vaig aconseguir que les ombres facials fossin més coherents amb l'estètica que buscava.

Imatge 85. Rostre amb normals personalitzades. *Font pròpia*

Imatge 84. Rostre sense normals personalitzades. *Font pròpia*

També vaig utilitzar vertex paint per afegir ombres personalitzades. El vertex paint permet assignar colors directament als vèrtexs d'un model, sense fer servir textures. Aquests colors després es poden aprofitar dins del shader per controlar zones concretes d'ombra. Així vaig poder reforçar detalls com el nas o algunes parts de la cara, aconseguint més control i expressivitat.

Imatge 86. Utilització de vèrtex paint per fer obres al nas. *Font pròpia*

Imatge 87. Outline del personatge. *Font pròpia*

Finalment, vaig experimentar amb la creació de l'*outline* (el contorn negre que envolta el model). Aquest efecte ajuda molt a remarcar el personatge i donar-li més presència dins l'escena. Per aconseguir-lo vaig utilitzar la tècnica de l'*Inverted Hull*, que consisteix a duplicar la malla, invertir-ne les normals i aplicar-li un material negre, creant així un contorn visible al voltant del model. En el meu cas, només volia remarcar el contorn exterior del personatge, no l'interior, ja que si els ulls, la boca o altres detalls interns també tinguessin *outline*, el resultat es veuria

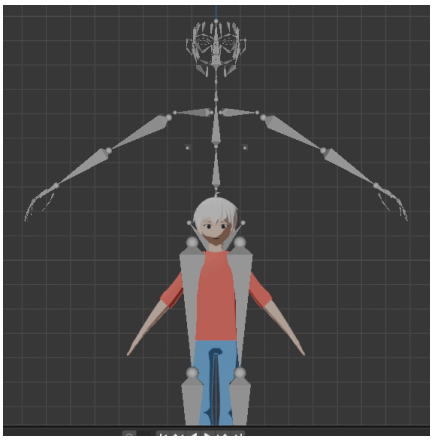
Imatge 88. Utilització de weight paint per treure outlines en parts com ulls o boca. *Font pròpia*

estrany i descurat. Tot i que aquest mètode duplica el nombre de polígons, en el meu projecte no representava cap problema pel tipus d'estil visual que buscava.

És important remarcar que tant el toon shader com l'outline els vaig crear només dins de Blender per comprovar si l'aspecte general del personatge funcionava visualment, ja que aquests efectes no es poden exportar directament a Unity. Per tant, sabia que després els hauria de recrear des de zero dins del motor. El que sí que vaig fer va ser bakejar les textures en imatges 2D perquè Unity les pogués entendre, mantenint els colors i l'estil del personatge.

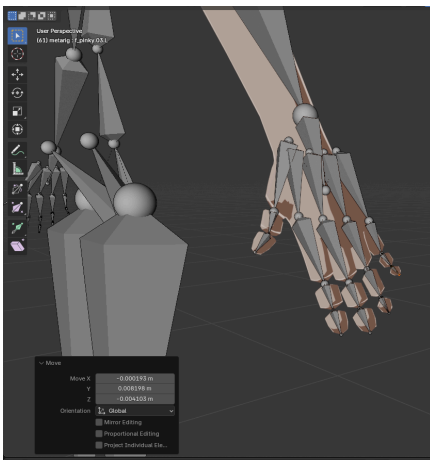
5.2.1.5 Rigging del personatge i animacions

Per fer el rigging del meu personatge vaig utilitzar Rigify, un complement que ja ve incorporat a Blender i que facilita molt aquest procés. El rigging consisteix a crear un esquelet virtual dins del model 3D, de manera que després es pugui animar. En altres paraules, és com donar-li "ossos" al personatge perquè pugui moure's. Fer-ho manualment seria molt complicat, ja que caldria moure cada os un per un, però Rigify permet generar una estructura bàsica anomenada metarig, que després es pot convertir automàticament en un rig complet i funcional.



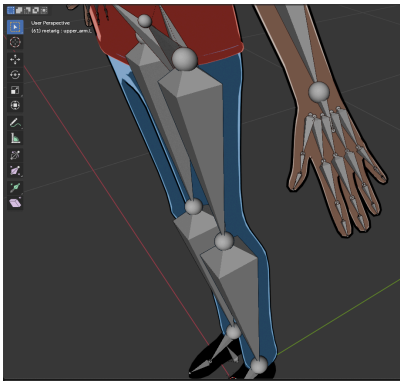
Imatge 89. Armature. *Font pròpia*

Aquest sistema utilitza una tècnica molt útil anomenada cinemàtica inversa (IK). Gràcies a això, no cal animar cada articulació per separat, simplement movent la mà, per exemple, la resta del braç s'ajusta automàticament seguint el moviment. Això fa que el procés sigui molt més ràpid i natural. Altres programes o sistemes també ofereixen eines de rigging, però vaig triar Rigify perquè és gratuït, ja ve amb Blender i, a més, té molta documentació i suport.



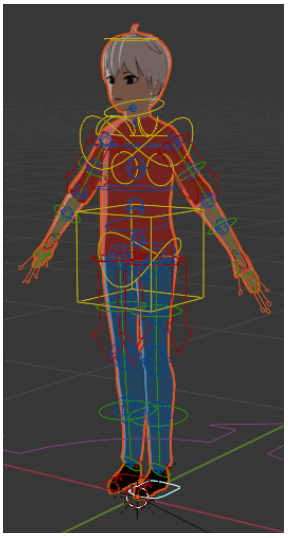
Imatge 90. Alinear ossos del metarig al model. *Font pròpia*

Un cop vaig generar el metarig, vaig dedicar força temps a alinear cada os amb la malla del personatge per assegurar que els moviments fossin coherents. També vaig eliminar els ossos de la cara, ja que per a les expressions vaig preferir fer servir shape keys, que permeten deformar la cara directament i controlar millor els gestos. A més, vaig afegir alguns ossos secundaris al cabell, especialment als mechons més visibles, per donar-los un moviment més suau quan el personatge es mogui.



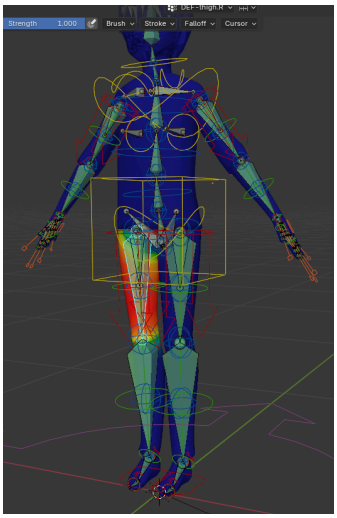
Imatge 91. Alinear ossos del metarig al model. *Font pròpia*

Per mantenir-ho tot organitzat, vaig crear grups d'ossos i vaig afegir els sufixos *.L* i *.R* per indicar el costat esquerre i dret. Això em va ajudar molt a mantenir la simetria i a accelerar el procés de *weight painting*. Finalment, vaig connectar cada cadena d'ossos amb el cos principal, de manera que, quan el personatge es desplaça o es gira, totes les parts segueixen correctament sense desplaçar-se fora del lloc.



Imatge 92. Creació d'un rig més compelt. *Font pròpia*

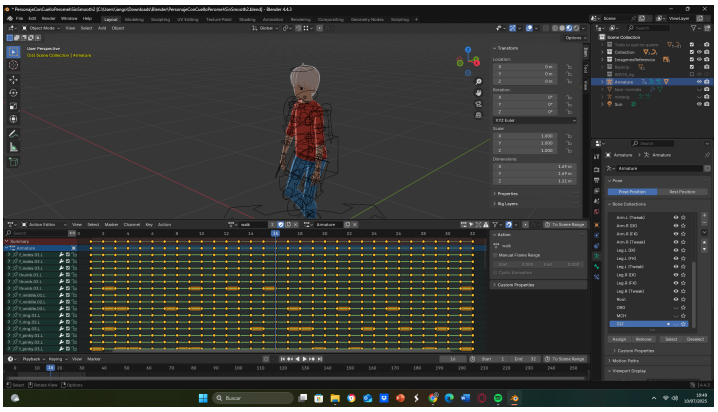
Quan ja tenia els ossos ben alineats, vaig prémer el botó Generate Rig, que crea automàticament el rig complex amb tots els controls i la cinemàtica inversa activada. Això fa que el rig estigui funcional, amb tots els ossos i controls preparats per animar. Tot i això, sense fer el weight painting, la malla no es mouria amb els ossos, és a dir, el personatge no es podria animar encara.



Imatge 93. Assignació de pesos a la malla. *Font pròpia*

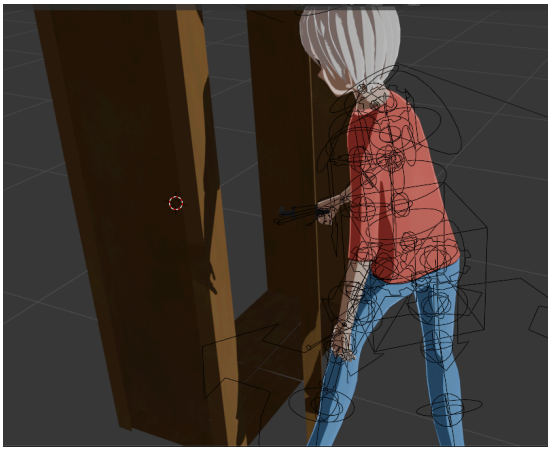
El *weight painting* és la tècnica que serveix per assignar quina part de la malla es mou amb cada os. Un valor de 1 significa que la zona es mou completament amb aquell os, mentre que un 0 significa que no es mou gens. Vaig separar les parts de la malla per treballar-les de manera individual i assegurar-me que cada os només afectés les

zones que tocava. Blender té una assignació automàtica que serveix de base, però després vaig ajustar-ho manualment per aconseguir moviments més naturals. Durant tot el procés, vaig provar el rig en Pose Mode per veure com es deformava la malla amb els moviments.

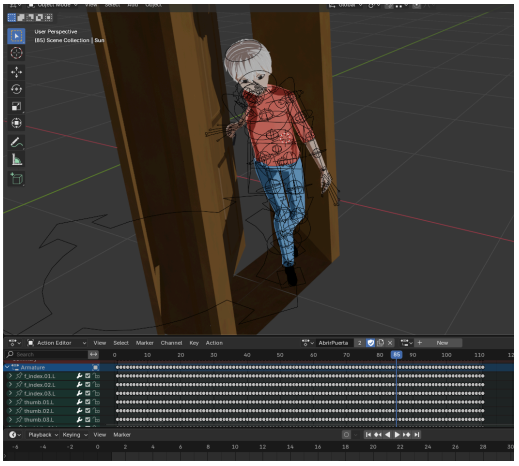


Imatge 94. Animacions en Blender. *Font pròpia*

Un cop vaig comprovar que la malla es deformava correctament amb el weight painting, vaig començar a treballar en les animacions del personatge. Primer vaig intentar crear-les directament a Blender, fent caminar, córrer i algunes poses bàsiques, però el resultat no m'agradava gens. Les animacions quedaven estranyes, poc naturals i no sabia com sincronitzar correctament els frames. Per exemple, en una animació de caminar cal que es pugui loopear i que el moviment entre el peu esquerre i el dret tingui el mateix nombre de frames. Al principi no ho tenia clar i això feia que tot quedés malament. Tot era molt d'assaig i error i, encara que repetís el procés, no aconseguia un resultat que em convences.



