Kharduk blade : A metroidvania video game with stylized 3D art

Ivan Ruiz Ribera

Final Degree Work
Bachelor's Degree in
Video Game Design and Development
Universitat Jaume I

July 20, 2022

Supervised by: Diego José Díaz García

Acknowledgments

First of all, I would like to thank my Final Degree Work supervisor, Diego José Díaz García, for his guidance during the project.

Thanks to my parents, for their support and letting me study this degree and achieve my dreams.

Thanks to Sara, for her unconditional support.

And thanks to my friends from the degree, Adrián, José Ignacio, Ignacio, Jorge, David and Roberto for helping me during this journey.

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

Kharduk Blade it's a metroidvania 3D video game, with a stylized game art, in which the player controls Darren, a priest devoted to Kharduk, who is decided to end with the users of necromancy. This project consists on creating the first level of the game, including three characters, two environments and the rest of the elements that form a video game. 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

1.1 Work Motivation

The main motivation for this work is to create a serious and structured project where I can test all the skills learned throughout the degree, specifically in the field of the 3D art, where I invested the most hours, both in the degree and in this project, even so, without leaving aside the rest of the elements that form a video game.

1.2 Objectives

There are 4 main objectives to accomplish:

- Create three characters, with an stylized and attractive design, and with a professional finish.
- Create two environments that suits the art style of the characters.
- Create a fluid gameplay.The game must be attractive not only by it's art, but also for it's gameplay.
- Complement the game with sounds and VFX to make it more appealing.

1.3 Environment and Initial State

The project has been made entirely by me, with my own equipment, all scripts and assets have been made from blank. I decided to use Blender for modeling, Unity as game engine, and Krita for 2D elements because we used this programs on the degree and I can get great results with them. For texturizing I have used Substance Painter, even so we haven't used it in the degree, it's a program I master, and it's the industry standard, so it allows to obtain professional results.

Chapter 2

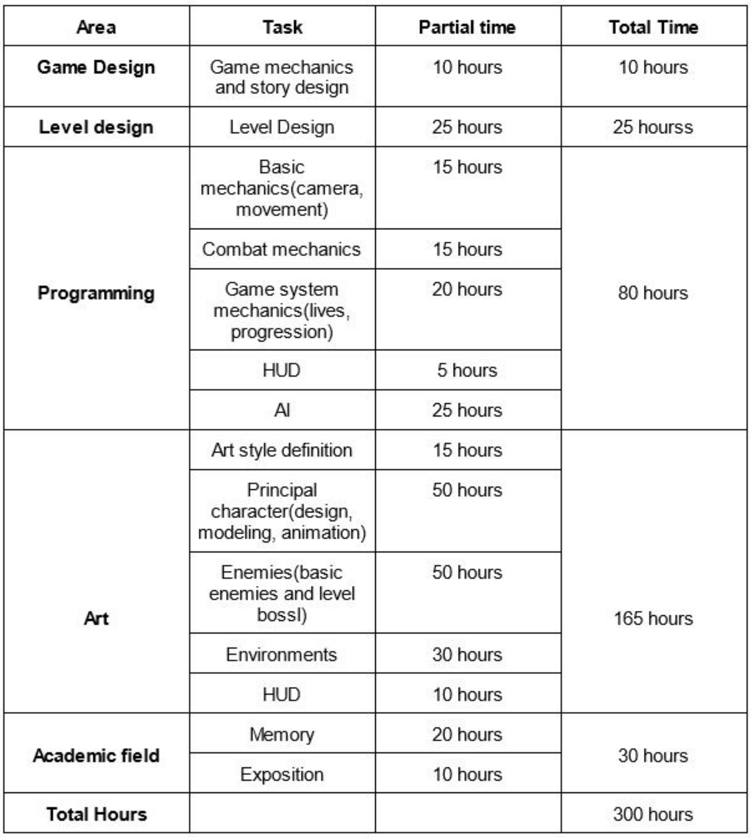
Planning and resources evaluation

Contents

2.1 Planning

In this section, the distribution of tasks and their extension in time is explained, is also explained the work methodology used to organize the project. First can be seen an estimation of how long would take each one, bearing in mind that it is an estimation, it doesn't reflect the real time, but it was useful to organize(see Table 2.1)

The work methodology used for organizing this project was Scrum, this methodology is based on development iterations(called sprints) that last a fixed time and are oriented to develop a new version of the game, in every iteration some tasks are selected, and the team develops them, at the end of the iteration there's a meeting where the tasks are reviewed to check if they are completed and the next iteration is planned.In this methodology there's also a daily meeting with all the team, as I am a single person instead of a team, the daily meeting was replaced with a work diary, where I made a note of every task and the time to complete it, and the meeting at the end of the sprint, was made with my supervisor. The tool used for organizing the sprints was, Trello, and was decided that every sprint would last for two weeks. All task were divided in three theme categories, art(purple), programming(light blue) and academic(pink), and used three time categories, to do(red), in progress(yellow), and finished(green), and added

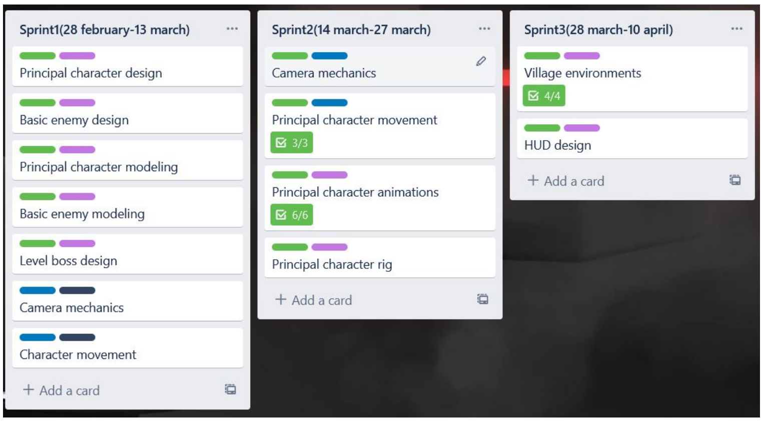


Area	Task	Partial time	Total Time
Game Design	Game mechanics and story design	10 hours	10 hours
Level design	Level Design	25 hours	25 hourss
Programming	Basic mechanics(camera, movement)	15 hours	80 hours
	Combat mechanics	15 hours	
	Game system mechanics(lives, progression)	20 hours	
	HUD	5 hours	
	AI	25 hours	
Art	Art style definition	15 hours	165 hours
	Principal character(design, modeling, animation)	50 hours	
	Enemies(basic enemies and level bossl)	50 hours	
	Environments	30 hours	
	HUD	10 hours	
Academic field	Memory	20 hours	30 hours
	Exposition	10 hours	
Total Hours			300 hours

Figure 2.1: Table showing the time estimations

one more categories to show task that haven't been completed in time.(see Figure 2.2, Figure 2.3 and Figure 2.4)

As these times were an estimation, here will be explained the difference between the estimated time and the real time dedicated to each part and to the whole project(see Table 2.5). The first remarkable difference is on the level design time, on the beginning was planned to do bigger map for the castle, with more paths and extension, but I realized that I had no time for that so I ended creating a smaller map, so there was no need to spend so much time on the design. On the programming area was invested less time than I thought I Would need, on the combat mechanics was invested more time than in the planning, but there have been included the combat AI of the enemies, so it is compensated with the AI area, in which have been invested less time than in the planning because only have been included there the non-combat AI, it took me more to program the HUD than I believed, and the progression mechanics took me less time because I made the map smaller, so there was no need to do the planned checkpoints. he artistic section, the definition of the artistic style have taken lees time than planned, the time to create the principal character was well-planned, but no the time to create



Sprint1(28 february-13 march)

- Principal character design
- Basic enemy design
- Principal character modeling
- Basic enemy modeling
- Level boss design
- Camera mechanics
- Character movement
- + Add a card

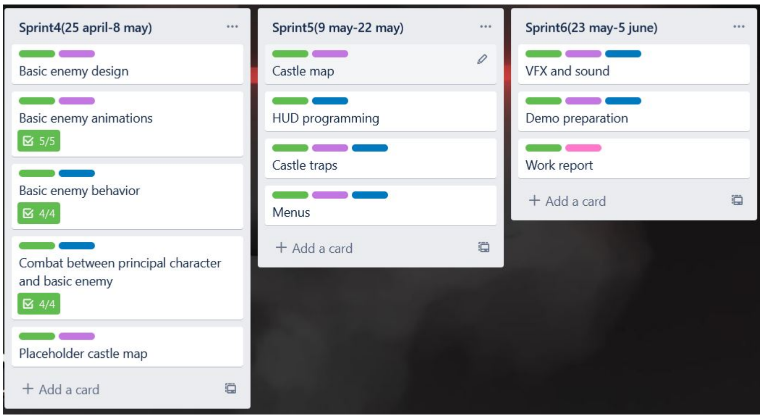
Sprint2(14 march-27 march)

- Camera mechanics
- Principal character movement ☑ 3/3
- Principal character animations ☑ 6/6
- Principal character rig
- + Add a card

Sprint3(28 march-10 april)

- Village environments ☑ 4/4
- HUD design
- + Add a card

Figure 2.2: Trello tables showing sprints 1, 2 and 3



Sprint4(25 april-8 may)

- Basic enemy design
- Basic enemy animations ☑ 5/5
- Basic enemy behavior ☑ 4/4
- Combat between principal character and basic enemy ☑ 4/4
- Placeholder castle map
- + Add a card

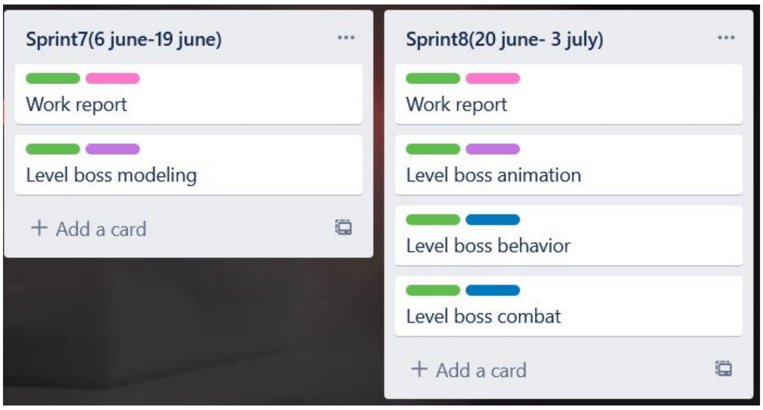
Sprint5(9 may-22 may)

- Castle map
- HUD programming
- Castle traps
- Menus
- + Add a card

Sprint6(23 may-5 june)

- VFX and sound
- Demo preparation
- Work report
- + Add a card

Figure 2.3: Trello tables showing sprints 4, 5 and 6



Sprint7(6 june-19 june)

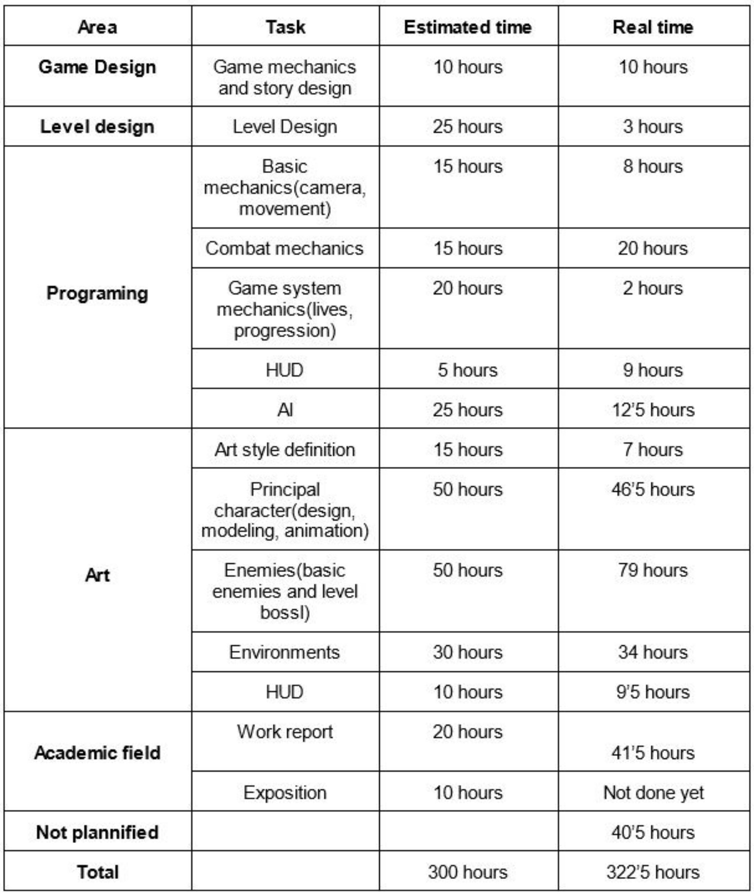
- Work report
- Level boss modeling
- + Add a card

Sprint8(20 june- 3 july)

- Work report
- Level boss animation
- Level boss behavior
- Level boss combat
- + Add a card

Figure 2.4: Trello tables showing sprints 7 and 8

the basic enemy and the level boss, they ended with more details than planned, so was invested more time on creating them. In creating the environments have been invested a bit more than planned, but it was almost all on creating the village. On the work report was invested twice as long as planned, a lot of time was invested on taking and preparing photos for the report. Finally, was added a section of things that weren't planned, here there are things like bugs fixing, that are unpredictable, and things like the environment trap, the VFX and the sounds, that although they were not in the planning table, they were in the Trello, which has been what was really used to organize the work.



Area	Task	Estimated time	Real time
Game Design	Game mechanics and story design	10 hours	10 hours
Level design	Level Design	25 hours	3 hours
Programing	Basic mechanics(camera, movement)	15 hours	8 hours
	Combat mechanics	15 hours	20 hours
	Game system mechanics(lives, progression)	20 hours	2 hours
	HUD	5 hours	9 hours
	AI	25 hours	12'5 hours
Art	Art style definition	15 hours	7 hours
	Principal character(design, modeling, animation)	50 hours	46'5 hours
	Enemies(basic enemies and level bossl)	50 hours	79 hours
	Environments	30 hours	34 hours
	HUD	10 hours	9'5 hours
Academic field	Work report	20 hours	41'5 hours
	Exposition	10 hours	Not done yet
Not plannified			40'5 hours
Total		300 hours	322'5 hours

Figure 2.5: Table showing estimated and real times

On the append A is included the work diary to consult all the time invested on the project(see Figure A.1, Figure A.2, Figure A.3).

2.2 Resource Evaluation

Human Resources

- **Programming** The salary of a junior video game programmer on Spain it's 21.500 per year[13], so it's 9'59€ per hour, on programing the game I invested 71'5 hours, so the cost is 683'54.
- **Art** The salary of a junior video game artist on Spain it's 24.734 per year[11], so it's 11'04 per hour, on the art I invested 196 hours, so the cost it's 2.163'84€.
- **Game design** The salary of a junior game designer on Spain it's 23.753 per year[10], so it's 10'6 per hour, on the game design I invested 13 hours, so the cost it's 137'85€.

The human cost of the project is 2.985'23€
Hardware

- **Laptop MSI GL62VR 7RFX:** It was worth 1300€ in 2018, now is discontinued, so it's impossible to know the price in the actuality.
- **Graphic Tablet Huion Kamvas Pro 13** 259€

Software

- **Blender** It's open source(Free)[8].
- **Krita** It's open source(Free)[12].
- **Substance Painter** It has a 19'35€ monthly subscription on it's web[1], or a yearly buy on Steam with a cost of 126€[2], this is the one I used.
- **Unity** Free for personal use[14].

The total cost of the project is 4.670'23€

Chapter 3

System Analysis and Design

Contents

3.1 Requirement Analysis

3.1.1 Functional Requirements

A functional requirement defines a function of the system that is going to be developed.

- **R1** The player can move horizontally with A, D.
- **R2** The player can jump pressing space.
- **R3** The player can double jump if he has unlocked the ability.
- **R4** The player can attack with left mouse click.
- **R5** The player can block enemy attacks with right mouse click.
- **R6** The player can evade attacks with Alt.
- **R7** The enemies can attack the player if he is in attack range
- **R8** The enemies will move to the player when detected.
- **R9** The player can change the volume of the game

- **R10** The player can exit the game.
- **R11** The player can pause the game.
- **R12** The player can resume the game.

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.

- **R13** The aesthetic of the game will be stylized and dark fantasy
- **R14** The gameplay must feel fun and smooth.
- **R15** The game must be well optimized

3.2 System Design

This section present the use cases(see Table 3.1 - Table 3.12)

Requirements:	R1
Actor:	Player
Description:	The player can move left pressing A key or right pressing D key
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused.
Steps normal sequence:	• 1.The player press A or D. • 2.The character moves in the desired direction. • 3.The character must be not attacking, evading or blocking.
Alternative Sequence:	None
Player moves horizontally	

Table 3.1: Case of use «CU1. Player move horizontally»

Requirements:	R2
Actor:	Player
Description:	The player can jump pressing space
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The player must be not attacking, evading, blocking or already jumping.
Steps normal sequence:	• 1.The player press Space key. • 2.The player jumps.
Alternative Sequence:	None
Player jumps	

Table 3.2: Case of use «CU2. Player can jump»

Requirements:	R3
Actor:	Player
Description:	The player can double jump pressing space when it's already jumping
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The player must had unlocked the double jump ability. • 4.The player must be already jumping.
Steps normal sequence:	• 1.The player press Space key. • 2.The player jumps again.
Alternative Sequence:	None
Player double jump	

Table 3.3: Case of use «CU3. Player can double jump»

Requirements:	R4
Actor:	Player
Description:	The player can attack the enemies pressing left click on the mouse
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The character must not be blocking, jumping, evading or already attacking.
Steps normal sequence:	• 1.The player press left click. • 2.The player attacks. • 3.The player hits the enemies on range.
Alternative Sequence:	• 3.1. The are no enemies on range.
Player attack	

Table 3.4: Case of use «CU4. Player can attack»

Requirements:	R5
Actor:	Player
Description:	The player block enemy attacks with right click
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The player must not be attacking, jumping, evading or already blocking.
Steps normal sequence:	• 1.The player press right click. • 2.The player block. • 3.The player don't receive damage from the enemy and the enemy gets weakened if the blocking it's done in time.
Alternative Sequence:	• 3.1. The blocking is not done in time and the player receives damage.
Player block	

Table 3.5: Case of use «CU5. Player block enemy attacks»

Requirements:	R6
Actor:	Player
Description:	The player rolls back and evade with Alt key
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The player must not be already evading.
Steps normal sequence:	• 1.The player press Alt. • 2.The rolls back and evades.
Alternative Sequence:	None
Player evade	

Table 3.6: Case of use «CU6. Player evade enemy attacks»

Requirements:	R7
Actor:	Enemy
Description:	The enemy can attack the player if it’s in range
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The player must be in range attack from the enemy.
Steps normal sequence:	• 1.The enemy attacks. • 2.The player receives damage.
Alternative Sequence:	• 3.1 The player is blocking and don’t receive damage and the enemy gets weakened.
Enemy attack	

Table 3.7: Case of use «CU7. Enemy attacks player»

Requirements:	R8
Actor:	Enemy
Description:	The enemy will move to the player when enters the vision range
Preconditions:	• 1.The player must be in game scene. • 2.The game must be not paused. • 3.The player must be in vision range of he enemy.
Steps normal sequence:	• 1.The enemy moves to the player.
Alternative Sequence:	None
Enemy moves to player	

Table 3.8: Case of use «CU8. Enemy moves to player»

Requirements:	R9
Actor:	Player
Description:	The player can change the volume of the game
Preconditions:	• 1.The player must be in options menu.
Steps normal sequence:	• 1.The player moves the volume slider to establish the desired volume.
Alternative Sequence:	None
Player selects volume	

Table 3.9: Case of use «CU9. Enemy change volume»

Requirements:	R10
Actor:	Player
Description:	The player can exit the game with the quit button
Preconditions:	• 1.The player must be in main menu or options menu.
Steps normal sequence:	• 1.The player presses the quit button. • 2.The games closes.
Alternative Sequence:	None
Player quits the game	

Table 3.10: Case of use «CU10. Player exit the game.»

Requirements:	R11
Actor:	Player
Description:	The player can pause the game
Preconditions:	• 1.The player must be in a game scene. • 2.The game must be not already paused.
Steps normal sequence:	• 1.The player presses escape key. • 2.The game pauses. • 3.The pause menu is showed.
Alternative Sequence:	None
Player pauses the game	

Table 3.11: Case of use «CU11. Player pauses the game.»

Requirements:	R12
Actor:	Player
Description:	The player can resume the game
Preconditions:	• 1.The player must be in a game scene. • 2.The game must be paused.
Steps normal sequence:	• 1.The player presses escape key or press the resume button. • 2.The games resume. • 3.The pause menu is closed.
Alternative Sequence:	None
Player resumes the game	

Table 3.12: Case of use «CU12. Player resumes the game.»

Chapter 4

Work Development and Results

Contents

4.1 Work Development

4.1.1 Characters

Three characters have been created for the game, the protagonist, a basic enemy, and the level boss, in this section will be explained the workflow followed to create each one of them.

Darren, The Priest of Kharduk

Design

This is the protagonist of the game, it's equipment was based on clerics of rol games like Dungeons And Dragons, being armored but not too heavily, and with a big weapon. In the image(see Figure 4.1) can be seen the first sketches for the equipment of the protagonist, the left one was used for creating the final design(see Figure 4.2).

