Final Degree Work Report
FATUM
A Tarot Based Puzzle Adventure

José Ignacio Valverde Ballester

Final Degree Work

Bachelor's Degree in
Video Game Design and Development

Universitat Jaume I

May 25, 2022

Supervised by: José Vte. Martí Avilés.

To my past self, thanks for being brave enough to start this path.

To my future self, keep going on. We still have a lot to walk.

To my present self, well done. You did a great job.

Acknowledgments

First of all, I would like to thank my Final Degree Work supervisor, José Vte. Martí Avilés, for his guidance and help in the development of this project.

I would like to thank my parents for allowing me to enter this degree. I know that the future will not be easy, but nothing in this life is. If I'm here it's because they trusted me.

I would also like to thank all the people who have helped me progress, both my friends here and my little brother and the old friends I have left. Thank you Ciro, Ignacio, Adrián, Iván, Javi, Jorge, Alicia, Fernando Ballester, Fernando Palau, Bruno and Glen.

And above all, I want to thank Esther Hidalgo for always being there for me, for everything that has supported me and lifted me up every time I've fallen. Without her I would never have come this far nor would I have dared to let my imagination run wild. I will always be in your debt.

I also would like to thank Sergio Barrachina Mir and José Vte. Martí Avilés for their inspiring LaTeX template for writing the Final Degree Work report, which I have used as a starting point in writing this report.

Abstract

Fatum is an unconventional puzzle and adventure game in which we will control a creature that wakes up in an unknown place in ruins without knowing who it is and how it got there. It will have to advance through short stages and solve the puzzles it finds to advance and escape. This work consists of the implementation of the first level of the game, which includes the tutorial and the design of a narrative whose backbone is the major figures of the *Tarot* known as the *Major Arcana.*[19] Academically, this document consists of the final degree project report of the Video game Design and Development bachelor's degree at the *Jaume I University.*

Contents

Chapter 1

Introduction

Contents

This chapter shows what the purpose of the work was in the beginning, why and how this project was going to be developed.

1.1 Work Motivation

This project is the culmination of a phase in my life. It is the way I have to express everything I have lived and learned during these last four years. Therefore, I decided to approach it in a special way.

When I conceived the idea that led to *Fatum*, what I was thinking was that the game should represent a path. And that's when I thought of tarot. The major cards of the *Tarot* are represented as a path a person travels. Each card symbolizes two stages that each person can go through in their life, both related, but often one positive and one negative. It was then the idea surfaced. An adventure game in which you must go through the tarot and the player

is the card of the *Fool,* which is the card that symbolizes both the beginning and the end of the journey.

Around this pillar, I was able to build a project that I was passionate about from the beginning because it encompassed the things I like the most about creating a video game: narrative, design, and programming. In addition, I set myself the goal of learning to create fantastic worlds, so this project was too easy for me to put that into practice as well.

This is a more personal work than I expected to do for a TFG, but I think I've taken the right path. In *Fatum* there will always be a fragment of me.

1.2 Objectives

In this project there are set several objectives related to different areas of video game creation. Classified according to their field they are the following:

Design:

- **Use the design by subtraction technique:**[2] *Design by subtraction* is a technique that consists of discarding everything that is not strictly necessary. This design model forces the developer to consider which elements of the game are left over and which are essential for its proper functioning.

- **Design mechanics that match the Tarot cards:**[19] The design of the main mechanics of the game must be linked to the *Tarot* cards. Specifically to the positive meaning of the cards.

- **Design bosses that match the Tarot cards:**[9] design of the bosses of the game must be linked to the negative meaning of the cards.

- **Design levels with puzzles:** The level design has to take into account the puzzles that make the player able to progress.
- **Design an intuitive tutorial:** The tutorial must be intuitive, since due to the design by subtraction there cannot be too many indicative to guide the player.

Narrative:

- **The narrative must be able to be told throughout the world:** Make a narrative flexible enough that it can be delivered in small doses.
- **The world must be simple but captivating:**Not having a very strong artistic section, the design of the world has to supply what cannot be achieved through more defined models.

Programming:

- **Design a robust inventory system:** The game's item system requires an inventory system that is capable of unlocking items, dynamically equipping them, and activating player abilities.
- **Programming a basic AI for enemies and a boss fight:**The game does not require very complex artificial intelligence, since they are not a fundamental part of the game. Still, for the world to feel alive it is necessary to have dynamic objects inhabiting it. The boss should also be simple, since it is the boss that should introduce the tutorial.

1.3 Environment and Initial State

At the beginning this project was a bit chaotic, since five proposals were previously presented and, far from deciding on one, it was decided to combine them all and create something new. That resulted in a couple of weeks of porting, brainstorming, discarding,

and re-conceptualizing until we had a solid core to work from. From there, after several weeks developing the narrative, which would be the skeleton of the project, everything was ready to start.

Regarding materials and resources, the project began on a PC that was on its last legs. The processor was slow, the graphics card worked badly, the RAM collapsed on its own... Fortunately, another device could be obtained a month after starting the project. The development tool is Unity, which is the game engine that is taught in the degree, so the starting point is pretty decent. Still, the use of tools that are not taught in the degree such as terrains was planned, so a challenge was expected. Also, some of the knowledge acquired doing external internships at *MadGear* was applied to this project.

Chapter 2

Planning and resources evaluation

Contents

2.1 Planning

This section contains the planning of the tasks carried out during the project together with the estimated time dedicated. These tasks are broken down into smaller and more specific tasks, also with the time spent on each part.

NARRATIVE: 55 hours

- World Creation: 20 hours
- Main Story: 20 hours
- Characters/Bosses: 15 hours

MECHANICS DESIGN: 30 hours

- Basic rules and mechanics: 5 hours
- Progression mechanics, relationship mechanics and strategy mechanics: 15 hours
- Economy mechanics: 5 hours
- Combat mechanics: 5 hours

LEVEL DESIGN: 30 hours

- In-game world design: 10 hours
- Maps design: 20 hours

PROGRAMING: 100 hours

- Basic mechanics programing: 15 hours
- Mask powers and mask selection: 20 hours
- Implementation of game systems: 30 hours
- HUD and Audio: 5 hours
- AI implementation: 30 hours

ART: 35 hours

- Character art design, animation and in-game implementation of character art: 15 hours
- Design of the environments and in-game implementation of the scenario artwork: 15 hours
- Level music and game sounds: 5 hours

ACADEMIC FIELD: 50 hours

- Memory :40 hours

- Presentation: 10 hours

Total Time: 300 h

Gantt Chart can be found on figure 2.1.

2.2 Resource Evaluation

Starting from the project budget, which is zero euros, all the resources used are free, except for the equipment with which it has been developed. That will not count in the budget since it was obtained before the project was carried out.

Specifications of the computer used for the elaboration of the project:

- **Processor:** Tiger Lake i7-11800H+HM570
- **Memory**: DDR IV 8GB*2(3200MHz)
- **Graphic Controller:** RTX3070, GDDR6 8GB
- **Storage:** 1TB NVMe PCle GEN4x4 SSD
- **OS:** Windows 10 Home

Total prize of the computer used: 1700€

Cost per hour of each field:

- **Programming**: 11.1€/h [14]
- **Narrative**: 24.97€/h [4]
- **Art**: 9.3€/h [5]
- **Mechanics Design**:9.3€/h [13]
- **Level Design**: 12.7€/h [8]

Total cost of the project: 3468.85€

Programs used for the elaboration of the project:

- **Unity(v.2020.3.3f):** The game engine used.[16]
- **Blender:** 3D modeling program used for character models.[1]
- **GitHub Desktop:** Github desktop app.[6]

The websites used to make the game are:

- **GitHub:**Virtual repository. The different versions of the game are saved here.[7]
- **Trello:** a website used to control the work. Divided on pending, doing and done it can be used to divide the whole work into tasks. Used to organize project tasks.[15]
- **Unity Asset Store:** a website for buying and selling assets. There are many free assets, so it is useful to get competent products without having a 3D modeler in the team.[17]
- **Mixamo:** a website with a lot of free animation for characters.[11]