Imatge 95. Animació d'obrir i tancar una porta. *Font pròpia*



Imatge 96. Animació d'obrir i tancar una porta. *Font pròpia*

Per això vaig decidir utilitzar Mixamo, una eina en línia que ofereix animacions predissenyades per personatges 3D i ja sincronitzades amb esquelets. Vaig baixar animacions base i les vaig importar a Blender, però no les vaig usar tal qual. Vaig ajustar alguns frames i ossos perquè els moviments s'adaptessin millor al que volia pel meu

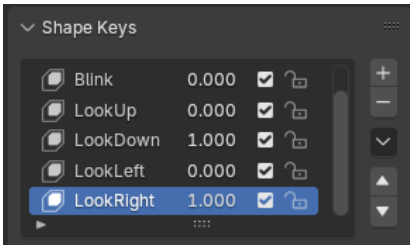
personatge. L'avantatge de Mixamo és que et proporciona una base correcta de keyframes, és a dir, l'esquelet ja està sincronitzat i les animacions es poden loopear sense problemes. Això em permetia començar amb una animació funcional i després modificar-la segons les meves necessitats.

Per assignar les animacions de Mixamo al meu rig de Rigify vaig fer servir Expy Kit. Així podia mantenir els ossos de control del meu rig i ajustar els moviments fins que quedessin naturals. Tot i així encara va ser un procés de molt assaig i error. Calia temps per aconseguir que cada moviment es veiés correcte i coherent, especialment en transicions com caminar, córrer o fer gestos.

Per les expressions facials vaig fer servir shape keys, que són una manera de moure parts concretes de la cara sense haver de tocar tota la malla. En el meu cas només vaig crear cinc shape keys essencials: mirar amunt, mirar avall, mirar a l'esquerra, mirar a la dreta i parpellejar. La idea era crear uns estats bàsics del personatge que es poguessin utilitzar dins Unity perquè es veiés natural quan es mirés a través de miralls en primera persona. Com que el joc és en primera persona, no calia fer expressions molt complexes, només calien les essencials per que el personatge tingués una mica de vida i realisme.

Imatge 97. Expressions facials. *Font pròpia*

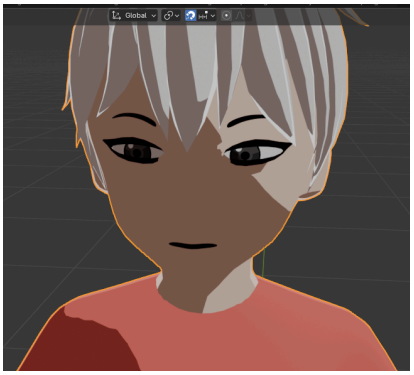
Cada shape key té un valor que va de 0 a 1. Quan està a 0 no hi ha cap deformació i quan està a 1 la deformació s'aplica completament. Això em permet controlar exactament com es mou cada part de la cara i ajustar la intensitat de cada moviment per fer-lo més suau. Per exemple, el parpelleig no és instantani sinó gradual, i els moviments dels ulls es poden combinar amb mirar amunt o a l'esquerra per crear gestos més naturals.



Imatge 98. Shape Keys. *Font pròpia*

També vaig duplicar i mirallejar les formes per mantenir la simetria de la cara i assegurar que totes les animacions facials funcionessin correctament amb el rig.

Així, qualsevol moviment del personatge es veu coherent i els petits detalls com el parpelleig o el moviment dels ulls no trenquen la naturalesa del model.



Imatge 99. Combinació de shape keys per crear noves expressions facials. *Font pròpia*

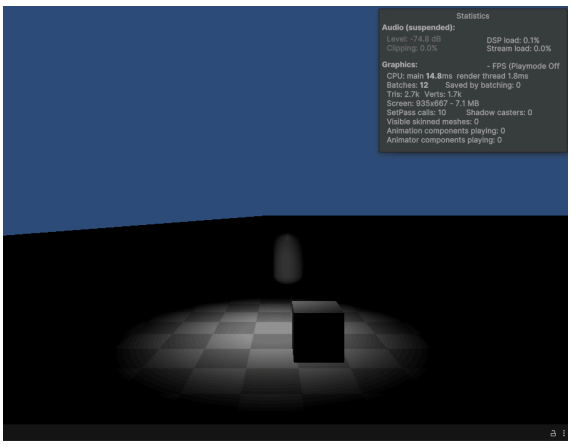
Tot aquest procés em va permetre tenir un rig complet i funcional dins de Blender, adaptat al meu personatge, amb els moviments bàsics i les expressions facials correctament configurades, llest per al joc i amb total control sobre com es mou i reacciona.

5.2.2 Implementació i mecàniques en Unity

5.2.2.1 Primer contacte amb Unity

Unity va ser la primera eina que vaig descarregar per començar aquest projecte, fins i tot abans de posar-me amb Blender. El motiu és que volia entendre bé com funcionava un motor de videojocs, ja que era el meu primer contacte amb aquest món. En aquell moment, Unity em va servir per familiaritzar-me amb la interfície, els menús i, sobretot, amb les possibilitats que ofereix a l'hora de crear jocs interactius.

Al principi vaig començar fent proves molt senzilles, com crear materials bàsics, col·locar llums, construir petits escenaris amb formes geomètriques simples i programar scripts en C#. El primer que vaig fer va ser un personatge amb forma de pastilla al qual li vaig donar moviment. Tot això ho vaig aprendre seguint tutorials de YouTube i sobretot amb molta pràctica. Encara que aquest experiment no formava part directa del videojoc final, em va servir per entendre les bases com crear scripts, afegir-los als objectes, configurar col·lisions i provar-ho tot dins de la mateixa escena. A partir d'aquell primer script vaig



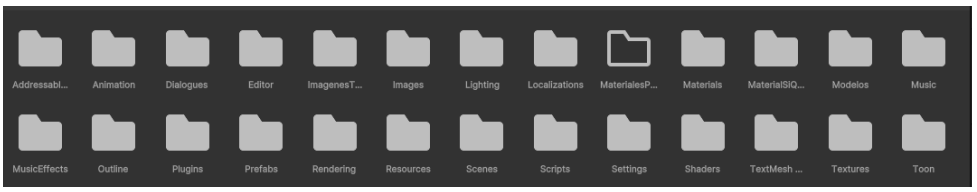
Imatge 100. Proves amb el funcionament del motor. *Font pròpia*

anar millorant el codi, corregint errors i afegint noves funcions fins que es va convertir en la base del sistema de moviment del personatge dins del joc.

També vaig anar experimentant amb sistemes que després sí que em serien útils per al joc real, com la interacció amb objectes, els diàlegs, l'outline per remarcar elements importants o fins i tot la creació d'una petita UI (per exemple, el punt al centre de la pantalla que indica cap a on mira el personatge). Molts d'aquests elements els vaig fer servir inicialment només com a proves, però em van donar la base per entendre el flux de treball a Unity i per poder aplicar-ho més endavant al projecte definitiu.

5.2.2.2 Organització del projecte i escenes

Quan vaig començar a treballar amb el projecte dins de Unity, una de les primeres coses que vaig fer va ser organitzar bé totes les carpetes. A mesura que un joc creix és molt fàcil que tot quedi desordenat, així que vaig separar-ho en carpetes com *Scripts*, *Models*, *Materials*, *Textures*, *Shaders*, *Prefabs*, *Sounds* i *Particles*. També hi guardava les imatges que havia exportat de Blender, com les textures bakejades. D'aquesta manera sempre sabia on havia d'anar cada element i no perdia temps buscant arxius.



Imatge 101. Organització per carpetes. *Font pròpia*

Pel que fa als models, tant el personatge com l'escenari els vaig exportar de Blender en format *.fbx* i després els vaig importar a Unity. Un cop dins, els vaig convertir en prefabs. Els prefabs serveixen per tenir una mena de plantilla reutilitzable d'un objecte i mantenir sempre la mateixa configuració. En el meu cas, l'escenari ja estava complet a Blender i no necessitava dividir-lo en peces petites ni duplicar portes o mobles, així que el vaig posar com un únic prefab. Tot i això, també en vaig fer servir per altres coses concretes, com el personatge o els diàlegs. Per exemple, en el cas dels diàlegs, tenia un prefab de text que ja portava la font i la mida

configurades, i així totes les línies quedaven iguals sense haver d'ajustar-les cada vegada.

Un altre punt clau van ser les escenes. A Unity, una escena és com un espai de treball o un nivell on tens tots els elements col·locats. Jo en vaig utilitzar tres:

- La primera només la vaig fer servir per experimentar: objectes geomètrics simples i alguns scripts molt bàsics per entendre com funcionava Unity.
- La segona era on provava la jugabilitat i la programació. Aquí no hi havia materials ni efectes, només servia per comprovar que els codis funcionaven bé.
- La tercera va ser l'escena definitiva del videojoc. Un cop veia que alguna cosa funcionava a la segona, ho afegia aquí amb tots els detalls: materials, il·luminació, partícules, shaders i la resta d'elements finals.

Treballar d'aquesta manera em va donar molta seguretat, perquè podia provar coses sense por de fer malbé la versió final. Entre l'organització de carpetes, l'ús dels prefabs i la separació d'escenes vaig aconseguir tenir el projecte molt més ordenat i fàcil de gestionar.

5.2.2.3 Mecàniques i comportament del videojoc

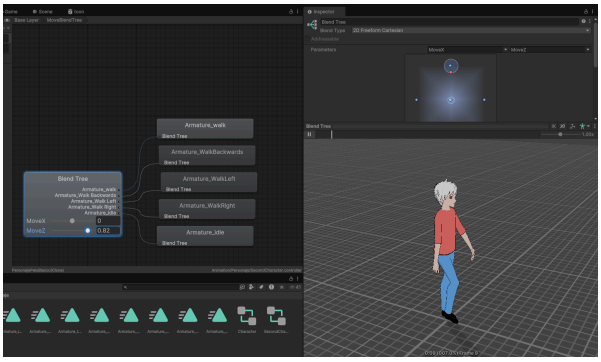
5.2.2.3.1 Jugador i control

Per controlar el jugador vaig crear diversos *scripts* que treballen de manera conjunta i que permeten que el personatge es mogui, interaccioni amb l'entorn i tingui un aspecte més realista.

El principal és el *FirstPersonController*. Aquest *script* gestiona tot el moviment del jugador: caminar cap endavant, enrere, a la dreta i a l'esquerra, així com girar la càmera segons els moviments del ratolí. També permet limitar la mirada vertical o bloquejar el gir horitzontal quan el joc està en mode cinemàtica.