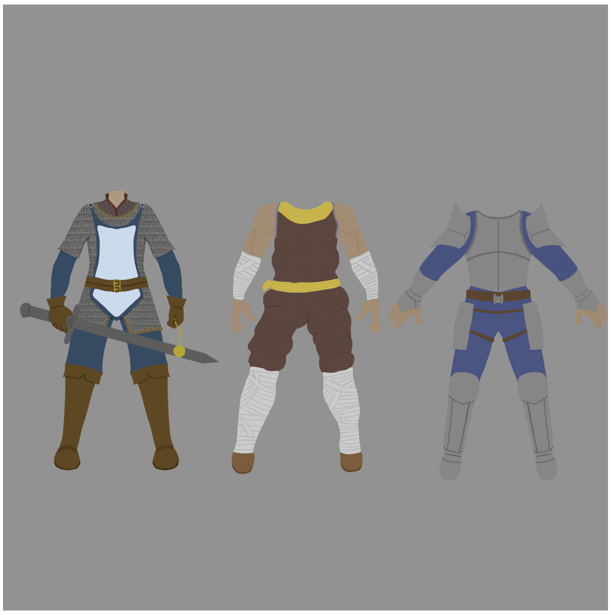


Figure 4.1: First designs for the protagonist

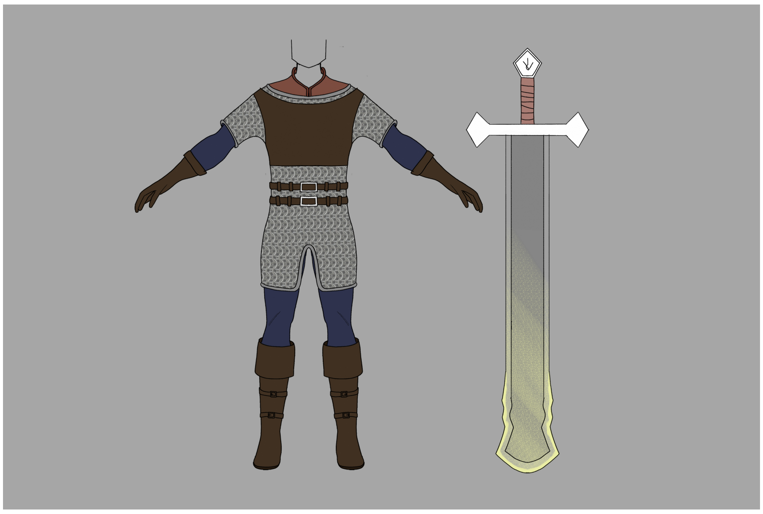


Figure 4.2: Final design for the protagonist

Modeling

The character was divided in five parts, that will work as a whole when they are united by the rig. Those objects are the head(see Figure 4.3), the armor(see Figure 4.4), the legs and arms(see Figure 4.5), the boots and gloves(see Figure 4.6), and finally the sword(see Figure 4.7). To create organic parts like the head, or the details on the armor, the technique used was sculpt modeling, which is similar to clay sculpting in real life but digitally, using the tools that Blender provides that simulates techniques of clay sculpting[7], to create the non organic parts like the base form of the armor or the sword, was used box modeling, which is to create a complex shape, starting from a primitive shape(box, cylinder,sphere) using different box modeling tools like extrude, bevel, inset, etc[5]; and modifiers like subdivision surface, solidify, etc[6]. With sculpt modeling, usually you end creating a mesh with millions of triangles, which are not optimized for real time rendering and animation so they need to be retopologized, retopology will be explained in the next section. The objects created with box modeling, unless you add more details sculpting on them, usually doesn't need to be retopologized.

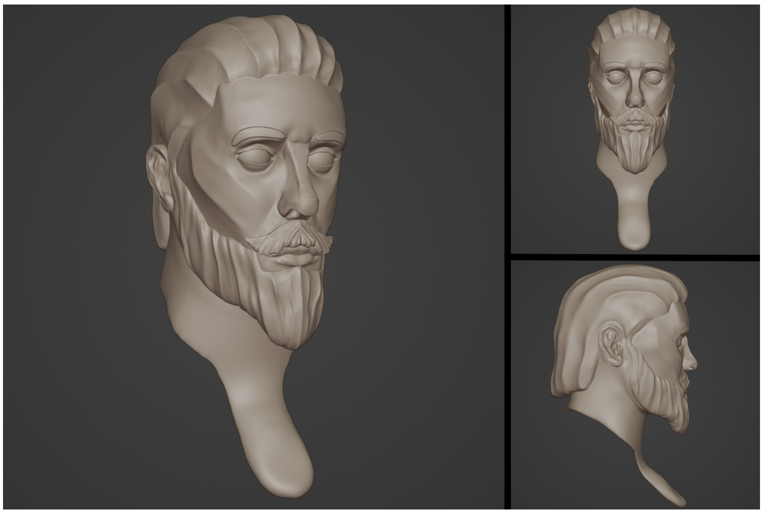


Figure 4.3: Head high poly

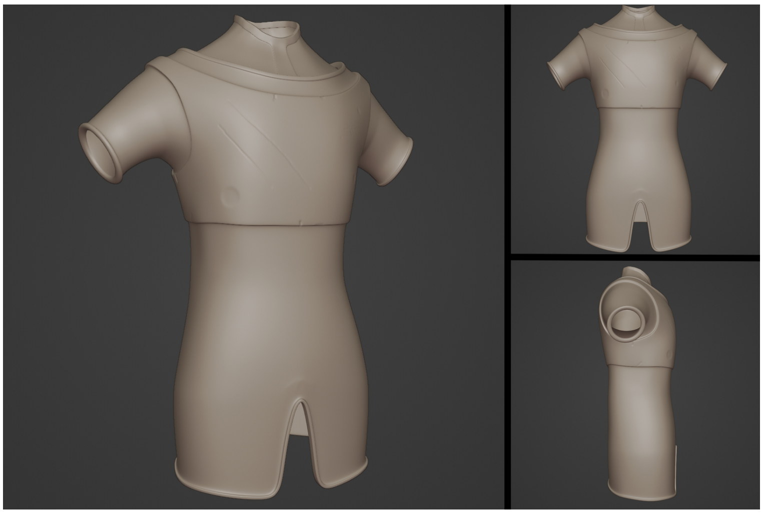


Figure 4.4: Armor high poly

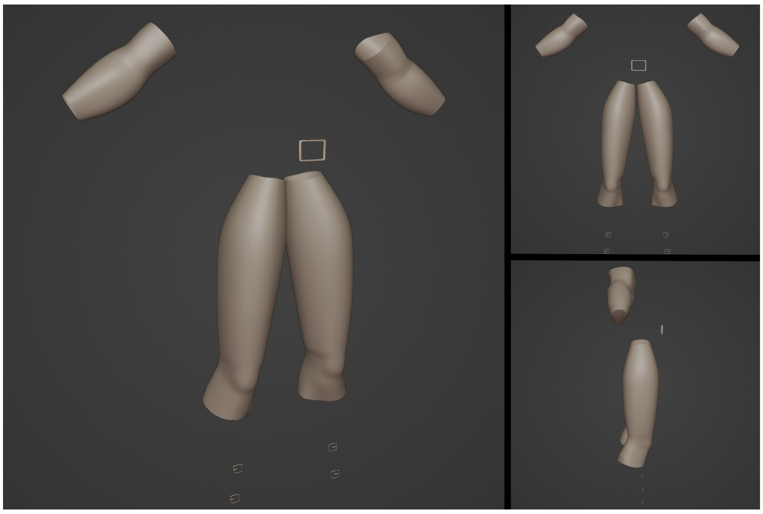


Figure 4.5: Legs and arms high poly

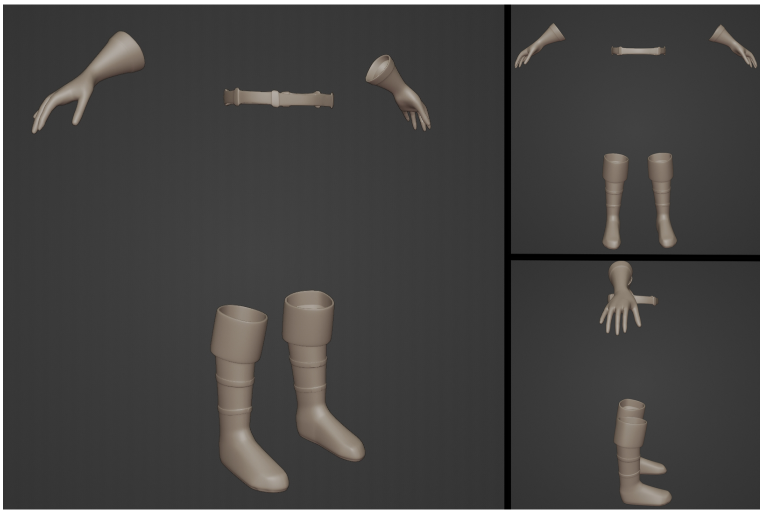


Figure 4.6: Boots and gloves high poly

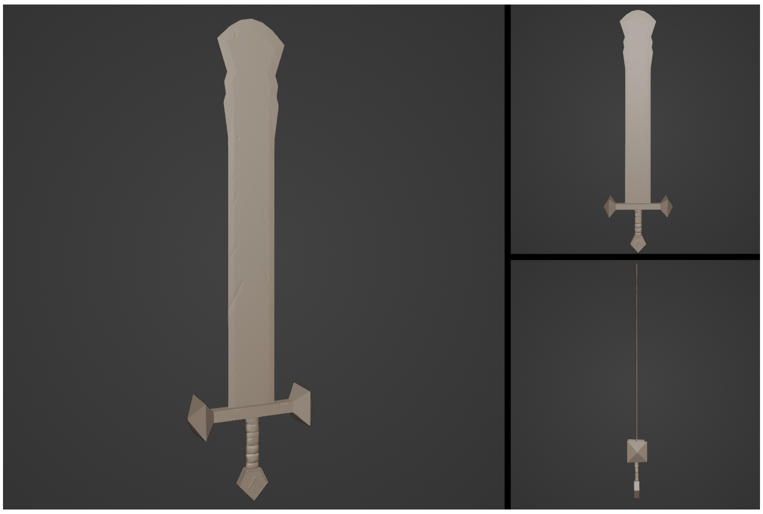


Figure 4.7: Sword high poly

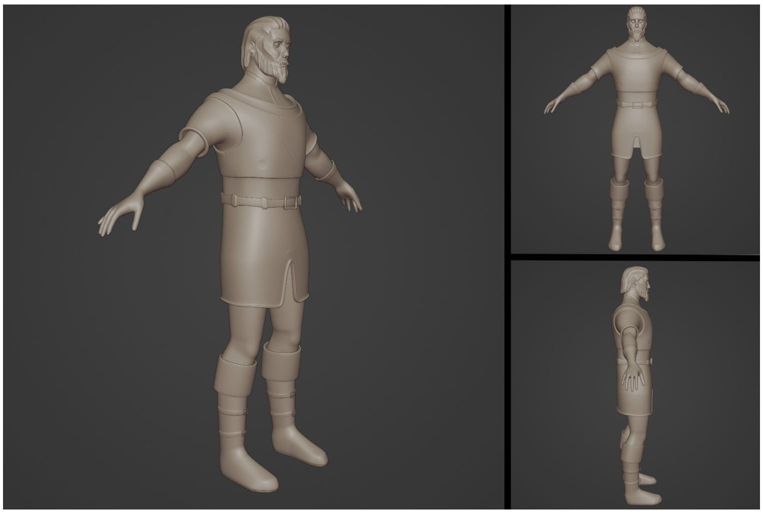


Figure 4.8: Character high poly

Retopology

Retopologize is to create a new low poly mesh, using as a start point the high poly mesh, using different techniques you create a new mesh with its vertices snapped to surface of the high poly mesh, you end up creating a mesh with relatively low triangles with the general silhouette of the high poly mesh. Later, the details from the high poly mesh will be added to the low poly mesh through different texture maps. To retopologize was used Blender with an add-on called Retopoflow[3], this add-on provides a new interface and tools to speed up the process of retopologizing. I tried to balance between quality and number of triangles.In this case only the head and the boots and gloves needed to be retopologized.

The head contains 6.574 triangles(see Figure 4.9).

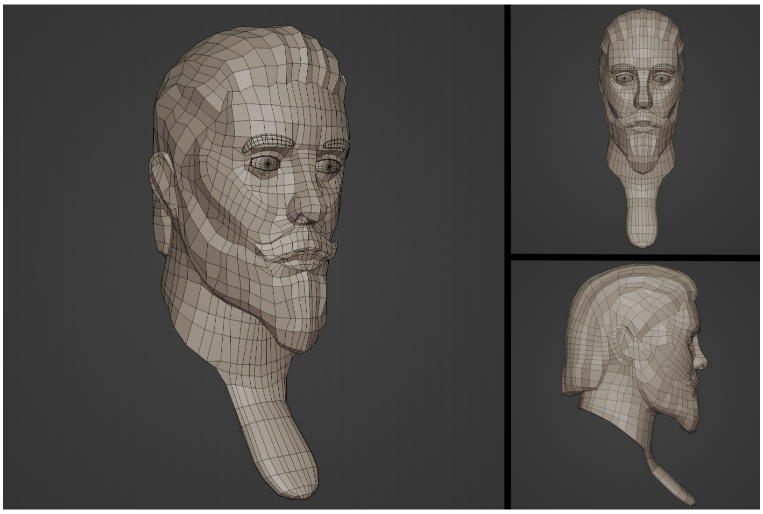


Figure 4.9: Head Wireframe

The armor contains 5.374 triangles(see Figure 4.10).

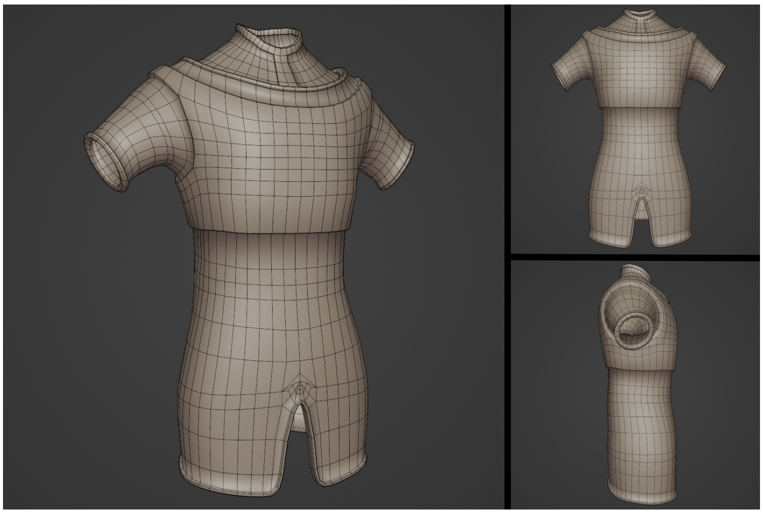


Figure 4.10: Armor Wireframe

The legs and arms contains 3.360 triangles(see Figure 4.11).

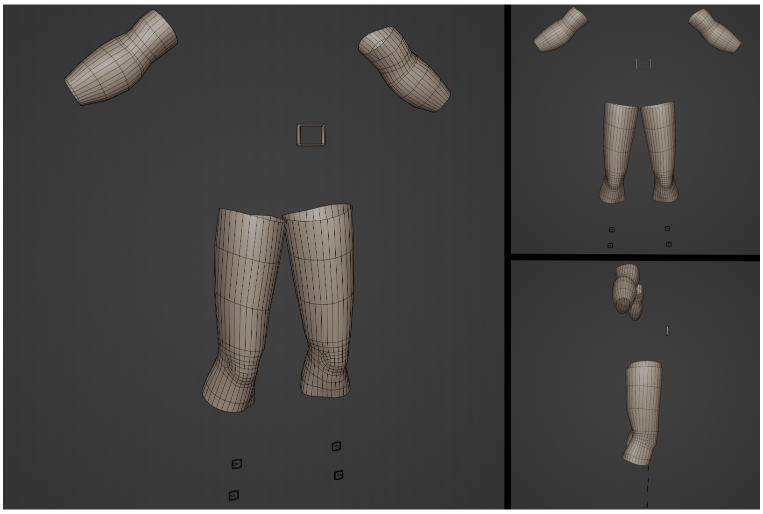


Figure 4.11: Legs and arms Wireframe

The boots and gloves contains 7.969 triangles(see Figure 4.12).

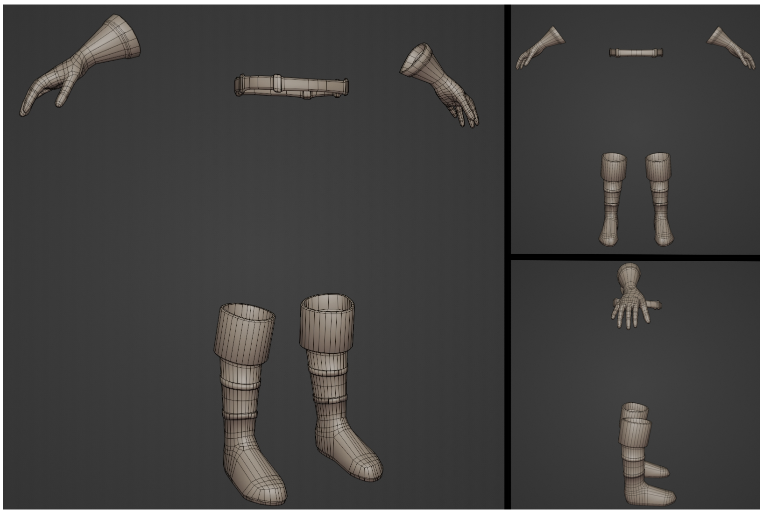


Figure 4.12: Boots and gloves Wireframe

The sword contains 1058 triangles(see Figure 4.13).

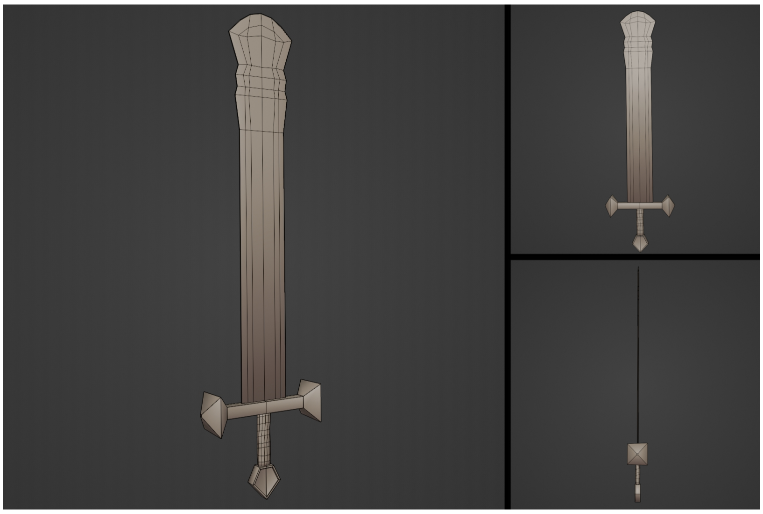


Figure 4.13: Sword Wireframe

In total the model contains 24.334 triangles, taking into account the level of detail, it's an acceptable number and as there will only be a few characters on screen at the same time, it's a fully manageable number for modern computers.

UV Unwrapping

UV unwrapping consists on creating cuts on the topology, to unroll the 3D mesh on to a 2D plane, so it can be properly texturized.

Here is the UV Unwrapping of each part(see Figure 4.14, Figure 4.15), this is a very important part, because it will help to create great materials.

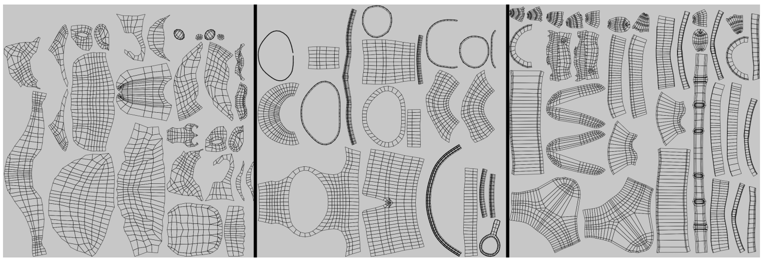


Figure 4.14: UV for head(left), armor(center), and boots and gloves(right)

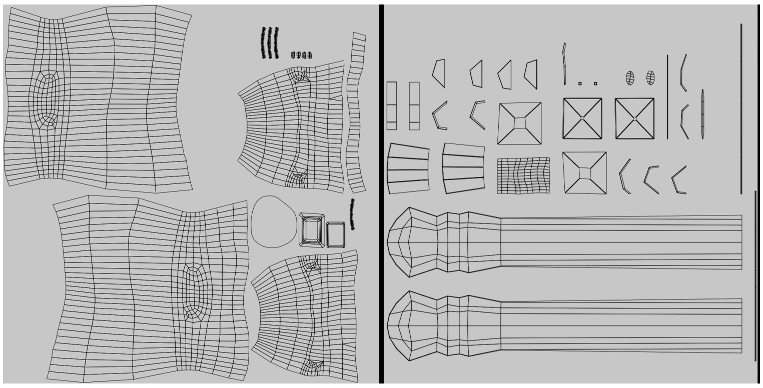


Figure 4.15: UV for legs and arms(left) and sword(right)

Textures

Substance Painter was the tool used for texturing , and the correct maps were exported for the Universal Render Pipeline(URP)[15], that is the Unity pipeline used for this project, the maps are Albedo, Normal, and Specular/Metallic. The Albedo map contains the information about color of the different objects of the mesh, the normal map contains information used to fake the lighting of bumps and dents, it's used to add details without using more polygons, Specular map are used to define the shininess/highlight on a surface, a Metallic map is a grayscale image where black pixels correspond to non-metals and white pixels correspond to metals, metals reflect much more light than non metals.

