# Dark Sorcerer: a visual novel video game created with the assistance of AI technologies

**Sergi Timoner Romero**

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

July 19, 2023

Supervised by: Supervised by: Diego José Díaz García

To Víctor

# Acknowledgments

I would like to thank my Final Degree Work supervisor, Diego José Díaz García , for his support and patience.

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

This document aims to cover the development of Dark Sorcerer, a Visual Novel videogame. This project relies heavily on Artificial Intelligence software to streamline the process as much as possible. A videogame prototype is created by applying these recently emerged technologies, verifying the potential of these tools for the game development field.

# Contents

Chapter 1

# Introduction

In this report, I will describe my Final Project, and the means that were used to develop it. In short, it consists on the creation of a Visual Novel called 'Dark Sorcerer'. The script, coding and assets are made through various Artificial Intelligence (AI for short) models.

A visual novel displays a story as text, but also accompanying it with sound effects, music, and graphics. It can be compared to a book that is read/viewed experienced, with a slightly higher amount of interactivity. Examples of visual novels include 'Fate/Stay Night', 'Umineko No Naku Koro Ni' and 'Doki Doki Literature Club!'

**Contents**

## 1.1 Keywords

Visual Novel, asset generation, AI art, AI tools.

## 1.2 Work Motivation

Starting in late 2021 and early 2022, the state of art of AI-related technologies has experienced a revolution. Language based models, image-generation models and even voice imitation tools. This field is evolving at a very fast pace.

These changes in the industry are opening new opportunities, while also upsetting the status quo, which brings dangers to already established businesses and start-ups.

In the field of video-game development, the bottle-neck that limits the learning and producing of projects are mainly the difficulty and cost of acquiring assets (art, music, script), and the time-consuming task of coding.

AI tools that automate the process of creating these assets and accelerate the coding process could further lower the entry barrier to video-game development.

Which is why, it is important to understand and measure how much impact these tools will have. How does the workflow change, once they are applied to the creation process? How much time is saved, compared to before?

And, how likely is it for AI-generated code and resources to become common-place? Will an AI-assisted developer outperform a traditional developer who eschews such tools?

Will there be a need for developers at all, once the AI develops the ability to handle all the parts of the process in an unified manner, just from a prompt from the average user? As of 2023, we are not at that point yet, but it is still worthwhile to consider such dilemmas early on, and acquiring data on the current limitations, drawbacks and capabilities of AI-assisted game development is a good starting point.

## 1.3 Objectives

This project aims to evaluate the feasibility of developing a video-game with minimal human intervention. All assets will be created from AI-related prompts, and that will include the coding of the game. If by the end of the semester, a fully functional and polished video-game is completed while following that limitation, this goal will be fulfilled.

Once that is accomplished, the process will be reviewed so that initial observations can be raised, and conclusions can be drawn from them. We want to answer the question of "How easy is it to make a video-game using AI?". The secondary objective beyond proving that it can be done at all, is finding out how much of a difference does applying these new technologies make, for the developer. It will also be worth comparing how well the finished game holds up against other games in the same genre, in terms of asset quality and gameplay.

In short, a video-game will be made using AI technologies to assist the development. Any difficulties, challenges or observations that arise during the development will be discussed, as well.

## 1.4 Environment and Initial State

Every work is developed under a set of given circumstances (the way of working, the technical and human team, etc.) and start from a given initial state. This section should reflect both. Furthermore, all decisions made before starting the work or externally imposed (those that have not been made by the author of the work) must be clearly stated.

It was decided early on that the genre of the video-game that would be made in this project would be a Visual Novel. This is because the gameplay loop is simple, which would raise the odds of ChatGPT (the AI technology used to generate the code of the game) successfully creating a playable result. It is also a type of game which requires of a decent amount of graphic assets, as well as a script/story, which should be generated with similar tools as well.

At the time of the project's start, there are already early implementations that attempt to integrate AI within the Unity editor, for ease of usage. This project will not be using them, because there is no access to these technologies to the public, and they are hardly polished enough to be used successfully. The equipment that is being used to develop the game lacks the required processing power for running those tools locally, which also limits our options in that regard.

The technologies and tools used on this project are all hosted online, and be free to use. That means, ChatGPT's free model will be used, instead of the paid GPT4, which is admittedly, much better for this task, but unavailable to us. Stable Diffusion, for image generation, will be run on Google Colab, or otherwise, on a free, hosted environment that will handle the VRAM processing.

CHAPTER 2

# Planning and resources evaluation

**Contents**

## 2.1 Planning

The project was scheduled to be split into different parts, which corresponded to the production of the resources necessary to make the game, and their implementation.

| | |
|---|---|
| Generating the story. | 120 hours |
| Creation of the Game's framework in Unity and coding. | 70 hours |
| Generating the Game Art: Character Sprites, background Images, and Interface. | 30 hours |
| Generating the Music and Sound Effects. | 30 hours |
| Project Memory and Other Documents | 50 hours |
| Total: | 300 hours |

Table 2.1: initial schedule of the project

Below is the initial plan to carry out this project, as it was proposed.

1- Generating the story. (120 hours)

ChatGPT or, if it is inaccessible, AI Dungeon will be used to generate the text of the story. Nonsensical results will be repeated until the results are consistent.

2- Creation of the Game's framework in Unity and coding. (70 hours)

Requests will be sent at ChatGPT to produce code that would serve for a Visual Novel game in Unity. Since it is unlikely to give fully functional code, the result will be adjusted manually until it works. It may also end up being more viable to do most of this part manually, and resort to ChatGPT to fill in gaps in the code.

3- Generating the Game Art: Character Sprites, background Images, and Interface. (30 hours)

For this, Stable diffusion will be used. Prompts will be tried and a large amount of images will be generated. From those, the suitable ones will be picked, and edited using the Img2Img and Inpainting functions of Stable Diffusion. For minor edits, if necessary, an art tool such as SAI will be used manually.

4- Generating the Music and Sound Effects. (30 hours)

Suitable genres and instruments will be selected and prompted into Soundraw. The resulting melodies will be listened, and the most fitting will be selected and saved, to be added to the game. VoiceAI will be used to generate voice acting for the characters. 5- Project Memory and Other Documents (50 hours)

Writing all the required material to present for reviewing, including this document.

## 2.2 Resource Evaluation

The project will be carried out in a laptop with a 6 VRAM GPU. Because it is not potent enough to run the required AI tools, online services offering them will be used instead. The software used is described next:

1- ChatGPT: ChatGPT is an AI Chatbot developed by Open AI. The chatbot has a language-based model that the developer fine-tunes for human interaction in a conversational manner. Effectively it's a simulated chatbot primarily designed for customer service; In this project, it is used to generate the storyline script of the Visual Novel game, as well as the source code in C#. It is also used to provide assistance or instructions for successfully using Unity.

2- Stable Diffusion: Stable diffusion is an open-source text-to-image model of Deep Learning published in 2022 by Stability AI. It creates images which are conditioned by textual descriptions or 'prompts'. The graphic components of the visual novel, such as the character sprites, background, and game art, is made using this tool.

3- Soundraw: Soundraw is an AI music generator that allows the user to create and compose original, royalty-free music. The visual novel's soundtrack is made using this tool.

4- Unity: Unity is an all purpose, cross-platform game engine that supports 2D and 3D graphics, drag and drop functionality and scripting through C#. It is the engine upon which this project's visual novel is built.

## 2.3 Time Verification

I will now compare the original plan, with how the project actually progressed in practice. The most important changes will also be explained and justified.

| Task | Planned Duration | Actual time spent |
|---|---|---|
| Generating the story. | 120 hours | 110 hours |
| Creation of the Game's framework in Unity and coding. | 70 hours | 74 hours |
| Generating the Game Art. | 30 hours | 56 hours |
| Generating the Music and Sound Effects. | 30 hours | 6 hours |
| Project Memory and Other Documents | 50 hours | 56 hours |
| Total: | 300 hours | 302 hours |

Table 2.2: Comparison between the initial schedule and the final schedule

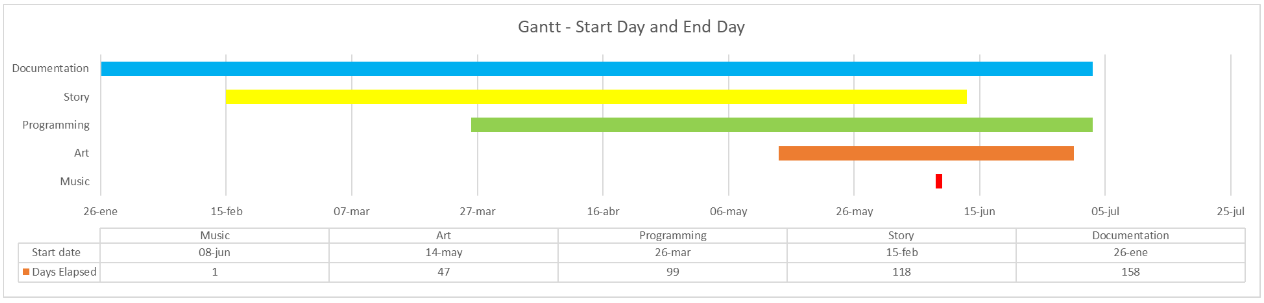
The biggest change is the amount of time invested in the game's music and audio. Generating the music turned out to be a quicker and easier affair than it was initially expected. In barely under 6 hours of work, There were already 10 tracks of music for the game, which was deemed sufficient. There had been plans to produce voice acting for the game's dialogue lines, but it became clear it was not going to be possible, and the idea was discarded.

The second biggest change is in the hours dedicated to generating the Game Art. Initially, the idea had been to create spritesheets for the characters, and overlay them over background images, which would depict landscapes. However, generating the spritesheets proved to be an issue, as Stable Diffusion did not produce consistent results. The amount and dimensions of the character portraits when laid in the spritesheet was different in each generation, and in many cases, the faces were not different from one sprite to the next. I switched from sprites to pictures depicting the events, instead. This required a much larger amount of pictures.