A més del moviment, aquest script s'encarrega de la interacció amb objectes i NPCs. Quan el jugador s'acosta a algun element interactuable, el sistema el detecta i permet que pugui fer-hi accions, com obrir portes o iniciar diàlegs. Tot això es

gestiona a través d'un sistema de detecció i un reticle al centre de la pantalla que indica quan hi ha alguna cosa amb la qual es pot interactuar.



Imatge 102. Moviment del jugador utilitzant *Blend Tree*. *Font pròpia*

Per fer que el moviment del jugador es vegi fluid i natural, vaig utilitzar un *Blend Tree* amb quatre animacions bàsiques: caminar cap endavant, cap enrere, a la dreta i a l'esquerra, fetes anteriorment en Blender. Això permet que, quan el personatge es mogui, les transicions siguin suaus i es vegi coherent amb la direcció que està agafant.

D'altra banda, vaig crear l'*script FaceIdleAnimator* per donar vida al rostre del personatge. Aquest controla el moviment dels ulls i el parpelleig, fent que el personatge sembli més viu i realista quan es mira en un mirall. També fa que els moviments de mirada siguin aleatoris dins d'un rang concret, aportant varietat i una mica de realisme sense necessitat d'animacions addicionals complexes.

Imatge 103. Animacions facials. *Font pròpia*

La combinació d'aquests dos scripts fa que el jugador tingui un control complet sobre el personatge, mentre que al mateix temps el personatge mostra detalls realistes com mirar al seu voltant o parpellejar, millorant molt la immersió dins del joc.

5.2.2.3.2 Sistema d'interaccions

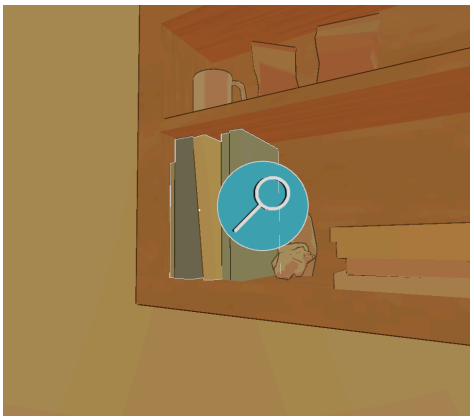
Una de les característiques principals del joc és la possibilitat que el jugador interactuï amb diferents elements de l'escenari. Per aconseguir-ho vaig programar un sistema d'interacció que està integrat directament dins del controlador del personatge (*FirstPersonController*).

Cada vegada que el jugador mira en una direcció, des de la càmera es llença un *raycast*, que és una línia invisible que comprova si al davant hi ha algun objecte amb el qual es pugui interactuar. Aquesta línia té una distància limitada, de manera que només es detecten els objectes que es troben prou a prop.

Quan el *raycast* colpeja un objecte, el codi comprova si aquest té implementada la interfície *IInteractable*. Aquesta interfície actua com una mena de contracte i assegura que qualsevol objecte que la tingui pugui oferir una acció d'interacció al jugador. D'aquesta manera puc afegir nous objectes sense haver de modificar tot el sistema, simplement aplicant aquest patró.

Imatge 104. Objectes interactuables. *Font pròpia*

Perquè l'experiència sigui clara i intuïtiva, vaig afegir un sistema visual de suport. Quan un objecte és interactuable i el jugador està prou a prop, aquest es ressalta amb un contorn (*outline*) i el reticle del centre de la pantalla canvia d'estat. Així el jugador sap en tot moment amb què pot interactuar. Si es mou o deixa de mirar l'objecte, el contorn i el reticle desapareixen automàticament i s'eviten confusions visuals.



Imatge 105. icona d'interacció en objectes interactuables. *Font pròpia*

A més, per reforçar encara més aquesta informació visual, alguns objectes tenen un *InteractionIcon*, un petit icona flotant que està lligada a l'objecte en 3D. Aquesta icona sempre gira cap a la càmera i només apareix quan l'objecte és interactuable i cap altre personatge està interactuant amb ell. D'aquesta manera, ajuda especialment els jugadors nous, ja que funciona com una mena de mini tutorial visual per entendre com funciona la interacció dins del joc. Tot i ser un element visible a la pantalla, l'*InteractionIcon* no forma part de la interfície tradicional, sinó que és una ajuda integrada dins del propi món del joc.

Quan el jugador prem la tecla E, el joc comprova quin tipus d'objecte té seleccionat. Si és un NPC, el moviment es bloqueja durant uns segons i s'activa el diàleg corresponent. En canvi, si és un objecte de l'escenari com una porta, un mecanisme o un element de *puzzle*, s'executa directament l'acció que tingui definida en el seu propi *script*.

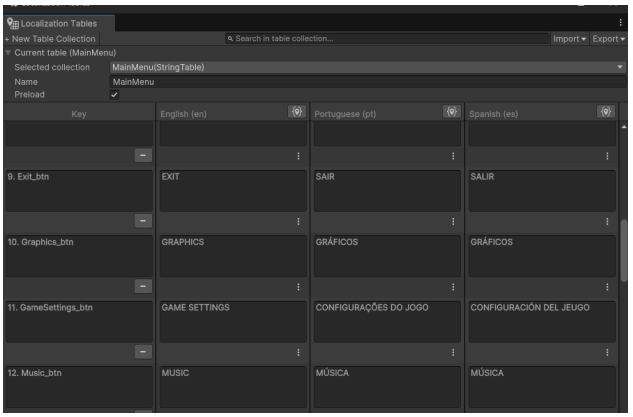
Aquest sistema té l'avantatge que és modular i escalable. Em permet utilitzar el mateix flux tant per als diàlegs com per als objectes de l'escenari o fins i tot per a elements que encara no he implementat. A més, mantenir totes les regles d'interacció centralitzades dins del controlador del jugador assegura que tot funcioni de manera coherent i sense inconsistències.

En conjunt, la interacció està pensada perquè sigui ràpida, intuïtiva i clara, tant a nivell tècnic com visual, i això ajuda a reforçar la immersió del jugador dins del món del joc.

5.2.2.3.3 Sistema de diàlegs i UI

Un altre element molt important del joc és el sistema de diàlegs, que permet que el jugador pugui parlar amb diferents personatges o interactuar amb objectes que mostren textos. Aquest sistema no només serveix per mostrar frases, sinó que també està pensat per donar opcions i, sobretot, per estar disponible en diferents idiomes.

El joc utilitza el sistema de localització de *Unity*, que permet tenir els textos guardats en taules i traduir-los de manera automàtica segons l'idioma escollit al menú de configuració. Això vol dir que una mateixa línia de diàleg no està escrita directament al codi, sinó que el programa busca la seva clau dins d'una taula i mostra la traducció



Imatge 106. Taules de localització. *Font pròpia*

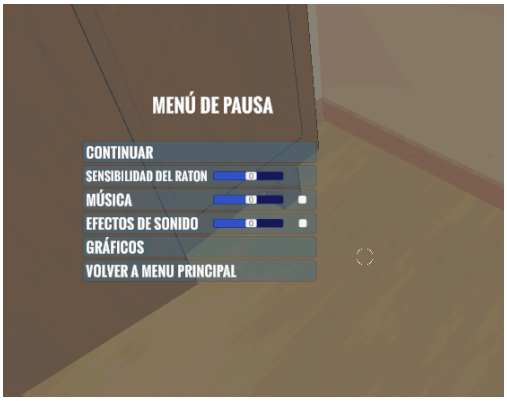
corresponent. D'aquesta manera, és molt fàcil afegir nous idiomes o modificar els textos sense tocar la programació.

Hi ha dos gestors de diàlegs principals. D'una banda, l'*ObjectDialogueManager*, que s'utilitza sobretot per objectes de l'escenari. Quan el jugador interactua amb un d'aquests objectes, apareix un panell amb el text que es va escrivint lletra a lletra, com si algú l'anés mecanografiant. Aquest efecte de "màquina d'escriure" ajuda a donar més dinamisme i fa que no sigui només un text estàtic. A més, el panell utilitza un sistema de *fade-in* i *fade-out*, és a dir, apareix i desapareix suaument, i això contribueix a fer que la presentació sigui més neta i professional.

Imatge 107. Diàlegs d'objectes. *Font pròpia*

De l'altra banda, tenim l'*NPCDialogueManager*, que està pensat per als personatges no jugables (NPCs). Aquest sistema és una mica més complex, ja que no només mostra frases, sinó que permet al jugador triar entre diferents opcions de resposta. Cada opció porta a un diàleg concret, i un cop una conversa s'ha completat, queda marcada per evitar que es repeteixi.

A nivell visual, els textos dels diàlegs dels NPCs també apareixen amb efecte de lletra a lletra i amb colors diferents segons qui parla (el jugador o un personatge). D'aquesta manera és fàcil identificar qui està parlant en cada moment. Quan comença un diàleg amb un NPC, el moviment del jugador queda bloquejat i la càmera gira suaument cap al personatge o un lloc determinat, de manera que la sensació és molt més natural, com si realment estigués parlant amb algú. Quan la conversa acaba, el control es retorna al jugador i el sistema es tanca automàticament.



Imatge 108. Menú de pausa. *Font pròpia*

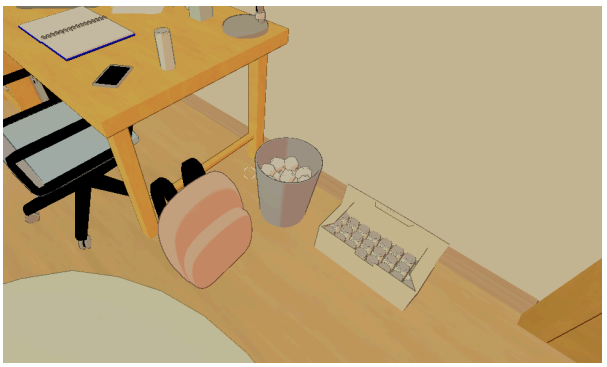
En paral·lel, tot aquest sistema de diàlegs forma part de la interfície d'usuari del joc, que actualment ja inclou el reticle del centre de la

pantalla per indicar els objectes amb què es pot interactuar i un menú de pausa accessible amb la tecla Escape. Aquest menú permet ajustar paràmetres com la sensibilitat del ratolí, el volum de la música i dels efectes de so, la qualitat gràfica o tornar directament al menú principal. Tot això està integrat amb el sistema de localització, de manera que els textos i opcions del menú sempre apareixen en l'idioma escollit.

En general, volia que el sistema de diàlegs fos fàcil d'ampliar i d'adaptar. Això em permet afegir nous textos, idiomes o millores visuals sense haver de modificar gaire la programació. D'aquesta manera puc continuar millorant el joc amb facilitat i mantenir una experiència coherent i completa per al jugador.

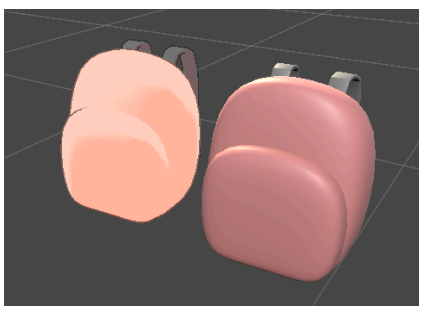
5.2.2.3.4 Estil visual i shaders

Un dels elements més importants de l'apartat visual del joc són els *shaders*. Els *shaders* són programes que indiquen com s'ha de representar cada objecte a la pantalla, definint coses com el color, la il·luminació o els efectes especials que pugui tenir.