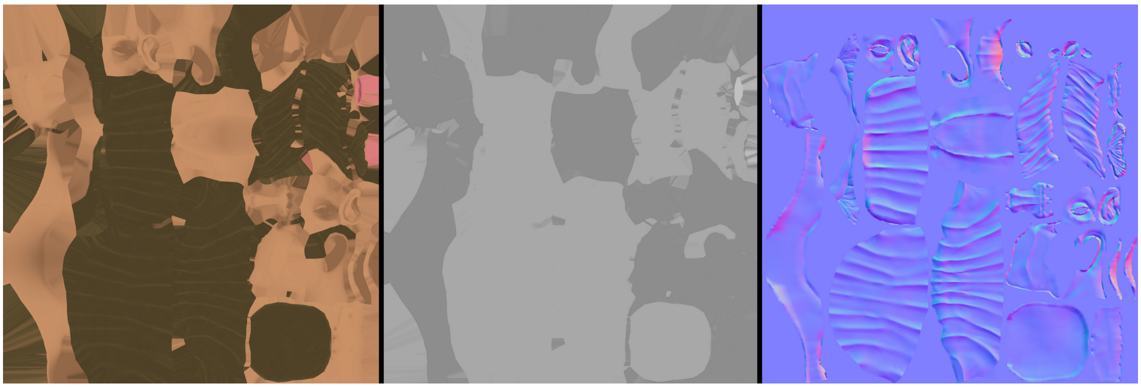


Figure 4.16: Head textures, Albedo(left), Specular(center) and Normal(right)

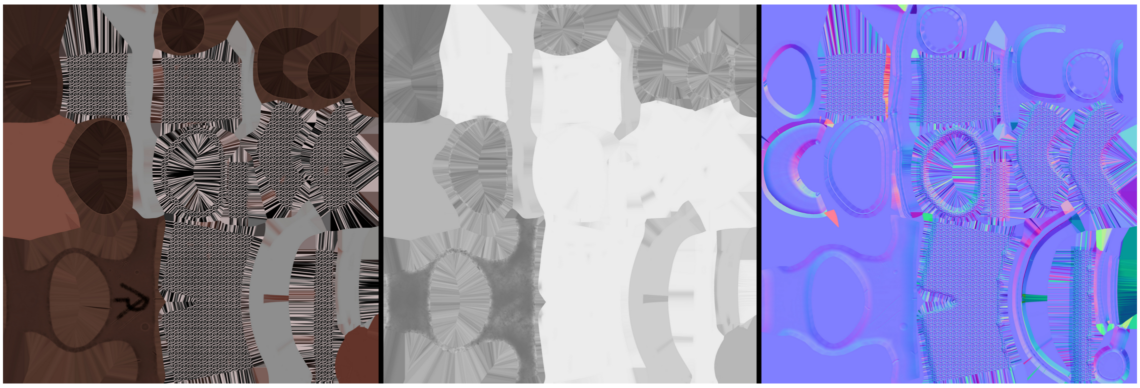


Figure 4.17: Armor textures, Albedo(left), Metallic(center) and Normal(right)

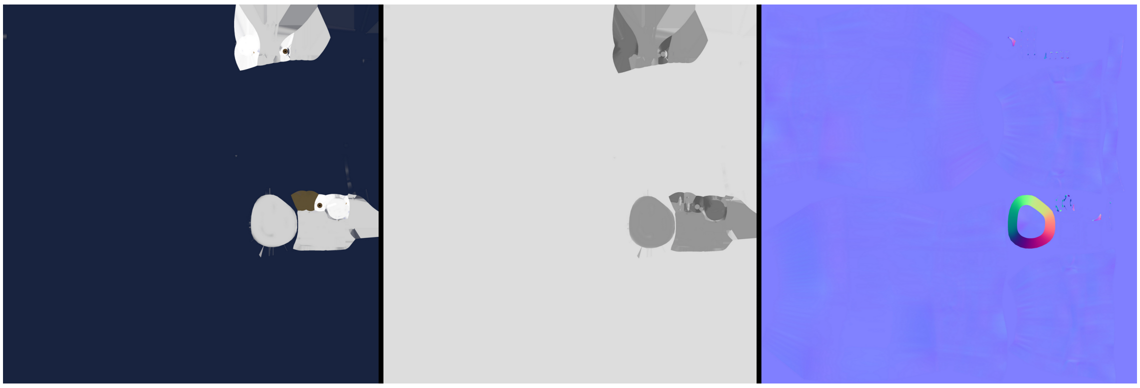


Figure 4.18: Legs and arms textures, Albedo(left), Specular(center) and Normal(right)

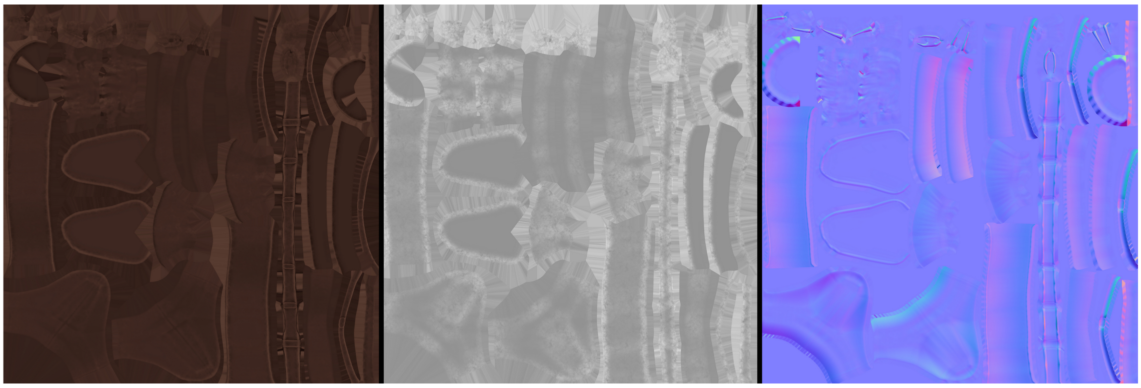


Figure 4.19: Head textures, Albedo(left), Specular(center) and Normal(right)

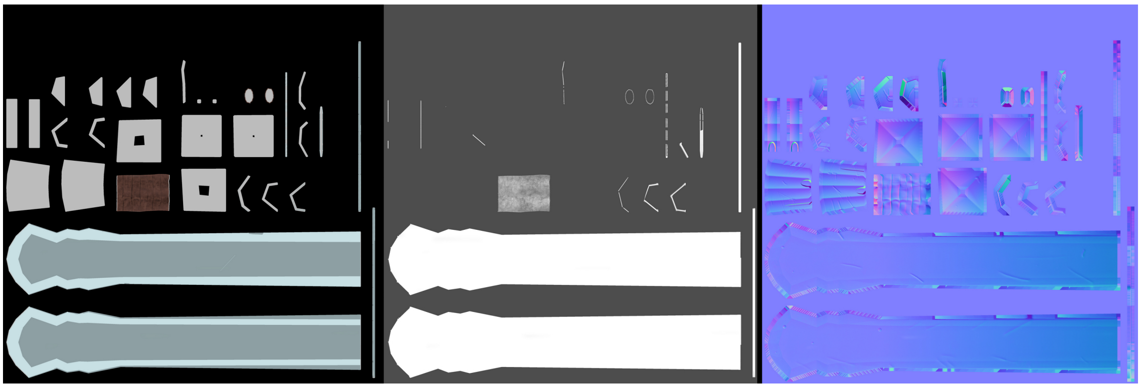


Figure 4.20: Head textures, Albedo(left), Metallic(center) and Normal(right)

Next you can see each part rendered using Blender with the Cycles render engine[4].

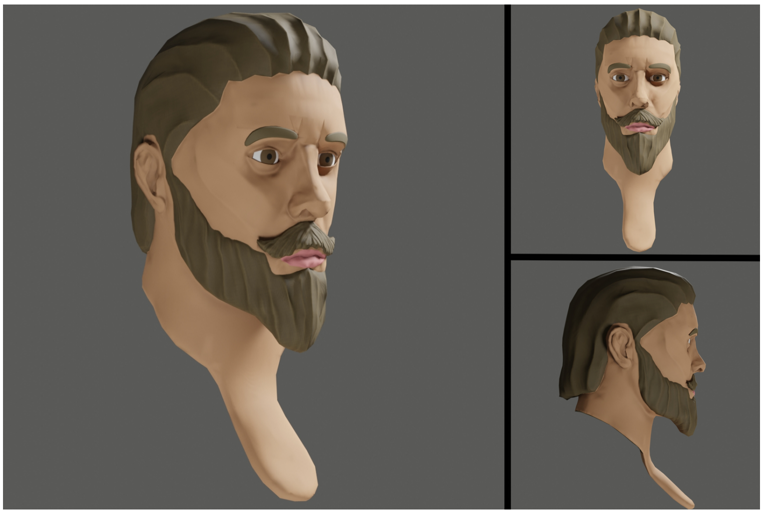


Figure 4.21: Head rendered

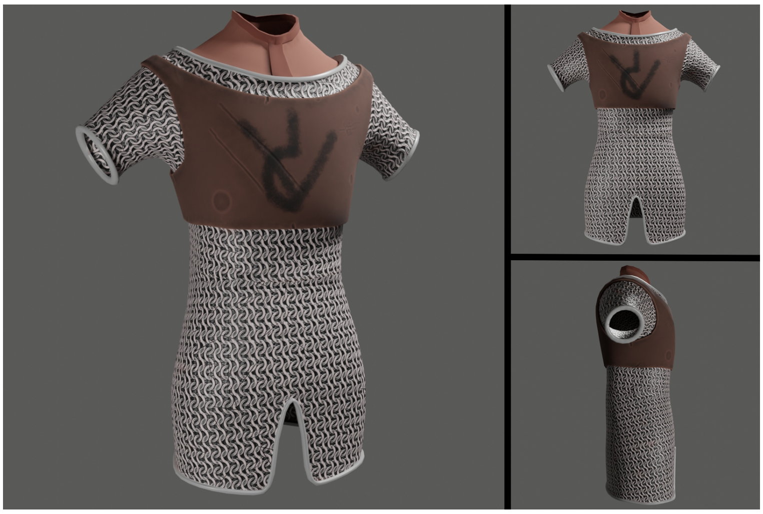


Figure 4.22: Armor rendered

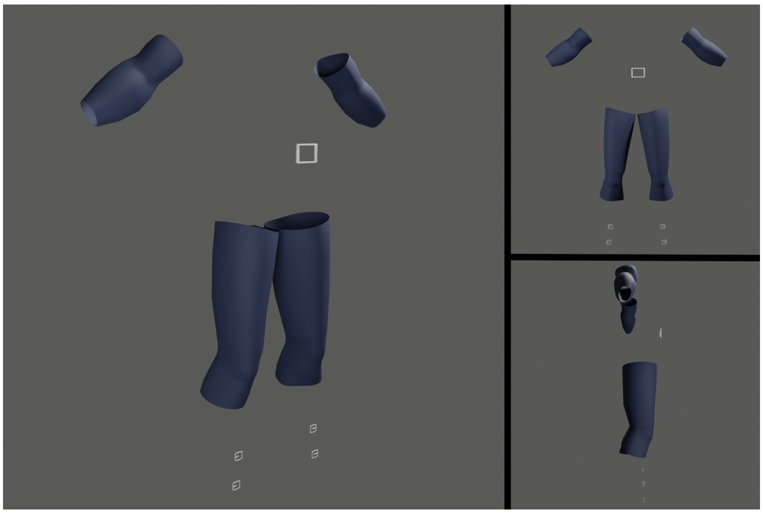


Figure 4.23: Legs and arms rendered

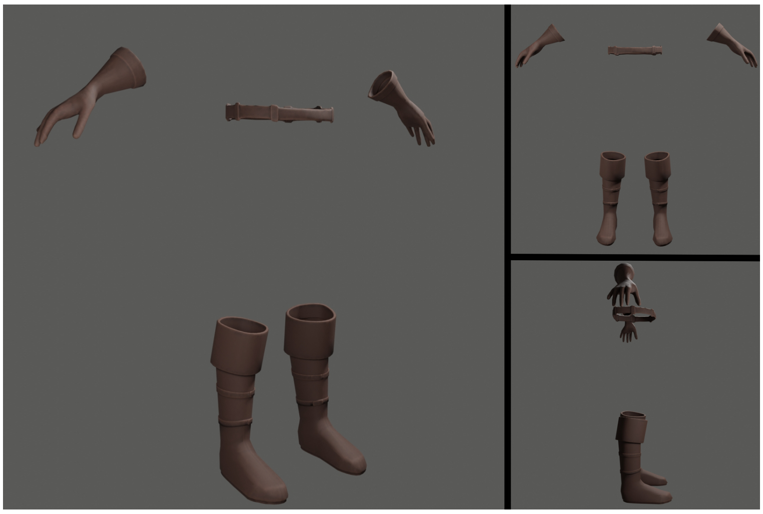


Figure 4.24: Boots and gloves rendered

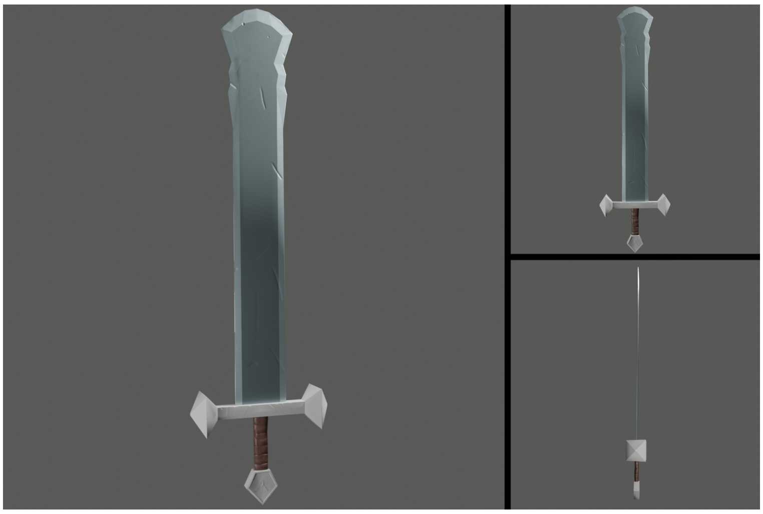


Figure 4.25: Sword rendered

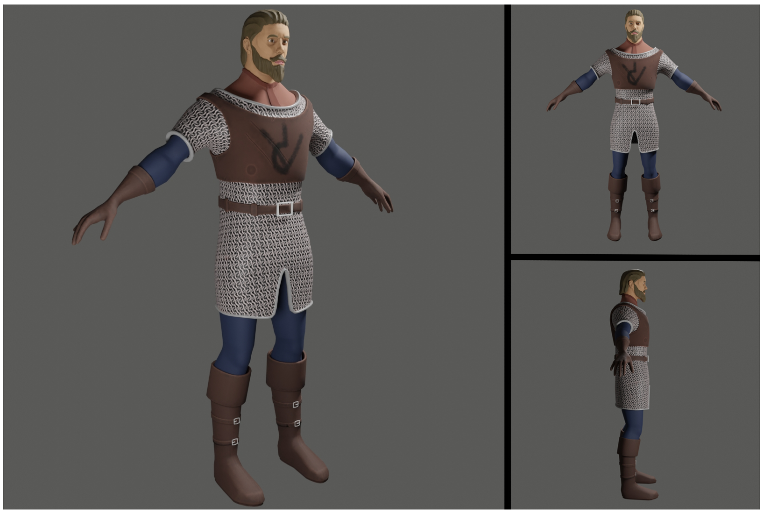


Figure 4.26: Character rendered

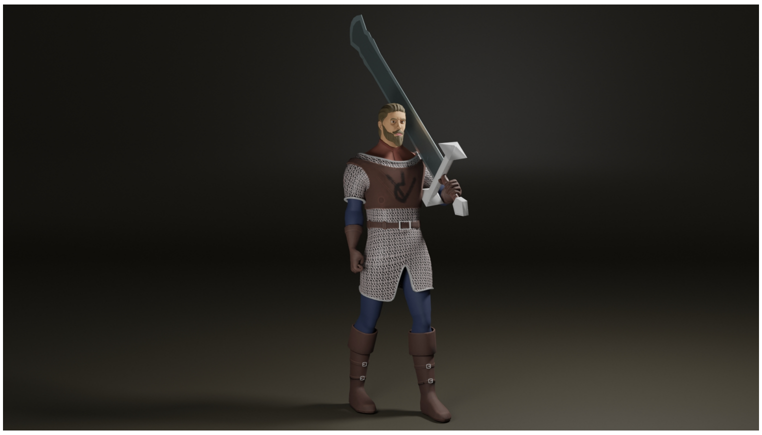


Figure 4.27: Beauty render

Rigging

The process of rigging consists on creating a skeletal structure on the model, that imitates the skeleton on the real life but simplified, allowing to move and animate the model changing the position, rotation or scale of each bone.The rigging process results in a hierarchical structure where each bone is in a parent/child relationship with the bones it connects to. This simplifies the animation process as a whole.

Blender was used for rigging, with the Rigify add-on[9], which allows you to create quickly the skeleton(see Figure 4.28, Figure 4.29), minimizing errors, and, once you put the skeleton in place, it creates a rig(see Figure 4.30, Figure 4.31), which already contains inverse kinematics for hands and foot.

In FK(Forward Kinematics), the animator specifies the position, rotation, and scale of each bone, and this will affect the childs, for example, if you move rotate a bone of the spine, it will affect the torso, head and arms. In IK(Inverse Kinematics), the animator moves a bone, called effector, and the computer will calculate the rotation, position or scale of the parents to create a realistic movement. This is used on extremities like the arms, legs or tails, for example, if the effector is on the hand, when the animator moves the hand, the computer will calculate the parameters for the bones of the arm and the forearm(or the number of bones that the animator specified that the hand effector should affect)

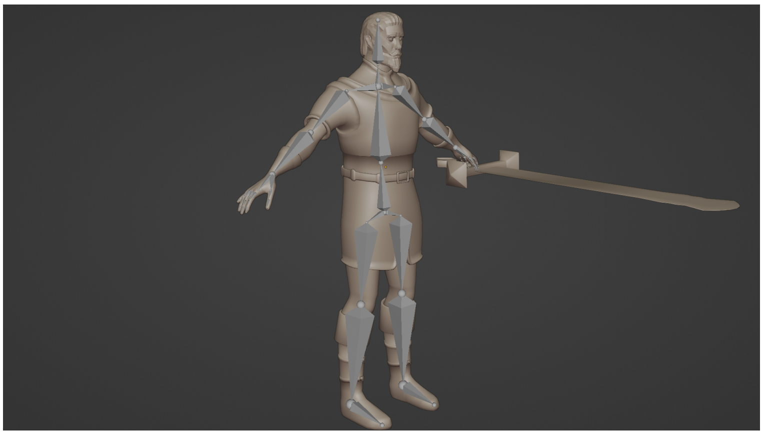


Figure 4.28: Skeleton view 1

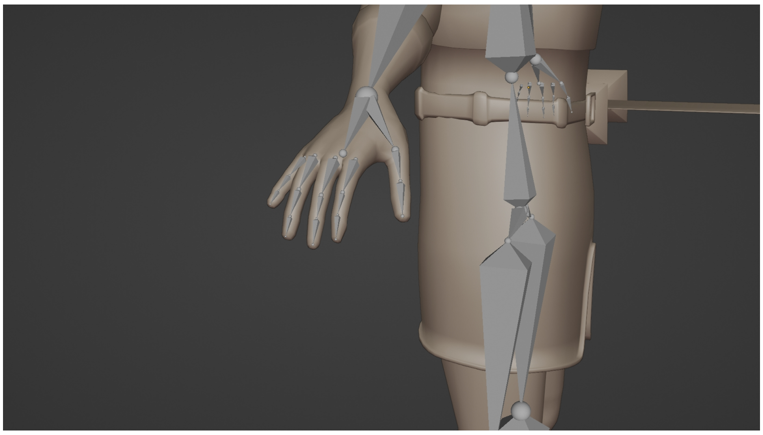


Figure 4.29: Skeleton view 2

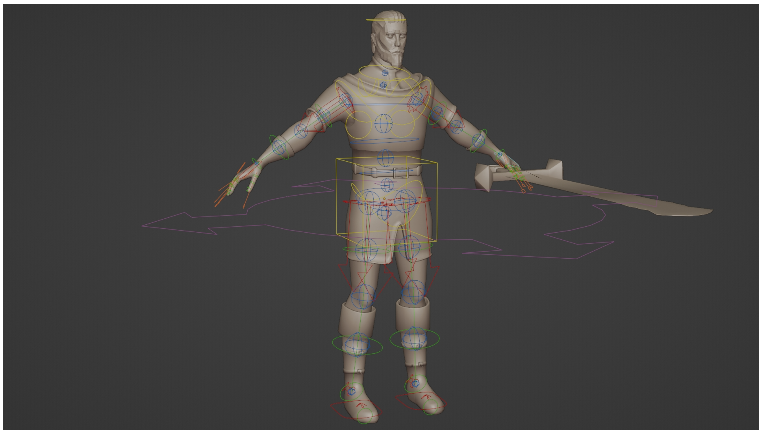


Figure 4.30: Rig view 1

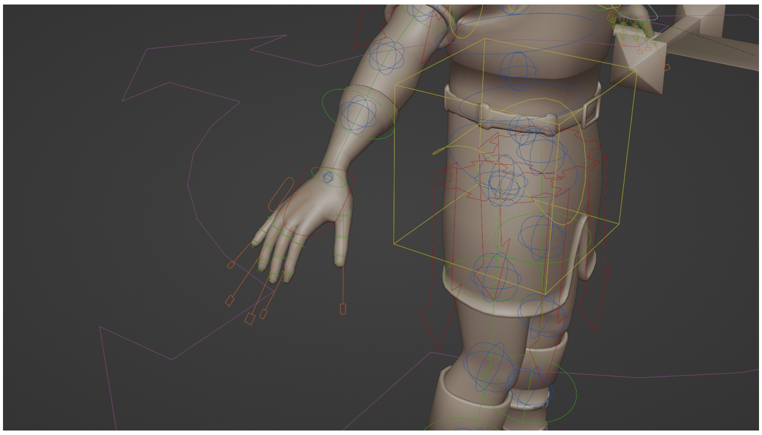


Figure 4.31: Rig view 2

Animation

Blender was also used for the animation, the animation is made by hand through key frames. This method of animation consists in changing the rotation or position of the selected bones for each key frame, and the rotation and position of the bones on the intermediate frames will be calculated by the computer using interpolation. Next can be seen the key frames for each animation.