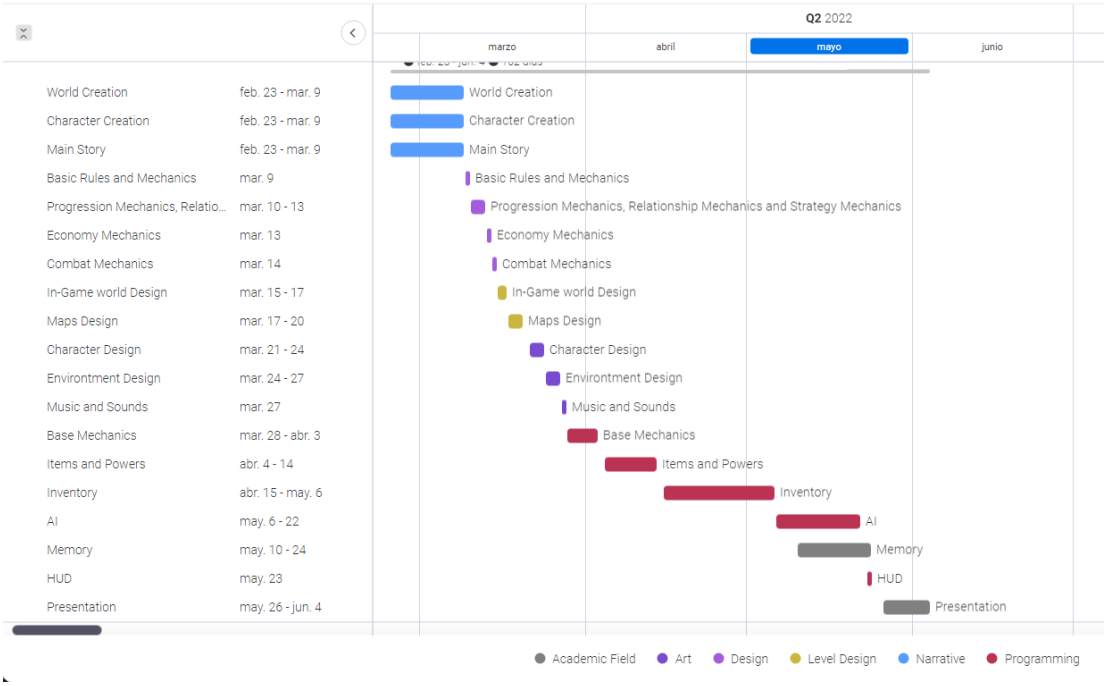

Figure 2.1: Gantt chart

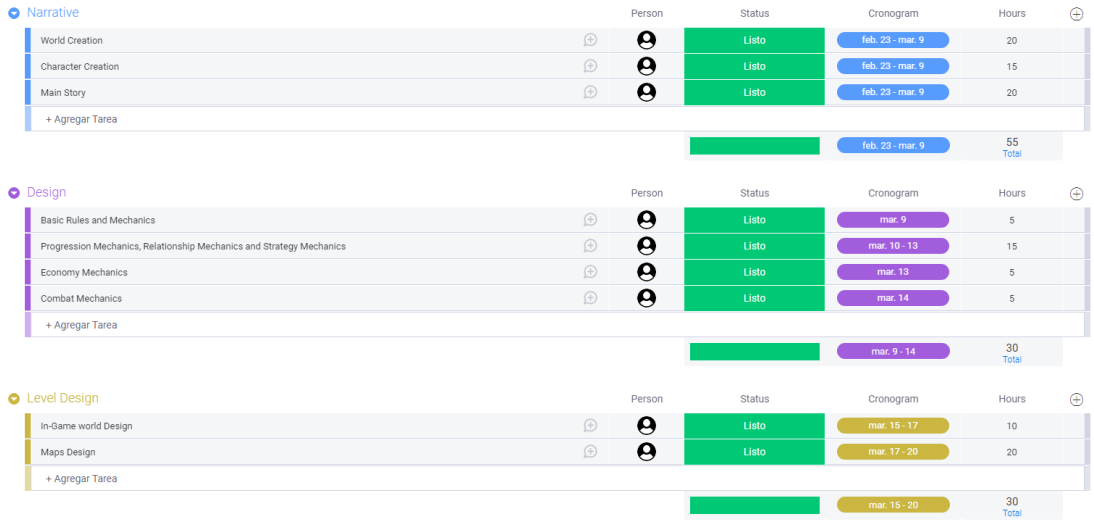

Figure 2.2: Gantt Organization 1

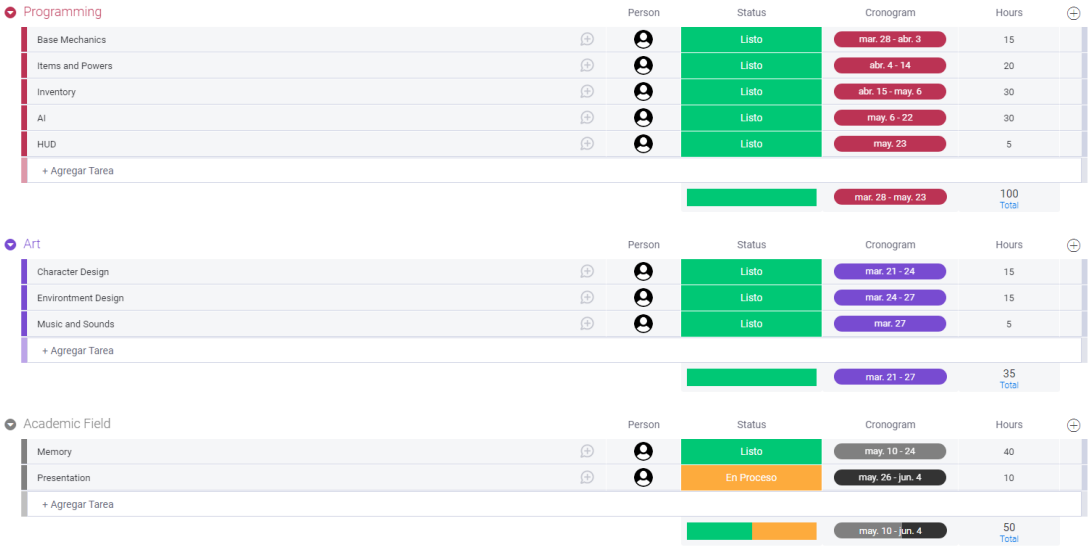

Figure 2.3: Gantt Organization 2

CHAPTER 3

System Analysis and Design

Contents

This chapter presents the requirements analysis, design and architecture of the proposed work, as well as, where appropriate, its interface design.

3.1 Requirement Analysis

To make the task of finding the functional requirements and the non-functional requirements easier, it's necessary to know how *Fatum* works. The game starts at a menu with four buttons: *New Game*, *Continue*, *Exit Game* and *Delete Data*. *New Game* loads a new game and takes the player to the opening scene. *Continue* loads the game from the last checkpoint the player took. *Exit Game* allows the player to leave the game. *Delete Data* on the other hand, is used to delete

the save data. In order for the Continue button to be enabled, the player must have previously played the game.

Once the game is loaded, the player will be able to move the character using the *W*,*A*,*S* and *D* keys, they will be able to jump by pressing the *Space* key and they will be able to rotate the camera by moving the mouse. If near an item, the player can press the *E* key to pick it up. In case of being on a storage platform, the player can open his inventory by pressing the *I* key. Within the inventory he can move by pressing *W*,*A*,*S* and *D*, he can change the description of the item by pressing *X* and he can equip items by pressing the *Space* key. Upon passing through a level end gate, the player will enter the next level.

Once an item is equipped, the player can use its primary ability by pressing the *Left Mouse Button*. In case of having a mobility ability, the player can press *Right Click* to use it.

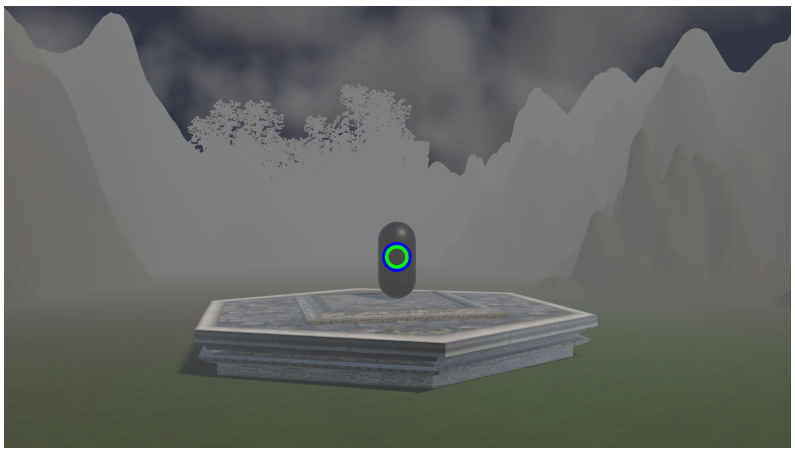

Figure 3.1: Example of a level

3.1.1 Functional Requirements

A functional requirement defines a function of the system that is going to be developed. These are the main requirements:

- **R1:** The player can pick up items with the E key.

- **R2:** The player can freely manage his inventory.
- **R3**: The player can use his primary ability with left click.
- **R4:** The player can use his mobility ability with right click.
- **R5:** The player can jump by pressing space.
- **R6:** Basic enemies can attack the player if the player is in their range.
- **R7:** Basic enemies can move to get within firing range.
- **R8:** The system can save the state of the player and his inventory.
- **R9:** The player can move out of sight of enemies.
- **R10:** The boss can fire electric shocks at the player.
- **R11:**The boss can throw energy spheres at the player.

3.1.2 Non-functional Requirements

Non-functional requirements are requirements that impose restrictions on design or implementation such as restrictions on design or quality standards. These are properties or qualities that the product must have:

- **R12:** The aesthetic must be low poly.
- **R13:**The UI has to be as simplistic as possible.
- **R14:** Controls should feel smooth and intuitive.
- **R15:** The enemies have to feel like an addition to the tension and not as a real threat.
- **R16:** The inventory must be eye-catching as well as practical.