Imatge 109. Toon shaders aplicats a objectes de l'escena. *Font pròpia*

En el meu cas vaig utilitzar un *toon shader*, que en lloc de buscar realisme aposta per un estil més semblant al dibuix animat. El funcionament es basa a simplificar les zones d'ombra i de llum, de manera que no hi ha transicions suaus sinó àrees molt més marcades i planes. Aquest tipus de representació canvia molt la sensació del joc, perquè passa de tenir una aparença bàsica i "realista barata" a tenir una personalitat pròpia i un estil molt més artístic.



Imatge 110. Comparativa d'un objecte amb i sense Toon shader. *Font pròpia*

Una de les característiques més visibles d'aquest *shader* és la possibilitat d'afegir contorns negres al voltant dels objectes i personatges. Aquests contorns

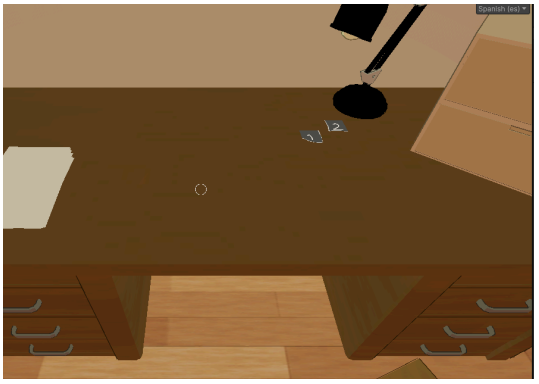
ajuden molt a diferenciar els elements de l'escena i reforcen l'estètica de dibuix animat, ja que recorden als traços d'una il·lustració. El gruix i la intensitat d'aquests contorns es poden modificar fàcilment, cosa que em dona flexibilitat per ajustar-los fins a aconseguir exactament l'efecte que busco.

A més del contorn, el *shader* permet jugar amb altres paràmetres com la duresa de les ombres, la força de la llum o fins i tot si algun objecte ha de destacar més amb un petit brillantor. Tot això es pot ajustar directament des de l'editor de *Unity*, sense necessitat de modificar el codi, i això em dona molta llibertat per experimentar ràpidament amb diferents estils.

No volia un joc que busqués realisme, sinó un que tingués una aparença pròpia i amb caràcter. El toon shader és el que em permet aconseguir-ho i, al mateix temps, transmetre millor la intenció artística darrere del projecte.

5.2.2.3.5 Puzzles i resolució d'enigmes

Un dels punts clau del joc són els puzzles, que no només serveixen per entretenir sinó també per fer avançar la història. El funcionament és bastant clar. Primer de tot, el jugador ha d'anar trobant peces amagades pel mapa. Aquestes peces es poden recollir i, un cop agafades, van directament a una mena d'inventari intern que les envia a la taula de puzzle.



Imatge 111. Resolució del trencaclosques. *Font pròpia*

Quan el jugador arriba a la taula i hi interactua, el joc entra en un mode diferent. La càmera es mou sola i es col·loca davant del puzzle, el personatge queda bloquejat i és el jugador qui, amb el ratolí, pot començar a arrossegar les peces per col·locar-les al lloc correcte. Tot això es fa dins d'una zona limitada perquè les peces no surtin disparades o quedin en llocs estranys, així que sempre es mantenen dins de l'espai pensat per resoldre'l. A més, vaig cuidar molt que tot aquest procés fos suau i natural, perquè

no volia que el canvi de càmera o el moviment de les peces trenquessin el ritme ni entorpissin l'experiència del jugador.

El sistema comprova de forma automàtica si les peces estan ben col·locades. No n'hi ha prou amb ajuntar-les una mica, sinó que el joc revisa que la posició coincideixi amb la solució que està guardada. Quan això passa i totes les peces són al seu lloc, el puzzle es dona per resolt i la porta vinculada s'obre. Les peces queden bloquejades en la seva posició final perquè el jugador no pugui moure-les un altre cop, i el puzzle queda marcat com a complet per evitar que es pugui repetir.

Si el jugador decideix sortir del puzzle abans d'acabar-lo, també pot fer-ho. En aquest cas, la càmera torna a la seva posició inicial i el personatge recupera el control de manera fluida i sense talls bruscos. Tot està pensat perquè el pas entre jugar i resoldre el puzzle sigui còmode i no trenqui la immersió.

El sistema està dissenyat de manera modular, de manera que el joc sap en tot moment quines peces té el jugador i quines encara falten. Quan totes s'han trobat i col·locat correctament, el puzzle es completa i la porta s'obre. Tot segueix la mateixa lògica d'interacció que la resta del joc, així que l'experiència és coherent i fàcil d'entendre des del primer moment.

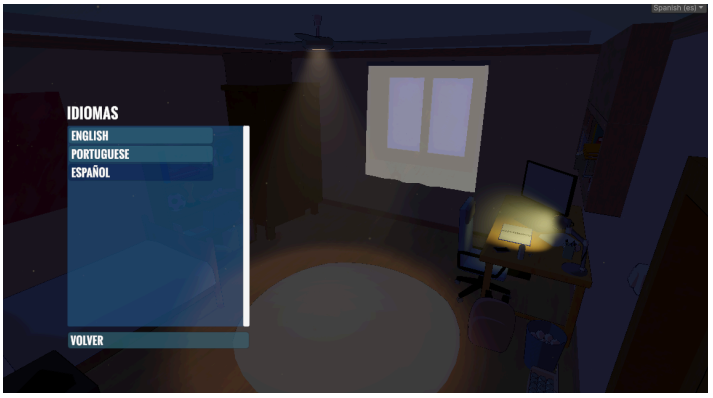
5.2.2.3.6 Menú principal

El menú és el primer que veu el jugador quan entra al joc, i per això vaig voler que fos una part molt cuidada, que transmetés bé l'estil i la sensació general del projecte. No volia fer un menú senzill només per començar a jugar ràpid, sinó una entrada que donés personalitat al joc i que fes que, des del primer moment, el jugador sentís que està entrant en un món coherent i viu.

Imatge 112. Menú Principal. *Font pròpia*

Una de les coses que més vaig cuidar va ser la sensació que dona al moure's per ell. Sempre m'ha semblat que un menú pot dir molt sobre un joc: si és brusc o poc polit, pot donar una mala impressió abans fins i tot de començar. Per això em vaig esforçar a fer que tot es notés suau, amb animacions i transicions agradables. Quan passes d'una secció a una altra, els elements no desapareixen de cop, sinó que s'esvaeixen poc a poc, i això fa que tot resulti més natural i professional. Volia que la sensació fos com si el menú t'acollís dins del joc, no com si fossin pantalles separades.

Una altra decisió important va ser fer que el fons del menú fos dinàmic. En lloc de posar-hi una imatge estàtica, vaig utilitzar la mateixa escena del joc, amb una càmera que es mou suaument i amb petits canvis de llum que donen vida al conjunt. Això crea la sensació que el joc ja està "respirant" abans d'entrar-hi, i ajuda molt a connectar el menú amb la resta del món. Aquesta idea va sorgir perquè no volia que el menú fos una pausa freda abans de jugar, sinó que formés part del mateix univers.



Imatge 113. Fons dinàmic. *Font pròpia*

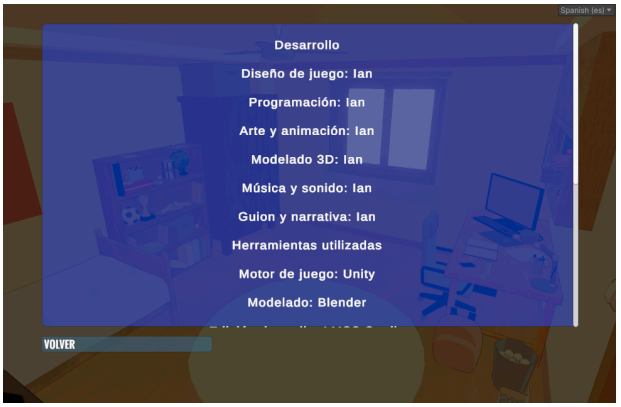
Imatge 114. Opcions de configuració. *Font pròpia*

Pel que fa a les opcions, per mi era important que el jugador pogués adaptar el joc al seu gust. Hi ha jocs on no pots tocar gairebé res, i crec que això et treu sensació de control. Aquí, en canvi, pots ajustar la brillantor, la resolució o el mode de pantalla, activar o desactivar el VSync o l'antialiasing, canviar el volum de la música i dels efectes, o escollir l'idioma. Són detalls que potser semblen secundaris, però fan que cada persona pugui jugar com li sigui més còmode. A més, totes aquestes preferències es guarden automàticament, de manera que quan tornes a entrar al joc

tot queda tal com ho havies deixat. Em semblava una manera de respectar el temps del jugador i oferir una experiència més personalitzada.

També vaig voler que el menú fos agradable d'utilitzar. Els botons tenen petits efectes visuals i sonors quan hi passes per sobre o hi fas clic. No és només per estètica, sinó perquè fa que tot sembli més viu i responsiu. Són detalls que, tot i ser subtils, ajuden molt a donar sensació de qualitat i de coherència amb la resta de l'experiència.

Finalment, vaig afegir una secció de crèdits que es mou automàticament, perquè es pugui llegir sense haver d'anar fent scroll manualment. Em semblava més elegant i d'acord amb el to general del joc. En conjunt, tot el sistema de menús busca mantenir una coherència visual i sonora, amb la mateixa tipografia, colors i ritme d'animacions que la resta del projecte. La idea és que, des del primer segon, el jugador senti que està dins del mateix món, i que fins i tot una cosa tan simple com un menú formi part de l'experiència i de la història que el joc vol explicar.



Imatge 115. Crèdits. *Font pròpia*

5.2.2.3.7 Pas del temps i transicions

Una de les parts que més vaig voler cuidar va ser el pas del temps dins del joc. Gran part de la història passa a l'habitació del protagonista i volia que aquest espai mostrés la seva rutina, que no fos sempre igual. La idea era que els canvis entre moments del dia ajudessin a transmetre la sensació de monotonia i repetició que viu el personatge. Tot el que passa en aquesta habitació reflecteix una vida que avança sense grans canvis, però amb petits detalls que van apareixent i que deixen intuir que alguna cosa està començant a canviar.

Per aconseguir-ho vaig dividir el dia en tres moments principals: matí, tarda i nit. Cada un té la seva pròpia il·luminació, ambient i sons, de manera que cada escena

té un to diferent. Al matí la llum és més clara i freda, a la tarda tot es torna més càlid i tranquil, i a la nit la llum baixa gairebé del tot i només queden uns punts de llum suaus que creen una atmosfera més íntima. Vaig intentar que aquests canvis fossin subtils, perquè el jugador els percebés sense que li trenquessin el ritme.