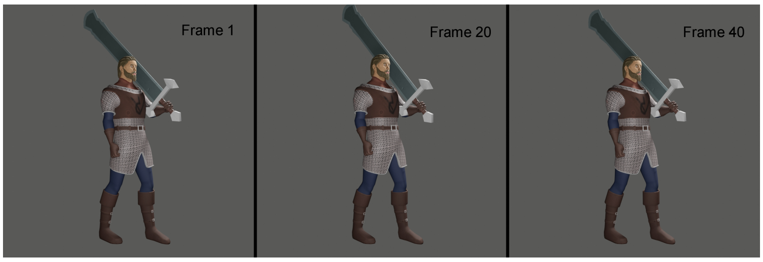


Figure 4.32: Idle animation

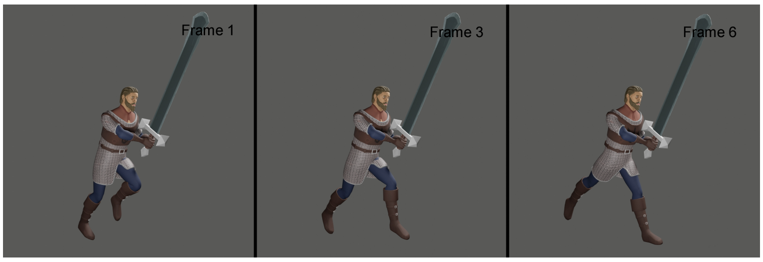


Figure 4.33: Run animation 1

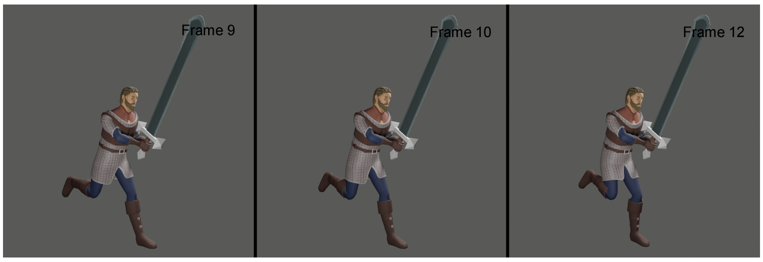


Figure 4.34: Run animation 2

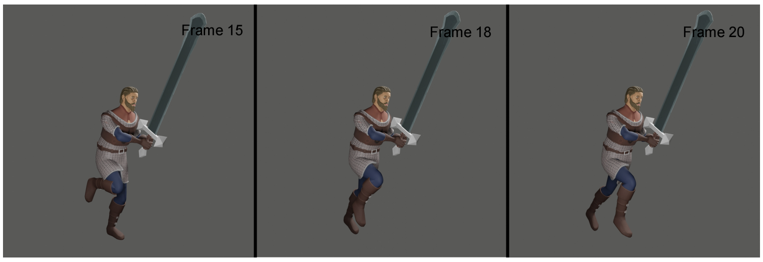


Figure 4.35: Run animation 3

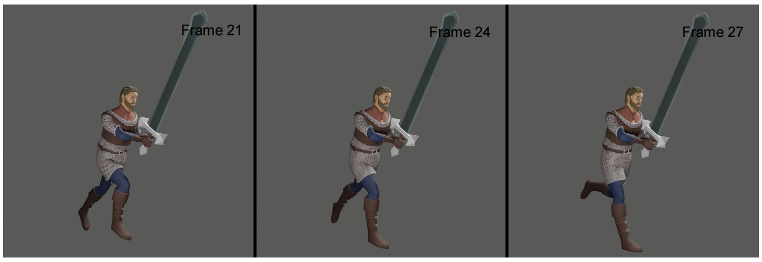


Figure 4.36: Run animation 4

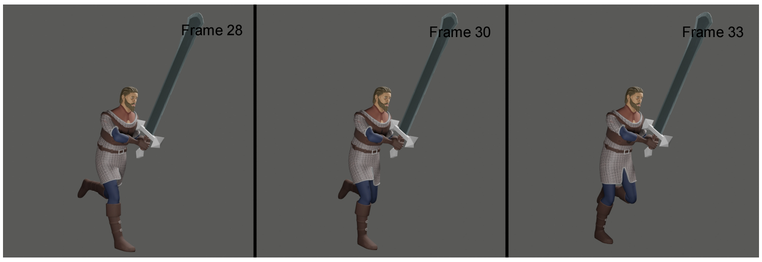


Figure 4.37: Run animation 5

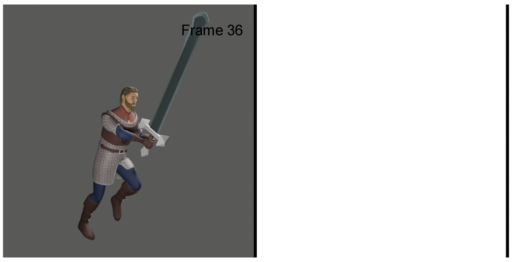


Figure 4.38: Run animation 6

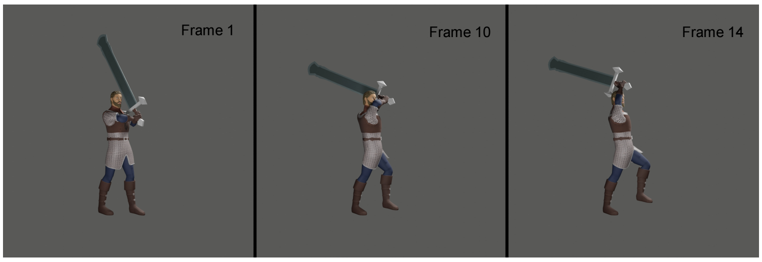


Figure 4.39: Attack animation 1

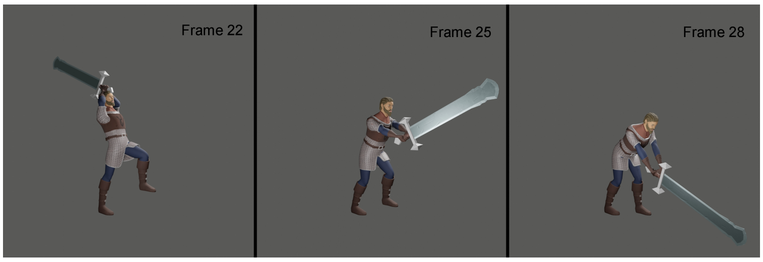


Figure 4.40: Attack animation 2

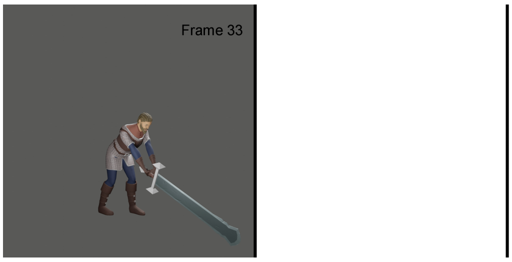


Figure 4.41: Attack animation 3

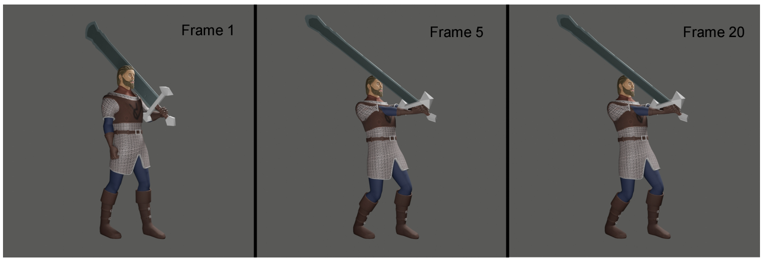


Figure 4.42: Block animation

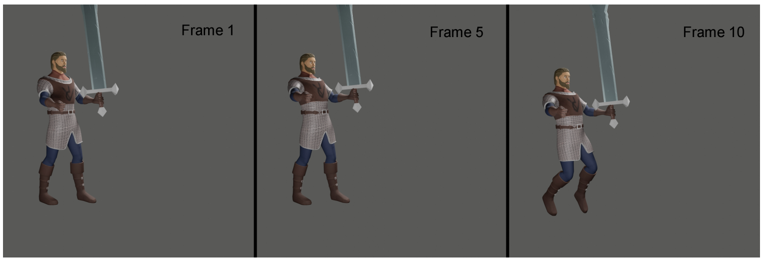


Figure 4.43: Death animation 1

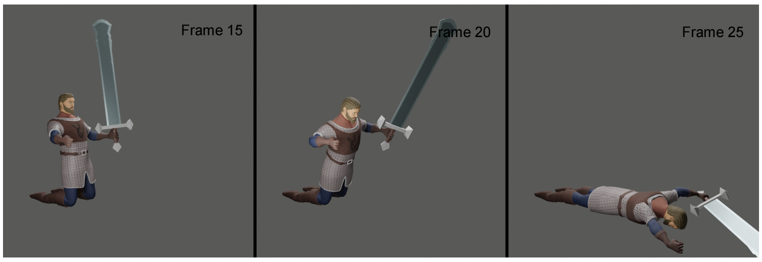


Figure 4.44: Death animation 2

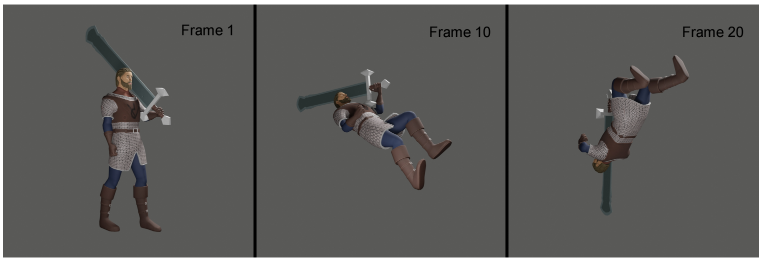


Figure 4.45: Evade animation 1

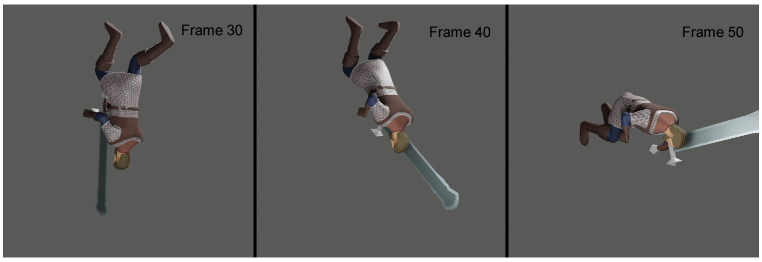


Figure 4.46: Evade animation 2

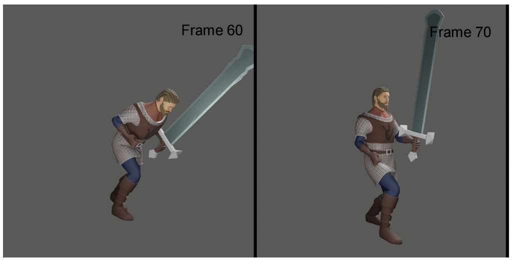


Figure 4.47: Evade animation 3

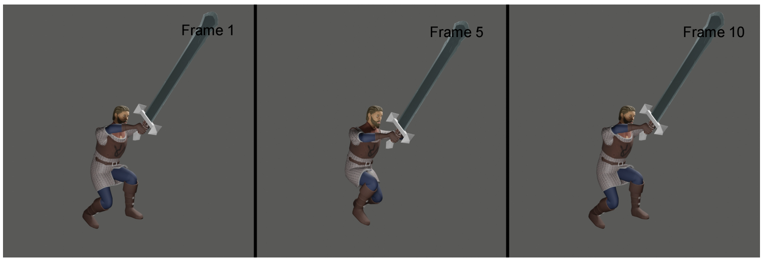


Figure 4.48: Jump Start Animation

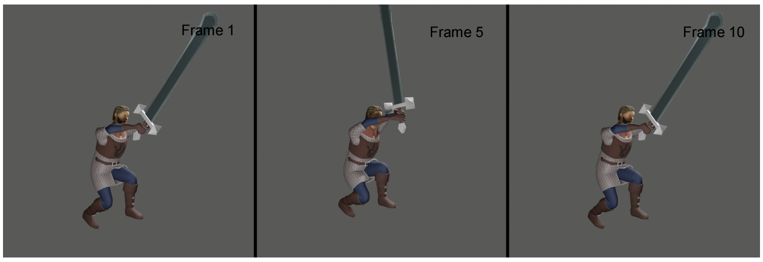


Figure 4.49: Jump Peak Animation

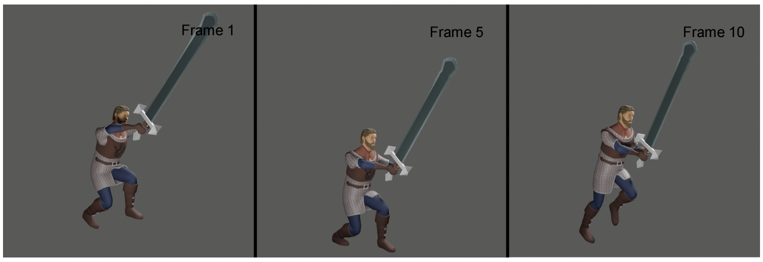


Figure 4.50: Jump Land Animation

Basic enemy

Is the enemy that will populate the level of the game, it's a dead warrior, revived by the necromancer, so it's designed very emaciated and with cracks on the armor(see Figure 4.51).

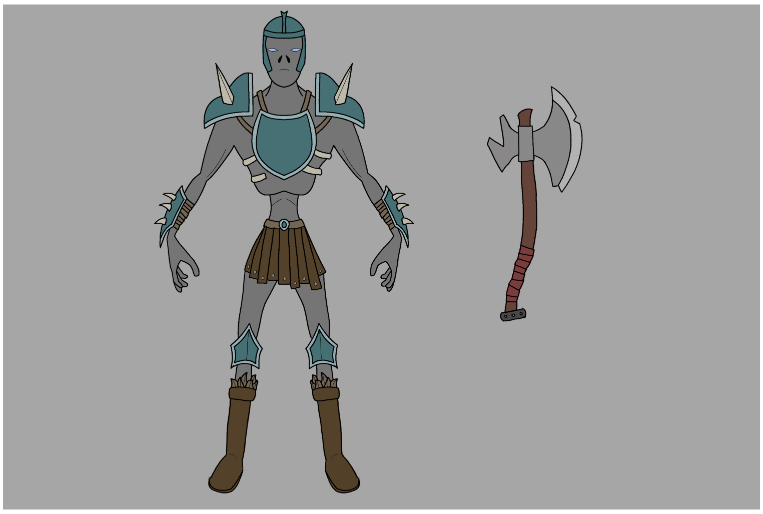


Figure 4.51: Enemy Design

Modeling

The model was divided in three parts, the armor(see Figure 4.52), the body(see Figure 4.53), and the axe(see Figure 4.54).As with the protagonist, Blender was used to model the high poly.

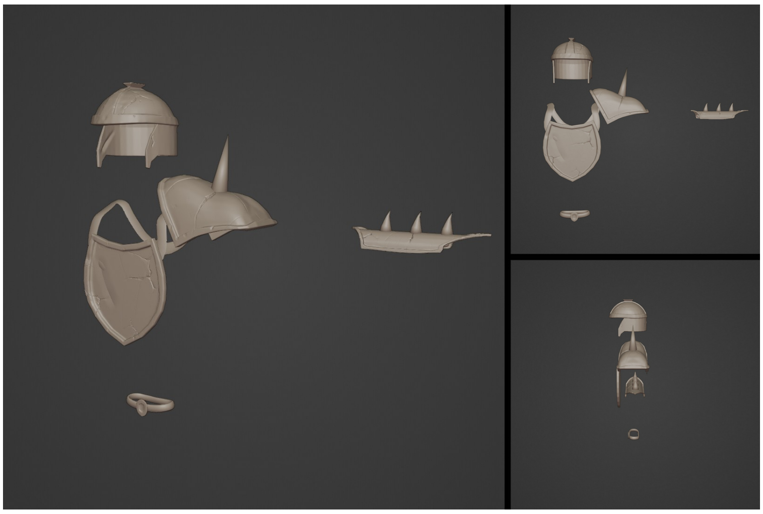


Figure 4.52: Armor high poly

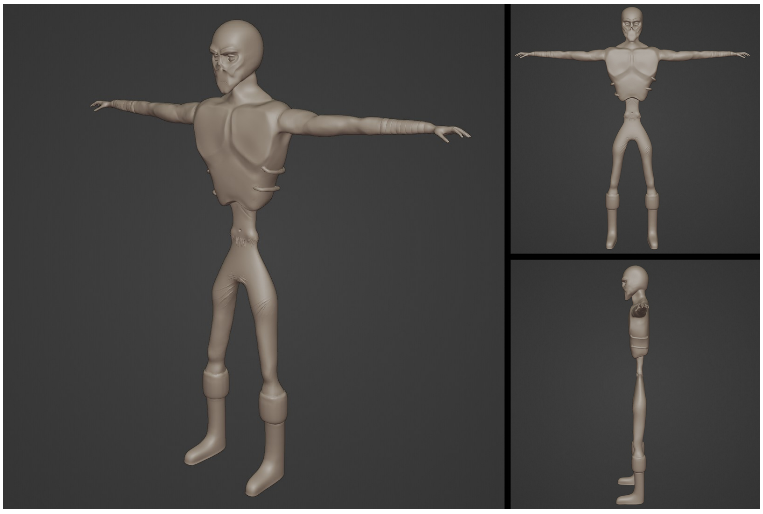


Figure 4.53: Body high poly

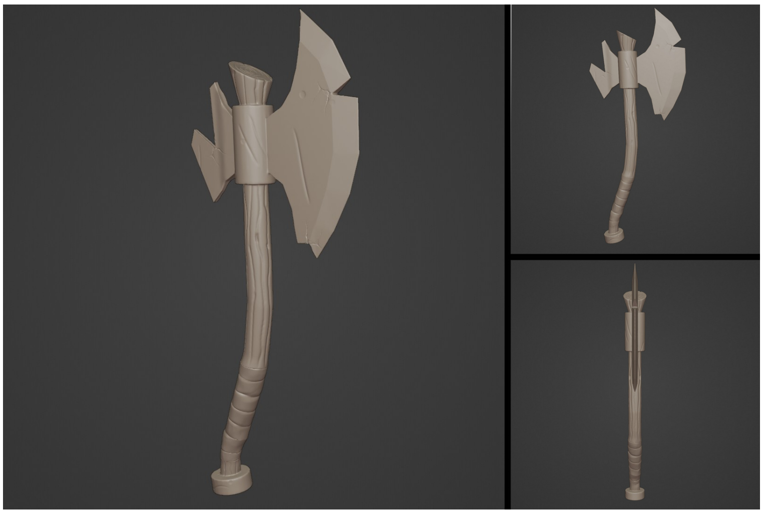


Figure 4.54: Axe high poly

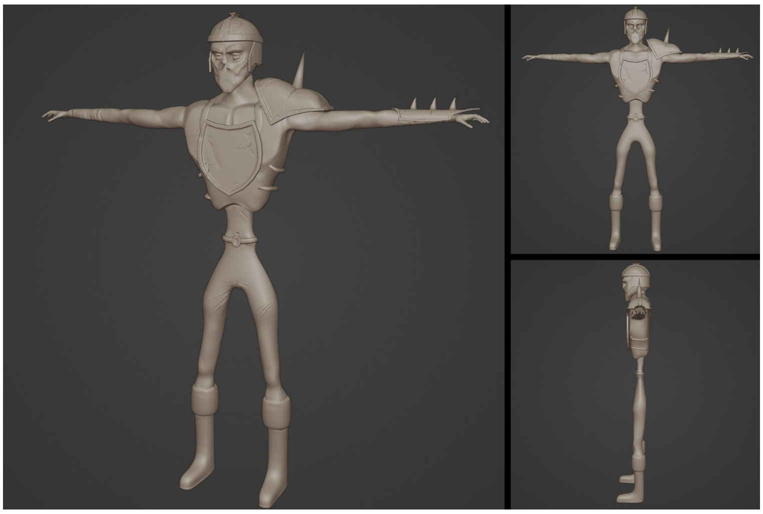


Figure 4.55: Enemy high poly

Retopology

As with the protagonist Blender was used for retopology with the retopoflow add-on, trying to balance quality and number of triangles.

The armor contains 8.046 triangles(see Figure 4.56).

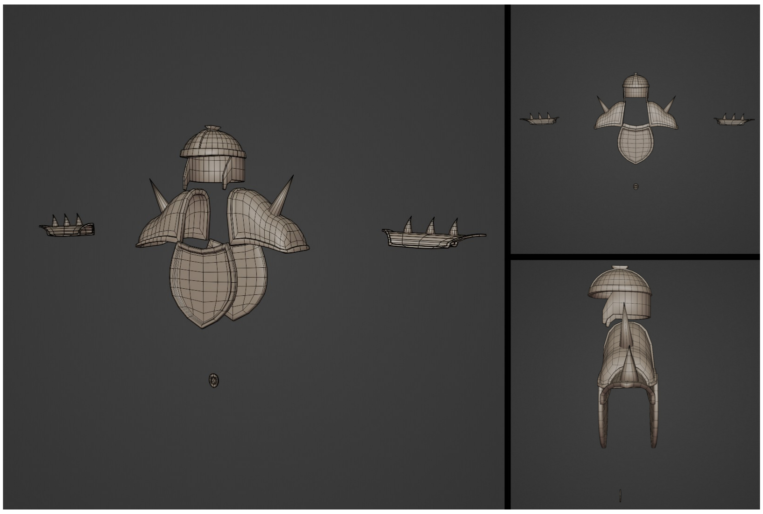


Figure 4.56: Armor Wireframe

The body contains 11.050 triangles(see Figure 4.57).

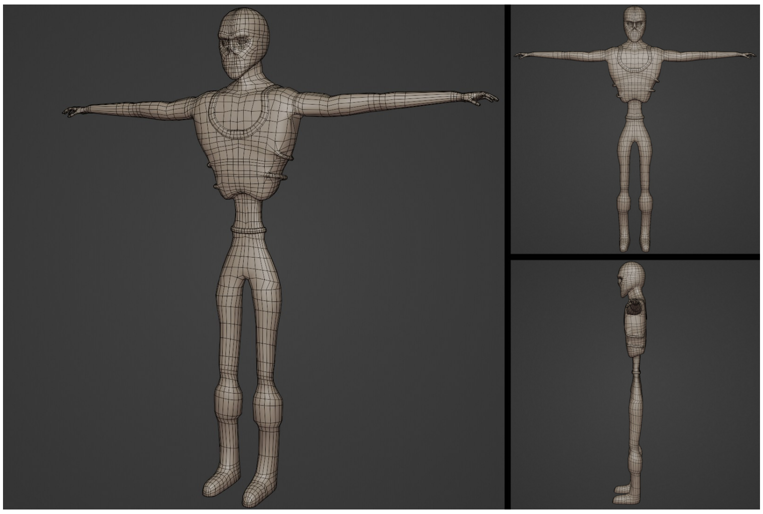


Figure 4.57: Body Wireframe

The axe contains 1.252 triangles(see Figure 4.58).