I populated the first chapter with images, averaging for a new image for every three or four lines. While this worked nicely, I greatly underestimated how long it would take me to give the rest of the story this treatment. Hence, I ended up spending a considerable amount of time making new images to make up for the lack of character sprites.

There were plans to make up to 8 paths, each leading to a different ending. In the end, only 6 of paths were made, as it was decided to spare those last remaining 10 hours of work on generating the story, which would have taken considerable longer, and invest them into further creating images to illustrate the paths that were already finished.

Out of the 6 final paths, only 2 are fully covered in images. The rest have a much smaller amount of images displayed per chapter.



| | Music | Art | Programming | Story | Documentation |
|---|---|---|---|---|---|
| Start date | 08-jun | 14-may | 26-mar | 15-feb | 26-ene |
| 🟧 Days Elapsed | 1 | 47 | 99 | 118 | 158 |

Figure 2.1: Gantt Chart depicting the start and end of every part of the project

## 2.4 Cost Evaluation

This project has been carried out by a single person, who was in charge of overseeing all the parts of it.

A videogame developer on average in Spain is paid around 1.740 € per month. This project has been in the works for roughly 5 months, so, should a developer be hired for a project such as this, for this length the cost would be around 8.700 €

Save for the price of the hardware, which is around 1400€, the software used for this project is entirely free, amounting to 0€ of cost. ChatGPT is free to use. So is Stable Diffusion, which has become open-source. All means used to develop the project have no cost for the developer.

There was a possibility of Google shutting down the Colab project housing the free web-ui interface used to generate the graphics of the game. While that did happen eventually, I found free alternatives I usd for the remainder of the images. Had no free alternatives become available, the same web-ui was also hosted in Runpod.com and Paperspace.com. The prices for renting GPU on Runpod ranged between \$0.44/hr to \$1.89 /hr. On the other hand, the subscription to the basic Paperspace services has the price of \$8.00/month. I was prepared to have to use these, but it ended up not being necessary.

Costs of the tools and human resources employed for the development of this project:

| | | |
|---|---|---|
| Videogame Developer | The human worker developing the game | 8700 € |
| HP ENVY Notebook | Gaming laptop in which the project is hosted | 1400€ |
| ChatGPT | Free online LLM for Generating code | 0€ |
| Kobold AI Lite | Generating story script (dialogue) | 0€ |
| Stable Diffusion (Google Collab) | Generating background images | 0€ |
| Soundraw | Generating the music | 0€ |
| Unity | Game engine on which the game is built on. | 0€ |
| VisualStudio | Code editor | 0€ |
| Overleaf | Free LaTex editor for the memory | 0€ |
| Total | - | 10100€ |

Table 2.3: Project Costs

Chapter 3

# System Analysis and Design

**Contents**

This chapter presents the requirements analysis, design and architecture of the project.

## 3.1 Requirement Analysis

Among the many genres of videogames, Visual Novels are structured in a rather simple way. A screen depicting an image, meant to be interpreted as a background or scenario. Character sprites adopting a variety of poses and expressions. And, either hovering over the sprites or placed in a dialogue box beneath them, the text representing the narration and the dialogue of the characters. Different configurations of text, background and character sprites are presented according to the player's choices, which are prompted to them via selectable buttons on key moments.

The player has to be able to click on the screen to progress the story and select the buttons to decide the path the story follows.

All other screens and functions within the game are minor, and provide quality of life to the player by allowing them to make adjustments to the settings, or to save the state of the game.

### 3.1.1 Functional Requirements

Here is a list of the functional requirements of the project:

- R1: The player can advance the story by right-clicking the screen
- R2: The player can select a story path by clicking a choice at key points of the story.
- R3: The player can open up a backlog that displays previously read lines.
- R4: The player can skip forward, passing over the text quickly.
- R5: The player can hide the text box to view the background picture at any time.
- R6: The player can save their progress, storing their progress on a permanent, loadable file.
- R7: The player can load a saved game to resume their previous playthrough where they left off.

| Requirement: | R1 |
|---|---|
| Input: | Player clicks on the main screen. |
| Output: | Script line displayed, visible sprites and background change. |
| In the main screen of the game, the player clicks on the screen to advance the story. | |

Table 3.1: Functional requirement «CLICK1. Click to advance»

| Requirement: | R2 |
|---|---|
| Input: | Player clicks on a storyline path button |
| Output: | A new chapter file is loaded, based on which button the player selected. |
| When the player is prompted to choose the actions of the visual novel's protagonist, two or more buttons will be displayed to them. Clicking them will determine which path the story will follow, and load the required files associated with that path. | |

Table 3.2: Functional requirement «CLICK2. Click to choose path»

| Requirement: | R3 |
|---|---|
| Input: | Player clicks the button labeled 'backog' on the UI box containing the text |
| Output: | The backlog opens up, if it wasn't already. If it was, it closes instead. |
| The player can open or close the backlog text box by clicking the button associated to it. The backlog allows the player to view lines that have already been seen. | |

Table 3.3: Functional requirement «BACKLOG1. Backlog button»

| Requirement: | R4 |
|---|---|
| Input: | Player clicks the button labeled 'skip' on the UI box containing the text. |
| Output: | The text fast-forwards, proceeding quickly until the player clicks on the screen, the script reaches a storyline path choice, or the story ends. |
| The player can cause the text to proceed much faster in order to skip ahead in the story without manually having to click repeatedly to advance the lines one at a time. | |

Table 3.4: Functional requirement «SKIP1. Skip text button»

| Requirement: | R5 |
|---|---|
| Input: | Player clicks the button labeled 'hide text' on the UI box containing the text. |
| Output: | The text box containing the currently displayed line disappears from the screen |
| The player can hide the text box to fully view the character sprites, background and art, which would normally be partially covered. Once they are done watching, a single click anywhere on the screen will cause the text-box to return | |

Table 3.5: Functional requirement «HIDE1. Hide Text button»

| Requirement: | R6 |
|---|---|
| Input: | Player clicks the button labeled ’save’ on the UI box containing the text. |
| Output: | The line in which the layer is currently becomes registered on a file |
| By clicking the save button, the game registers the current line in which the player is. This allows the player to mark a spot of the story in order to return to it at a later time, through the Load action. | |

Table 3.6: Functional requirement «SAVE1. Save the game»

| Requirement: | R7 |
|---|---|
| Input: | Player clicks the button labeled ’load’ on the UI box containing the text. |
| Output: | The player is brought to the game state that was registered last time the ’save’ button was pressed |
| By clicking the load button, the game brings the player to the line that is registered in the save file. This allows them to continue reading from the point they left off, when they last played the game. | |

Table 3.7: Functional requirement «LOAD1. Load the game»

### 3.1.2 Non-functional Requirements

Non-functional requirements impose conditions on the design or implementation. In this project, the non-functional requirements are as follow:

- R8: The Visual Novel’s assets must be generated by AI technology.
- R9: All the contents of the story script must be reachable in-game. That is, the player must be able to actually reach all available endings.
- R10: The script of the game must be coherent. The narrative must be consistent and not present plot-holes.
- R11: The art assets must be consistent in style. Furthermore, the character art may not present anatomy flaws or disfigured shapes, which AI-generated art is known to produce.
- R12: The game must be playable on a Windows 10 computer.

## 3.2 System Design

Initially, there was going to be a more complex system of game states. This was the description I proposed when I was yet to begin working on the project:

From the Main menu, the player can access to the story screen, enter the screen to change the settings, enter the loading screen and exit the game.

From the story screen, the player will view the story, and access the pause screen. The pause screen lets the player interface with the backlog, the save screen and the loading screen, as well as return to the main menu or to the story screen when they are done there. The loading screen allows the player to go to a saved point of the story. It can be exited to return the screen from which the player entered it.

In the final version of the game, the pause screen is un-necessary, because all of those buttons are added to the text box of the Story Screen. Now the backlog can be accessed directly from the Story Screen without needing to go to a different scene. Lastly, the load and save screen have been replaced. A few buttons on the text box, and in the title screen suffice to do the same function.