Imatge 116,117 i 118. Matí, tarda i nit. *Font pròpia*

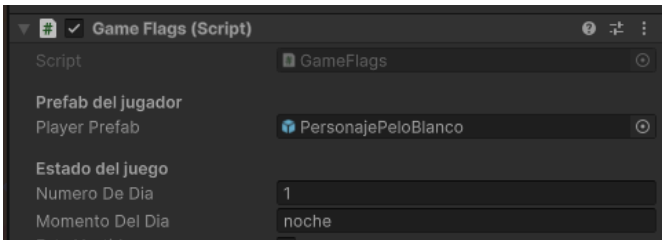
El sistema que gestiona aquests canvis és el *DayEnvironmentController*, responsable de controlar la intensitat i el color de la llum, els sons ambientals i altres paràmetres visuals. En paral·lel, i com vaig explciar abans, al menú principal hi ha una versió adaptada d'aquest sistema, anomenada *MenuDayCycle*, que s'encarrega de donar vida al fons i mantenir aquesta mateixa sensació de món viu fins i tot abans de començar la partida.

Tot aquest sistema serveix per representar la rutina del protagonista i fer que el pas del temps sigui una part més de la narrativa. No volia que fos només una qüestió estètica, sinó una manera de mostrar com el personatge viu atrapat en un cicle que es repeteix. A mesura que avança la història, alguns canvis s'aprecien una mica més i preparen l'ambient per al moment en què tot començarà a transformar-se.

Crec que aquest sistema dona molta identitat al projecte. Ajuda a que l'habitació tingui vida pròpia, a que el jugador senti el pas dels dies i entengui millor la situació del protagonista. Al final, és una manera senzilla però efectiva de transmetre la seva rutina i com, a poc a poc, aquesta comença a trencar-se.

5.2.2.3.8 Sistema global de gestió del joc

Dins del joc hi ha molts sistemes que treballen alhora: el pas del temps, els diàlegs, les interaccions, els puzzles i altres elements que han d'estar sempre coordinats entre si. Perquè tot això funcioni de manera estable i coherent, cal una base comuna capaç de guardar i organitzar tota la informació que necessita el joc. Aquesta funció la compleix el *GameFlags*, un script intern que, tot i no ser visible per al jugador, és fonamental per al correcte funcionament de tot el projecte.



Imatge 119. Informació que pot guardar *GameFlags*. *Font pròpia*

El *GameFlags* s'encarrega de guardar informació relacionada amb l'estat general del joc. Desa dades com si el jugador ha realitzat determinades accions, si ha completat algun puzzle, si és de dia o de nit o si certs esdeveniments ja han passat. Tot això queda registrat automàticament i permet que el joc recordi l'estat de cada element, fins i tot quan es canvia d'escena o es tanca la partida.

Aquest sistema és essencial per mantenir l'ordre intern i assegurar que tot el que passa tingui sentit dins del conjunt. Gràcies al *GameFlags*, cada objecte, diàleg o esdeveniment pot comprovar en quin punt es troba la partida i actuar en conseqüència. Això evita contradiccions, errors o repeticions, i permet que cada situació es desenvolupi de manera coherent amb el progrés del jugador.

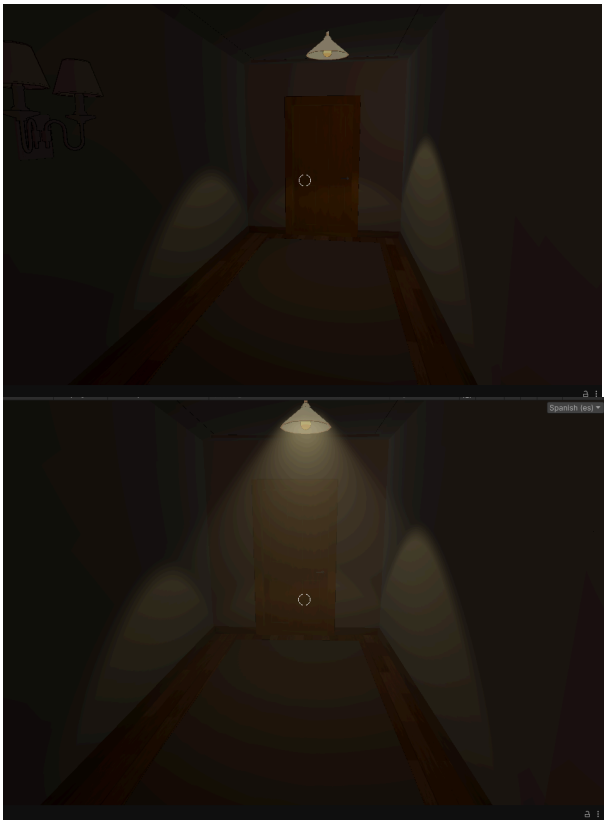
El *GameFlags* treballa constantment en segon pla, sense mostrar res a la pantalla, però gestionant una gran quantitat d'informació. És el que connecta tots els sistemes del joc i els permet comunicar-se entre ells. D'aquesta manera, tot el conjunt funciona com una sola estructura coordinada i estable.

Encara que el jugador no el vegi directament, el *GameFlags* és una de les parts més importants del projecte. Gràcies a ell, el joc pot mantenir una continuïtat real, recordant decisions, estats i accions, i aconseguint que tot el que passa dins del món tingui coherència i resposta. És, en definitiva, la base invisible que fa possible que l'experiència funcioni amb solidesa i que cada element del joc tingui sentit dins del conjunt.

5.2.3 Il·luminació a Unity

Imatge 120. Il·luminació a l'escena. *Font pròpia*

En un videojoc de terror psicològic, l'ambientació és una de les parts més importants. El jugador ha de sentir aquesta tensió i incomoditat constant, i per aconseguir-ho la il·luminació té un paper fonamental. A través de la llum podem dirigir l'atenció del jugador, crear contrastos entre seguretat i perill o donar la sensació que hi ha alguna cosa amagada. Jo volia que el meu joc tingués una il·luminació que ajudés a transmetre aquestes emocions, però que alhora es veiés bé i tingués una aparença més professional.



Imatge 121 i 122. Diferència entre llum volumètrica i llum no volumètrica. *Font pròpia*

El meu projecte estava fet amb URP (Universal Render Pipeline), que és una de les maneres que té Unity de renderitzar o mostrar els gràfics. El problema és que URP no permet fer servir llum volumètrica de manera nativa. Per entendre-ho fàcilment, la llum volumètrica és aquella en què es pot veure el feix o el recorregut de la llum, com el con d'una llanterna o d'un focus enmig de la boira. Sense això, només es veu el punt final il·luminat, però no el camí de la llum, i això trenca una mica l'atmosfera que volia aconseguir.

Després de buscar diferents opcions, vaig trobar un recurs fet pel mateix Unity que permetia utilitzar llums volumètriques dins d'URP. Crear aquest efecte des de zero

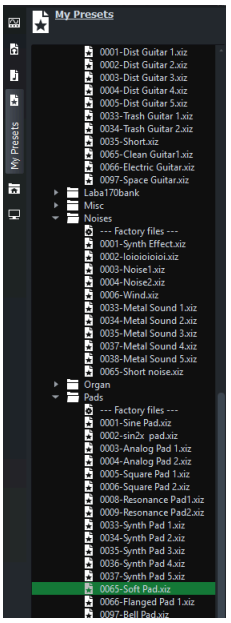
hauria requerit molt de temps i coneixements tècnics, així que vaig decidir fer servir aquesta eina per poder centrar-me més en la part artística i narrativa del joc.

Un cop vaig aconseguir integrar correctament aquestes llums volumètriques, el canvi es va notar molt. Al principi vaig tenir alguns problemes perquè les llums no es mostraven com volia, però després d'investigar i provar diferents configuracions vaig aconseguir solucionar-ho i fer que tot es veiés correctament dins del joc.

Gràcies a això, la il·luminació va millorar de manera notable i l'ambient del joc va guanyar molta més profunditat i realisme. Les escenes ara transmeten millor aquesta sensació d'inquietud i misteri que buscava des del principi. Encara hi ha aspectes que es podrien perfeccionar, però el resultat final és molt satisfactori i aconsegueix recrear l'ambient de terror psicològic que volia.

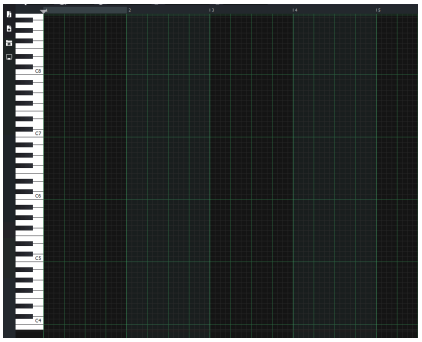
5.2.4 So i ambientació

La música és un dels elements més importants dins dels videojocs. És capaç de transmetre emocions i sensacions sense necessitat de veure res directament. De vegades, una melodia pot fer-nos entendre què està passant o fer-nos sentir el que el personatge viu, encara que no s'expliqui amb paraules. Per això, des del principi vaig voler donar-li una importància especial, ja que crec que la música pot donar molt de significat a cada escena i ajudar a crear l'ambient que el joc necessita.



Imatge 123. Biblioteca d'instruments. *Font pròpia*

Quan vaig començar aquesta part del projecte estava completament desorientat. No sabia per on començar ni com fer que res sonés bé. Simplement tenia LMMS obert i intentava entendre com funcionava. El primer que vaig fer va ser mirar una mica la seva interfície i escoltar la biblioteca d'instruments per veure quins sons podien servir-me millor segons el tipus de sensació que volia aconseguir. També vaig



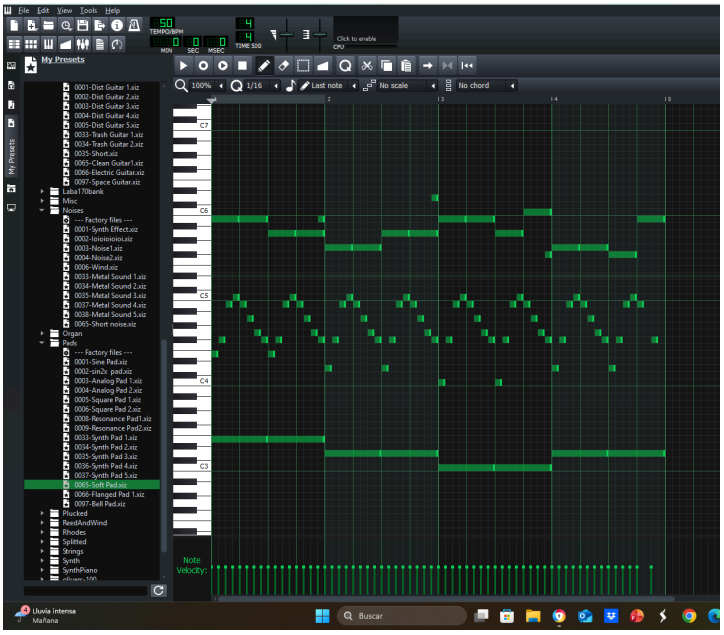
Imatge 124. *Piano Roll. Font pròpia*

començar a experimentar amb el *piano roll*, provant a col·locar notes i veure què sortia. Però, al principi, res sonava com volia. Tot semblava desordenat o sense coherència, i això em va fer adonar que necessitava entendre una mica més com funciona la música.