3.2 System Design

Requirements:	R1
Actor:	Player
Description:	The player can pickup an item pressing E key
Preconditions:	1. The player is near an Item
Setps normal sequence:	1. The player presses E 2. The item disappears from the scene. 3. The item gets added to the inventory.
Alternative Sequence:	None
Item Pickup	

Table 3.1: Case of use «CU1. Player Picks up Item»

Requirements:	R2
Actor:	Player
Description:	The player can manage it's inventory at will
Preconditions:	1. The player is on an inventory slate
Setps normal sequence:	1. Player presses Key I 2. Inventory Gets open 3. Player moves towards the inventory with WASD keys. 4. Player Selects an item with Space key. 5. Player Equips an item.
Alternative Sequence:	None
Inventory Management	

Table 3.2: Case of use «CU2. Player manages inventory»

Requirements:	R3
Actor:	Player
Description:	The player can use main skill using left click
Preconditions:	1. The player has a main skill item equipped 2. The player has enough mana to use the skill
Setps normal sequence:	1. The player presses left click 2. The skill is used
Alternative Sequence:	None
Main Skill	

Table 3.3: Case of use «CU3. Player Uses Main Skill»

Requirements:	R4
Actor:	Player
Description:	The player can use it's mobility skill pressing right click
Preconditions:	1. The player has a mobility skill item equipped 2. The player has enough mana to use the skill
Setps normal sequence:	1. The player right click 2. The skill is used
Alternative Sequence:	None
Mobility Skill	

Table 3.4: Case of use «CU4. Player Uses Mobility Skill»

Requirements:	R5
Actor:	Player
Description:	The player can jump pressing the Spacebar
Preconditions:	1. The player is on the ground
Setps normal sequence:	1. The player presses Space 2. The character jumps
Alternative Sequence:	None
Jump	

Table 3.5: Case of use «CU5. Player Jump»

Requirements:	R6
Actor:	Basic Enemy
Description:	The enemy can attack the player when it's in range
Preconditions:	1. The player is in range
Setps normal sequence:	1. The enemy attacks
Alternative Sequence:	None
Enemy Attack	

Table 3.6: Case of use «CU6. Enemy Attack»

Requirements:	R7
Actor:	Basic Enemy
Description:	The enemy can re position itself to get into attack range
Preconditions:	1. The player is in sight range 2. The player is not in shoot range
Setps normal sequence:	1. The enemy moves 2. The enemy checks if the player is in attack range
Alternative Sequence:	None
Enemy Kite	

Table 3.7: Case of use «CU7. Enemy Kite»

Requirements:	R8
Actor:	Game System
Description:	The system can save player and inventory state
Preconditions:	1. The player is on an inventory slate or checkpoint 2. There is an existing data file
Setps normal sequence:	1. The system gets player data 2. The system gets inventory data 3. The system saves all the data in a .txt file
Alternative Sequence:	The system creates a data file, then saves the data
System save	

Table 3.8: Case of use «CU8. System Save»

Requirements:	R9
Actor:	Player
Description:	The player can escape the enemy vision range
Preconditions:	1. The player is in enemies vision range
Setps normal sequence:	1. The player runs getting far from the enemy 2. The enemy checks if the player is in its vision range 3. The enemy returns to its normal path
Alternative Sequence:	1. The player runs getting far from the enemy 2. The enemy checks if the player is in its vision range 3. The enemy keeps tracking the player
Player Hides from the enemy	

Table 3.9: Case of use «CU9. Player hides»

Requirements:	R10
Actor:	Boss
Description:	The boss uses its lightning attack
Preconditions:	None
Setps normal sequence:	1. The boss uses its lightning attack
Alternative Sequence:	None
Boss Lightning	

Table 3.10: Case of use «CU10. Boss Lightning Attack»

Requirements:	R5
Actor:	Boss
Description:	The boss uses its tracking ball attack
Preconditions:	None
Setps normal sequence:	1. The boss uses its tracking ball attack
Alternative Sequence:	None
Boss Tracking ball attack	

Table 3.11: Case of use «CU11. Boss tracking sphere»

3.3 System Architecture

This section describes the architecture of the projected system. The video game is made with *Unity3D* engine, specifically with 2020.3.1f version. For running games made with this engine the minimum system requirements are:

- OS: *Windows* 7 SP1+, mac-OS 10.13+, Ubuntu 18.04+
- Graphics card with DX10 (shader model 4.0) capabilities.
- CPU: SSE2 instruction set support.

This information has been taken from the official *Unity* website. It's generally reliable, but it is not a guarantee that the game will work on devices of similar power. Therefore, after testing with different specifications, the recommended specifications are:

- OS: *Windows* 7, mac-Os 10.13+, Ubuntu 18.04+
- Graphics card DX10(shader model 4.0) capabilities.
- CPU: *Intel Core i3*
- RAM: 8 GB
- Graphic Card: *NVIDIA GTX 760.*

3.4 Interface Design

The interface of the game has been created following the model of design by substraction. Therefore, it is very minimalistic, since it has only the elements that are strictly necessary for the player. It should be noted that the entire interface that refers to the player's status has been implemented as one more element of the character.

The first interface found in the game is the menu interface. It consists of four buttons, a title and a subtitle. The design is simple

and austere, giving the player the first clue that the game will follow the same thematic(see figure3.2).

Figure 3.2: Menu

The second interface is the one the player sees but doesn't realize is in front of them. It deals with the life and mana of the main character. These, as has been mentioned before, are not on a canvas as would be normal. They are integrated into the character himself, in rings that he wears on his back, chest and shoulders. When the player takes damage, the green ring changes to a red hue, indicating the character's remaining health. As you spend mana, the blue ring gets lighter until it turns white, indicating that there are no reserves left. There are four rings located on each side of the character so that the player never loses sight of this interface even when rotating the camera.

The third interface to talk about is the inventory (see figure 3.2). This is the most complex interface, since it consists of several parts. The first part is the equipped items section, on the left hand side. This section consists of several images that represent the items that are equipped. The central image corresponds to the character's mask, the image above it is used to indicate the mobility ability that the player has equipped, the two on the sides are the passive abilities and the five below correspond to the main abilities. Just below is a text window where the description appears. On the right side of the interface is the list of items along with instructions on how to use the inventory. This part of the interface also shows the progress of the game, since the main objective of the game is to recover them all and the slots are visible from the beginning.

Figure 3.3: Inventory

There is a fourth interface in the pause menu, which is accessed by pressing the *Esc* key. It consists of three buttons placed one on top of the other. The layout follows the patterns of the menu interface.

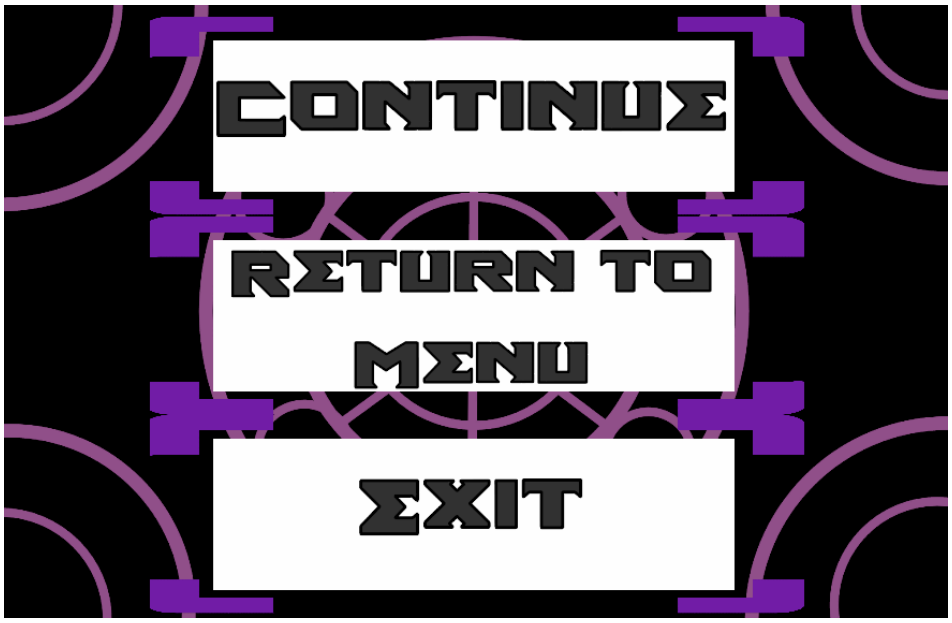

Figure 3.4: PauseMenu

Chapter 4

Work Development and Results

Contents

This chapter explains how this project has been carried out. All the work done and the issues occurred are explained in chronological order. The final conclusions of the author will also be given at the end together with the results.

4.1 Work Development

This section will explain the development of the video game and the phases it has gone through. It should be noted that the work done is only part of the game's tutorial, despite the fact that all items and areas of the game have been designed.

Before developing the game, the design of the narrative and the worldbuilding was done, since it is the part that has the most weight in the project, although it is perhaps the most difficult to see. It started from a medieval world with touches of fantasy and it was

shaped at the same time with folklore, introducing the god of destiny into it. This god ends up divided into the items that the player has to collect and, although he is not told at any time, the main character is also an object from the god of destiny, the madman's mask. Of all the *tarot* cards, the *Fool* was chosen for its meaning. The *Fool* card is usually considered a link between the rest, since, from the path meaning point of view, it is the traveler who travels the road that the rest of the cards make up. It is also the one that represents the one who reads the cards seeking to predict fortune, which in the context of the game is the player himself.