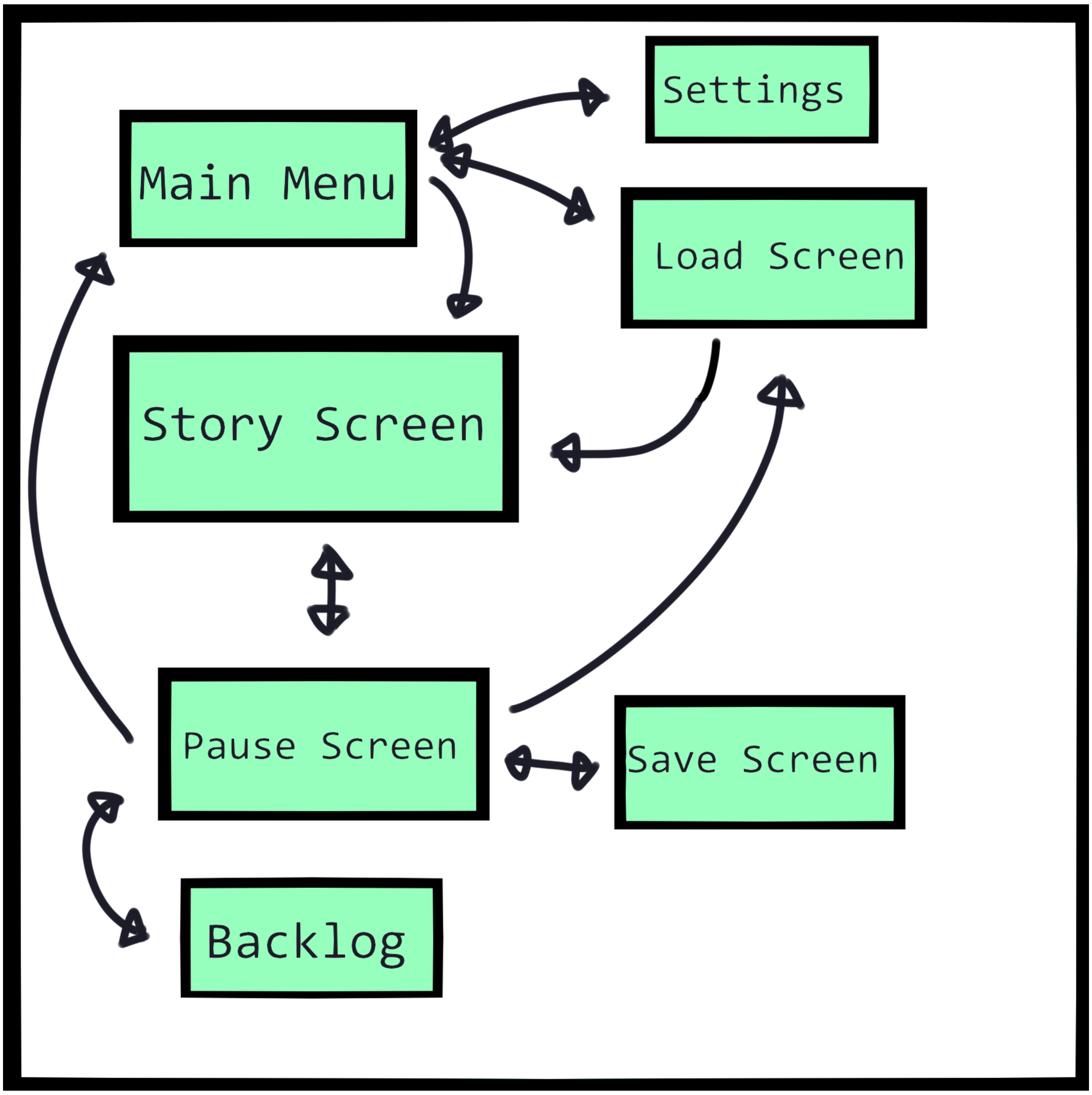


Figure 3.1: initially intended game states

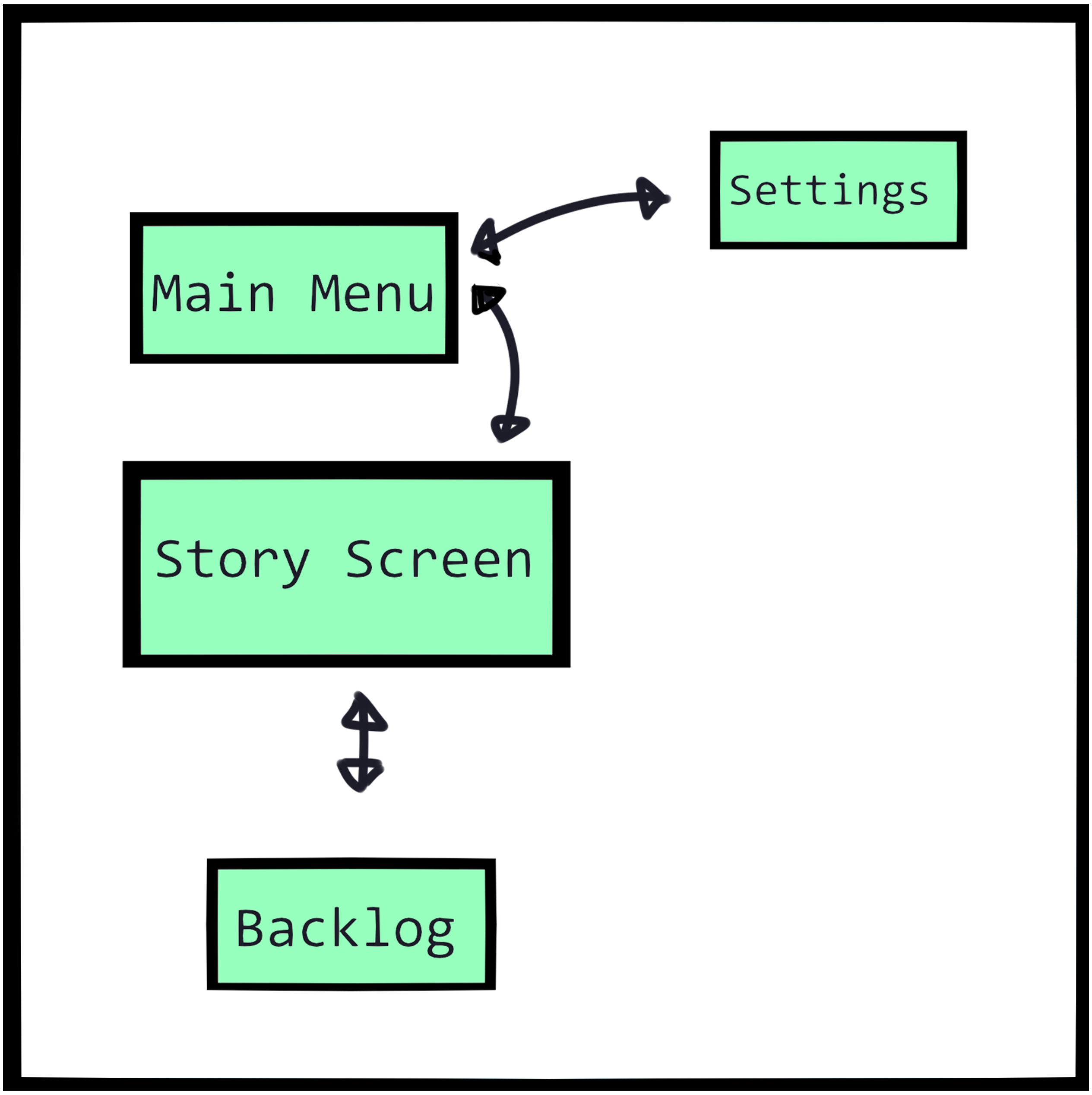


Figure 3.2: game states in the final version of the game

## 3.3 System Architecture

Minimum System Requirements to run the visual novel made during the project:
Operating system version: Windows 7 (SP1+) , Windows 10 and Windows 11

CPU: x86, x64 architecture with SSE2 instruction set support.

Graphics API: DX10, DX11, DX12 capable.

Additional requirements: Hardware vendor officially supported drivers. For development: IL2CPP scripting back-end requires Visual Studio 2015 with C++ Tools component or later and Windows 10 SDK.

## 3.4 Interface Design

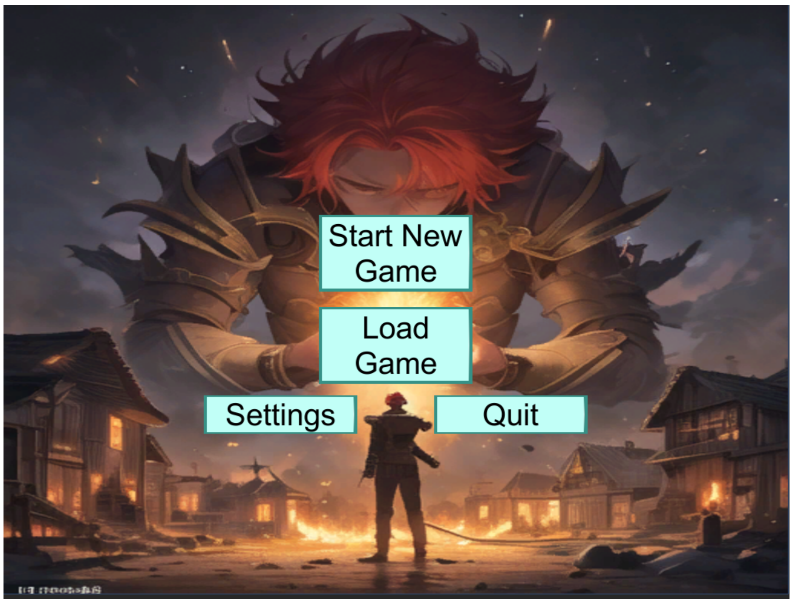
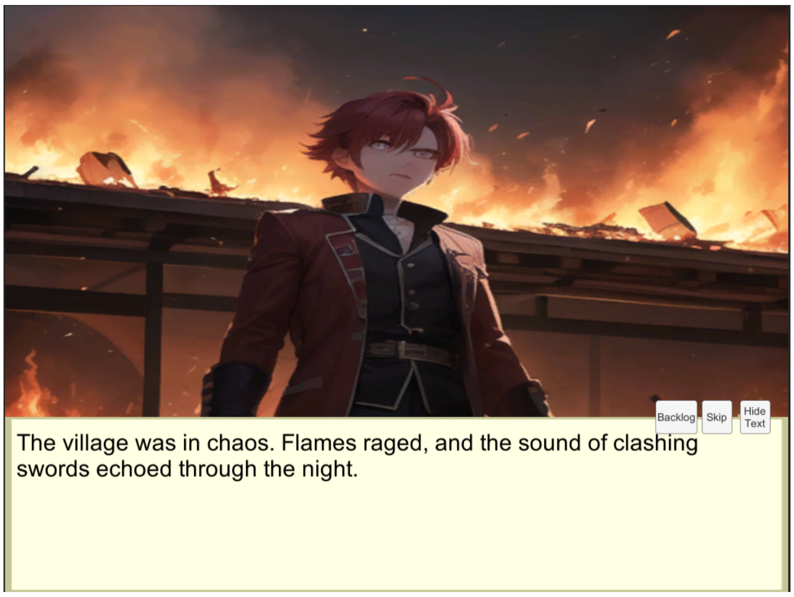


Figure 3.3: Arrangement of buttons of the title screen



The village was in chaos. Flames raged, and the sound of clashing swords echoed through the night.