Vaig començar a informar-me sobre teoria musical, que és un camp molt ampli i ple de conceptes. Vaig aprendre què és una escala i com es forma, quins tipus d'acords hi ha i quines sensacions poden transmetre. Tot i això, encara em considero molt principiant, i sé que el que he après només és una petita part del que hi ha darrere de la música. Tot aquest coneixement, però, em va servir per entendre millor per què certes notes encaixen entre si i d'altres no, i per començar a donar una mica més de sentit a les meves melodies.

Imatge 125. Peça musical pròpia. *Font pròpia*

A mesura que anava aprenent, vaig començar a crear les meves pròpies peces per al videojoc. Cada cançó tenia un objectiu concret: transmetre una sensació o acompanyar un moment determinat. Volia que cada cançó transmetés exactament el que jo volia expressar i que encaixés bé en el moment en què s'utilitza dins del joc. El piano va ser un dels instruments principals que vaig utilitzar, ja que pot transmetre emocions molt diferents segons com es toqui. A més, hi vaig afegir sons com veus suaus o instruments més ambientals, que aportaven profunditat i ajudaven a crear l'atmosfera adequada.

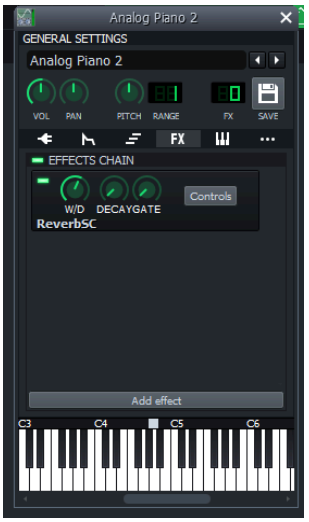


Imatge 126. Composició musical amb piano. *Font pròpia*

Tot i així, no ha estat gens fàcil aconseguir que la música sonés com volia. Encara em falta molt coneixement per poder fer que tot soni més net o més professional. Hi ha peces que no

acaben d'encaixar com m'agradaria o que podrien estar millor treballades, però per al nivell en què estic, estic satisfet amb el resultat. Al final, el més important era aconseguir transmetre alguna cosa, i crec que això sí que ho he pogut fer.

També he après a utilitzar alguns efectes bàsics dins del programa, com el *reverb* o el *delay*, que ajuden a donar sensació d'espai i profunditat. Això va ser útil per fer que els sons no sonessin tan plans i que les peces fossin més agradables d'escoltar. A més, vaig fer que les cançons fossin *loopejables*, de manera que poguessin repetir-se sense que es notés el final ni el principi, ja que això és important en un videojoc per mantenir una sensació constant.



Imatge 127. Efectes de so bàsics. *Font pròpia*

Mirant enrere, tot aquest procés m'ha ensenyat molt. Al principi no sabia ni per on començar ni tenia ningú a qui preguntar, però amb paciència i moltes proves he aconseguit fer música que, tot i les seves limitacions, expressa el que volia. He après que fer música és molt més complex del que sembla, però també molt gratificant. Encara hi ha molt camp per investigar i molt per millorar, però el més important és que ara tinc una base sobre la qual puc seguir creixent i explorant. I, tot i ser principiant, he descobert que el més valuós no és només el resultat, sinó el procés d'arribar-hi.

Conclusions

Després de tot aquest temps treballant en el meu projecte, puc dir que he après moltíssim més del que m'esperava. Quan vaig començar, realment no sabia si seria capaç de crear un videojoc 3D complet des de zero, i aquesta era justament la hipòtesi del treball: veure si seria capaç de fer-ho tot jo sol, aprenent a programar, modelar, animar, crear la música i integrar-ho tot en un mateix projecte. Ara, després de mesos de feina, puc dir que sí, he estat capaç de fer-ho, encara que no ha estat gens fàcil.

He après molt sobre programació en C# dins de Unity. Al principi tot em sonava molt tècnic i complicat, i moltes vegades el codi no funcionava o feia coses diferents de les que volia. Em passava hores intentant entendre per què no anava bé, buscant solucions en fòrums i provant diferents opcions fins que, al final, aconseguia que tot funcionés com volia. Aquesta part m'ha fet entendre molt millor com pensen els programes i com es pot aconseguir que tot estigui connectat dins del joc.

També he après a modelar i crear escenaris en 3D amb Blender. Al principi em costava molt fer formes més complexes o aconseguir que tot quedés proporcionat, però amb el temps vaig anar millorant i vaig poder fer escenaris complets, objectes i fins i tot el meu propi personatge. Vaig aprendre a fer el "unwrap", posar textures, treballar amb materials i nodes, i aconseguir un estil visual propi. Després vaig passar a la part del rigging i les animacions, que també va ser un repte. Fer que el personatge es mogués de manera natural no és gens fàcil, però combinant el que vaig fer jo amb les animacions de Mixamo vaig aconseguir uns resultats força bons.

A Unity vaig haver d'aprendre com importar-ho tot, configurar les animacions, fer els shaders i aconseguir que el joc es veiés com volia. Aquesta part va ser molt interessant perquè vaig començar a entendre com tot el treball que fas a Blender cobra vida dins del motor del joc. Encara així, hi ha coses que m'hauria agradat treballar més, sobretot la il·luminació, perquè és una part molt important en un joc de terror i crec que encara em falta experiència per aconseguir l'ambient perfecte.

Una de les parts que més em va costar, però també una de les més interessants, va ser la creació de la música. Al principi em sentia completament perdut, ja que mai

havia fet música abans i no sabia per on començar. Vaig utilitzar el programa LMMS, i primer vaig explorar els instruments i l'interfície per entendre com funcionava. Després vaig començar a investigar una mica sobre teoria musical: què són les escales, com es formen, quins tipus hi ha i com poden transmetre diferents emocions. Tot i ser un camp molt ampli i complex, vaig aconseguir entendre'n les bases i aplicar-les a les meves composicions.

Amb aquesta base vaig poder crear diverses peces per al joc. Cada cançó tenia la seva funció i havia de transmetre una sensació concreta segons el moment: calma, misteri o inquietud. Tot i que el resultat no és perfecte i encara em falta molt per aconseguir que soni d'una manera més neta o professional, estic satisfet amb el que he fet. Sent principiant, és difícil aconseguir que tot soni com voldries, però crec que per al meu nivell actual he aconseguit resultats més que correctes. A més, he descobert que la música és una part molt poderosa dels videojocs, capaç de transmetre allò que les imatges soles no poden expressar.

En general, estic molt satisfet amb el resultat. El joc és jugable, té una base sòlida i mostra tot el que he après. Òbviament, encara es podria millorar molt, però el més important era demostrar-me que podia fer-ho i que podia aprendre tot el necessari per aconseguir-ho. He après que crear un videojoc no és només fer que les coses funcionin, sinó entendre cada part del procés, tenir paciència i saber organitzar-se. També he après a no rendir-me quan alguna cosa no surt bé, perquè gairebé sempre hi ha una solució, encara que triguis a trobar-la.

Mirant enrere, crec que la hipòtesi sí que s'ha complert. He estat capaç de crear un videojoc 3D funcional i he après molt sobre totes les àrees implicades. La part de la música m'ha ajudat a entendre millor la importància del so dins dels videojocs i m'ha despertat interès per continuar aprenent en aquest camp tan ampli. Potser el joc no és perfecte, però el que m'emporto és tot el coneixement, l'experiència i la satisfacció de veure com una idea meva s'ha convertit en alguna cosa real. Si alguna cosa he après amb aquest projecte, és que amb temps, esforç i constància pots arribar molt més lluny del que penses, i que, encara que siguis principiant, sempre pots crear alguna cosa que transmeti el que vols.

Agraïments

Vull donar les gràcies al meu professor Jaime Morcillo per ajudar-me a organitzar el treball i donar-me consells quan m'encallava amb algunes parts del projecte. També vull agrair a la meva família, que sempre m'ha recolzat i m'ha animat a seguir endavant, sobretot en els moments en què tot es complicava. I finalment, a la gent dels fòrums i comunitats online, que tot i no conèixer-me, sempre van intentar ajudar-me i explicar-me les coses amb claredat. Gràcies a tots ells he pogut tirar endavant aquest projecte.

Bibliografia i webgrafia

A continuación, te explicamos cómo puedes entender la psicología del jugador en el diseño de juegos. (2024, 21 junio). *www.linkedin.com*. https://es.linkedin.com/advice/0/heres-how-you-can-grasp-player-psychology-game-design-8qzbc?lang=es (Data de consulta: 18/4/2025)

Admin, & Admin. (2019, 13 julio). *Consejos de Game Design: ¿Qué son las mecánicas y cómo explotarlas?* Pixel Perfect Studio - *Dream it, build it!* https://www.pixelperfectstudio.mx/2019/07/12/consejos-de-game-design-que-son-las-mecanicas-y-como-explotarlas/ (Data de consulta: 18/4/2025)

Blender Foundation. (s. f.). *blender.org - Home of the Blender project - Free and Open 3D Creation Software.* https://www.blender.org/ (Data de consulta: 17/4/2025)

Campus, C. (2024, 18 marzo). *La evolución de los videojuegos a lo largo de la historia.* Universidad Europea Creative Campus. https://creativecampus.universidadeuropea.com/blog/evolucion-videojuegos/ (Data de consulta: 16/4/2025)

Colaboradores de Wikipedia. (2025, 1 abril). *Historia de los videojuegos. Wikipedia, la Enciclopedia Libre.* https://es.wikipedia.org/wiki/Historia_de_los_videojuegos (Data de consulta: 16/4/2025)

Colaboradores de Wikipedia. (2025a, 18 marzo). *Atari 2600. Wikipedia, la Enciclopedia Libre.* https://es.wikipedia.org/wiki/Atari_2600 (Data de consulta: 16/4/2025)

Contributors to Wikimedia projects. (2025, 7 marzo). *Història dels videojocs. Viquipèdia, L'enciclopèdia Lliure.* https://ca.wikipedia.org/wiki/Hist%C3%B2ria_dels_videojocs (Data de consulta: 17/4/2025)

Coslada, R. (2025, 17 marzo). *20 Videojuegos que marcaron la década de los 90 | Por Rodrigo Coslada. Colossus Gamers.*

https://colossusgamers.com/20-videojuegos-que-marcaron-la-decada-de-los-9 (Data de consulta: 16/4/2025)

Danielparente. (2022, 18 diciembre). *El proceso de diseño de niveles: una guía para principiantes. Daniel Parente Blog.* https://www.danielparente.net/es/2022/12/18/el-proceso-de-diseno-de-niveles-una-guia-para-principiantes/ (Data de consulta: 18/4/2025)