After finishing the world creation, the setting was next. It was necessary to find a visual style that managed to captivate the player without being too complex due to the lack of personnel specialized in 3D modeling. In the end, relatively open settings were chosen with trees, ancient ruins and a somewhat thick fog that reinforced the mystic and mysterious ambience while hiding a bit the textures used. In addition, this mist also does a good job of helping with ambient occlusion and lessening the popping sensation of items furthest from the player's view.

After that, the next step was the creation of the items. Although most of them would not be implemented, having a simple design of all of them would help later to pose the challenges that the player will have to overcome. This tables shows all the objects, along with the card that corresponds to them, their use and the type of objects they are (Table 4.1) (Table 4.2).

Name	Card	Type	Obtention zone	Effect
Fool's Mask	The Fool	Core/Mask	El Sendero del Caminante	None. It's the base mask.
Magician's earrings	The Magician	Mobility	El Otro Lado	Teleport
Holy Mask	The High Priestess	Core/Mask	El Otro Lado	This mask increases max aura and regen, but lowers health.
The Empress	The Empress	Skill	El Templo Real	
King's crown	The Emperor	Core/Mask	El Templo Real	This mask increases max health, but lowers movement speed.
Spiritual ring	The Hierophant	Passive	El Otro Lado	Gives the player a shield when uses any type of active skill
Lover's rings	The Lovers	Passive	El Sendero del Caminante	Unlocks the healing ability
Winged boots	The Chariot	Mobility	El Templo Real	Dash
Strength bracelets	Strength	Passive	El Panteón de los Puros	Allows the player to push heavy blocks
Hermit's talisman	The Hermit	Skill	El Sendero del Caminante	When used, it creates an astral copy of the character. The copy can move freely, jump and use movement skills, but can't alejarse more than x meters from the body.
Wheel of Fortune	Wheel of Fortune	Passive	El Cosmos	When the character takes damage it recovers aura. If it uses skills heals part of the aura consumed.
Justice	Justice	Core/Mask	El Panteón de los Puros	Justice's Mask increases movement

Figure 4.1: Items Table 1

				speed and health a little bit, but lowers max aura.
Raven claws	The Hanged Man	Passive	El Bosque Oscuro	Allows the player to jump using walls
Death	Death	Core/Mask	El Bosque Oscuro	Death's mask brings the character a second life while dying, but decreases both health and max aura.
Wings of Temperance	Temperance	Passive	El Panteón de los Puros	Allows the player to double jump
The Devil	The Devil	Skill	El Bosque Oscuro	Lanza una cadena del jugador hacía adelante. Puede servir para engancharse, tirar de objetos etc.
The Tower	The Tower	Passive	El Templo Real	
The Star	The Star	Core/Mask	El Cosmos	
Medallón de la luna.	The Moon	Skill Half	El Cosmos	Cuando se combina con el medallón del sol confiere la habilidad de detener el tiempo durante unos segundos.
Medallón del sol	The Sun	Skill Half	El Cosmos	Cuando se combina con el medallón de la luna confiere la habilidad de detener el tiempo durante unos segundos.
Judgement	Judgement	Skill	El Panteón de los Puros	Smite
The World	The World	Key	El Cosmos	Allows the Destiny God to reunite(game ends)

Figure 4.2: Items Table 2

The next part was the conceptual creation of the different areas of the game, although in the end only the first would be implemented for reasons of time. Each one was created individually, thinking that everything should make sense while creating a narrative path for the player and that they should encompass card archetypes. The different zones created are the following.

- **The Way of the Wayfarer:** It is the first of the environments and the one that constitutes the tutorial. It is a peaceful place, where nature abounds without too many large constructions, since its main theme is travel and therefore the area constitutes a gigantic path with certain obstacles. In this area the artifacts of the *Fool*, the *Lovers* and the *Hermit* are confined.

- **The Royal Temple:** It is a huge construction that connects with The Way of the Wayfarer. It tries to represent a royal palace, since the artifacts of the *Emperor*, the *Empress*, the *Chariot* and the *Tower* are enclosed in it. It is a place loaded with decoration in which flora is scarce.

- **The Pantheon of the Pure:** It is a round temple similar to a coliseum with several doors. Each of the doors leads to a gigantic room where anyone who dares to enter it will be put to the test. Each of these tests will be determined by the artifact that is locked in it, these being *Strength*, *Temperance*, *Justice* and *Judgment*.

- **The Darkest Forest:** This place, once abundant in life and vegetation, has been corrupted and consumed by the evil energies of the three artifacts that are locked within it. The few creatures that manage to survive the place have become very aggressive, charging anything that moves in front of them. The artifacts that rest in this forest are the *Hanged Man*, the *Devil* and *Death*.

- **The Other Side:** This area defies physical laws. The earth is divided into fragments that float on an infinite abyss. Time passes differently and gravity varies depending on the island. All these strange events are due to the influence of the three artifacts that rest here: the *Magician*, the *High Priestess* and the *Hierophant*.
- **The Cosmos:** The final zone. It exists on another plane of existence, for the powers of the artifacts that reside here are too dangerous to be contained in the material world. It is a place made up of a gigantic platform suspended in space, with various celestial bodies floating around it. This is where the artifacts of the *Wheel of Fortune*, the *Star*, the *Moon*, the *Sun* and the *World* are kept. It is an area in which there is no fauna or flora, since given its conditions nothing is capable of surviving there for a long time. It is thought that this is the dimension where the *God of Fate* used to reside.

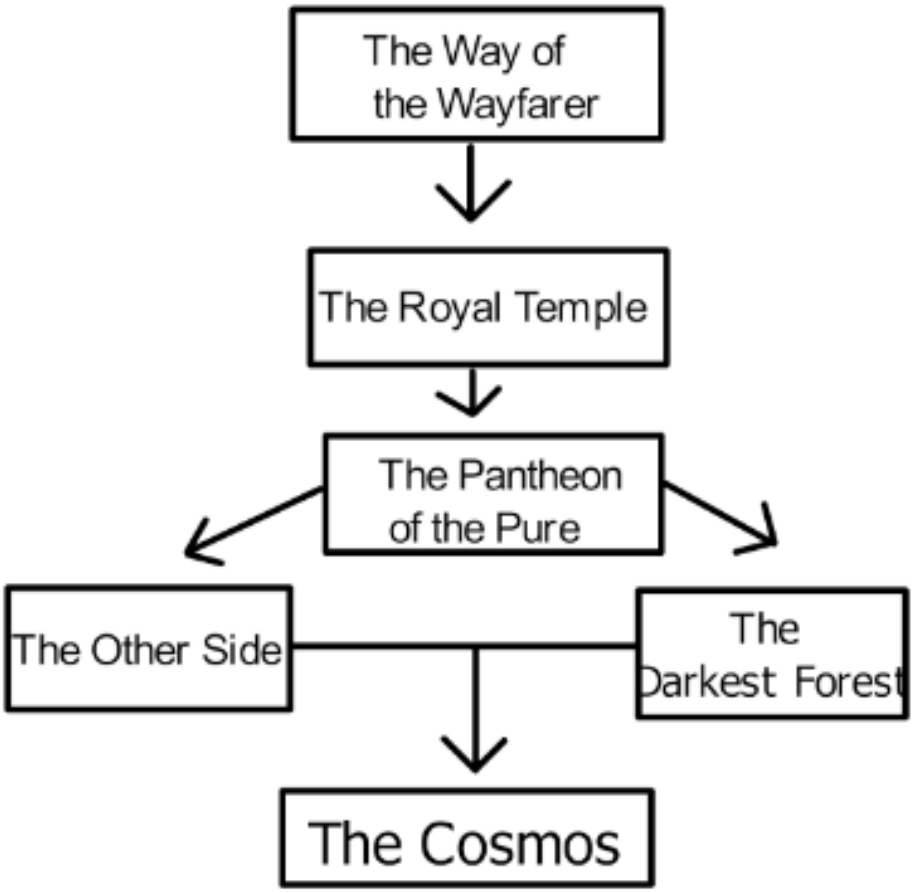

Figure 4.3: Map areas

It was decided that the first level(see here4.4) would be linear and would serve to introduce the player to the most basic mechanics of the game.[12][3][10] It is a straight corridor that starts in a circular area where the player learns to move and rotate the camera. Upon investigation, it will see that there is a way forward: a small opening in the stone wall. However, in order to get through there, you will have to jump over a fallen column that is blocking your path. This is how it learns to use the jump. After leaving the area it will find itself in a gigantic forest flooded with a dense fog. In the forest there are a couple of basic enemies, who when they see the player will go for it. Here the player will be able to learn how the enemies move, how they shoot and, if he gets too far away, they lose track of him. Being a straight corridor, it is easier for the latter to happen and the player can recognize patterns. At the end of the hallway is a stone platform in front of a stone wall. When the player presses the platform, the wall descends and opens the way to the next level. In this way, the player has been able to see all the mechanics that will help him overcome the following challenges in a completely intuitive way.