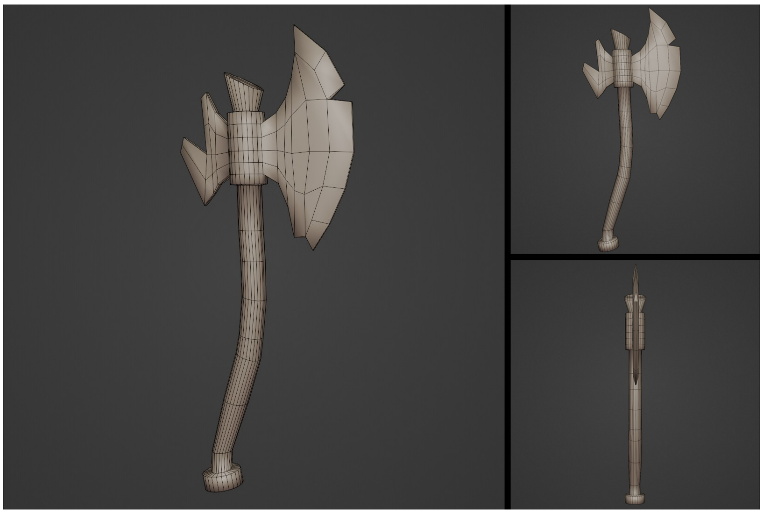


Figure 4.58: Axe Wireframe

The model in total contains 20.348 triangles, it’s a bit less than the character so it’s a great number.

UV Unwrapping

The UV were made in Blender(see Figure 4.59)

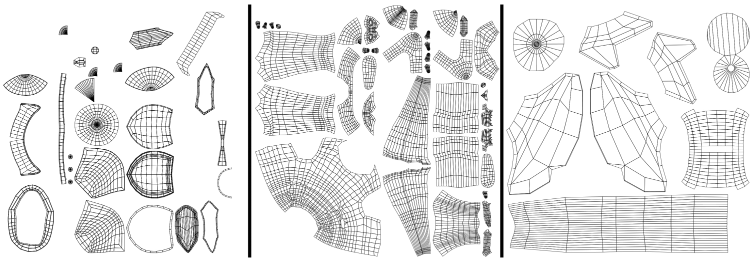


Figure 4.59: Basic enemy UV, armor(left), body(center), axe(left)

Textures

With the UVs done, the textures were made on Substance Painter, one material for each part, the armor(see Figure 4.60)), the body(see Figure 4.61), and the axe(see Figure 4.62). As with the protagonist, each material has three textures, the Albedo, the Specular/Metallic and the Normal.

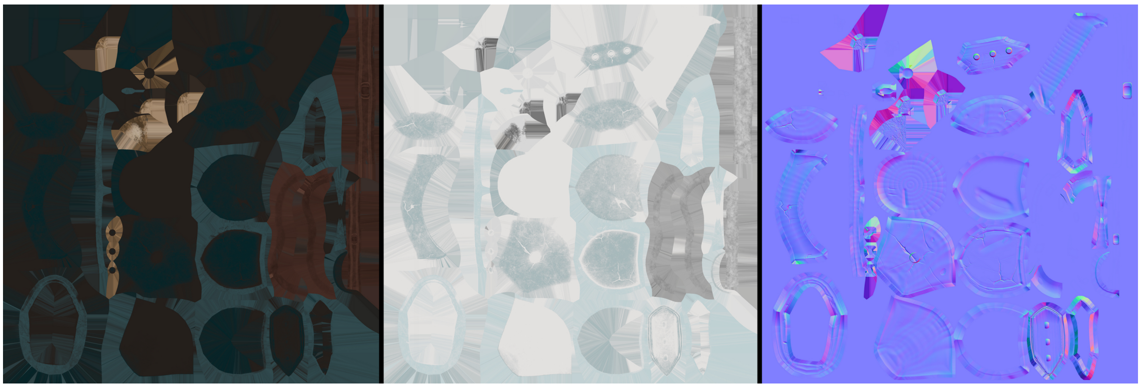


Figure 4.60: Armor texturess, Albedo(left), Specular(Center) and Normal(left)

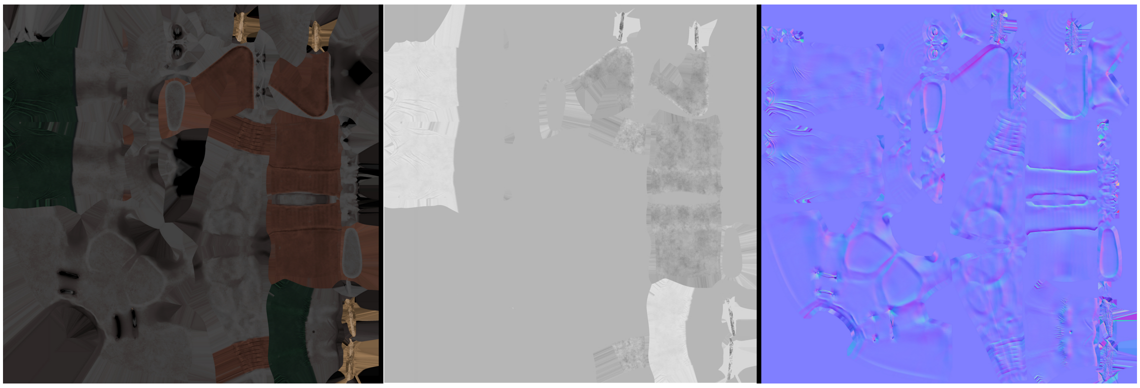


Figure 4.61: Body textures, Albedo(left), Specular(Center) and Normal(left)

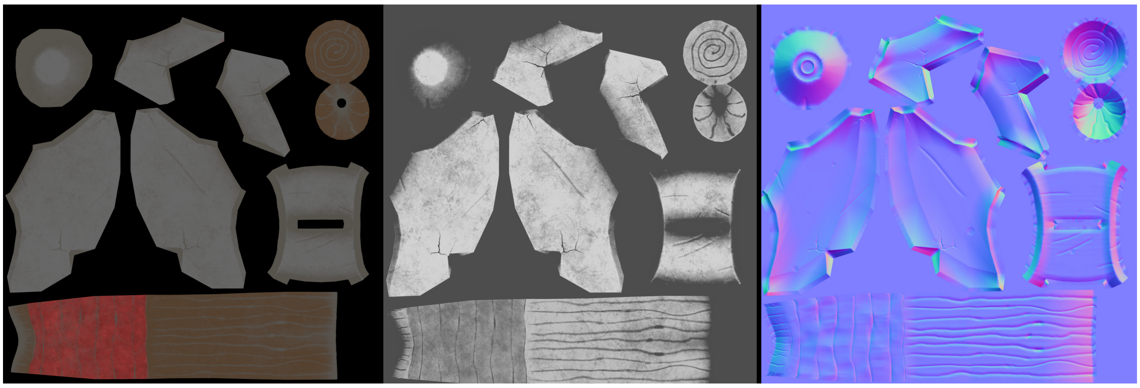


Figure 4.62: Axer tetures, Albedo(left), Metallic(Center) and Normal(left)

Below you can see the renders

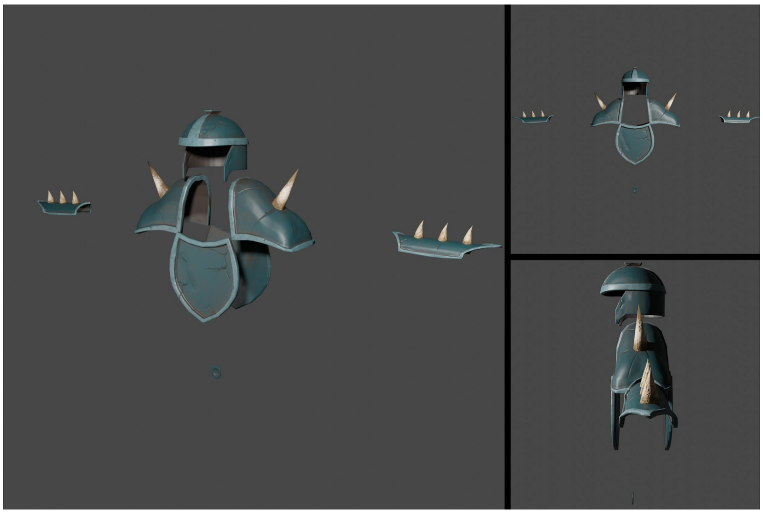


Figure 4.63: Armor render

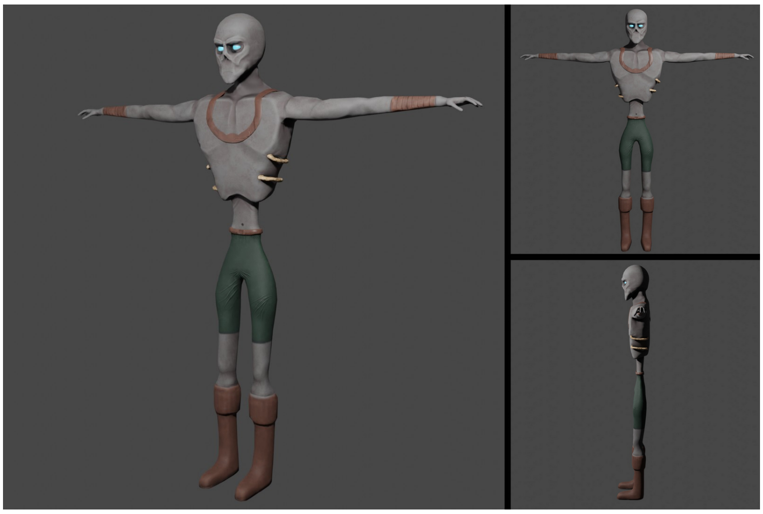


Figure 4.64: Body render

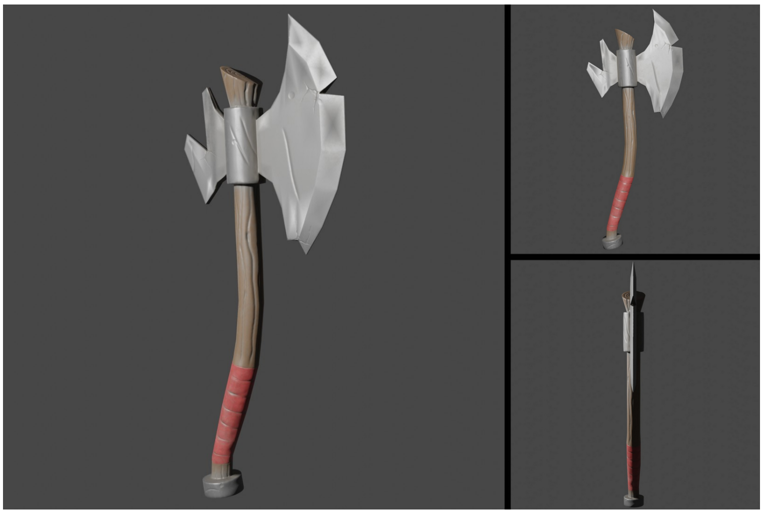


Figure 4.65: Axe render

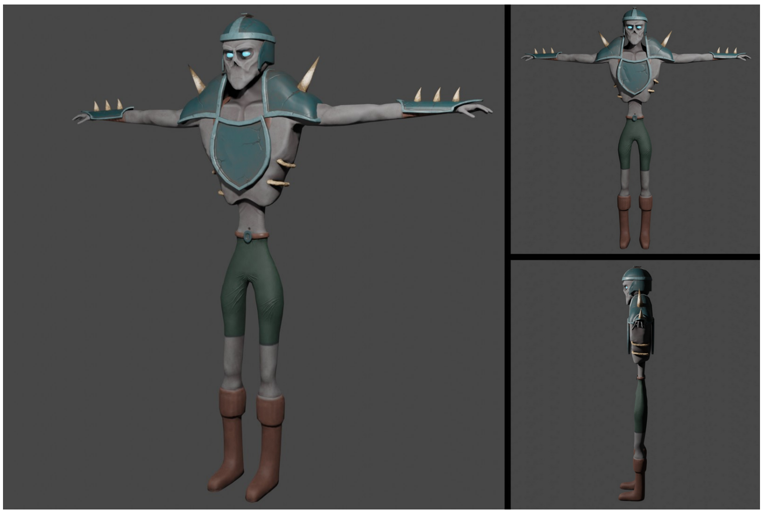


Figure 4.66: Basic enemy render

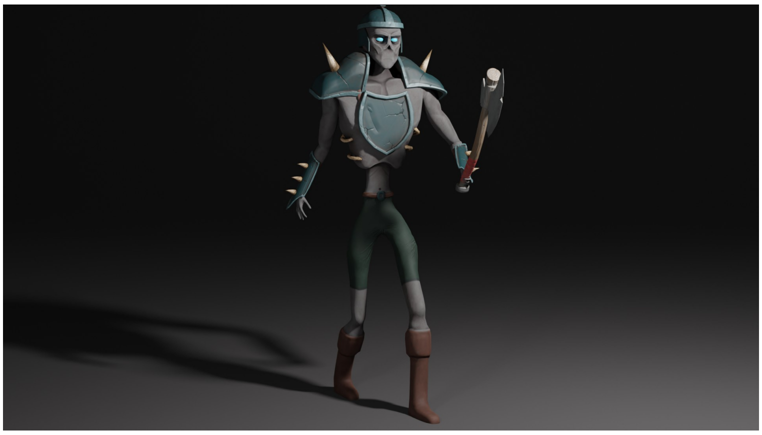


Figure 4.67: Basic enemy beauty shot

Rigging

For the basic enemy was used the same workflow as with the protagonist, Blender with the rigify add-on for the skeleton(see Figure 4.68, Figure 4.69) and rig(see Figure 4.70, Figure 4.71).

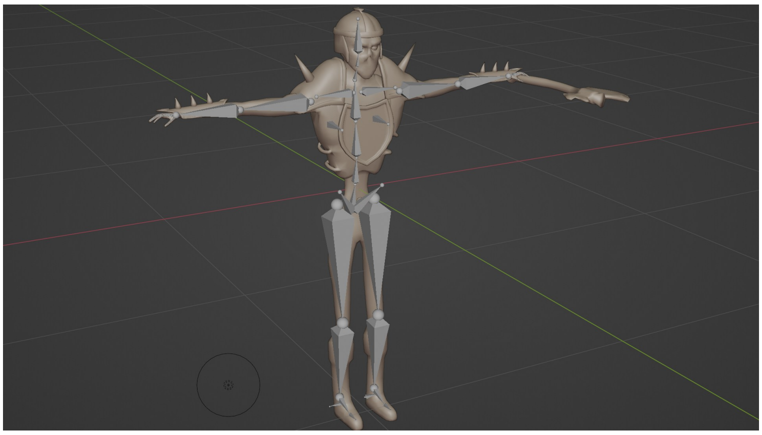


Figure 4.68: Basic enemy skeleton 1

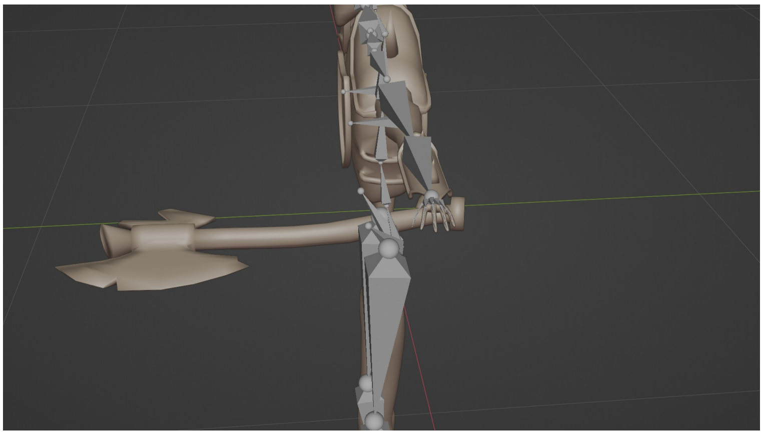


Figure 4.69: Basic enemy skeleton 2

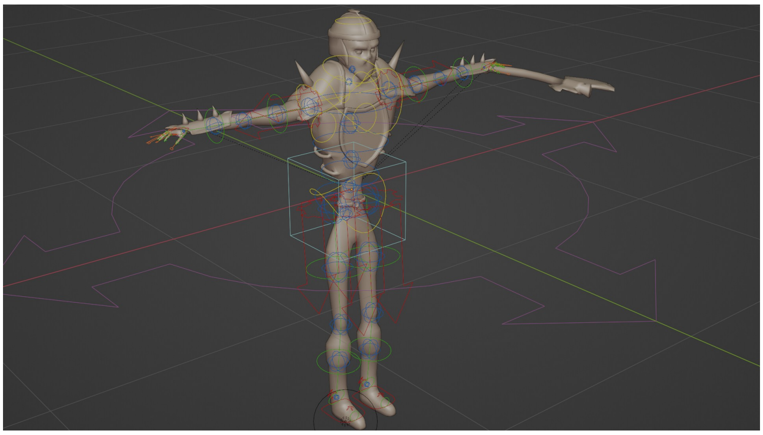


Figure 4.70: Basic enemy rig 1

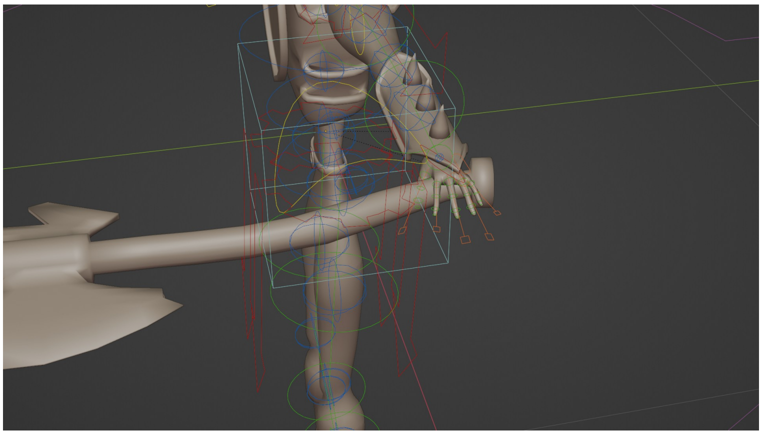


Figure 4.71: Basic enemy rig 2

Animation

For the animation was also used Blender, the animation is made by hand through key frames. Next can be seen the key frames for each animation.