De Marino, M. E. (2024, 5 julio). *Texturizar en blender. Euroinnova International Online Education.* https://www.euroinnova.com/arquitectura-y-diseno/articulos/texturizar-en-blender (Data de consulta: 17/4/2025)

Deepin en Español. (2021, 17 febrero). *Linux Multimedia Studio (LMMS) - Wiki de Deepin en español. Deepin En Español.* https://xn--deepinenespaol-1nb.org/wiki/linux-multimedia-studio-lmms/ (Data de consulta: 18/4/2025)

Foro3D. (s. f.). https://www.foro3d.com/showthread.php?t=149525 (Data de consulta: 17/4/2025)

GM, A. (2024, 10 enero). *¿Cuál fue el primer videojuego? Esta es la historia de un pasatiempo universal. Historia National Geographic.* https://historia.nationalgeographic.com.es/a/cual-fue-primer-videojuego-esta-es-historia-pasatiempo-universal_20123 (Data de consulta: 16/4/2025)

González, A. C. (2022, 20 febrero). *Blender, qué es y para qué se utiliza. Profesional Review.* https://www.profesionalreview.com/2022/02/20/blender-que-es-y-para-que-se-utiliza/ (Data de consulta: 17/4/2025)

Historia de los videojuegos: Década de los 70 - *ElOtroLado.* (s. f.). https://www.elotrolado.net/wiki/Historia_de_los_videojuegos:_Decada_de_los_70 (Data de consulta: 16/4/2025)

Historia de los videojuegos: Década de los 80 - *ElOtroLado.* (s. f.). https://www.elotrolado.net/wiki/Historia_de_los_videojuegos:_D%C3%A9cada_de_los_80 (Data de consulta: 16/4/2025)

Instructables. (2018, 5 octubre). *Blender: Basic rigging process. Instructables.* https://www.instructables.com/Blender-Basic-Rigging-Process/ (Data de consulta: 17/4/2025)

MasterD. (2023, 27 abril). *¿Qué es el Game Design? ¿Cómo llegar a ser Game Designer? MasterD.* https://www.masterd.es/blog/game-design-que-es (Data de consulta: 18/4/2025)

Merino, L. J. (2024, 12 diciembre). *La batalla por el trono: PlayStation vs. Sega Saturn. LOS40.* https://los40.com/2024/12/12/la-batalla-por-el-trono-playstation-vs-sega-saturn/ (Data de consulta: 16/4/2025)

Moisés. (2022, 28 diciembre). *Los videojuegos de los 80 y 90 más influyentes de la historia. MiArcade.* https://miarcade.com/los-videojuegos-de-los-80-y-90-mas-influyentes-de-la-historia/ (Data de consulta: 16/4/2025)

Moyano, M. (2024, 29 noviembre). *¿Qué es Blender? The Core School.* https://www.thecoreschool.com/blog/que-es-blender-y-como-puedes-sacarle-partido-para-la-animacion-3d/ (Data de consulta: 17/4/2025)

Sandoval, C. I. L. (2019). *Unity: aprende a desarrollar videojuegos*. Espanya: RC Libros

Plataforma de desarrollo en tiempo real de Unity | Motor 3D, 2D, VR y AR. (s. f.). *Unity.* https://unity.com/es (Data de consulta: 17/4/2025)

Ramírez, L. (2023, 23 marzo). *Game Design: ¿Qué es y cómo convertirte en un diseñador de videojuegos? Thinking For Innovation.* https://www.iebschool.com/hub/game-design-que-es-disenador-de-videojuegos-innovacion/ (Data de consulta: 18/4/2025)

Reyes, A. (2021, 10 marzo). *Los videojuegos más importantes de la historia (Parte 2). UAMedia Blog.* https://uamedia.org/blog/los-videojuegos-mas-importantes-de-la-historia-parte-2/ (Data de consulta: 17/4/2025)

Tones, J. (2021, 22 enero). *Cuál ha sido el mejor año de la historia de los videojuegos: analizamos tecnología y éxitos para responder a la pregunta definitiva. Xataka.* https://www.xataka.com/videojuegos/cual-ha-sido-mejor-ano-historia-videojuegos-analizamos-tecnologia-exitos-para-responder-a-pregunta-definitiva (Data de consulta: 17/4/2025)

Welcome to LMMS | User manual. (s. f.). *User Manual.* https://docs.lmms.io/user-manual (Data de consulta: 18/4/2025)

Imatge de portada: Creació d'un videojoc en Unity. *Font pròpia*

Annexos

Els annexos següents recullen una selecció de les parts més rellevants del codi desenvolupades durant el projecte.

No s'hi mostra tot el contingut complet dels scripts, sinó fragments representatius que permeten entendre el funcionament general dels principals sistemes del joc: el control del jugador, les interaccions, el sistema de diàlegs, el menú i la configuració, la gestió del pas del temps i els efectes visuals mitjançant shaders.

I. Control del jugador i sistema d'interacció

```
using UnityEngine;

public class FirstPersonController : MonoBehaviour
{
    public float speed = 4f;
    public float sensitivity = 2f;
    public Camera playerCamera;
    private CharacterController controller;
    private float rotationX;
    private Vector3 moveDirection;
    private bool canMove = true;

    void Start()
    {
        controller = GetComponent<CharacterController>();
        Cursor.lockState = CursorLockMode.Locked;
    }

    void Update()
    {
        if (!canMove) return;

        float moveX = Input.GetAxis("Horizontal");
        float moveZ = Input.GetAxis("Vertical");
        Vector3 move = transform.right * moveX + transform.forward * moveZ;
        controller.Move(move * speed * Time.deltaTime);

        rotationX -= Input.GetAxis("Mouse Y") * sensitivity;
        rotationX = Mathf.Clamp(rotationX, -80f, 80f);
        playerCamera.transform.localRotation = Quaternion.Euler(rotationX, 0, 0);
        transform.Rotate(Vector3.up * Input.GetAxis("Mouse X") * sensitivity);
    }
```

```
    public void LockMovement(bool state)
    {
        canMove = !state;
        Cursor.lockState = state ? CursorLockMode.None : CursorLockMode.Locked;
    }
}

public interface IInteractable
{
    bool CanInteract();
    void Interact();
}

public class InteractionManager : MonoBehaviour
{
    public float interactDistance = 2f;
    public LayerMask interactLayer;
    public Camera playerCamera;
    private bool isInteracting = false;
    private static InteractionManager instance;
    public static InteractionManager Instance => instance;

    void Awake()
    {
        instance = this;
    }

    void Update()
    {
        if (Input.GetKeyDown(KeyCode.E) && !isInteracting)
        {
            Ray ray = new Ray(playerCamera.transform.position,
playerCamera.transform.forward);
            if (Physics.Raycast(ray, out RaycastHit hit, interactDistance, interactLayer))
            {
                var interactable = hit.collider.GetComponent<IInteractable>();
                if (interactable != null && interactable.CanInteract())
                {
                    StartInteraction();
                    interactable.Interact();
                }
            }
        }
    }

    public static void StartInteraction()
    {
        if (instance != null) instance.isInteracting = true;
```

```
    }

    public static void EndInteraction()
    {
        if (instance != null) instance.isInteracting = false;
    }
}

public class CamaInteractuable : MonoBehaviour, IInteractable
{
    public AudioClip dormirClip;
    private bool isSleeping = false;

    public bool CanInteract()
    {
        return !isSleeping && GameFlags.Instance.momentoDelDia == "noche";
    }

    public void Interact()
    {
        InteractionManager.StartInteraction();
        isSleeping = true;
        TransitionHelper.Instance.StartDayTransition("manana", dormirClip, 0.6f, 0.3f);
        StartCoroutine(ResetSleep());
    }

    private IEnumerator ResetSleep()
    {
        yield return new WaitForSeconds(1.5f);
        isSleeping = false;
        InteractionManager.EndInteraction();
    }
}
```

II. Sistema de diàlegs

```
using UnityEngine;
using TMPro;
using System.Collections;
using System.Collections.Generic;

public class DialogueUI : MonoBehaviour
{
    public TextMeshProUGUI dialogueText;
    public GameObject dialoguePanel;
    public bool NextPressed { get; private set; }
```

```
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Space))
            NextPressed = true;
    }

    public void ShowLine(string line)
    {
        dialoguePanel.SetActive(true);
        dialogueText.text = line;
        NextPressed = false;
    }

    public void Hide()
    {
        dialoguePanel.SetActive(false);
        dialogueText.text = "";
    }
}

public class ObjectDialogueManager : MonoBehaviour
{
    public DialogueUI dialogueUI;
    private Queue<string> dialogueQueue = new Queue<string>();
    private bool isPlaying = false;

    public void StartDialogue(string[] lines)
    {
        if (isPlaying) return;
        isPlaying = true;
        dialogueQueue.Clear();

        foreach (string line in lines)
            dialogueQueue.Enqueue(line);

        StartCoroutine(PlayDialogue());
    }

    private IEnumerator PlayDialogue()
    {
        while (dialogueQueue.Count > 0)
        {
            string nextLine = dialogueQueue.Dequeue();
            dialogueUI.ShowLine(nextLine);
            yield return new WaitUntil(() => dialogueUI.NextPressed);
        }

        dialogueUI.Hide();
```

```
        InteractionManager.EndInteraction();
        isPlaying = false;
    }
}

public class DialogueTrigger : MonoBehaviour, IInteractable
{
    public string[] dialogueLines;
    private ObjectDialogueManager dialogueManager;

    void Start()
    {
        dialogueManager = FindAnyObjectByType<ObjectDialogueManager>();
    }

    public bool CanInteract()
    {
        return !InteractionManager.Instance.Equals(null);
    }

    public void Interact()
    {
        InteractionManager.StartInteraction();
        if (dialogueManager != null)
            dialogueManager.StartDialogue(dialogueLines);
    }
}

public class LocalizationManager
{
    public static string GetText(string key)
    {
        return PlayerPrefs.GetString("lang", "es") == "es" ? key : "[EN] " + key;
    }
}
```