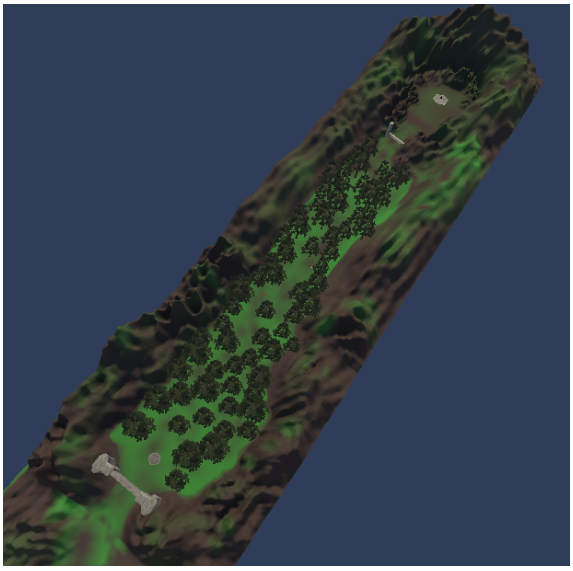

Figure 4.4: First level

The second level is more open. It is an initial corridor behind which a gigantic area is hidden through which the player can move freely. In its center there will be a temple with a closed door that will keep behind it the first unlockable object: the talisman of lovers. To open this area, you must first go to two areas in the form of a corridor on the sides of the level, which will contain puzzles. At the end of the puzzle will be the pressure plate that will open the area. Hitting both of them will open the door. When the player returns for the item, they will also trigger the pressure plate that allows access to the boss area.

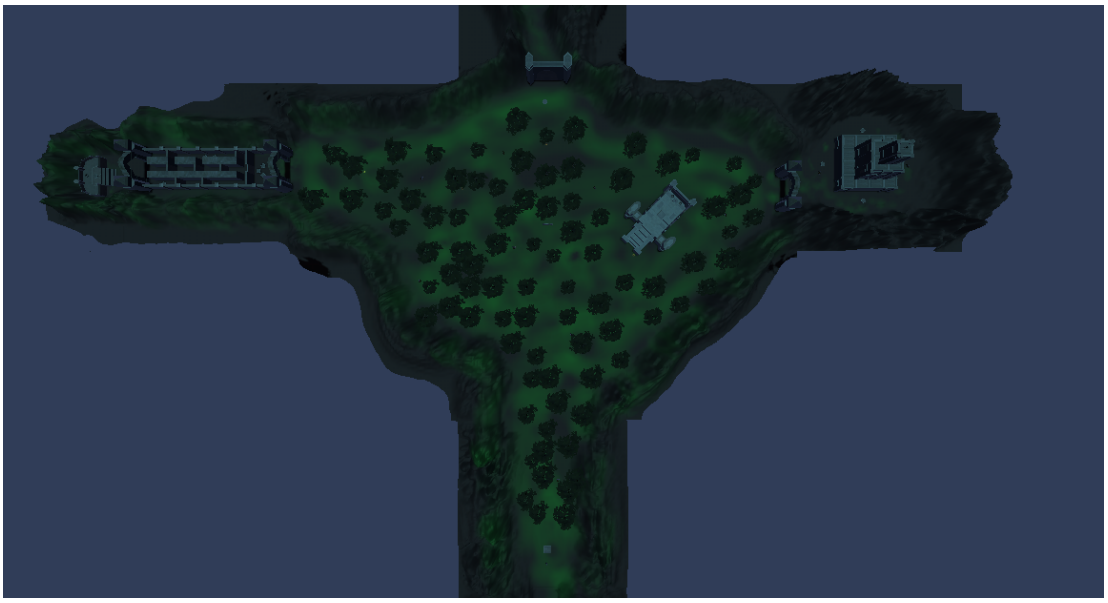

Figure 4.5: Second Level

The boss area was designed as a circular platform with several walls and special platforms. These platforms have the objective of accumulating energy that the boss launches with his attacks. When charged the special walls will rise. These serve to bounce an attack to the boss, with which the player will manage to hurt him. When the boss is defeated, his item will appear and the next area will open.

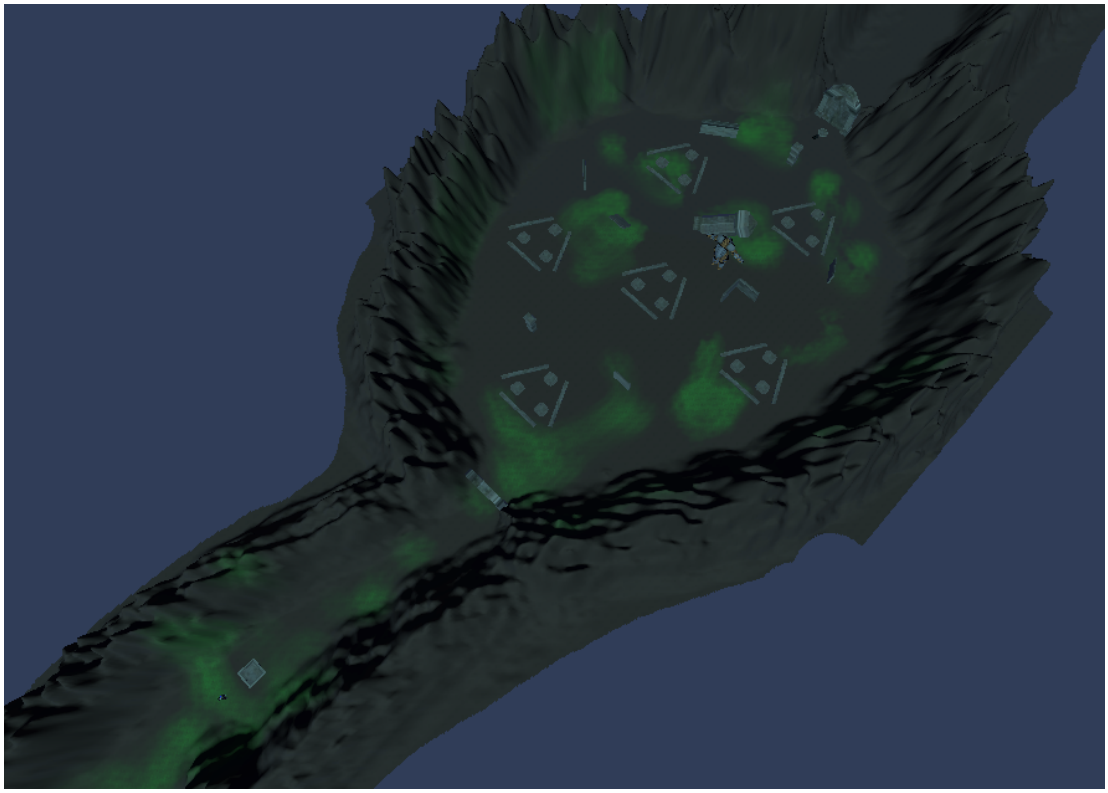

Figure 4.6: Boss Level

For the fourth zone it returns to linearity. This area, within the concept of the full game, serves as an interlude between *The Way of the Wayfarer* and *The Royal Temple*. It introduces the mechanics of the ghost player and consists of two obstacles and a puzzle. The first obstacle is a door with a timer. When the player presses the switch, the door opens and after a few seconds it closes. The timer always closes the door before the player can reach it, so in order to proceed they must use the spirit to hit the switch, return to their body and go through the door. The second obstacle is a wall of force through which only the spirit can pass. Behind it is a switch that disables it. When the player has passed the obstacles, they must use what they just learned to solve a small puzzle and thus advance to the next area, which in this case is the end screen of the game.

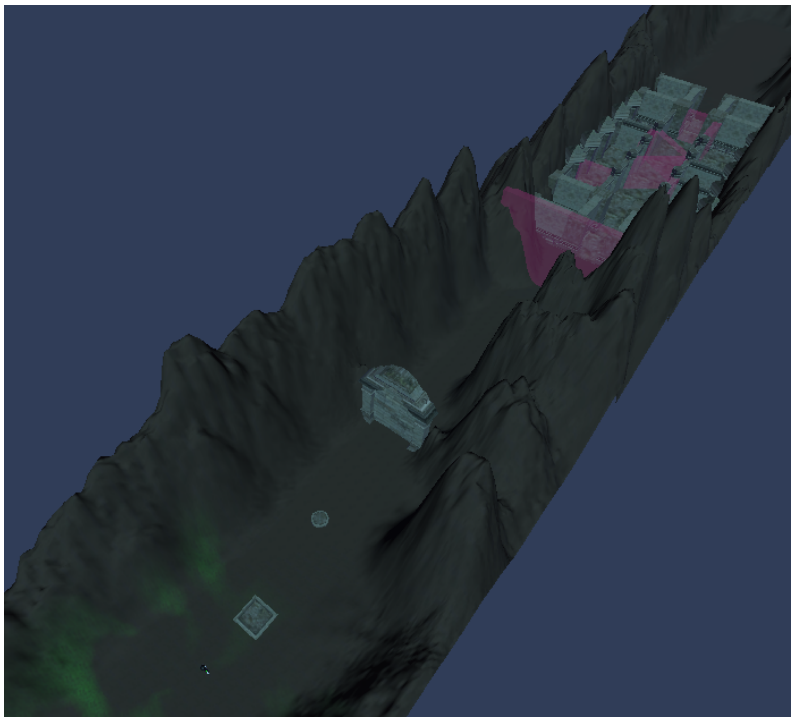

Figure 4.7: Third Level

Once the levels were designed, the next step was to create a temporary scene and start developing the basic mechanics. This design phase was divided into several sub-phases: player phase, inventory phase, save phase, AI phase, and boss phase.