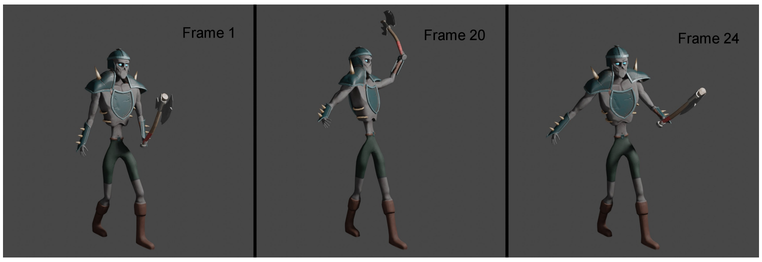


Figure 4.72: Basic enemy attack 1

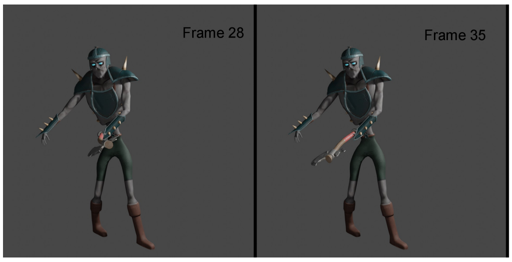


Figure 4.73: Basic enemy attack 2

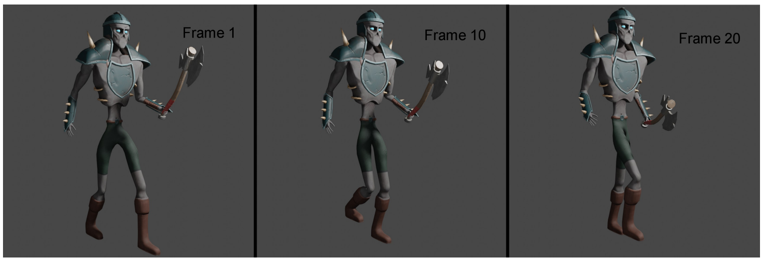


Figure 4.74: Basic enemy walk 1

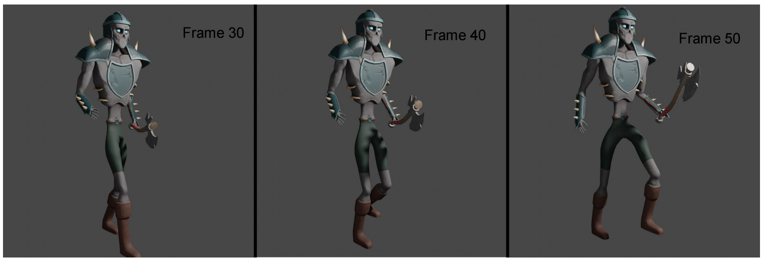


Figure 4.75: Basic enemy walk 2

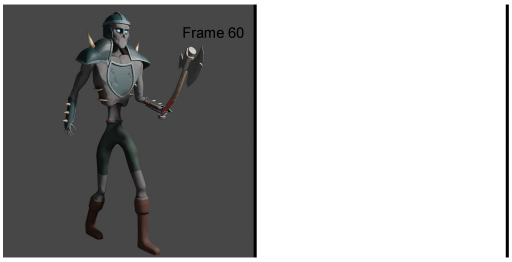


Figure 4.76: Basic enemy walk 3

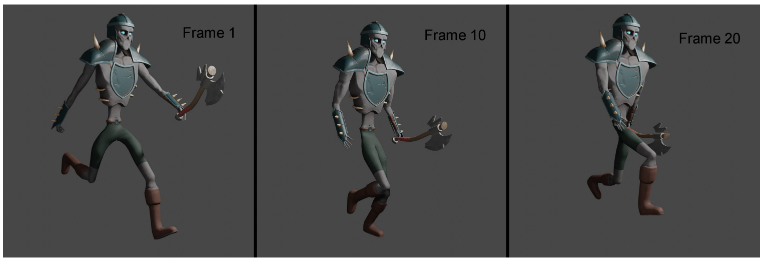


Figure 4.77: Basic enemy run 1

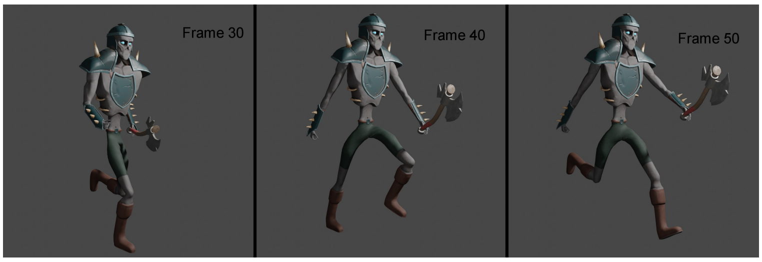


Figure 4.78: Basic enemy run 2

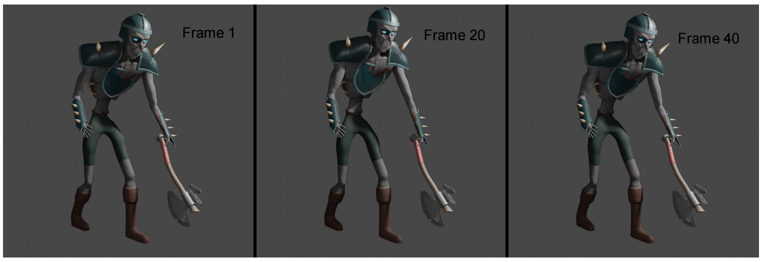


Figure 4.79: Basic enemy weakened

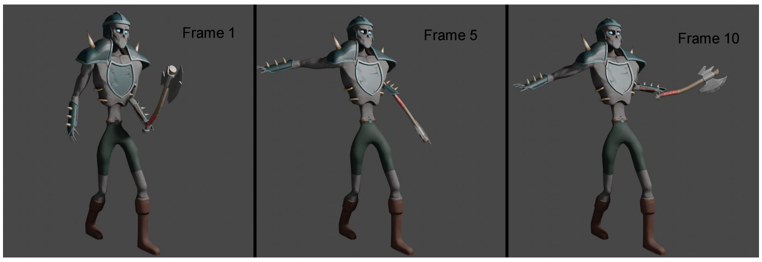


Figure 4.80: Basic enemy hurt

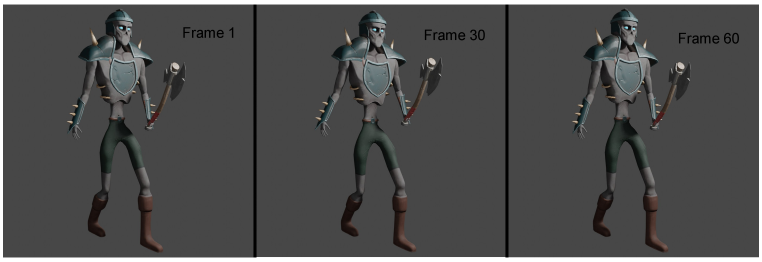


Figure 4.81: Basic enemy idle

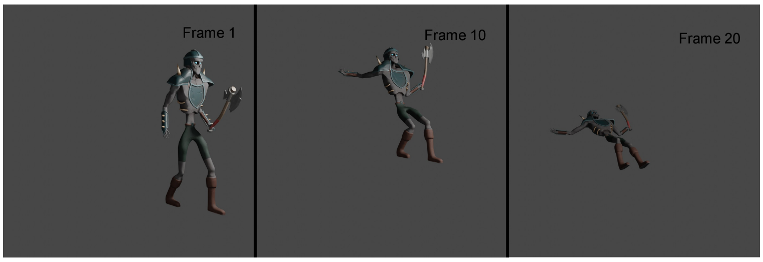


Figure 4.82: Basic enemy death

Final Boss

It's the necromancer that terrifies the village and it's the level boss, he's designed big and strong, his body type is based on the body of some athletes from Strongman competitions, and gave him a more tribal look with tattoos(see Figure 4.83).

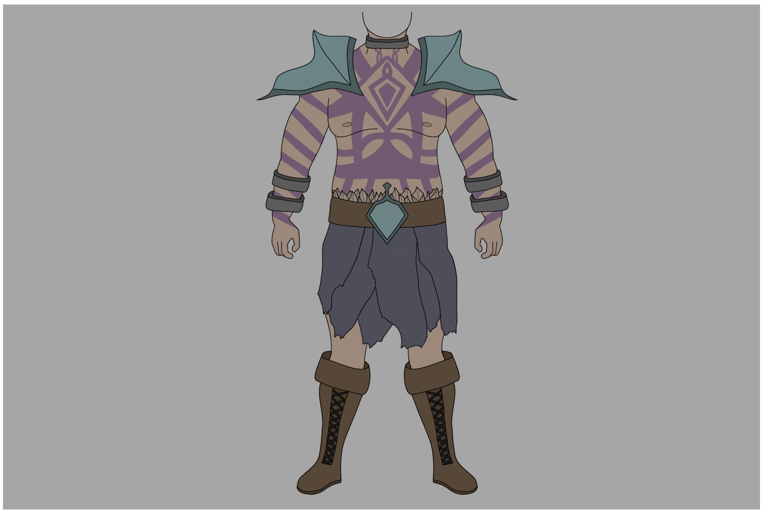


Figure 4.83: Level Boss design

Modeling

As with the others characters the high poly was made on Blender, the model was divided in two parts, the body(see Figure 4.84) and the armor(see Figure 4.85).

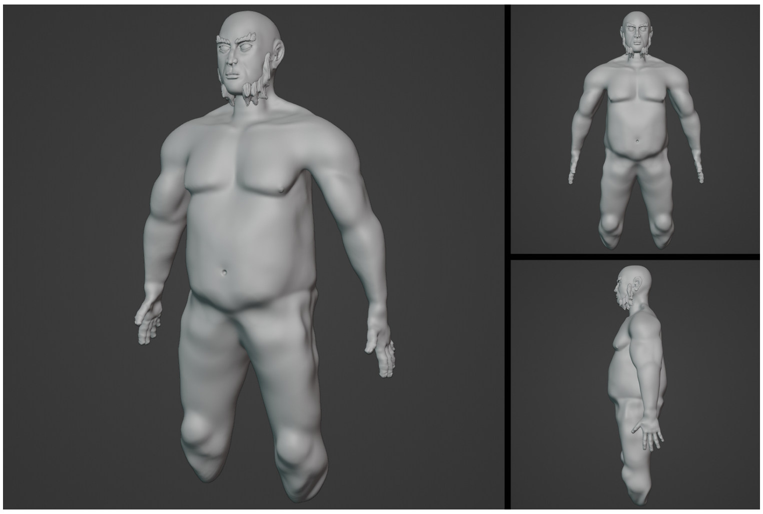


Figure 4.84: Level Boss body high poly

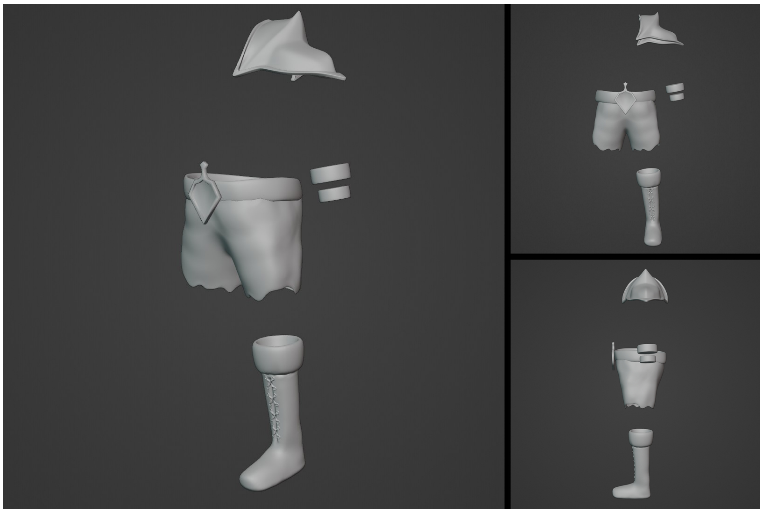


Figure 4.85: Level Boss armor high poly

Retopology

As with the other characters for the retopology was used Blender with the retopoflow add-on, trying to balance quality and number of triangles.

The body contains 21.766 triangles.

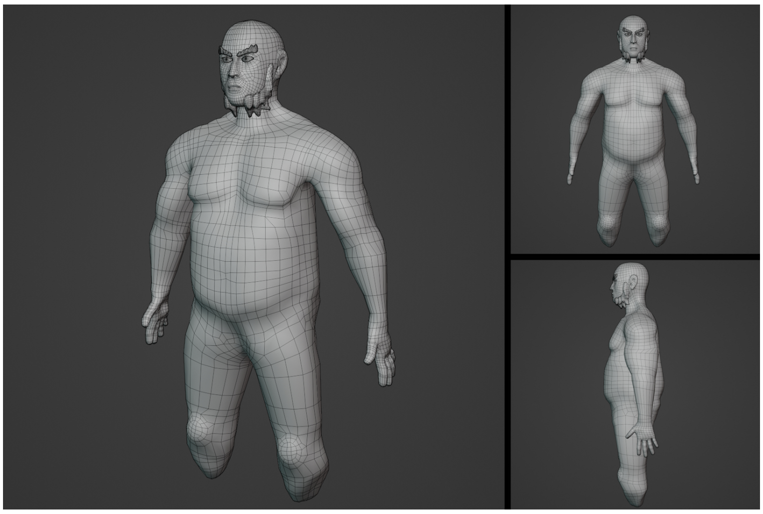


Figure 4.86: Level Boss body wireframe

The armor contains 18.060 triangles.

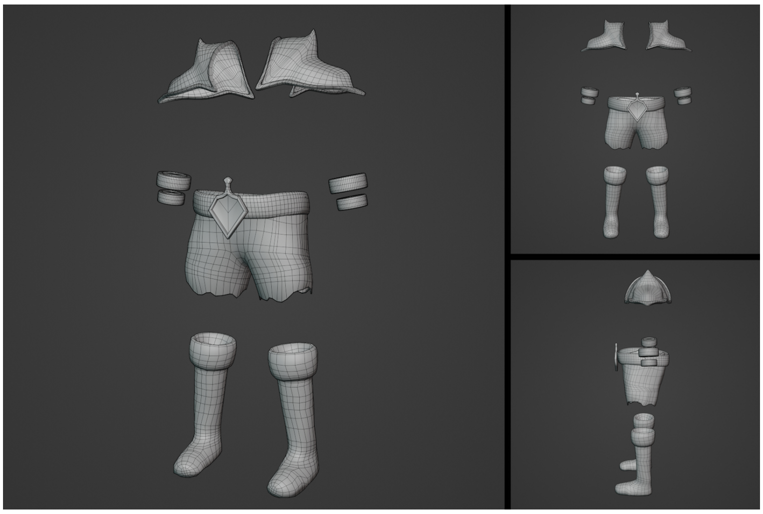


Figure 4.87: Level Boss armor wireframe

In total the model 39.826 triangles, is a little more than what it should be, but as when the boss appears, it will only be him and the protagonist, it is nothing to worry about.

UV Unwrapping

Next, following the same workflow as with the other characters, the UV Unwrapping was made.

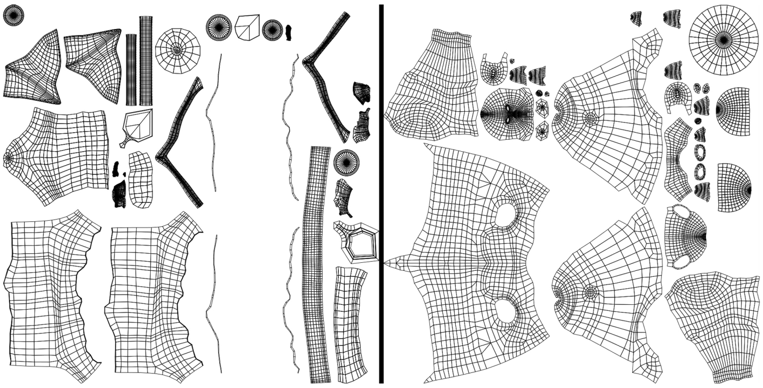


Figure 4.88: Level Boss UV, armor(left), body(right)

Textures

With the UVs done, the textures were made on Substance Painter, one material for each part, the body(see Figure 4.89) and the armor(see Figure 4.90).

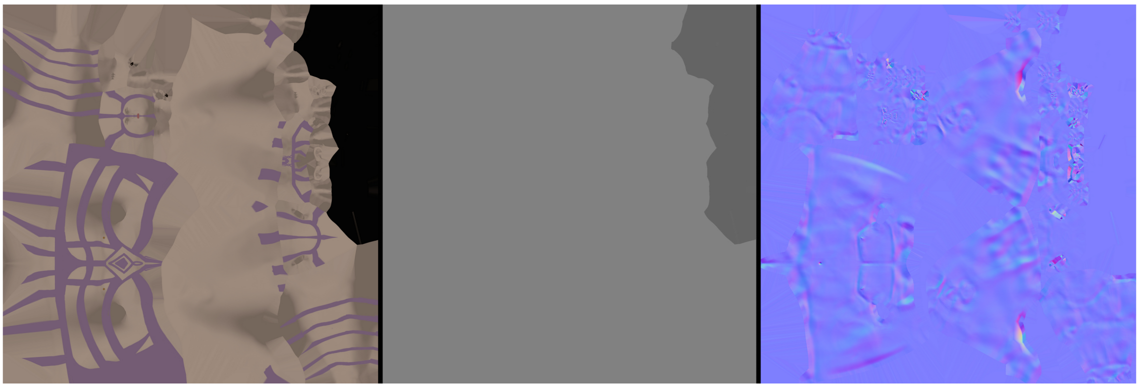


Figure 4.89: Level Boss body textures, albedo(left), specular(center), normal(right)

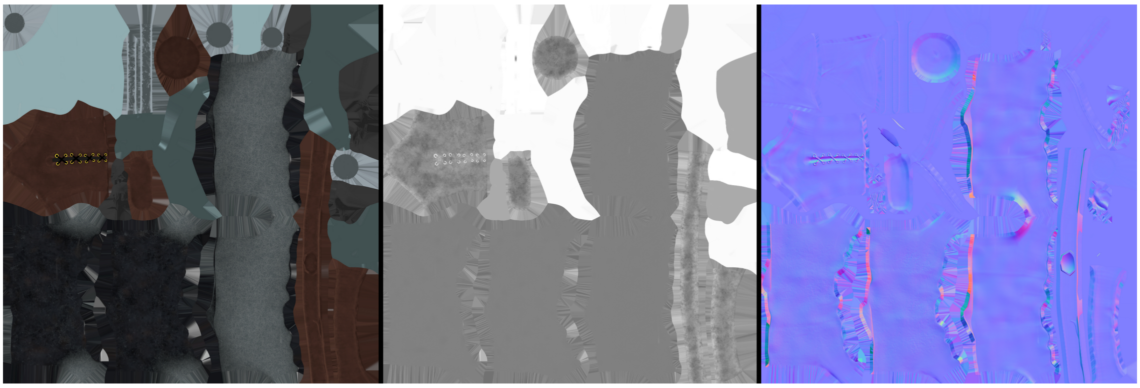


Figure 4.90: Level Boss armor textures, albedo(left), metallic(center), normal(right)

Below you can see the renders

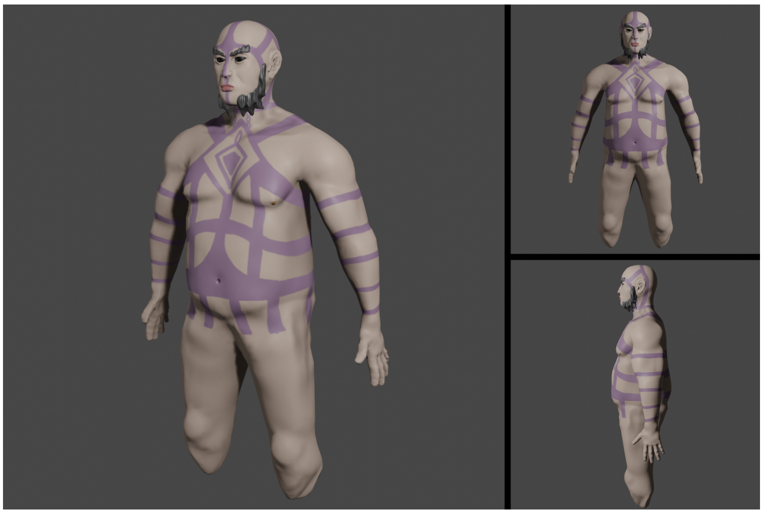


Figure 4.91: Level Boss body render

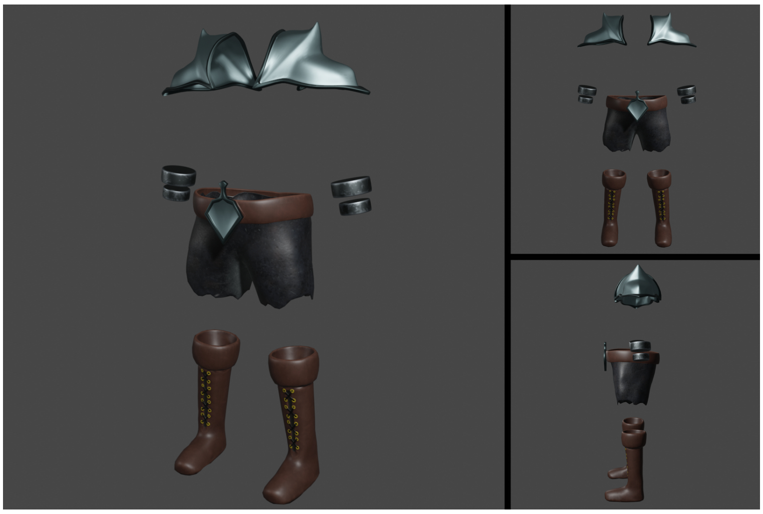


Figure 4.92: Level Boss armor render

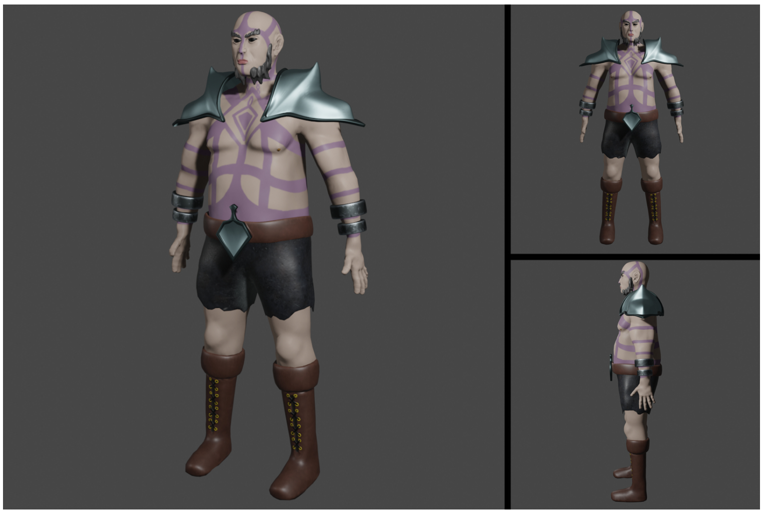


Figure 4.93: Level Boss render

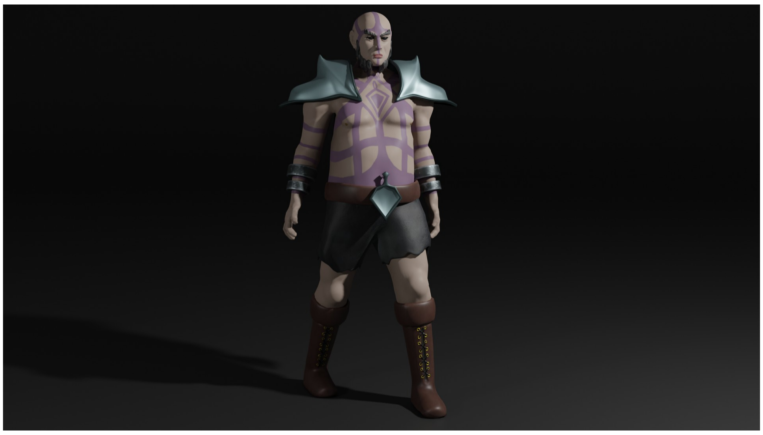


Figure 4.94: Level Boss body beauty render

Rigging

Here is used the same workflow as with the other characters, Blender with rigify add-on for the skeleton(see Figure 4.95, Figure 4.96) and rig(see Figure 4.97, Figure 4.98)

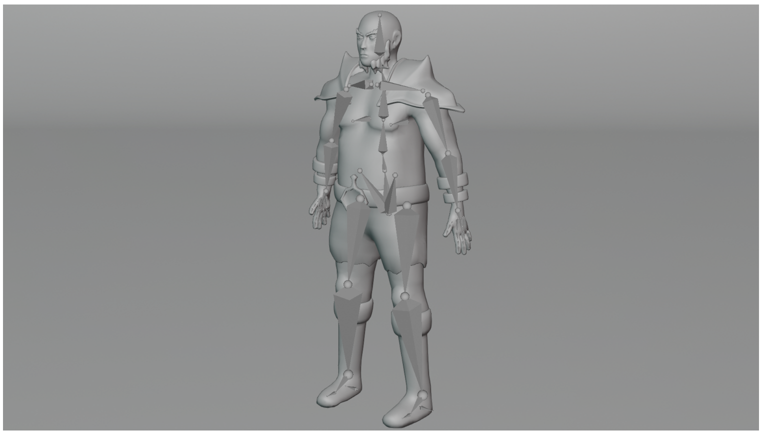


Figure 4.95: Level Boss skeleton

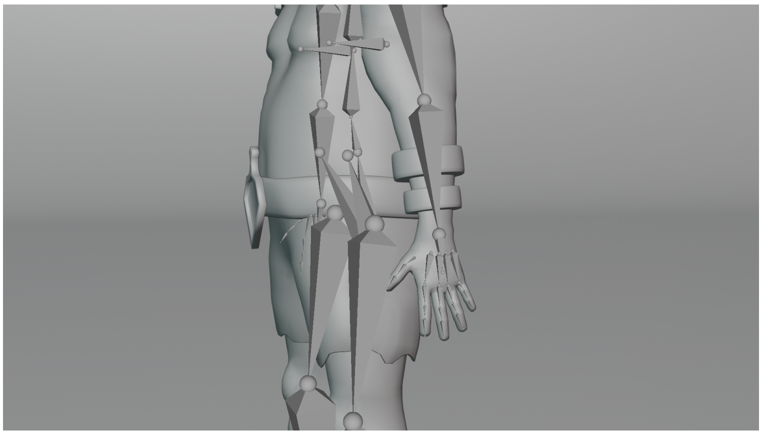


Figure 4.96: Level Boss skeleton

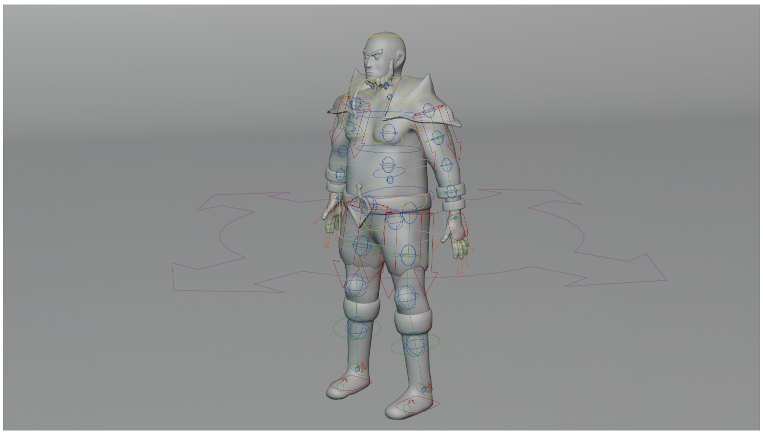


Figure 4.97: Level Boss rig

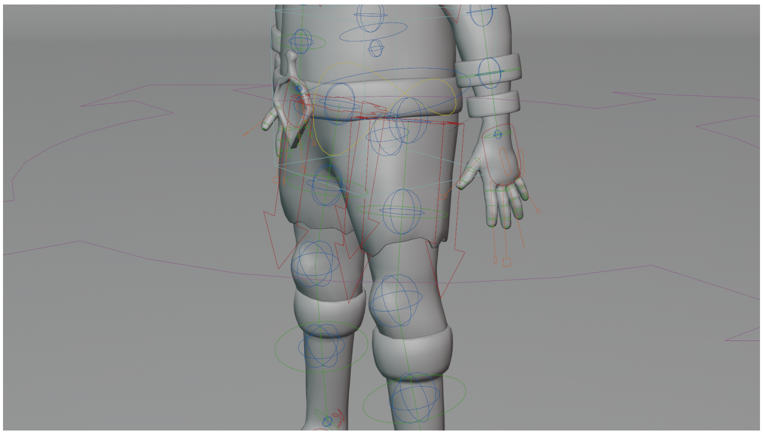


Figure 4.98: Level Boss rig

Animation

For the animation was also used Blender, the animation is made by hand through key frames. Next can be seen the key frames for each animation.