III. Menú i configuració

```
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
using TMPro;
using System.Collections;

public class MenuController : MonoBehaviour
{
    public GameObject mainPanel;
```

```
public GameObject settingsPanel;
public GameObject loadingScreen;
public Slider loadingBar;
public CanvasGroup menuCanvasGroup;
public CanvasGroup loadingCanvasGroup;
private bool isTransitioning = false;
public string sceneToLoad = "Mihabitación";

void Start()
{
   settingsPanel.SetActive(false);
   mainPanel.SetActive(true);
   loadingScreen.SetActive(false);
   menuCanvasGroup.alpha = 1f;
   loadingCanvasGroup.alpha = 0f;
}

public void OnNewGameClicked()
{
   if (!isTransitioning)
      StartCoroutine(LoadNewGame());
}

IEnumerator LoadNewGame()
{
   isTransitioning = true;
   yield return StartCoroutine(FadeCanvas(menuCanvasGroup, 1, 0, 1f));
   mainPanel.SetActive(false);
   loadingScreen.SetActive(true);
   yield return StartCoroutine(FadeCanvas(loadingCanvasGroup, 0, 1, 1f));
   AsyncOperation operation = SceneManager.LoadSceneAsync(sceneToLoad);
   operation.allowSceneActivation = false;

   while (!operation.isDone)
   {
      loadingBar.value = Mathf.Clamp01(operation.progress / 0.9f);
      if (operation.progress >= 0.9f) operation.allowSceneActivation = true;
      yield return null;
   }
}

IEnumerator FadeCanvas(CanvasGroup cg, float start, float end, float duration)
{
   float elapsed = 0f;
   while (elapsed < duration)
   {
      elapsed += Time.deltaTime;
      cg.alpha = Mathf.Lerp(start, end, elapsed / duration);
```

```
            yield return null;
        }
        cg.alpha = end;
    }

    public void OpenSettings()
    {
        mainPanel.SetActive(false);
        settingsPanel.SetActive(true);
    }

    public void BackToMenu()
    {
        settingsPanel.SetActive(false);
        mainPanel.SetActive(true);
    }
}

public class AudioManager : MonoBehaviour
{
    public static AudioManager instance;
    public AudioSource musicSource;
    public AudioSource[] effectsSources;
    public float musicVolume;
    public float effectsVolume;

    void Awake()
    {
        if (instance == null) instance = this;
        musicVolume = PlayerPrefs.GetFloat("MusicVolume", 100f);
        effectsVolume = PlayerPrefs.GetFloat("EffectsVolume", 100f);
        ApplyVolumes();
    }

    void ApplyVolumes()
    {
        musicSource.volume = Mathf.Pow(musicVolume / 100f, 1.2f);
        foreach (var s in effectsSources)
            s.volume = Mathf.Pow(effectsVolume / 100f, 1.2f);
    }

    public void SetMusicVolume(float value)
    {
        musicVolume = value;
        PlayerPrefs.SetFloat("MusicVolume", value);
        ApplyVolumes();
    }
```

```
    public void SetEffectsVolume(float value)
    {
        effectsVolume = value;
        PlayerPrefs.SetFloat("EffectsVolume", value);
        ApplyVolumes();
    }
}

public class BrightnessController : MonoBehaviour
{
    public Volume worldVolume;
    public Slider brightnessSlider;
    public TextMeshProUGUI brightnessValueText;
    private ColorAdjustments worldColorAdjust;

    void Start()
    {
        worldVolume.profile.TryGet(out worldColorAdjust);
        float saved = PlayerPrefs.GetFloat("BrightnessLevel", 0f);
        brightnessSlider.value = saved;
        ApplyBrightness(saved);
        brightnessSlider.onValueChanged.AddListener(ApplyBrightness);
    }

    void ApplyBrightness(float value)
    {
        worldColorAdjust.postExposure.Override(Mathf.Lerp(-1f, 1f, (value + 10f) / 20f));
        brightnessValueText.text = value.ToString("0");
        PlayerPrefs.SetFloat("BrightnessLevel", value);
    }
}

public class ResolutionManager : MonoBehaviour
{
    public TextMeshProUGUI resolutionText;
    public bool defaultFullscreen = true;

    void Start()
    {
        string saved = PlayerPrefs.GetString("SelectedResolution", "1920x1080");
        bool fullscreen = PlayerPrefs.GetInt("FullscreenMode", defaultFullscreen ? 1 : 0) == 1;
        ApplyResolution(saved, fullscreen);
    }

    public void ApplyResolution(string res, bool fullscreen)
    {
        string[] parts = res.Split('x');
        int width = int.Parse(parts[0]);
```

```
        int height = int.Parse(parts[1]);
        Screen.SetResolution(width, height, fullscreen);
        resolutionText.text = res;
        PlayerPrefs.SetString("SelectedResolution", res);
        PlayerPrefs.SetInt("FullscreenMode", fullscreen ? 1 : 0);
    }
}

public class VSyncManager : MonoBehaviour
{
    public Toggle vsyncToggle;

    void Start()
    {
        bool enabled = PlayerPrefs.GetInt("VSyncEnabled", 1) == 1;
        QualitySettings.vSyncCount = enabled ? 1 : 0;
        vsyncToggle.isOn = enabled;
        vsyncToggle.onValueChanged.AddListener(SetVSync);
    }

    public void SetVSync(bool state)
    {
        QualitySettings.vSyncCount = state ? 1 : 0;
        PlayerPrefs.SetInt("VSyncEnabled", state ? 1 : 0);
    }
}
```

IV. Pas del temps

```
using UnityEngine;
using UnityEngine.Rendering;
using System.Collections;
using System.Linq;

public class MenuDayCycle : MonoBehaviour
{
    [System.Serializable]
    public class TimeOfDay
    {
        public string nombre;
        public Color luzColor;
        public float luzIntensidad;
        public Vector3 luzRotacion;
        public VolumeProfile volumeProfile;
        public AudioClip ambientClip;
        public float duracionTransicion = 30f;
        public float tiempoEnFase = 20f;
```

```
    }

    public TimeOfDay[] momentosDelDia;
    public Light directionalLight;
    public Volume globalVolume;
    public AudioSource ambientAudioSource;

    private int indiceActual = 0;
    private int siguienteIndice = 1;
    private float tiempo = 0f;

    void Start()
    {
        if (momentosDelDia.Length < 2) return;
        AplicarConfiguracion(momentosDelDia[indiceActual]);
    }

    void Update()
    {
        var actual = momentosDelDia[indiceActual];
        var siguiente = momentosDelDia[siguienteIndice];
        tiempo += Time.deltaTime;
        float t = Mathf.Clamp01(tiempo / actual.duracionTransicion);

        if (directionalLight != null)
        {
            directionalLight.color = Color.Lerp(actual.luzColor, siguiente.luzColor, t);
            directionalLight.intensity = Mathf.Lerp(actual.luzIntensidad, siguiente.luzIntensidad,
t);
            directionalLight.transform.rotation = Quaternion.Lerp(
                Quaternion.Euler(actual.luzRotacion),
                Quaternion.Euler(siguiente.luzRotacion),
                t
            );
        }

        if (globalVolume != null)
            globalVolume.profile = t > 0.5f ? siguiente.volumeProfile : actual.volumeProfile;

        if (ambientAudioSource != null)
        {
            AudioClip clip = t > 0.5f ? siguiente.ambientClip : actual.ambientClip;
            if (ambientAudioSource.clip != clip)
            {
                ambientAudioSource.clip = clip;
                ambientAudioSource.Play();
            }
        }
```

```
        if (tiempo >= actual.duracionTransicion + actual.tiempoEnFase)
        {
            tiempo = 0f;
            indiceActual = siguienteIndice;
            siguienteIndice = (siguienteIndice + 1) % momentosDelDia.Length;
            AplicarConfiguracion(momentosDelDia[indiceActual]);
        }
    }

    void AplicarConfiguracion(TimeOfDay config)
    {
        if (directionalLight != null)
        {
            directionalLight.color = config.luzColor;
            directionalLight.intensity = config.luzIntensidad;
            directionalLight.transform.rotation = Quaternion.Euler(config.luzRotacion);
        }

        if (globalVolume != null && config.volumeProfile != null)
            globalVolume.profile = config.volumeProfile;

        if (ambientAudioSource != null && config.ambientClip != null)
        {
            ambientAudioSource.clip = config.ambientClip;
            ambientAudioSource.Play();
        }
    }
}

public class DayEnvironmentController : MonoBehaviour
{
    [System.Serializable]
    public class DaySetting
    {
        public int numeroDeDia;
        public string momentoDelDia;
        public Color luzColor;
        public float luzIntensidad;
        public VolumeProfile volumeProfile;
        public AudioClip ambientClip;
    }

    public Light directionalLight;
    public Volume globalVolume;
    public AudioSource ambientAudioSource;
    public DaySetting[] configuraciones;
```

```
    public void CambiarMomentoDelDia(int dia, string momento)
    {
        var config = configuraciones.FirstOrDefault(c => c.numeroDeDia == dia &&
c.momentoDelDia == momento);
        if (config == null) return;

        directionalLight.color = config.luzColor;
        directionalLight.intensity = config.luzIntensidad;
        globalVolume.profile = config.volumeProfile;
        ambientAudioSource.clip = config.ambientClip;
        ambientAudioSource.Play();
    }
}

public class GameFlags : MonoBehaviour
{
    public static GameFlags Instance;
    public int numeroDeDia = 1;
    public string momentoDelDia = "manana";
    public bool estaVestido = false;

    void Awake()
    {
        if (Instance == null)
        {
            Instance = this;
            DontDestroyOnLoad(gameObject);
        }
        else Destroy(gameObject);
    }

    public void AvanzarMomento()
    {
        if (momentoDelDia == "manana") momentoDelDia = "tarde";
        else if (momentoDelDia == "tarde") momentoDelDia = "noche";
        else
        {
            momentoDelDia = "manana";
            numeroDeDia++;
        }
    }
}
```

V. Shader

```
Shader "Custom/WindowEmission_URP"
{
    Properties
    {
        _BaseColor("Base Color", Color) = (1,1,1,1)
        _EmissionColor("Emission Color", Color) = (1,1,1,1)
        _EmissionStrength("Emission Strength", Range(0,5)) = 1
        _Smoothness("Smoothness", Range(0,1)) = 0.5
        _Metallic("Metallic", Range(0,1)) = 0
        _MainTex("Base Map", 2D) = "white" {}
    }

    SubShader
    {
        Tags { "RenderType"="Opaque" "Queue"="Geometry" }
        LOD 300

        Pass
        {
            Tags { "LightMode" = "UniversalForward" }
            HLSLPROGRAM
            #pragma vertex vert
            #pragma fragment frag
            #include "Packages/com.unity.render-pipelines.universal/ShaderLibrary/Core.hlsl"

            struct Attributes
            {
                float4 positionOS : POSITION;
                float3 normalOS : NORMAL;
                float2 uv : TEXCOORD0;
            };

            struct Varyings
            {
                float4 positionHCS : SV_POSITION;
                float3 normalWS : NORMAL;
                float2 uv : TEXCOORD0;
            };

            TEXTURE2D(_MainTex);
            SAMPLER(sampler_MainTex);
            float4 _BaseColor;
            float4 _EmissionColor;
            float _EmissionStrength;
            float _Smoothness;
            float _Metallic;
```

```
        Varyings vert(Attributes IN)
        {
            Varyings OUT;
            OUT.positionHCS = TransformObjectToHClip(IN.positionOS.xyz);
            OUT.normalWS = TransformObjectToWorldNormal(IN.normalOS);
            OUT.uv = IN.uv;
            return OUT;
        }

        half4 frag(Varyings IN) : SV_Target
        {
            half3 baseCol = SAMPLE_TEXTURE2D(_MainTex, sampler_MainTex, IN.uv).rgb *
_BaseColor.rgb;
            half3 emission = _EmissionColor.rgb * _EmissionStrength;
            return half4(baseCol + emission, 1);
        }
        ENDHLSL
    }
  }

  FallBack "Hidden/Universal Render Pipeline/FallbackError"
}
```