The player phase, as the name suggests, is the phase in which all the player mechanics take place. This includes movement, camera rotation, and health and mana stats. The player's movement is carried out through a *Rigidbody* and a script , which takes the inputs made by the player and applies a speed depending on where the player is pressing. The camera is controlled by *Cinemachine*. It is a free look type camera that is tied to the player and rotates around it. It has three rings at three different heights, so that they form a hemisphere that restricts the movement of the camera. The player's resources are managed by a script that saves their statistics and updates the life and mana bars in the update function so they are always at the corresponding value.

The inventory development phase was perhaps the longest, since it is the one that uses the most subsystems. The inventory system is made up of two large blocks: the inventory block, which is the block that manages the inventory per se, and the skills block, which is the one in charge of managing the skills that the player can use. Let's start by looking at the second block. The first thing found in this block is the skill class. This class stores the data that a skill can have.

The next script you have is *SkillDB*. It is a database with all the skills in the game. It contains a list of all the skills in the game, along with a function that assigns the relevant image to the skill. The third item is *Skill Logic*. In *skillLogic* are all the functions that execute the skills. For example, the *Hermit* skill would have all its data stored in *SkillDB*, but the function that is activated when using the skill is in *SkillLogic*. Thanks to an attribute within the skill class, a skill can be related to its logical function. The fourth Item is responsible for

carrying out this work: *SkillManager*. This script finds which skill is trying to be activated and activates the skill in question.

The inventory block is responsible of managing the visual part of the inventory and equipping and unequipping items. To begin with, the main interface that would be used as an inventory was developed. After that, the script called *InventoryManager* was created, which would be in charge of managing the different inventory objects and the *InventorySlot* script, which would work as an object and would be assigned to each inventory slot.

Each inventory slot stores which slots it has around it and the skill it corresponds to. The *InventoryManager* is able to select a slot and move between the different slots on the right side of the inventory. When a slot is selected, it looks at what type of skill it has and automatically selects a slot on the left side that can contain the type of skill. The slots on the left are connected to the player. When an item is equipped to a slot on that part, the player gains the ability to use that ability.

The save phase was fairly straightforward, as much of the inventory work was done with the inventory in mind being able to scale from level to level. To perform the saving task, a *GameManager* was created to manage the most important inventory and player data and a *SaveDataManager*. The *SaveDataManager* is a script that saves information in a text document saved in the *persistentDataPath* of the computer and is responsible for reading this information when a load is required. The information saved by the *SaveDatamanager* is: the player's current health and mana, the current level, the last checkpoint the player stepped on, a boolean indicating which items are unlocked and which items are not, the items equipped in each slot(none if there is none).

The information is loaded in three phases. The first loads the level and the player at the position of the last past checkpoint, the second

loads the player's stats, and the third loads the inventory. Each of these phases has a different function and can be called depending on what is necessary for each situation.

The AI phase was also very simple, since the *NavMesh* system that *Unity* has implemented as standard was used. Each enemy has a *NavMeshAgent* and a couple of very basic functions. The first function allows them to move from *WayPoint* to *WayPoint*, waiting a few seconds before getting on their way after reaching one.

The second function is linked to its line of sight, which is made up of a *CapsuleCollider* that acts as a trigger. In this way, when the player enters the *CapsuleCollider* a signal is sent to the enemy to warn them that the player has been detected. When it detects the player, another function is activated to chase them and get within shooting range. This distance is determined by a floating number that can be adjusted for each enemy independently. When the distance is correct, the enemy will call another function to fire a projectile in a parabola calculated by hand on the spot, with an internal cooldown of one second. If the player gets too close, the AI has one last function to move away until the shot is possible again.

The boss phase was the easiest as it does not have an artificial intelligence per se, but is governed by Randoms. The boss teleports every 15 seconds to a new location of the 5 that it has saved (same operation as the AI *waypoints*) and every 10 seconds invokes a random attack between two. The first is a chain of four lightning bolts falling from the sky. The second is a sphere directed towards the player. The former has an 70 percent chance of occurring, while the latter is only 30 percent. However, there is a controller that forces the boss to throw a ball if he hasn't thrown a ball in 6 attacks. In addition, the rate of the ball increases by 10 percent for each point of health that the boss is missing, up to a maximum of 60 percent.

This is due to the design of the boss battle, which, as explained

previously, consists of a phase where you have to force the boss to shoot at platforms to charge up the mirror walls, and another where the player must guide the sphere to the mirror walls so that they bounce and go towards the boss.

With all phases of the core mechanics done, the next step was to make the scenarios, and with the scenarios came the last element in this game: the puzzles. The puzzles in this section of the game are very simple. They consist of several pressure plates that, when pressed, trigger stone blocks and walls to allow the player to move forward. In this way, the only challenge that exists is to find how to get to that platform. They are managed by a puzzle controller that checks if all the slates are pressed and opens the door if they are.

There are several types of puzzles with their respective scripts. The first and simplest is the one that only moves one block and needs a single switch. The second is a bit more complex, as it allows to add multiple switches and has a timer that returns the block to its original position several seconds after the last switch has been activated. The third type of puzzle controller allows to move an unlimited number of blocks in any direction and add an unlimited number of switches.

When the levels were finished, a simple main menu was implemented to finish off the gaming experience. The menu consists of four buttons: *New Game*, which loads a new game, *Continue*, which loads the game from the last save point, *Quit Game*, which closes the game, and *Delete Data*, which is used to delete the save file . Everything is managed with functions of *Unity's SceneManager*.

The *Scripts* have been stored in different groups, these being *General*, *Skills*, *Inventory*, *SaveData*, *EnvItems*, *Menu*, *Puzzle*, *Enemy* and *Inventory*. In the *Skills* group are all the scripts related to the skills that the player has at his disposal and the database that saves them. In the *Inventory* group are all the *Scripts* in charge of man-

aging the inventory and communicating the *Skills* group with the player. In *SaveData* is the *Script* that saves the player's data and the level in which he is.

In the *Puzzle* group are the different *Scripts* that manage the types of *Puzzles* in the game, including the boss puzzle. *EnvItems* is a helper group. Scripts that manage the game environment are stored in it, such as triggers that activate the game instructions or the object collection system. They are very simple scripts. In the Enemy group are both the controllers of the basic enemies, as well as the boss and the projectiles they use.

Finally, in the *General* group all those *Scripts* that do not belong to any other group such as the *CharacterController* or the *AudioManager* are saved.

4.2 Results

This section will present the results obtained with respect to the initial proposal and those that have not been achieved. To open the section, this is the current version of the game:

- Link to repository: Fatum Repository
- Google Drive Build Folder: Fatum Build Folder

The results obtained have been quite satisfactory. All the objectives of the project have been achieved quite satisfactorily. It should be noted that the design of the game in general terms has been carried out for a video game of a much larger scope than what has been implemented, since it was agreed to only implement the first phase for reasons of time.

In terms of design, all the tarot cards have been transformed into objects and bosses with simple mechanics but that pose a challenge in the form of a puzzle and quite differentiated and unique areas have been created, at least in theory.

In terms of development, it was possible to implement everything planned and also optimize the game a bit through different "tricks" that have been in the video game industry for many years. For example, to hide the effect of the popping of the trees in the levels, fog is used, which not only serves this cause, but also helps to generate the desired atmosphere of the area. It must also be said that the results obtained by the *Terrain tool* were quite satisfactory although the textures could perhaps be better.

In general, the work has been quite round. Many of the teachings received in the degree and those obtained by own means and during external internships, which have been carried out at the same time as this project, have been used. It is an experience where you can lose a few minutes while you think about your things or completely immerse yourself in the atmosphere of the video game.

Chapter 5

Conclusions and Future Work

Contents

5.1 Conclusions

The truth is that at first I was not very clear what kind of project this was going to be. I had several initial ideas, but I didn't like any of them enough to want to dedicate three months of my life to it. However, there was something in each one that I liked, so I tried to put them all together and they ended up converging on this project idea. It was not an easy task but it has been achieved.

The design phase was perhaps the most fun for me. Especially the part where I had to take the tarot cards and turn them into something else. It's a bit complex trying to create 22 items with different abilities that work well with each other and don't overlap. I didn't just have to figure out what I wanted each one to do. I had to carefully read the different meanings of the cards and, through long

brainstorming, build the effect for each one and polish it, keeping in mind that I had to create objects of each type so that no archetype was left without too many options. In the end the mobility skills were off the hook but I had no other option to maintain the general balance of the game.

Also, the added challenge of using *design by subtraction* kept me grounded. When part of your design is based on only displaying what is strictly necessary on the screen, it forces you to consider things more carefully and puts a stop to designing. The game must look austere and the scenarios must not feel overloaded. In short, the general feeling had to be similar to that caused by the *Shadow Of The Colossus*[18] scenarios but it also had to have its own essence. In this aspect I think I have done a good job.

In the development phase I experimented with *Unity* tools that I hadn't touched until now. The most notorious is undoubtedly the terrain. When I set out to make the scenarios, I wasn't very sure if I was going to try to model them in *Blender* or using the editor that *Unity* has integrated, but when I saw that there was another option and that it also allowed me to do the work as if I were painting with a brush, I decided that it was worth a try. With a little more practice, perhaps the result would have been better, but I am quite satisfied with the work done.