Figure 3.4: Default interface of the main screen

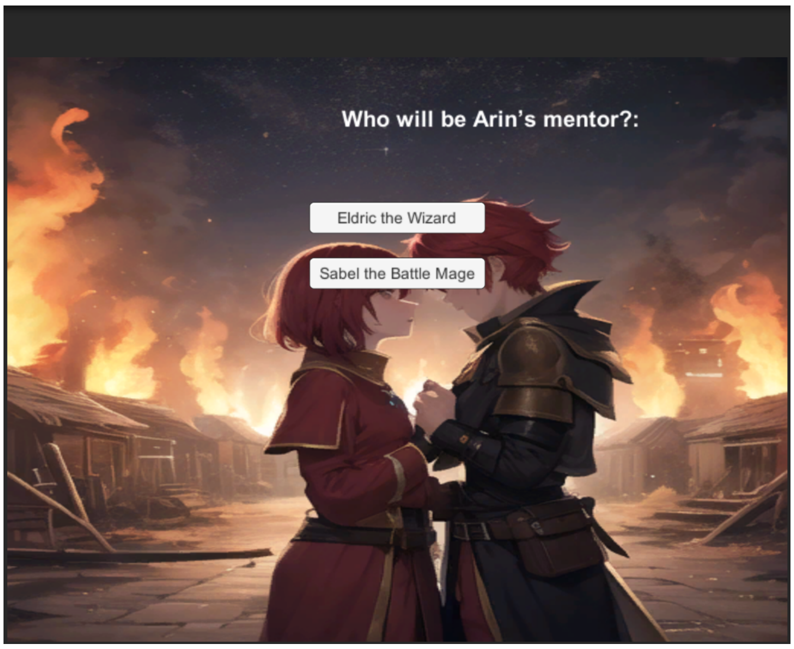


Figure 3.5: Arrangement of buttons of the main screen when there is a choice.

Chapter 4

# Game Design Document

**Contents**

## 4.1 Game Overview

Dark Sorcerer tells the tale of Arin, a young boy born with the talent to use magic. The day his village is attacked by the minions of the Dark Sorcerer, he awakens to his power and flees from his home with his family. Once they make it to safety, Arin will have to find a way to master his powers, so that he may use them to oppose the Dark Sorcerer and protect his people from him. Thus, he embarks a journey to save his kingdom from the evil plans of his enemy.

### 4.1.1 Genre

The game is a visual novel, a sequence of text and images that present a story to the player/reader, and allow them to make choices, which determine the outcome of the story.

### 4.1.2 Target Audience

The intended audience for the game is early to late teenagers, possibly young adults, as well. It will be played by regular players of visual novels or casual readers of epic fantasy novels.

## 4.2 Game Mechanics

The gameplay is minimal. A screen will display the background, and character sprites will appear, switching expressions whenever the story has them act or talk. Occasionally, artwork depicting the events in the story will be shown to the player. Meanwhile, a text box in front of those will show the text of the narration and the dialogue.

The player will be able to click or hit space to make the next lines advance. It will also be possible to right-click to open up a menu, which will allow viewing the images without text, skip to the next chapter, read the backlog or save their game, marking the spot at which they are at, for returning to it at a later time.

In chapters that present the player a choice, the text will give way to buttons depicting the options available. Hovering over a choice will highlight it, and clicking it will select it.

The majority of the playtime will be spent in the story screen. This one will have a plain text-box to allow reading, located at the bottom half of the screen, occupying about 20 to 30% of the view. It will cover the background, but it will be possible to conceal it with a button press, should the player wish to see the image below. The background placed behind this text box will contain AI art depicting either events from the story, or a picture of the environment in which the characters are. Along with the background, sprites of the characters will be visible, being highlighted to represent one of them is talking.

## 4.3 Game Story

Dark Sorcerer's story has branching paths, which lead to several possible endings. The story is divided in three parts. Each part will have several chapters, and a single major choice, in which two alternatives will be presented.

Depending of his choices, he will train under the tutelage of a sagely wizard called Eldric, or a battle-mage called Sabel. With what he learns from them, he will have to prepare for a difficult quest. Whether he will go through it with alone with his mentor, or accompanied by his little sister Irina, will be the second choice, and the options available for him in the end will depend on what he chose in these two moments of the story.

Like it was mentioned previously, the first choice has the player choose who the protagonist's mentor will be.

In the second act, only the path of the Battlemage mentor has a choice. This choice has the player decide whether his sister joins him on his journey to defeat the Dark Sorcerer or not. Lastly, in the third part, at the climax of the story, the protagonist will have a final choice, on what he is willing to sacrifice to defeat the villain and save an endangered village.

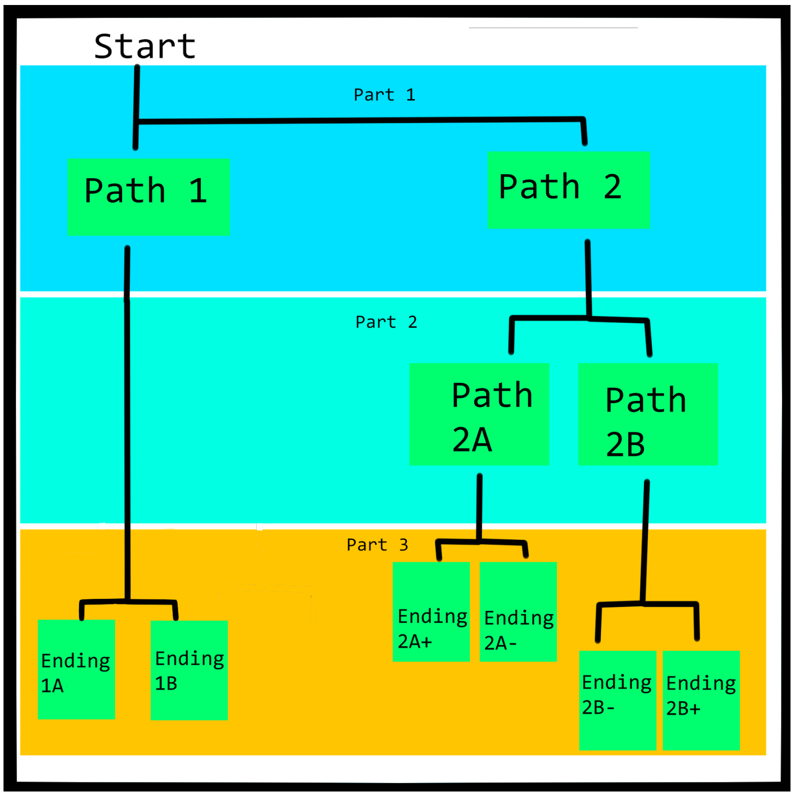


Figure 4.1: Chart depicting story routes and the endings they lead to

That makes for a total of six paths which have different endings at the end of each of them.

## 4.4 Character design

In Dark Sorcerer there are several important characters which appear in all acts, no matter what path the player chooses:

Arin

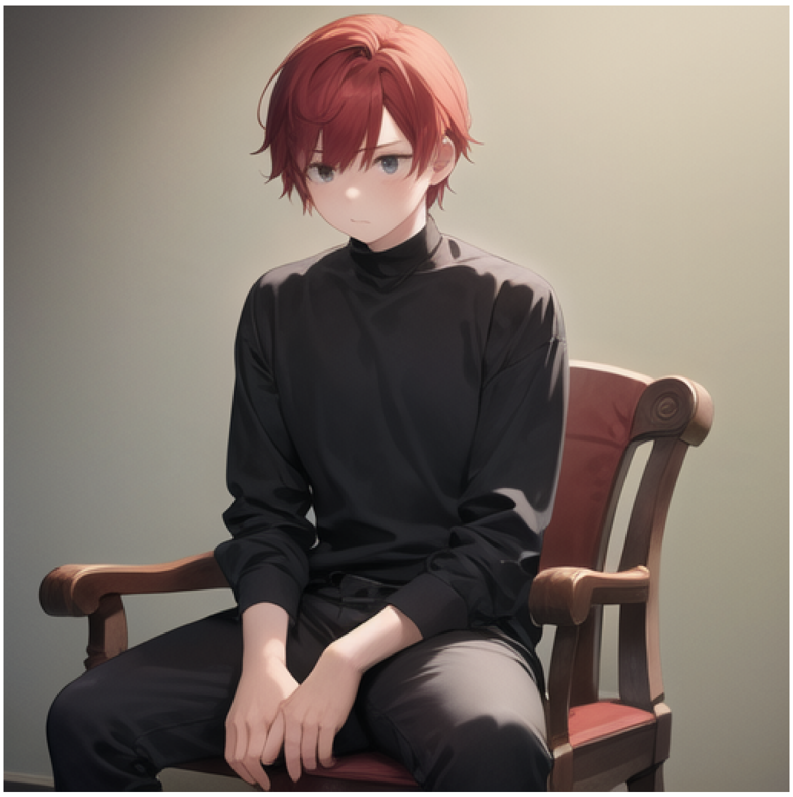


Figure 4.2: Arin

The protagonist of the game. He awakens to his magical power while the Black Guard raids his village, and uses those to defend his family. He is hard-working and serious, with a strong sense of justice.