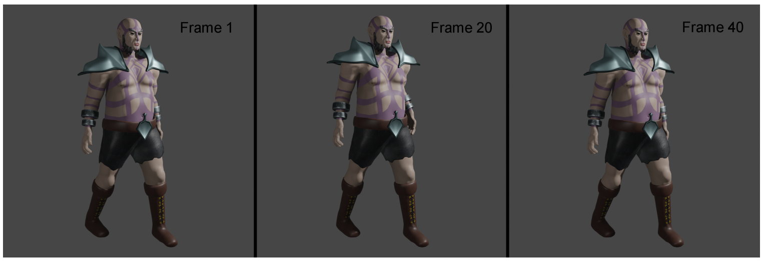


Figure 4.99: Level Boss Idle animation

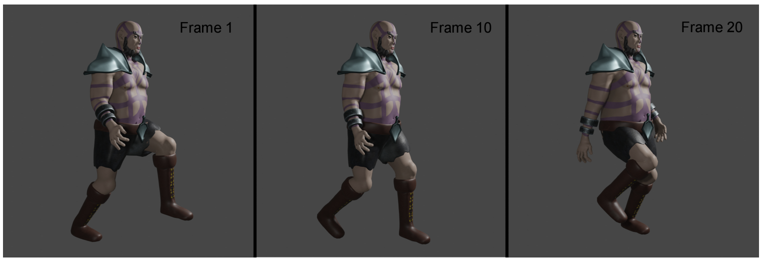


Figure 4.100: Level Boss walk animation 1

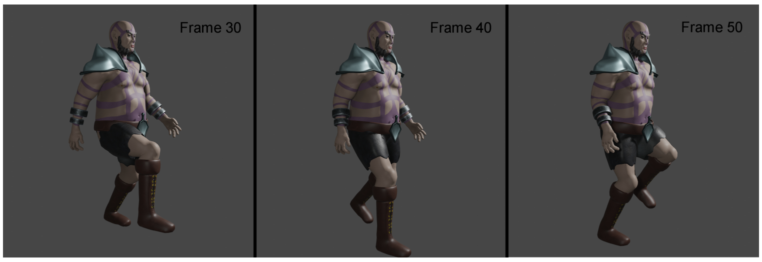


Figure 4.101: Level Boss walk animation 2

Figure 4.102: Level Boss walk animation 3

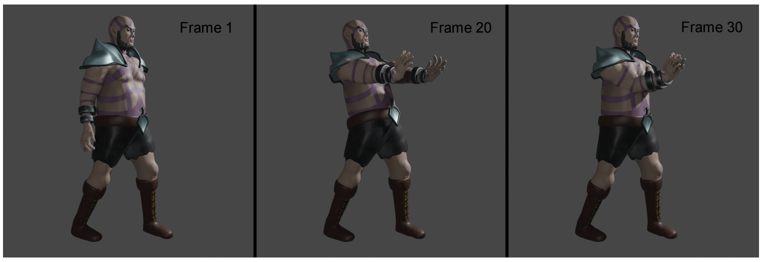


Figure 4.103: Level Boss spell attack animation

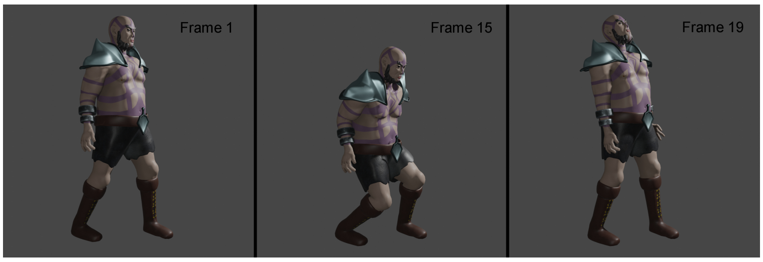


Figure 4.104: Level Boss jump attack animation

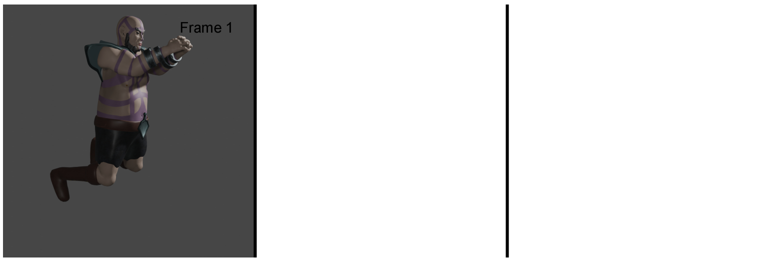


Figure 4.105: Level Boss jump attack falling animation

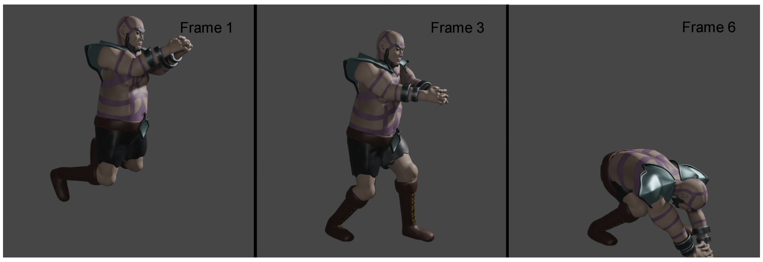


Figure 4.106: Level Boss jump attack land animation

4.1.2 Environments

The game contains two environments, a village, where you start and where you get the main mission, and a castle, where the mission happens, here is the workflow for each one.

Castle

First, you can see the skecth(see Figure 4.107), it's designed to be brief but entertaining, also, being a metroidvania videogame there must be backtracking, that means paths that you can't go through until you get some ability later in game, so it was designed in a way that you can't get to the boss until you don't find the double jump power up.

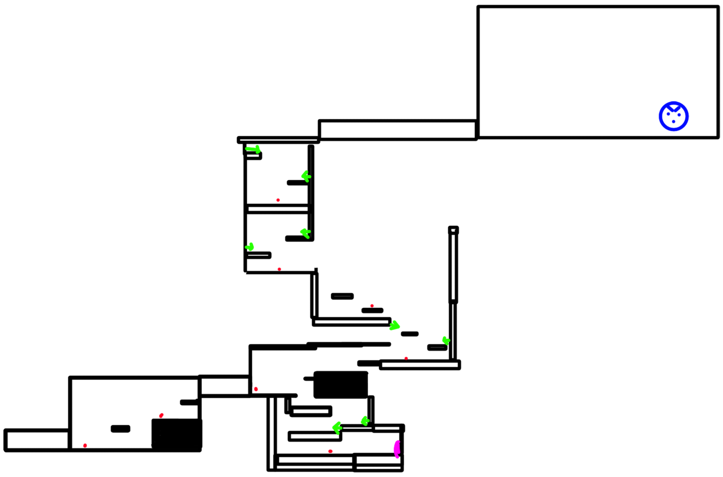


Figure 4.107: Skecth of the map

To make the map modular, only one asset was created, with rectangular shape(see Figure 4.108), this asset was repeated to create the map, the same material was used for the background, but more obscured, to differentiate it.

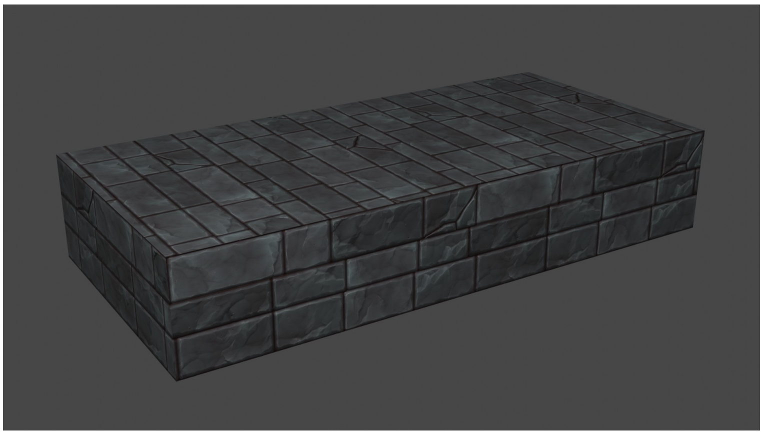


Figure 4.108: Modular asset

Following the design the map was created(see Figure 4.109).

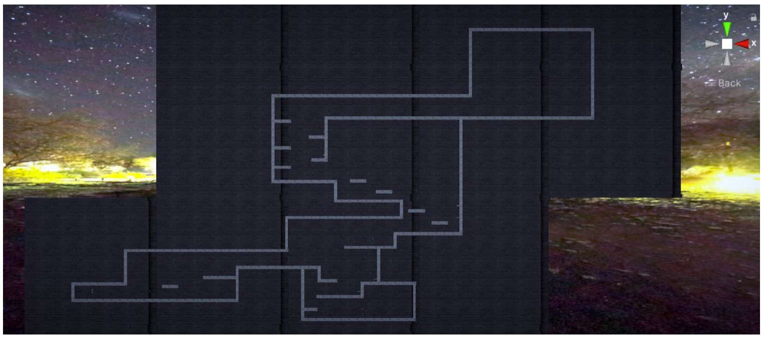


Figure 4.109: Castle map in Unity

Village

The village has quite a lot of parts so to keep on the limit of pages, here only be showed some of the most important assets.

The same workflow was used for the houses and the stairs,first box modelling the low poly, and then sculpting on a copy of the low poly but with the remesh sculpt tool of blender to create the high poly, which add more polygons but keeping the shape, so there's no need for the retopology, because there's already a copy of the low poly .Next you can see the wireframe(see Figure 4.110, Figure 4.111, Figure 4.112), high poly(see Figure 4.113, Figure 4.114, Figure 4.115) and renders(see Figure 4.116, Figure 4.117, Figure 4.118) of the two houses and the stairs.

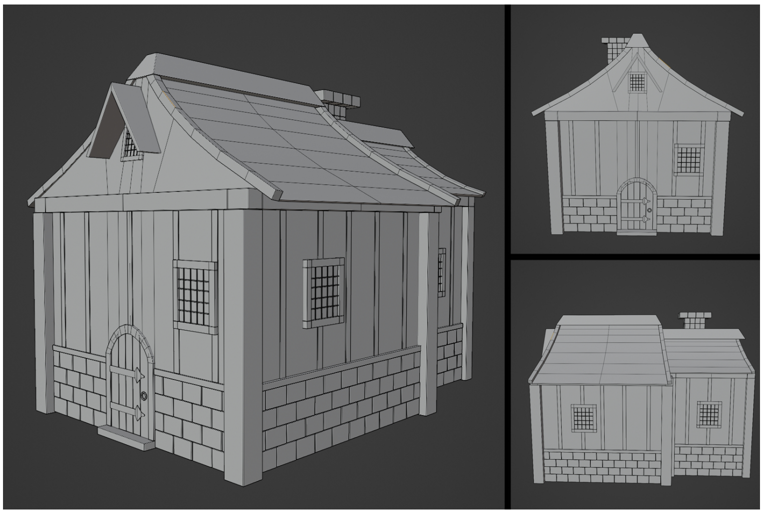


Figure 4.110: House 1 Wireframe

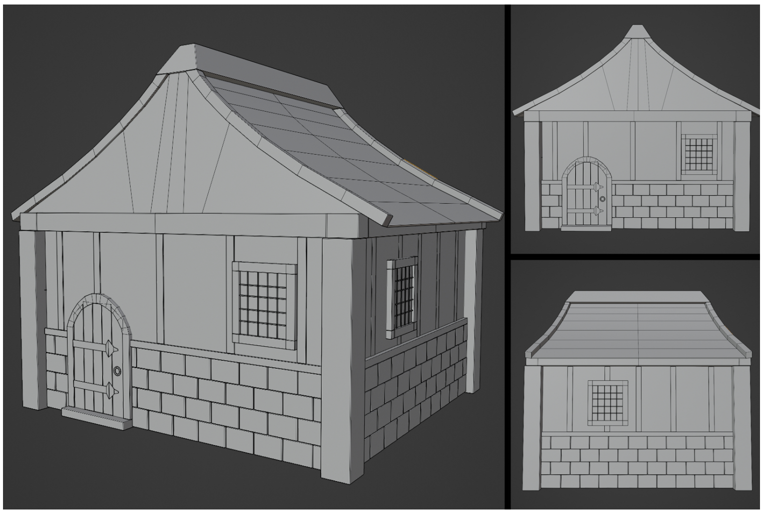


Figure 4.111: House 2 Wireframe

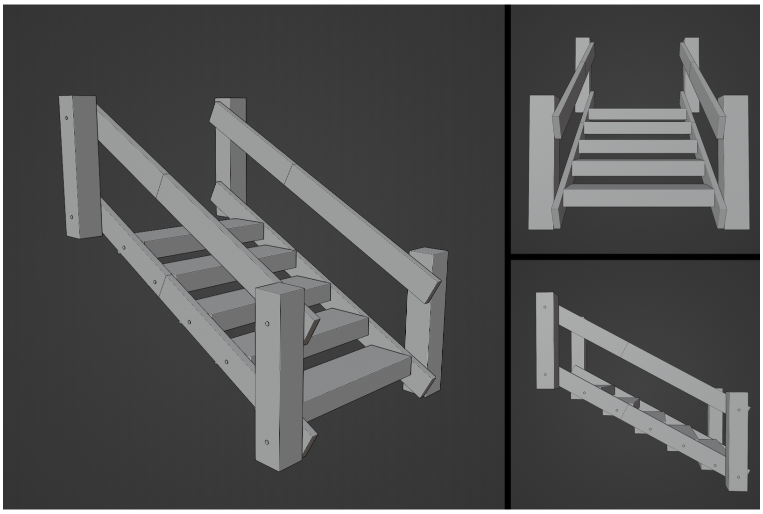


Figure 4.112: Stairs Wireframe

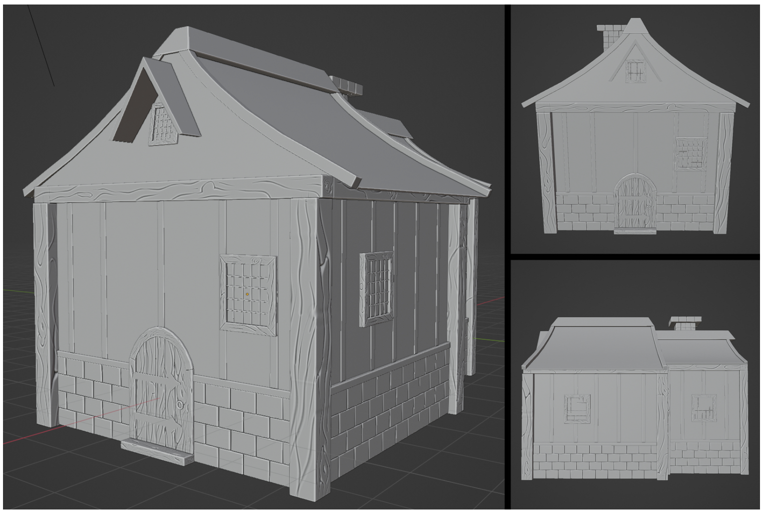


Figure 4.113: House 1 high poly

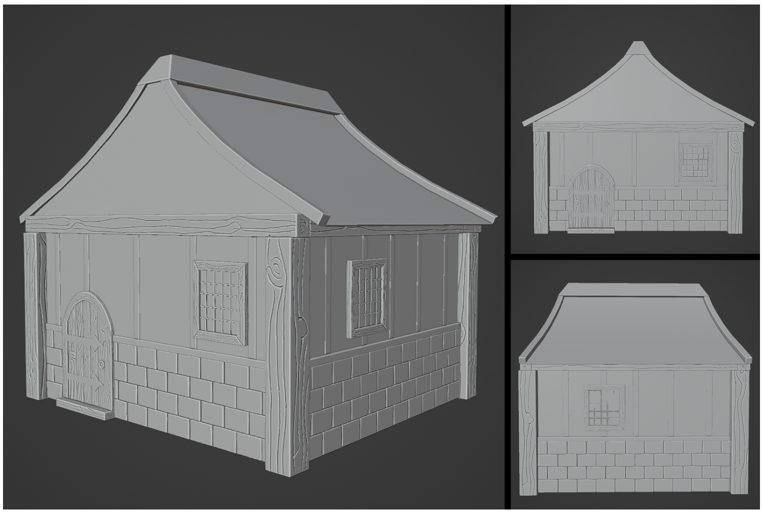


Figure 4.114: House 2 high poly

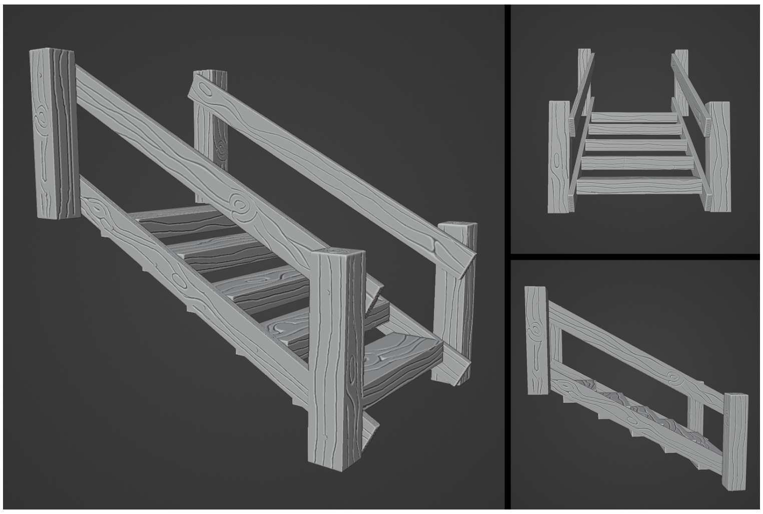


Figure 4.115: Stairs high poly

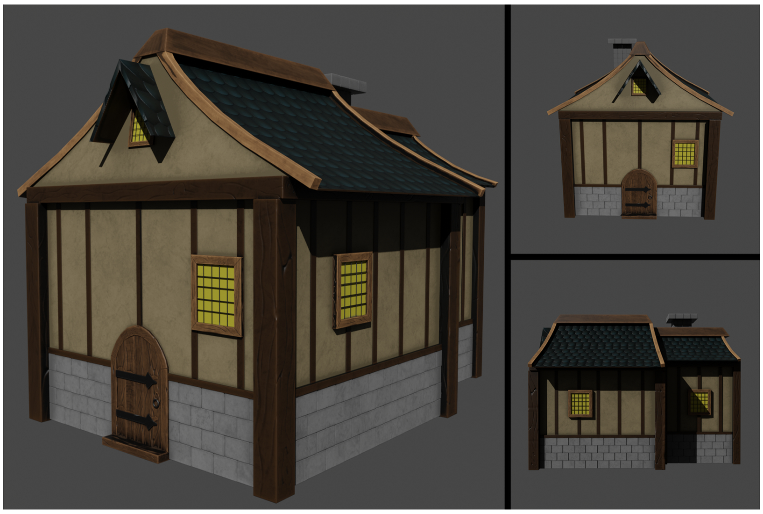


Figure 4.116: House 1 render

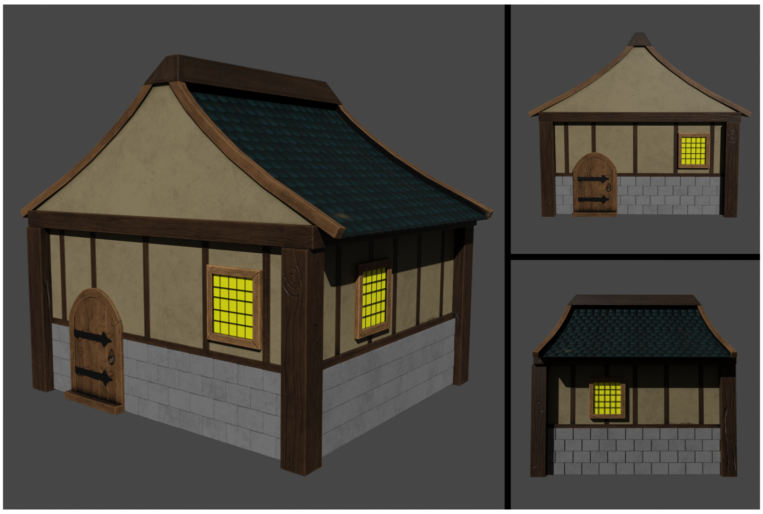


Figure 4.117: House 2 render

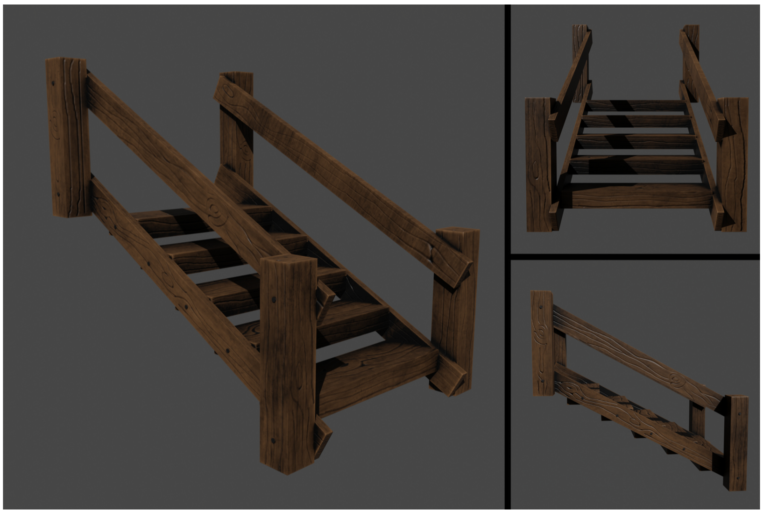


Figure 4.118: Stairs render

The workflow for the tree it's a mix between the workflow for organic parts and the workflow for non-organic assets, because the bricks didn't need retopology, so they were made with box modeling, but the tree needed retopology because it was sculpted from blank.Next you can see the wireframe(see Figure 4.119), the high poly(see Figure 4.120) and the render(see Figure 4.121) of the tree.