The inventory system is perhaps the most complex part of the project. In part it is its core, since everything revolves around objects. It's the part I spent the most time on in the end, but I think it was the right decision. After all, if the inventory failed, nothing in the game would work afterwards. In part I was lucky, since in my external internships I was working on something similar, so I wasn't starting from scratch. Even so, this inventory is much more complex than others that I have been able to develop and the truth is that I am proud of it.

In general, I think the project has turned out quite well for someone who is not specialized in the artistic side of video game development. It has everything I wanted it to have.

5.2 Future work

This project is planned to be much bigger. Therefore, I intend to continue it and try to go all the way with it. Once it is evaluated, I intend to upload it to *Itch.io* as a free demo and improve it based on the feedback it may receive.

Apart from this, several of the tools that I have learned to use thanks to this project will be useful to me in the future. *Terrains* have massive potential when it comes to designing settings with textures, grass, prefab trees, and decoration. I know that I have not taken full advantage of them and I intend to continue learning to use them. Just like *Cinemachine* cameras. I am aware that they can be used to make cinematics in game and I would like to learn now that I have managed to lay the foundations.

As for future projects, I've always liked narrative, so chances are that if I develop something new on my own it will have a great narrative weight.

Bibliography

[1] Blender. Blender. https://www.blender.org. Accessed: 2022-05-11.

[2] Sara Borondo. Design by subtraction. https://www.elcorreo.com/tecnologia/fs-gamer/lanzamientos/diseno-sustraccion-conectar-20200421102241-nt.html. Accessed: 2022-05-11.

[3] Silver Cats. Nature assets. https://es.wikipedia.org/wiki/Shadow_of_the_Colossus. Accessed: 2022-05-11.

[4] Historia de la Empresa. Narrative salary. https://historiadelaempresa.com/como-convertirse-en-guionista-de-videojuegos. Accessed: 2022-05-11.

[5] EIM. Artist salary. https://eniyimeslekler.com/es/cuanto-gana-un-artista-del-juego-salarios-que-hace. Accessed: 2022-05-11.

[6] GitHub. Github desktop. https://desktop.github.com. Accessed: 2022-05-11.

[7] GitHub. Github webpage. https://github.com. Accessed: 2022-05-11.

[8] GlassDoor. Level designer salary. https://www.glassdoor.es/Salaries/level-designer-salary-SRCH_KO0,14.htm?countryRedirect=true. Accessed: 2022-05-11.

[9] Nix Hydra. The arcana main page. https://thearcanagame.fandom.com/wiki/The_Arcana_(game)_Wiki. Accessed:2022-05-11.

[10] Aquarius Max. Building assets. https://assetstore.unity.com/packages/3d/environments/landscapes/p-w-temple-edition-33637. Accessed: 2022-05-11.

[11] Mixamo. Mixamo webpage. https://www.mixamo.com. Accessed: 2022-05-11.

[12] Qianyuez. Skybox. https://assetstore.unity.com/packages/2d/textures-materials/sky/free-night-sky-79066. Accessed: 2022-05-11.

[13] TokioSchool. Designer salary. https://www.tokioschool.com/noticias/sueldo-disenador-videojuegos-cuanto-cobra/. Accessed: 2022-05-11.

[14] TokioSchool. Programmer salary. https://www.tokioschool.com/noticias/cual-es-sueldo-programador-videojuegos/. Accessed: 2022-05-11.

[15] Trello. Trello webpage. trello.com/. Accessed: 2022-05-11.

[16] Unity. Unity. https://unity.com/pages/unity-pro-buy-now?ds_rl=1295837&ds_rl=1295837&gclid=CjwKCAjwve2TBhByEiwAaktM1JZUHiwuTp6YV63ZKCjtEWsF9sUBgOtESryzipr06t0yl6cbWjNSPBoCRBwE&gclsrc=aw.ds. Accessed: 2022-05-11.

[17] Unity. Unity asset store. https://assetstore.unity.com/?gclid=CjwKCAjwve2TBhByEiwAaktM1MVjXh60n0Ja2Legpe-5dNy1-vmkvrURMrBwYAreVhOUdS1woyTNPxoCBwE&gclsrc=aw.ds. Accessed: 2022-05-11.

[18] Wikipedia. Shadow of the colossus. https://es.wikipedia.org/wiki/Shadow_of_the_Colossus. Accessed: 2022-05-11.

[19] Wikipedia. Tarot explained. https://es.wikipedia.org/wiki/Tarot_(adivinación). Accessed:2022-05-11.

Appendix A

Other considerations

A.1 Puzzles solution

This section explains how to solve all the puzzles in *Fatum*. They will be resolved in the order in which the player is intended to face them, since the second level, being an open level, has the possibility of being resolved non-sequentially.

The first puzzle that will be discussed is *The Tower*. It is a rectangular tower with two floors located on the second level. The goal is to reach the highest floor. To do this, the first thing to do is press the four buttons on the ground floor. There is one on each side of the tower**(1)**. After pressing it, some stairs will be activated that will allow access to the next floor. Once on top, the player has to look for the switch that opens the next stairs, which is in the upper left **(2)**. When you press it, blocks will come out of the walls. The player has to use them to climb to the second floor. At the top there is a switch**(3)**. This turns on an elevator that takes the avatar directly to the top of the map.

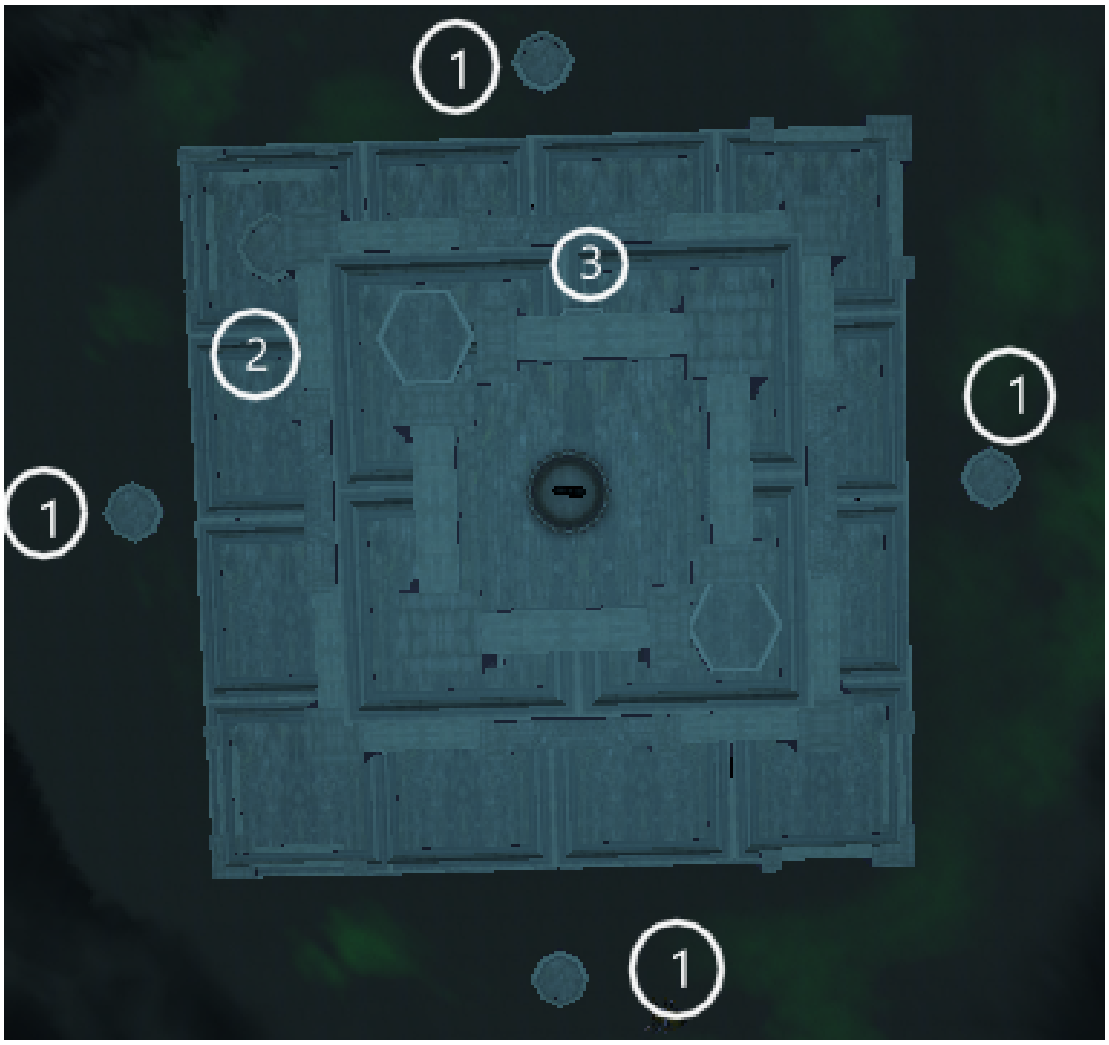

Figure A.1: The Tower

The second puzzle is on the second level as well. This is a labyrinth, in which you will have to look for switches to move walls and form a path to the end. Upon entering the maze, the player will be trapped, being forced to choose one of three paths. To advance in the puzzle you have to follow the following sequence:

1. Go down the left path and hit the switch. A wall will open to the right.
2. Continue to the right and go down again to the area at the beginning. Take this time the path to the right and press the switch. A wall will open up on the center path.
3. Return to the center path and continue to the end. Once there turn to the path to the right and press the switch. A door will open to the left.
4. Return to the central path again. This time take the left path and climb up. The last door in the center will open.
5. Go back to the central path and go up to claim the reward. Taking it will open the door at the beginning of the labyrinth.