Irina

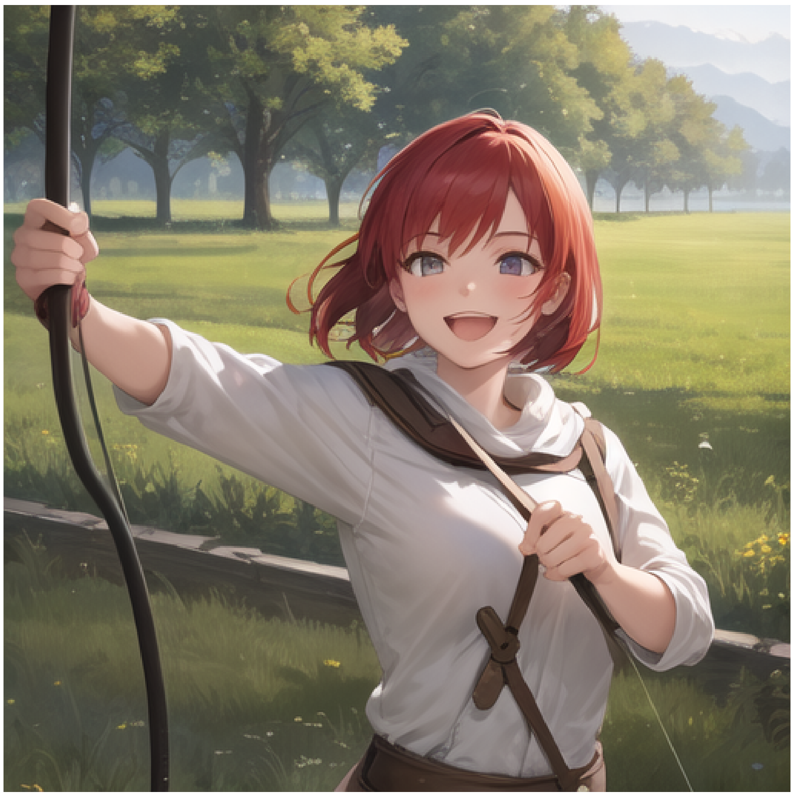


Figure 4.3: Irina

Arin's little sister. She moves with Arin and his parents to Ralz village after their home village is raided. While Arin learns how to use magic, Irina trains to become an archer. In one of the routes, she accompanies Arin in his adventures.

Eldric

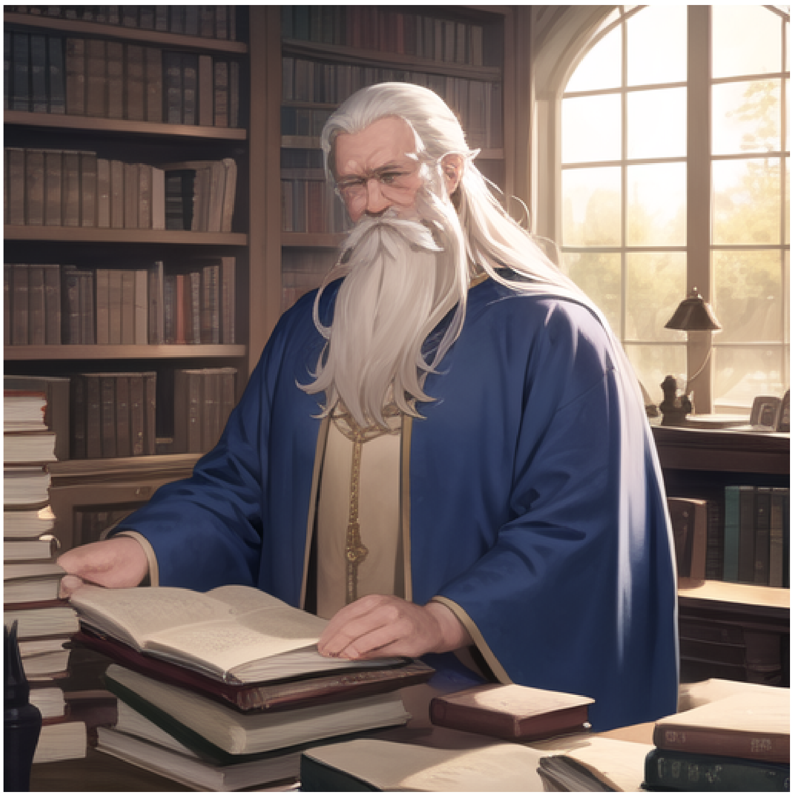


Figure 4.4: Eldric

A wise wizard, and an important member of the Magic Council, which work to keep the kingdom safe from malicious magic users. He lives in Ralz village, and offers to teach Arin how to use magic adequately. He prefers to use long range spells that require time to prepare.

Sabel

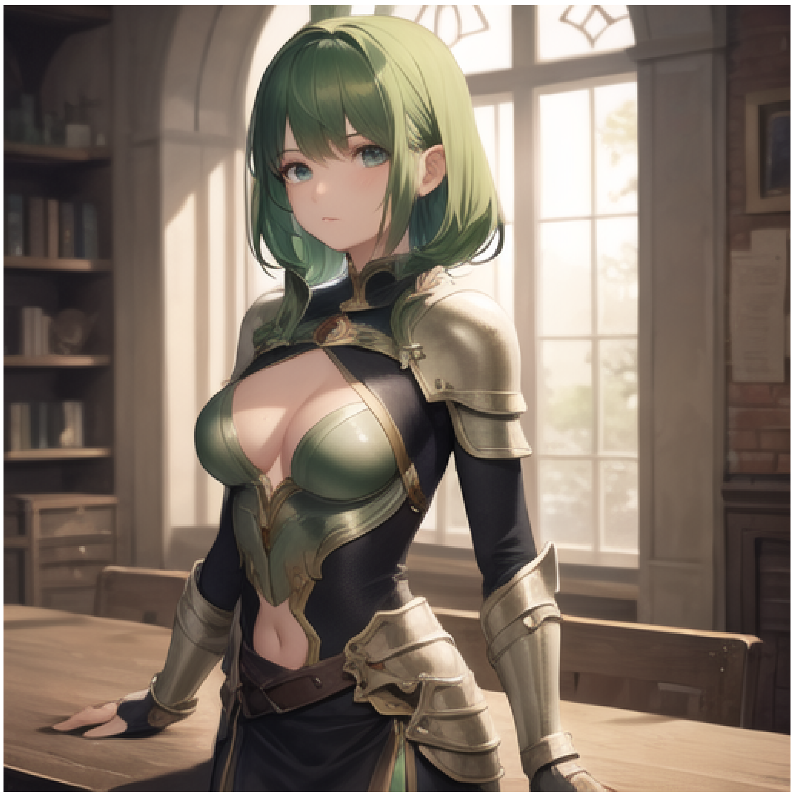


Figure 4.5: Sabel

A battle mage who lives in Ralz village. She has extensive martial training and has mastered a lot of weapons, in addition to her magic, which she uses during combat to support her mixed fighting style. Along with Eldric, she involves herself on the search for the Dark Sorcerer, and volunteers to train Arin.

Eryndor

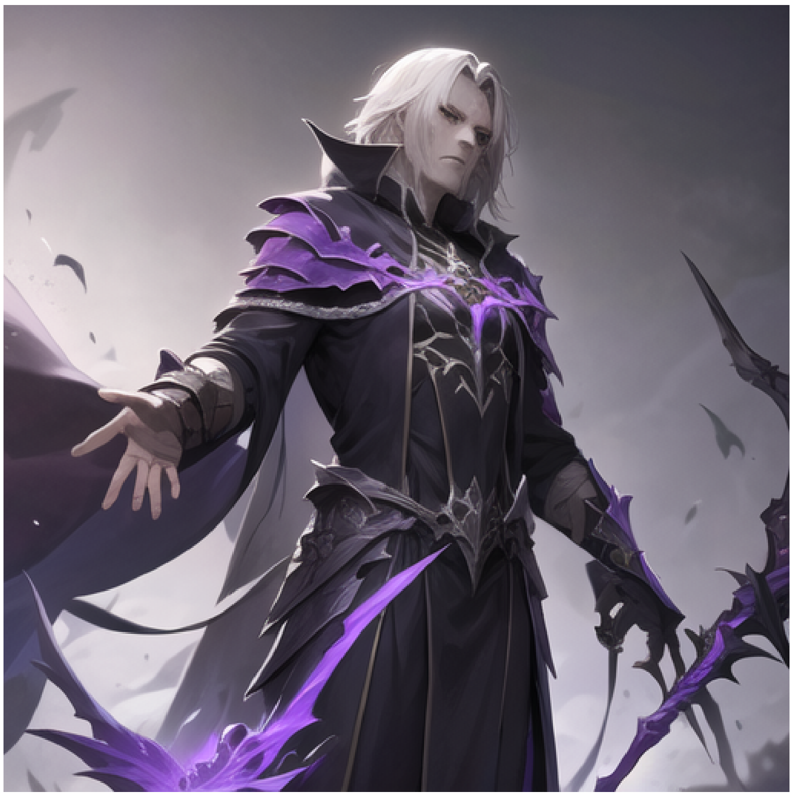


Figure 4.6: Eryndor

The Dark Sorcerer. He leads the Black Guard, and is responsible for the destruction of many villages and the summoning of countless monsters and demons into the kingdom. Resentful and cruel, he will stop at nothing to achieve his ambitions.

## 4.5 Art Direction

The art of the game is AI generated. the aesthetic of the game is anime style. The AI art models AnythingV3 and Anylora have been used in Stable Diffusion to make all of the graphics of the game.

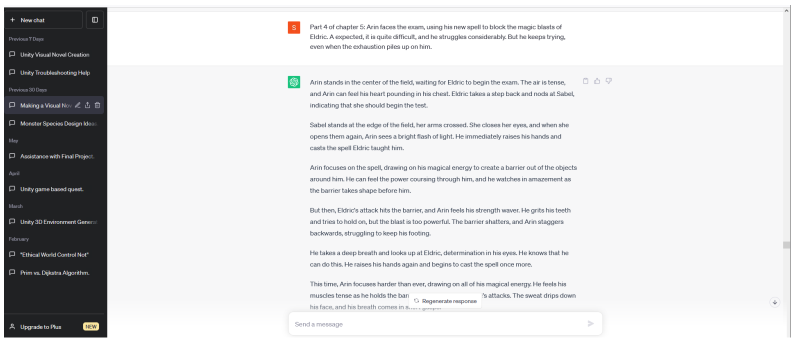
CHAPTER 5

# Work Development and Results

**Contents**

## 5.1 Work Development Overview

The Work Development section of this report details the development of a Visual Novel in Unity, leveraging the use of several AI tools to enhance the project's overall quality. Notably, ChatGPT was utilized to generate the script for the story and the code for the game, streamlining the development process and saving valuable time.