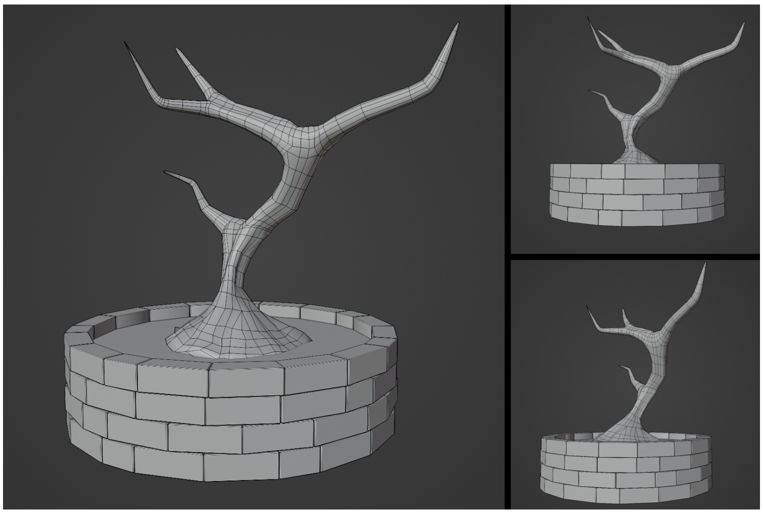


Figure 4.119: Tree wireframe

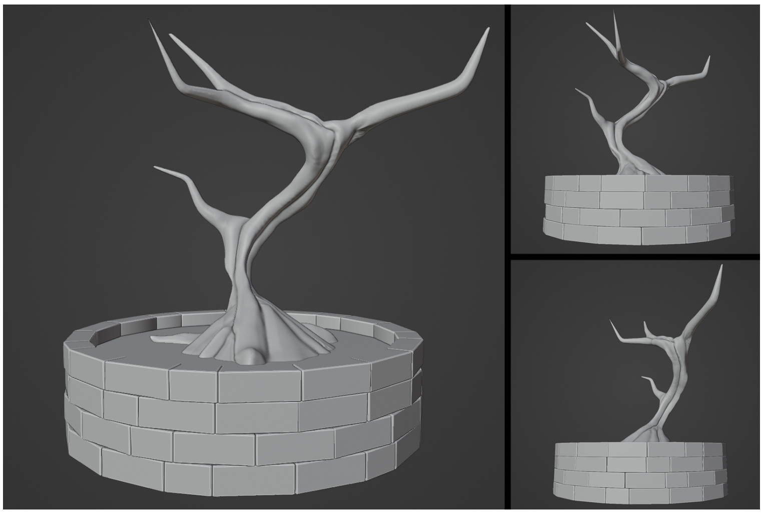


Figure 4.120: Tree high poly

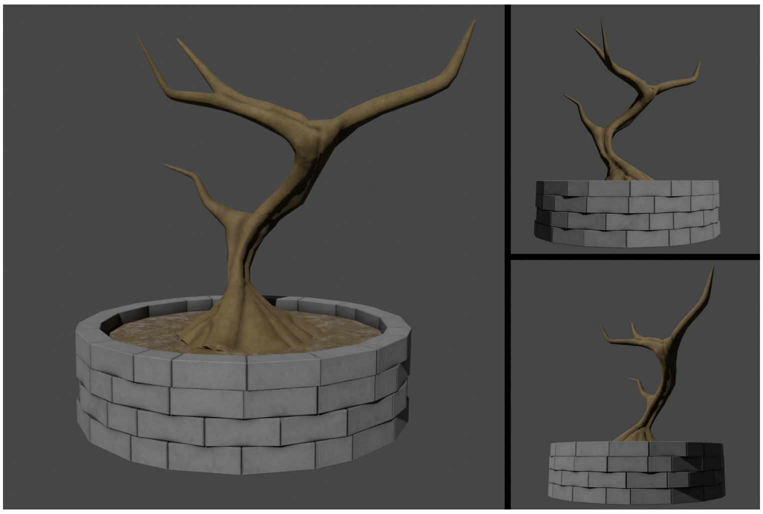


Figure 4.121: Tree render

4.1.3 Enemy AI

Another important part of the game is the enemy AI, so it will be explained on this section

Basic Enemy AI

The basic enemy AI is based on two states, the first one, when he haven't detected the player, so he patrols between two waypoints, and has a line of sight to detect the player, on the image (see Figure 4.122)can be seen the two waypoints of an enemy(green boxes) and his line of sight(red line) , when the player is detected it enters the second state, where he runs to the the player, and when he gets on attack range, stops and attack each few seconds, on the image (see Figure 4.122)can be seen the stopping range(green circle) and the attack range(blue circle).

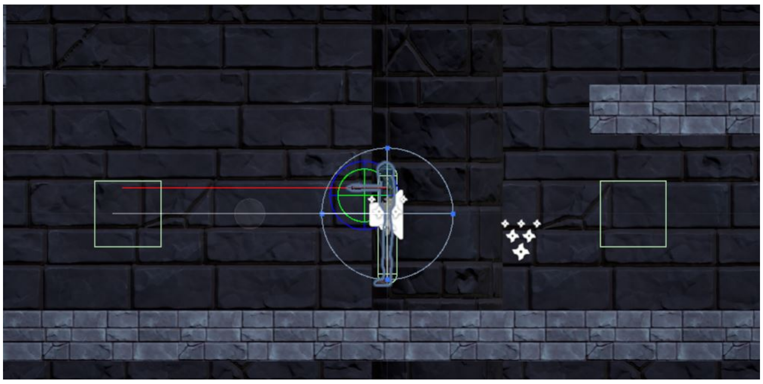


Figure 4.122: Basic enemy gizmos

On the image(see Figure 4.123) is showed an scheme of the basic enemy AI.

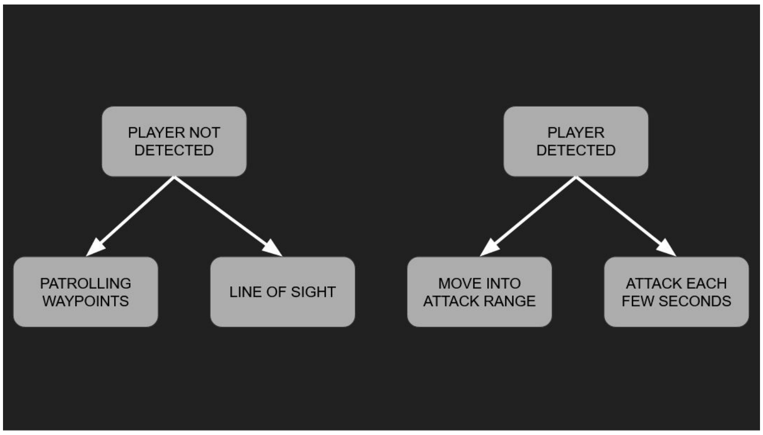


Figure 4.123: Basic enemy AI scheme

Level Boss AI

The level boss AI consist on three attacks, an a time between these attacks so the player can attack him. Those three attacks are, first, a jump attack, the level boss jumps and falls on the position of the player, the next attack are five missile spell that are directed to the player position, and the third attack are three missile that falls on the player position. This loop can be seen on the image below(see Figure 4.124).

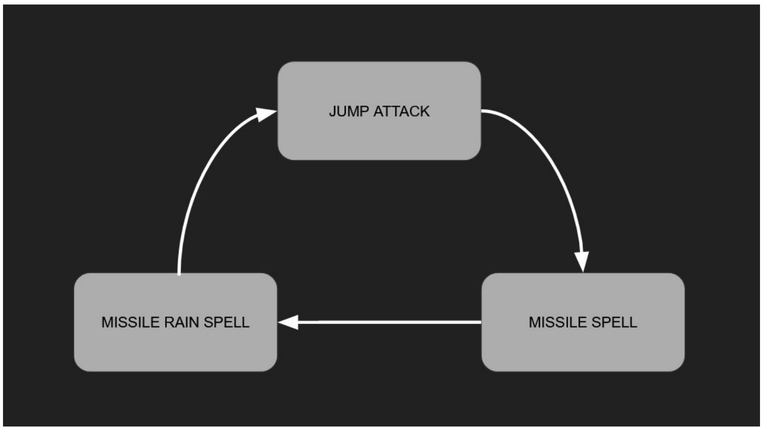


Figure 4.124: Level Boss AI scheme

4.2 Results

In general, the results have been quite satisfactory, I have been able to achieve all the objectives to a greater or lesser extent, the characters have ended looking great and I'm very happy with them, the environments are great, but I think I could have used them better and they could be more detailed, but taking into account that I have limited time, and that my focus was mostly on the characters, I think they have ended looking good, I'm also happy with the gameplay, is much better than the results I have achieved on previous works in the degree so that's a great achievement, with the VFX and sounds I'm satisfied, although they were actually something secondary, they look great and they give the game a more appealing result. There have been deviations compared to the initial planning, so I have been obligated to remove things and give less details to others, but still, the project have ended looking really great and I really think that with more time and polishing some things it could end being a viable project for commercialization.

Chapter 5

Conclusions and Future Work

Contents

5.1 Conclusions

. This was a very ambitious project, the quality and quantity of assets that I wanted to include in the project was too much for the time I had, as I was progressing on the project, the more I realized that I underestimate the time I would need to achieve my objectives. With that being said, the experience have been very gratifying and I have learned a lot, if I compare my 3D skills at this point, and at the beginning of the project, I can see that I have made a lot of progress, and the project have ended looking very great. I have enjoyed the most part of the creation of this project, especially the modeling and animation of the characters, creating three characters with the same style and animating them have been quite a challenge but very enjoyable. With the knowledge that I have now, if I had to do the project now, I would change some things to adjust to the time, the type of the game would be different, and I would only do two characters and one environment so they could be more detailed, but that doesn't mean that I'm not happy with the results of the project. In conclusion, event with the lack of time I have enjoyed creating this project and I'm happy with the result.

5.2 Future work

I don't think this project has a future as hole, I don't think I would be able to create a complete game from this project, not because of the quality but because of the lack of time of resources that are needed to complete a project like that. But I will polish the characters and parts of the environments and use them to enhance the portfolio, and maybe I upload the game to a web like to itch.io to use it also as a demo for my portfolio.

Bibliography

[1] Adobe. Substance painter on adobe web. https://www.adobe.com/es/products/substance3d-painter.html.

[2] Adobe. Substance painter on steam page. https://store.steampowered.com/app/1775390/Substance$_3D_Painter_2$022/.

[3] CG Cookie. Retopoflow github. https://github.com/CGCookie/retopoflow.

[4] Blender Foundation. Blender cycles render engine manual. https://docs.blender.org/manual/en/latest/render/cycles/index.html.

[5] Blender Foundation. Blender manual on box modeling tools. https://docs.blender.org/manual/en/2.80/modeling/meshes/toolbar/index.html.

[6] Blender Foundation. Blender manual on modifiers. https://docs.blender.org/manual/en/2.80/modeling/modifiers/index.html.

[7] Blender Foundation. Blender manual on sculpting tools. https://docs.blender.org/manual/en/2.80/sculpt$_paint/sculpting/tools.html$.

[8] Blender Foundation. Blender official web. https://www.blender.org/.

[9] Blender Foundation. Rigify manual. https://docs.blender.org/manual/en/2.81/addons/rigging/rigify.html.

[10] Glassdoor. Glasdoor page on game designer salaries on spain. https://www.glassdoor.es/Sueldos/junior-game-designer-sueldo-SRCH$_KO0,20.htm$.

[11] Glassdoor. Glassdoor page on 3d artists salaries on spain. https://www.glassdoor.es/Sueldos/artista-3d-sueldo-SRCH$_KO0,10.htm$.

[12] Krita. Krita offcial web. https://krita.org/es/.

[13] Tokio School. Tokio school blog on videogame programmer salaries. https://www.tokioschool.com/noticias/cual-es-sueldo-programador-videojuegos/.

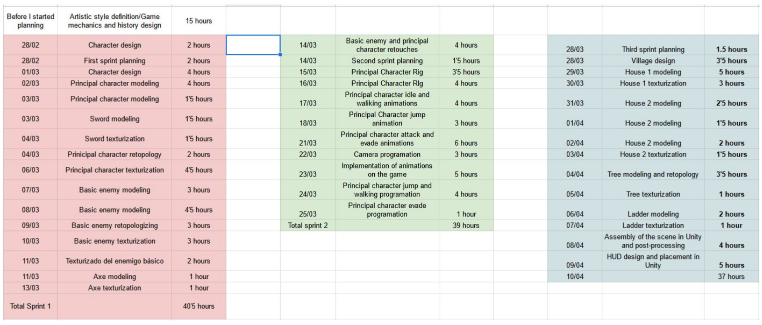
[14] Unity Technologies. Unity official web. https://unity.com/es.

[15] Unity Technologies. Unity urp manual. https://docs.unity3d.com/Packages/com.unity.render-pipelines.universal@13.1/manual/lit-shader.html.

APPENDIX A

Work diary

Here is included the work diary.

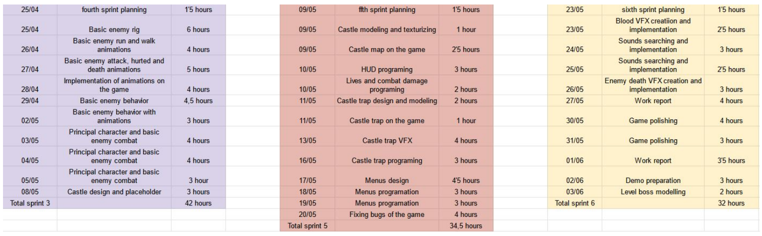


Before I started planning	Artistic style definition/Game mechanics and history design	15 hours
28/02	Character design	2 hours
28/02	First sprint planning	2 hours
01/03	Character design	4 hours
02/03	Principal character modeling	4 hours
03/03	Principal character modeling	1'5 hours
03/03	Sword modeling	1'5 hours
04/03	Sword texturization	1'5 hours
04/03	Prinicipal character retopology	2 hours
06/03	Principal character texturization	4'5 hours
07/03	Basic enemy modeling	3 hours
08/03	Basic enemy modeling	4'5 hours
09/03	Basic enemy retopologizing	3 hours
10/03	Basic enemy texturization	3 hours
11/03	Texturizado del enemigo básico	2 hours
11/03	Axe modeling	1 hour
13/03	Axe texturization	1 hour
Total Sprint 1		40'5 hours

14/03	Basic enemy and principal character retouches	4 hours
14/03	Second sprint planning	1'5 hours
15/03	Principal Character Rig	3'5 hours
16/03	Principal Character RIg	4 hours
17/03	Principal character idle and waliking animations	4 hours
18/03	Principal Character jump animation	3 hours
21/03	Principal character attack and evade animations	6 hours
22/03	Camera programation	3 hours
23/03	Implementation of animations on the game	5 hours
24/03	Principal character jump and walking programation	4 hours
25/03	Principal character evade programation	1 hour
Total sprint 2		39 hours

28/03	Third sprint planning	**1.5 hours**
28/03	Village design	**3'5 hours**
29/03	House 1 modeling	**5 hours**
30/03	House 1 texturization	**3 hours**
31/03	House 2 modeling	**2'5 hours**
01/04	House 2 modeling	**1'5 hours**
02/04	House 2 modeling	**2 hours**
03/04	House 2 texturization	**1'5 hours**
04/04	Tree modeling and retopology	**3'5 hours**
05/04	Tree texturization	**1 hours**
06/04	Ladder modeling	**2 hours**
07/04	Ladder texturization	**1 hour**
08/04	Assembly of the scene in Unity and post-processing	**4 hours**
09/04	HUD design and placement in Unity	**5 hours**
10/04		37 hours

Figure A.1: Table showing work diary from sprint 1, 2 and 3

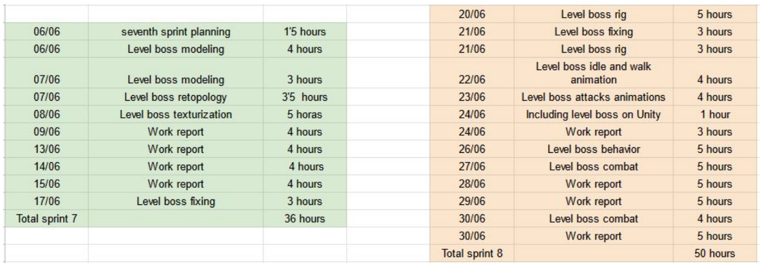


25/04	fourth sprint planning	1'5 hours
25/04	Basic enemy rig	6 hours
26/04	Basic enemy run and walk animations	4 hours
27/04	Basic enemy attack, hurted and death animations	5 hours
28/04	Implementation of animations on the game	4 hours
29/04	Basic enemy behavior	4,5 hours
02/05	Basic enemy behavior with animations	3 hours
03/05	Principal character and basic enemy combat	4 hours
04/05	Principal character and basic enemy combat	4 hours
05/05	Principal character and basic enemy combat	3 hour
08/05	Castle design and placeholder	3 hours
Total sprint 3		42 hours

09/05	ffth sprint planning	1'5 hours
09/05	Castle modeling and texturizing	1 hour
09/05	Castle map on the game	2'5 hours
10/05	HUD programing	3 hours
10/05	Lives and combat damage programing	2 hours
11/05	Castle trap design and modeling	2 hours
11/05	Castle trap on the game	1 hour
13/05	Castle trap VFX	4 hours
16/05	Castle trap programing	3 hours
17/05	Menus design	4'5 hours
18/05	Menus programation	3 hours
19/05	Menus programation	3 hours
20/05	Fixing bugs of the game	4 hours
Total sprint 5		34,5 hours

23/05	sixth sprint planning	1'5 hours
23/05	Blood VFX creatiion and implementation	2'5 hours
24/05	Sounds searching and implementation	3 hours
25/05	Sounds searching and implementation	2'5 hours
26/05	Enemy death VFX creation and implementation	3 hours
27/05	Work report	4 hours
30/05	Game polishing	4 hours
31/05	Game polishing	3 hours
01/06	Work report	3'5 hours
02/06	Demo preparation	3 hours
03/06	Level boss modelling	2 hours
Total sprint 6		32 hours

Figure A.2: Table showing work diary from sprint 4, 5 and 6



				20/06	Level boss rig	5 hours
06/06	seventh sprint planning	1'5 hours		21/06	Level boss fixing	3 hours
06/06	Level boss modeling	4 hours		21/06	Level boss rig	3 hours
07/06	Level boss modeling	3 hours		22/06	Level boss idle and walk animation	4 hours
07/06	Level boss retopology	3'5 hours		23/06	Level boss attacks animations	4 hours
08/06	Level boss texturization	5 horas		24/06	Including level boss on Unity	1 hour
09/06	Work report	4 hours		24/06	Work report	3 hours
13/06	Work report	4 hours		26/06	Level boss behavior	5 hours
14/06	Work report	4 hours		27/06	Level boss combat	5 hours
15/06	Work report	4 hours		28/06	Work report	5 hours
17/06	Level boss fixing	3 hours		29/06	Work report	5 hours
Total sprint 7		36 hours		30/06	Level boss combat	4 hours
				30/06	Work report	5 hours
				Total sprint 8		50 hours

Figure A.3: Table showing work diary from sprint 7 and 8

Appendix B

Source code

The code of this project can be accessed through this link: `https://al385762@bitbucket.org/ivan_ruizUJI/tfgkhardukblade.git`