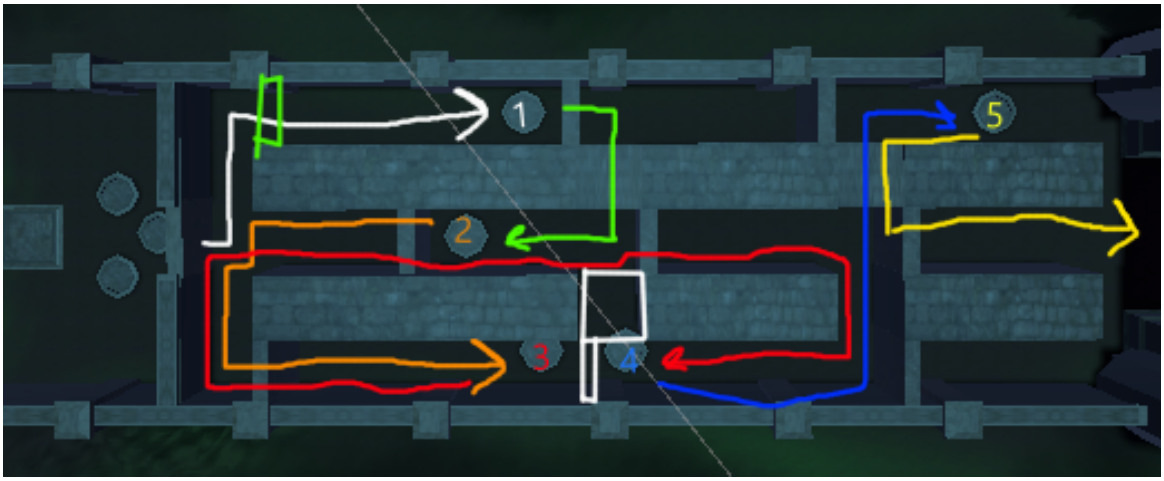

Figure A.2: The Labyrinth

The next puzzle is on the boss level. In fact, it's about the boss fight itself. Throughout the map there are a few triangular areas with three platforms. These platforms absorb the lightning that the boss uses to attack. When all three platforms have absorbed a hit, the three outer walls will rise. Once the walls have been raised, the player must attract the projectile that the boss throws, his second attack, to the walls so that it hits and ricochets towards the boss. Upon receiving a projectile impact the structure unloads and the walls lower. Structures can only be charged once, so more will need to be charged to beat the boss. You have to repeat this process three times to defeat the boss.

Figure A.3: The Hermit

The last puzzle is found on level four. This puzzle requires having the skill that is unlocked by defeating the boss and picking up the item it drops. This ability allows the player to create a spectral copy of the avatar and control it. With this copy, certain switches must be pressed to remove the force fields that prevent the player from moving forward. The sequence to solve the puzzle is as follows:

1. Use the ability to create the copy and with it enter the first entrance on the left. Go down the hall and press the switch. The path is blocked by a wall.
2. Undo the copy and create another. This time go to the last entry on the right. Go down the hall and press the switch.
3. Once activated, return to the initial position with the copy to access the first entry on the right side. Activate the switch.
4. Return to the central area with the copy and go up to the second entrance on the left. Press the switch and the puzzle is complete.

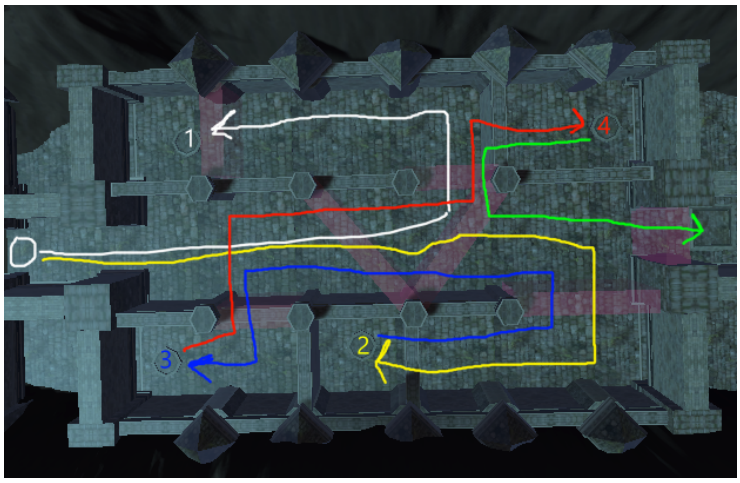

Figure A.4: The Magic Labyrinth

A.2 The Wanderer

The Wanderer is the character that the player will control. Its design is a fundamental section of the project, so in this appendix an exhaustive analysis from different areas will be devoted to it.

Starting with the meaning within the game, *The Wanderer* has been designed for the sole purpose of being an avatar and not a character. This means that it lacks will, character and personality. It is an empty shell controlled by the player, a tool to overcome the tests that the environment proposes. Its design differs with the entire aesthetic of the game to highlight it and convey the feeling that it's in a world that does not belong it. This avatar also serves as a link between the player and *The Narrator*, since the latter transmits its advice and instructions to the avatar. However, there is a twist to this, as *The Narrator* is not only aware that *The Wanderer* is a puppet, but is also aware that there is someone behind a screen controlling it. Therefore, The Narrator does not only control the avatar with his requests. It also controls the player indirectly without them realizing it.

The next section to analyze is its visual design. Opening this section it is the form. The Wanderer features the characteristics of a small child, with big head and spindly arms and legs. The intention behind this decision is to try to make the character to transmit innocence. Contrary to this idea is the mask that the avatar wears. It is a mask with a pointed nose and empty eyes, with an opening where the mouth should be. Its function is double. The first is to give the avatar a gloomy as well as picturesque touch. The second is to hide the avatar's face to emphasize its impersonality.

The main colors chosen are white and especially black. The white color often represents purity, in this case, the purity of a creature without a mind to think, a will to break and a voice to cry suffering. It also helps emphasize the feeling of innocence. On the other hand,

black is an elegant color as well as mysterious and gives the avatar the mystical and melancholic aura that the game seeks to convey.

To end this section, it should be noted that the decorative elements of the armor on the shoulder pads and chest, as well as the plates on its spine, are used in game as indicators of the character's remaining health and mana.

From the narrative point of view, this avatar has been created from the relic of *The Fool*, a tarot figure that is linked to the others, since it is who symbolizes the traveler who travels the path of destiny. It is the avatar itself that walks this same path in the video game, gathering the rest of the artifacts formed from the remnants of the god of destiny, who speaks to it through its mask.

Figure A.5: Character model from the front

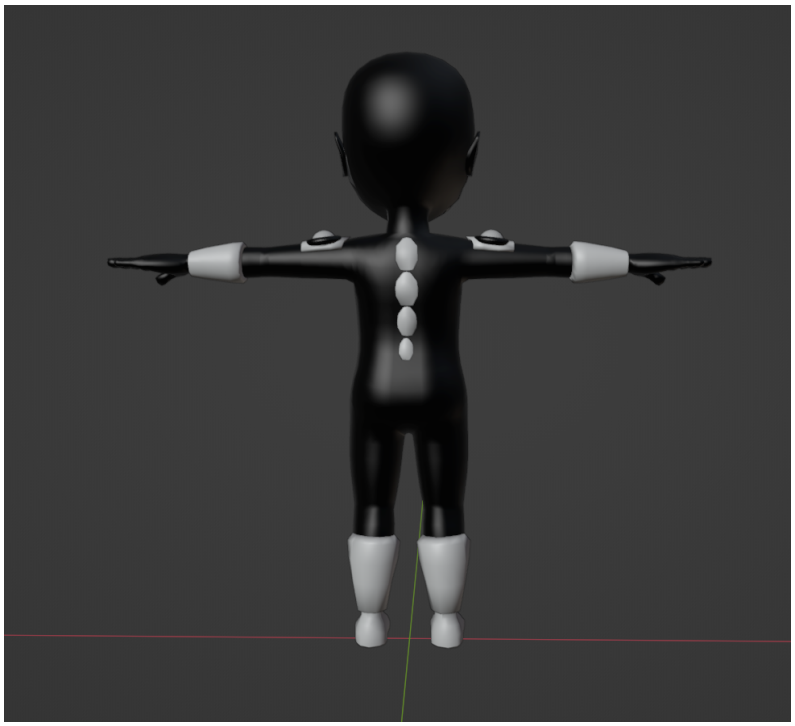

Figure A.6: Character model from the back

Appendix B

Source code

Many scripts have been developed in this project to make the whole world of *Fatum* work. It can be accessed through this link:

`https://github.com/JoseIgnacioValverde/FatumScripts.git`