New chat
Previous 7 Days
Unity Visual Novel Creation
Unity Troubleshooting Help
Previous 30 Days
Making a Visual Nov
Monster Species Design Ideas
May
Assistance with Final Project.
April
Unity game based quest.
March
Unity 3D Environment Generat
February
"Ethical World Control Not"
Prim vs. Dijkstra Algorithm.
Upgrade to Plus
NEW
Regenerate response
Send a message

Part 4 of chapter 5: Arin faces the exam, using his new spell to block the magic blasts of Eldric. A expected, it is quite difficult, and he struggles considerably. But he keeps trying, even when the exhaustion piles up on him.

Arin stands in the center of the field, waiting for Eldric to begin the exam. The air is tense, and Arin can feel his heart pounding in his chest. Eldric takes a step back and nods at Sabel, indicating that she should begin the test.

Sabel stands at the edge of the field, her arms crossed. She closes her eyes, and when she opens them again, Arin sees a bright flash of light. He immediately raises his hands and casts the spell Eldric taught him.

Arin focuses on the spell, drawing on his magical energy to create a barrier out of the objects around him. He can feel the power coursing through him, and he watches in amazement as the barrier takes shape before him.

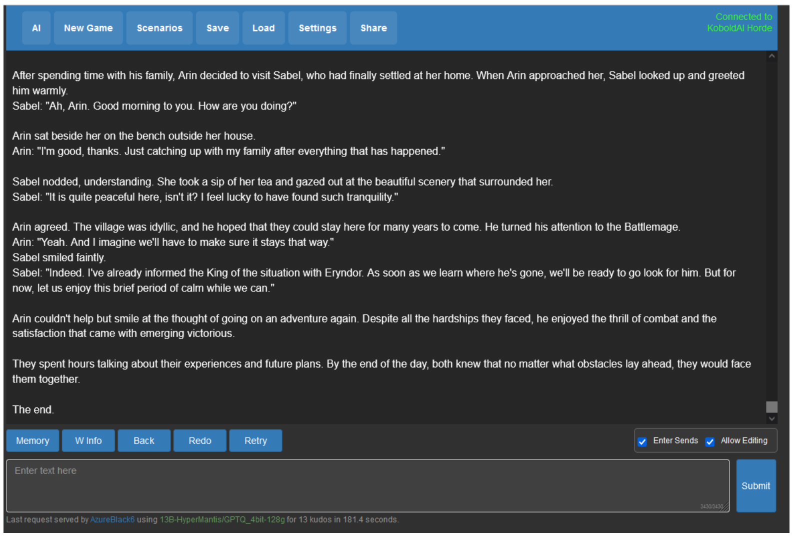
But then, Eldric's attack hits the barrier, and Arin feels his strength waver. He grits his teeth and tries to hold on, but the blast is too powerful. The barrier shatters, and Arin staggers backwards, struggling to keep his footing.

He takes a deep breath and looks up at Eldric, determination in his eyes. He knows that he can do this. He raises his hands again and begins to cast the spell once more.

This time, Arin focuses harder than ever, drawing on all of his magical energy. He feels his muscles tense as he holds the bar 's attacks. The sweat drips down his face, and his breath comes in s

Figure 5.1: ChatGPT generating the script of the game's story

However, it should be noted that while the use of ChatGPT was successful in generating the game's code, the generated script required extensive editing to maintain coherence and ensure that the plot progressed as intended. This was less noticeable during the early chapters of the story, in which ChatGPT could recall clearly all the key points from conversations and prompts. But the more the story advanced, the more frequently it would forget important things established early on. It would also introduce new characters without being prompted to do so. Its prose had also a tendency to repeat itself. The gradual decrease of internal coherence in the text stalled the project for a while.



After spending time with his family, Arin decided to visit Sabel, who had finally settled at her home. When Arin approached her, Sabel looked up and greeted him warmly.
Sabel: "Ah, Arin. Good morning to you. How are you doing?"

Arin sat beside her on the bench outside her house.
Arin: "I'm good, thanks. Just catching up with my family after everything that has happened."

Sabel nodded, understanding. She took a sip of her tea and gazed out at the beautiful scenery that surrounded her.
Sabel: "It is quite peaceful here, isn't it? I feel lucky to have found such tranquility."

Arin agreed. The village was idyllic, and he hoped that they could stay here for many years to come. He turned his attention to the Battlemage.
Arin: "Yeah. And I imagine we'll have to make sure it stays that way."
Sabel smiled faintly.
Sabel: "Indeed. I've already informed the King of the situation with Eryndor. As soon as we learn where he's gone, we'll be ready to go look for him. But for now, let us enjoy this brief period of calm while we can."

Arin couldn't help but smile at the thought of going on an adventure again. Despite all the hardships they faced, he enjoyed the thrill of combat and the satisfaction that came with emerging victorious.

They spent hours talking about their experiences and future plans. By the end of the day, both knew that no matter what obstacles lay ahead, they would face them together.

The end.

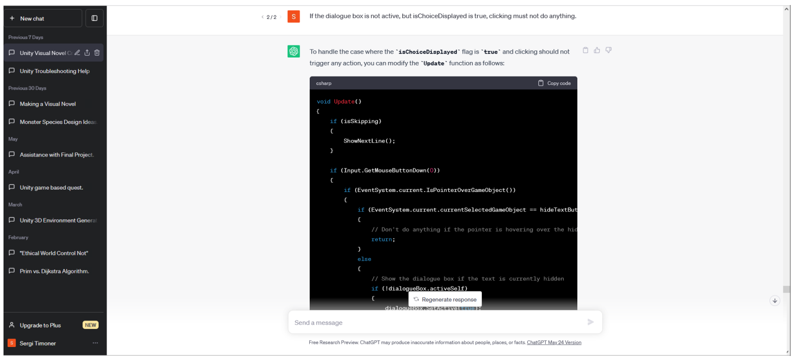
Last request served by AzureBlack6 using 13B-HyperMantis/GPTQ_4bit-128g for 13 kudos in 181.4 seconds.

Figure 5.2: KoboldAI Horde used for producing the script of the game's story

On regards to the code, the convenience of the AI assistant was considerable. Not only it could analyze code and find the errors, but it would suggest corrected alternatives to that code, which solved it successfully most of the time. If it ever generated faulty code, telling it the error message the Unity Editor returned would allow it to pinpoint what it did wrong and amend it. The main challenge was explaining with words, within the prompt, what exactly was the intended functionality of the code. While writing the prompt, the developer has to make sure to use unambiguous language that can't be misinterpreted. The more vague the prompts of the developer are, the most likely that the AI will misunderstand what is being asked of it.

Not long after I ran into this issue, I switched to a different software to produce the text for the game. I used KoboldAI Horde for it. It provides free models which are optimized for narration, and it features a "memory" textbox, which allows the user to remind the AI of important things. It can be adjusted on the fly, which makes it easier to guide the narration in the desired direction. The main issue of this new feature is that, even though the quality of the output is much greater, the speed at which it is produced is much slower, going from ChatGPT 5 seconds of generation to KoboldAI's

120 seconds of generation. Having to wait that long before a prompt gave any results, only for it to be inaproppriate and requiring to start the generation all over added a considerable amount of tedium to the project.



If the dialogue box is not active, but isChoiceDisplayed is true, clicking must not do anything.

To handle the case where the `isChoiceDisplayed` flag is `true` and clicking should not trigger any action, you can modify the `Update` function as follows:

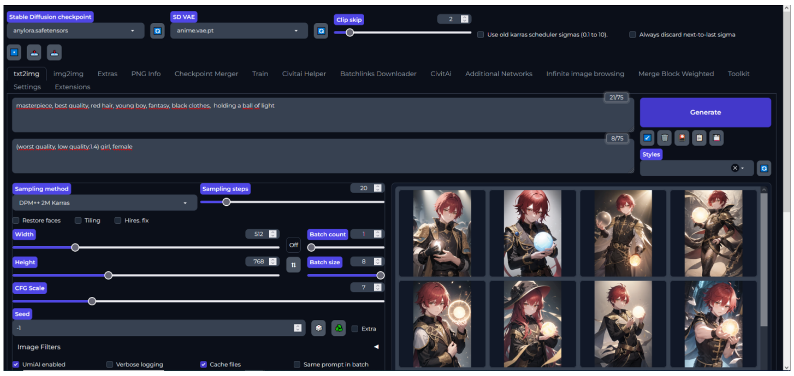
```csharp
void Update()
{
    if (isSkipping)
    {
        ShowNextLine();
    }

    if (Input.GetMouseButtonDown(0))
    {
        if (EventSystem.current.IsPointerOverGameObject())
        {
            if (EventSystem.current.currentSelectedGameObject == hideTextBut
            {
                // Don't do anything if the pointer is hovering over the hid
                return;
            }
            else
            {
                // Show the dialogue box if the text is currently hidden
                if (!dialogueBox.activeSelf)
                {
                    dialogueBox.SetActive(true);
```

Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT May 24 Version

Figure 5.3: ChatGPT used for producing the code of the game's framework

ChatGPT could provide step-by-step instructions on how to use the editor, to make all the processes that didn't rely on coding also be performed more smoothly. If I needed to know how to find an option in the editor, or what kind of component was I required to add, and where, the usual method would have been to invest time searching google for an answer. Being able to ask ChatGPT for help, and it producing a valid answer instantly let me skip on extensive searches through the official documentation, and browsing queries left by other users which had similar issues, but not exactly the same ones as I did.

The incorporation of Stable Diffusion for the procedural generation of text proved to be an effective tool for the project. However, challenges were encountered in the generation of fine details, such as the characters' hands, which often required multiple iterations to achieve the desired result.



Stable Diffusion checkpoint: anylora.safetensors
SD VAE: anime.vae.pt
Clip skip: 2
☐ Use old karras scheduler sigmas (0.1 to 10).
☐ Always discard next-to-last sigma

masterpiece, best quality, red hair, young boy, fantasy, black clothes, holding a ball of light (21/75)

(worst quality, low quality:1.4) girl, female (8/75)

Sampling method: DPM++ 2M Karras
Sampling steps: 20
☐ Restore faces ☐ Tiling ☐ Hires. fix
Width: 512
Height: 768
Batch count: 1
Batch size: 8
CFG Scale: 7
Seed: -1
☐ Extra
☑ UmiAI enabled ☐ Verbose logging ☑ Cache files ☐ Same prompt in batch

Figure 5.4: Stable Diffusion generating the background images that populate the Visual Novel.

As a brief description of how it works, positive and negative prompts are delivered, specifying keywords on what must and must not appear on the image. By setting the batch count to 8, for each prompt 8 images are generated. Once a satsifying one is found, it gets sent to the In-Painting tab, where small edits can be done. For example, selecting a character's hair and prompting "shorter hair, brighter color, red", would fix a hair that is black-colored and too long.

Img2Img is an additional mode that allows creating a new image while using another one as a loose reference.

The CFG parameter below determines how closely the image's result must come to the description provided. The lower it is, the more leeway the AI has to improvise, giving as a result more creative compositions and images.

In the case of In-Painting and Img2Img modes, there is a parameter called Denoising strength, which determines how closely the resulting image must mimic the original image. Setting it at 0, no changes happen. Setting it at 1, the result has nothing to do with the source image, ignoring it entirely. Generally, when I had to use a base image from which to produce a new one, I set CFG Scale at the low value of 7, while keeping the Denoising value between 0.5 and 0.7.

Soundraw, as a service, still seems to be in its infancy. Producing music that fit closely to my description wasn't often possible. Still, it produced a more than adequate result. The soundtrack needed to sound fitting, and the results obtained were deemed good enough, even if they left something to be desired.

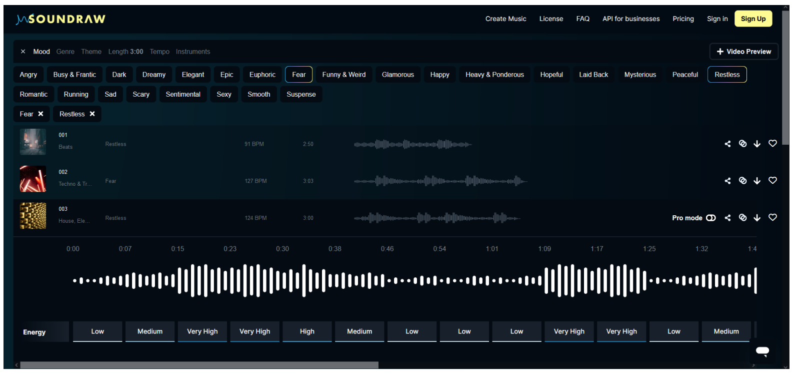


Figure 5.5: Soundraw producing music that plays during the game

Despite these challenges, the overall development process was successful, with the use of AI tools proving to be a valuable asset in the creation of a high-quality Visual Novel. Through careful planning, efficient implementation, and effective problem-solving, the project was able to achieve most of its objectives and overcome any obstacles encountered along the way.

## 5.2 Development Process

After the project I had proposed was given the green light, I created a new 'conversation' with ChatGPT, in which I explained to it my goal to write a story, and my need for it to assist me in the process. It offered a series of ideas for the story, suggesting several different genres. After considering all genres it proposed, I decided to go for an epic fantasy. This caused ChatGPT to provide seven ideas for the story, taking the newly selected genre into account. The one I picked read like "A story where the player takes on the role of a powerful sorcerer who must master their powers and use them to defeat a dark sorcerer who threatens to destroy the kingdom."

It was also the AI's idea to split the story in three acts, each one with a different set of choices. Afterwards, we developed the antagonist and the protagonist, detailing their motivations and the resources each one of them would have. Afterwards, secondary characters were discussed.

Once all the characters had been developed sufficiently, I told ChatGPT to make a list of chapters in which the whole action would be neatly summarized. Then, I had it write chapter 1.

This is where I found my first setback. The intention had been to write the entirety of chapter 1, which would have around 2000 words, with just one generation. Even after I specified I wanted the chapter to be around 2000 words long, ChatGPT tried to wrap up the chapter in an amount of words that was closer to 300-400 words. This caused every event within the story to feel rushed, and there were no instances of dialogue between the characters.

In a visual novel, dialogue is very important, more so than actual narration. So the first few generations were just not good. I asked ChatGPT whether it was capable of making replies of 2000 words, and it confidently replied that it could.

After some investigating, I discovered the problem. ChatGPT can, in theory, give replies with the length requested. In practice, however, it is set to not exceed a set limit of words, so no matter how much one pleads, the reply won't reach the desired length. In short, the AI lied. When asked about its capabilities, it may assure the user it can do something it actually can't do. Likewise, it can claim itself incapable of doing something that it can easily do.

This required a change in strategy. Instead of making the entire chapter in one sitting, it would split the chapter into several parts, and then write each part separately, in order. This worked just fine, at first. Though an issue that was subtle initially, but that would grow more noticeable the longer work progressed began popping up.

Each generation took the narrative in a different direction. there was a loose connection between the two replies, but the narration was disrupted. This could be smoothed over by manual editing, and by telling the AI to repeat the generation after being reminded of what it did wrong.

Guiding the AI so that it would write a compelling narrative was challenging, requiring me to edit constantly parts of the text. After five hours of progress, I concluded the first work session, with a half-finished first chapter, amounting to 1336 words. It became clear that leaving the software to generate the entire story unsupervised was not going to work. It would be a much more involved process.

Over the days of that week, I progressed following the same modus operandi, and eventually arrived at the first milestone of the project. The first choice which separates the main paths of the story. The common part of the story ended at this point.

The choice was made to focus on one path first, and once it was complete, switch to the next. By March 5th, the story had 3460 words, and the story began to take shape.

The amount of generations I had to make at this point to get the story back on track was higher than before. It appears ChatGPT has a limited memory for all that is discussed, which means that the longer that a conversation goes on, the more trouble ChatGPT has to remember the information that was discussed in the beginning. At this point, it began to forget about some characters altogether, or it introduced new ones unprompted.

Eventually, the difficulty to generate coherent text made it look like writing the story by myself without the input of the AI would be faster and produce better results, but that would defy the point of the project.

I went through an intensive revision of the story as it was at that time. The hope was that, if I was able to tweak everything written so far to make it read more consistent, the software would have an easier time understanding everything, and the quality of the text generations would increase.

By the 19 of March, I finished the partial rewrite and had the AI keep going from there. I did notice an improvement, and that same day, the first act of the story following 'Route A' was close to completion.

I decided to begin preparing the framework of the game then, and take a break from the tedium that was generating the story, correcting its mistakes telling it to start over repeatedly.

Creating a new Unity Project, I decided to start a separate conversation with ChatGPT, which would handle the programming aspect of the creation process. ChatGPT, which had struggled making a compelling story, had a surprisingly easy time not only generating the C# scripts, but also giving instructions for me to follow while using the game editor. The instructions were accurate, and so clear that even a beginner could follow them. Occasionally, a code would not work, and by copying and pasting the error code, ChatGPT was able to find out what had gone wrong, and either adjusted the script, or told me where to go in the editor to address the issue.

In a single sitting, an intensive work session lasting twelve hours, there was a functional title screen, a settings menu, and a scene in which the story would play.

The text is contained in '.txt' files on the Resources folder, which the code accesses to read and store each text line, then displays it on the screen. Each click causes the text to advance.

As for the settings menu, at this stage of the creation it allowed to change the game's resolution and switch between windowed and full-screen mode.

With the game's skeleton already working as intended, I returned to the generation of the story, by the 9th of April, Act 1 of path A was finished. Five days later, and after a short break, I added the story script so far into the game's txt files.

Small additions and adjustments took place over the rest of the month. At the start of May, the barebones buttons without graphics from the title screen were outfitted with actual button sprites.

A couple of days were set aside to advance working on the first draft of the current document.

After this, I began implementing non-core functionalities to the game. Quality of life changes that made the experience better for the player. In particular, the button to skip forwards was useful to speed up work on the game. Having to click over one hundred times to progress to the same spot in the story repeatedly was delaying the progress quite a bit. Once the button worked properly, testing the game became much faster and easier. That day, a "hide text" button was also added to the game, and the first art for the game, he background of the title screen, was generated using Stable Diffusion.

With the delivery of the Memory of the project close, I ceased progress on the game to fix the issue the first draft had. On the 22th of May, I resumed work on the story of the visual novel. However, I kept running into the issue of ChatGPT's responses becoming less creative and more nonsensical the further the story advanced. I had run into a wall, and there wasn't much time left to work on the project.

Rather than pushing ahead with chatGPT, I dedicated some time to search for alternatives that could give me better results. While I had kept AI-Dungeon as my plan B, this one wasn't good enough either.

I found KoboldAI Lite, which unlike ChatGPT, it was trained for storytelling and for roleplay purposes. It also had an editable memory field, which made it easier to ensure all the relevant details in each scene were kept into account by the AI. The dialgues too, were consistent and not repetitive.

The downside of this tool is that it was much slower than ChatGPT, having a queue that all the currently online users had to wait through, before it was their turn to get a generation. The generations of text themselves were much shorter, vaguely equivalent to two or three lines of narration, or one lengthy line of dialogue.

But they were of much higher quality, so I endeavored to use this tool for the project from then on. ChatGPT was relegated to work on the programming aspect of the project. Because the transition in narrative style was so abrupt, I realized I had to start over the story from scratch.

I spent 12 hours straight trying to catch up to where I had been so far with the story. And in these, I generated over 4000 words, barely having to discard any of the text that was generated. Contrast that with the 6742 words of the complete "Act 1A" at this point, which took 35 hours to make using ChatGPT.

While discarding over six thousands lines of script felt like a waste, the bump in quality the project received after doing so was worthwhile. And at the rate KoboldAI was going, the decision would pay dividends soon.

But there was one more aspect of the game which had not even been worked on yet: The generation of AI images. In a sense, this is the core of the project, even more so than the text generation is.

The day after the intensive story-writing session, I booted Stable Diffusion from the official Automatic11111 Google Collab, and after loading the AnythingV3 model, I started testing things out.

Making spritesheets was impossible using this tool, though, so I scrapped the idea of the conventional sprites over a background, and decided to make illustrations depicting the actions themselves.

This was done by copying and pasting lines from the story in the '.txt' files into the prompt field. I also added some keywords to distinguish each character. Associating a different hair color to each character was more than enough to get consistent results.

Over the course of the afternoon, 8 images were generated.

By the 3rd of June, I was able to finish all three acts of route B, reaching the first completed ending of the story. Because the final player choice happens right at the end, I quickly finished the second ending before that day was finished.

At this time, I still had the hope that I'd be able to deliver a complete game before the deadline for the first turn of the Final Project's defense, so I increased the hours I dedicated to the game every day, and the progress of the game naturally benefited from that.

On the 4th of july, I made 10 more images, and used KoboldAI to progress further on the story of the second act of route B, trying to get to the 3rd and 4th endings. By the next day, I arrived to the final choice point, and made a brief stop to actually implement the story into the game, including the first and second endings. Once that was done, I compiled and created the first playable prototype, which worked without issues outside of the Unity Editor.

The deadline for the first turn arrived and, while the game was in a state that could be considered playable, it left a lot to be desired. Only two endings, out of the initial eight that had been promised. It was lacking in the game art as well, with only 18 pictures, which barely covered half of the common route of the first act. It lacked features that I had promised in the original project proposal. And it lacked music, as well.

After requesting a shift to the second turn and acquiring it, I set to work to further advance the project into a complete state.

On the 8th of June, I opened Soundraw and recorded the sound of my computer while sending the online Music AI tool the required settings for the genre music I wanted. seven recordings which, after being changed to a mp3 format, were ready to add to the game.

One day later, I finished generating the 3rd ending of the game. The day after that, the 4th one was finished as well. Both of these were added into the game, and I had ChatGPT generate more quality of life features that the game was lacking. The one that took most work to get it done was the Backlog Button, which allowed to view text of lines that had been already passed.

This was handled using a different index pointing always at the line of text immediately before the current one. Two buttons signifying arrows would allow the player to go back or forward. Clicking anywhere other than those buttons would close the backlog and display the current story text again.

Besides that, I added an aesthetic change which caused the name of the character who was talking during dialogue to appear separately from the rest of the text. Each character had a color assignated to them, and whenever they spoke, the text displayed on the text box would change to their color. It bears repeating that this was all code that ChatGPT generated, and I only prompted it to correct itself when it didn't work on the first try. Before I called it a night, on that same session I implemented the 3rd and 4th endings into the game, and expanded the code that displays the text so that each of the images I had made so far would show up exactly in the line they were meant to show up.

The day after, I did the same for the music. Both the images and for the music rely on a couple of nested dictionaries, which stored all the images and music, and associated a text line to each. Whenever the index of the currently selected text is reached, the dictionaries are checked to see if they match with a key associated with an image or a sound file. If there is a match in the music dictionary, the Audio Source object is updated, changing the track. If there is a match in the image dictionary, the background image object is updated, changing it's source image to the new one.

Once I was done adjusting all the images and music tracks to their respective lines, I switched to story generation, finished the 5th ending, the first one that belonged to route A, and created the second playable prototype of the game. That was on the 12th of June.

The day after, I finished the 6th and final ending, making the decision to invest the rest of the time I would have dedicated to make a 7th and 8th ending, into ensuring enough images were made to populate the Visual Novel with it. With that, the story generation part of the game was finished, with a script that is composed of 25379 words. The entire text, when pasted on a word file, is 62 pages long.

From there, I worked on the game non-stop everyday, settling into a routine. Start Stable Diffusion on Google Collab, and spend an entire session creating as many pictures as possible. When the session expired, which took around 5 hours on average, I would dedicate the next few hours to add all the images created into the game. From the 14th of June until the 24th of june, I kept making new images and adding them to the game every day.

The only exceptions were on the 21st and the 22nd, in which the Google Collab session would not load for me. I took this chance to further refine the game's programming, taking on the last big challenge the game had in store for me: Implementing a Saving and Loading system, which would allow the player to continue playing from the same spot they were at last time they booted up the game.

Unlike most of the other features in the game, which I had an idea on how to do even by myself, I had no clue how to add such a thing to a game, having never worked on a project which involved storing data in to a file, which would persist between game sessions. ChatGPT handled it for me, and it turned out it was easier than expected. Just like with the text lines the game loaded, I could store the important variables the game would need to load (such as the current line index, the currently displayed image, the current act/route of the game, and the song currently playing) into a txt file, which the game could read and adjust itself according to the text written in there.

Not only that, I expanded the save system the day after, to allow it to store settings as well. The resolution at which the player sets his window, whether it's in full-screen or not, and, after a quick implementation, the volume slider which determined how loudly the music is meant to sound within the game.

At the time of writing, the game has 103 images generated with AI. They are barely enough to cover the first and second acts of route B. I intend to keep spending every day until the time to deliver this document creating and adding more images to the game.

## 5.3 Results

The initial goal for the visual novel was that there would be three acts, with two branching paths each. This would lead to a total of eight routes, each one with their own unique ending. The script generation was the bottleneck of this project, requiring more work than expected to produce satisfying results. As a consequence of time constraints, the total amount of routes has been lowered to 6. Of these 6, only two routes are completely populated by images.

Still, all the functionalities a typical visual novel may include are present within the project's game, and they are functional too. It becomes proven, then, that using AI-tools to produce simple videogames is viable. It remains to be seen how the AI handles more complex genres, such as RPGs, shooting games, or racing games.

The story that was generated for the Visual Novel talks about a young boy who awakens to magic powers while his village is being attacked. He learns to master his powers, and embarks on a journey to stop the one responsible behind the attack on his village. It is a pretty generic theme for a story, and the narration is not particularly groundbreaking. ChatGPT is a multi-purpose AI, which may not be entirely suited for narration and script-writing.

As of the time of writing this report, open-source, Deep-learning language models have emerged, which specialize in dialogue and narration and achieve a greater immersion than ChatGPT. Future attempts to do the same process I did for this project should yield a greater quality of script, provided those models are used instead of the generic, but highly versatile ChatGPT.

It should also be noted that all the tools were used independently, and then their outputs were brought together manually to produce the game. In the future, dedicated software that integrates all these technologies will become available, and the time spent manually using each one of them should decrease as a result, as well.

Chapter 6

# Conclusions and Future Work

**Contents**

## 6.1 Conclusions

The amount of time, effort and money AI assistants, and automatic asset generation tools will allow developers to save will only keep increasing the more this technology is developed. At a personal level, the procedure I've used on this project will be adopted as my new default way of designing and creating videogames. Developers will find a way to tailor these tools for their own needs, making them more sophisticated and accurate over time. I have no doubt that, if allowed to grow to its full potential, Artificial Intelligence will become a cornerstone that all future developers of software will rely on, not just videogames.

There will still be a market for completely non-AI assisted products, similar to how traditional artists that make paintings on physical canvases and don't reply on image editing software still have their niches, but the majority of developers will sooner or later adopt these tools to make their work easier.

There exists a worry in the industry that content creators and programmers will eventually be replaced, with the average user being able to produce material of similar quality with just a few clicks and typed sentences. However, my experiences using these tools for this project tell me that this won't be the case for a very long time. Code produced by ChatGPT may be functional, but it won't do anything if the person generating it lacks the expertise to know how it works, read it, and apply it where it should be applied. Similarly, art created by AI will be lacking if the user doesn't have enough artistic skills and knowledge to know how to guide it into the desired result. The results generated need to be revised, tweaked and altered, providing only a base from which the creator can work with ease. But the need to actually work on the piece is still there, even if the more tedious parts have been trimmed away from the process.

It is my prediction that, should exercises like this one be repeated once a year, I expect newer software, which will leave the one used for this project entirely obsolete, will keep being produced, and with it, increasingly promising results will be shown. Which is why I'd be glad to see more projects of a similar thematic as this one to be made, comparing their results with mine.

Perhaps the next AI-generated Visual Novel is very close to being published, and it will be interesting to see how much better it is compared to 'Dark Sorcerer', the one that was made in this project.

## 6.2 Future work

As for Dark Sorcerer, the story endings will leave the story open enough to make possible a sequel, or an expansion. Perhaps a new protagonist, exploring the same setting, and learning about the events that took place in the first one, and carrying on the work of the protagonist. Or maybe the protagonist will continue his adventures! I can definitely see a Dark Sorcerer 2 being developed at some point in the future. I may try to repeat the steps followed in this projct in a few years, if only to see how much the status quo of the AI-Developed technology has grow since 2023. I spent roughly 5-6 month making Dark Sorcerer. Maybe the sequel, following a much more perfected pipeline of softwares and AI assistance, will be of much higher quality, and be finished in as little as a month or two.

# Bibliography

APPENDIX A

# Other considerations

## A.1 Bibliography

Software employed during the project:
Unity: https://unity.com/es
ChatGPT: https://chat.openai.com/
Stable Diffusion: https://stablediffusionweb.com/
- Google Colab project I used : https://colab.research.google.com/github/Linaqruf/sd-notebook-collection/blob/main/cagliostro-colab-ui.ipynb (Will probably not work anymore, usage is limited to those with a premium Google Colab account)
- Free crowd-sourced Stable Diffusion hosted by Kobold Horde:
https://aqualxx.github.io/stable-ui/
Soundraw: https://soundraw.io/
KoboldAI: https://lite.koboldai.net/#
Visual Novels referenced as examples in this work:
- The video-game Fate Stay Night. (adventure, visual novel).
https://en.wikipedia.org/wiki/Fate/stay_night
- The video-game Umineko no Naku Koro Ni (mystery, fantasy, visual novel).
https://en.wikipedia.org/wiki/Umineko_When_They_Cry
- The video-game Doki Doki Literature Club! (horror, visual novel).
https://en.wikipedia.org/wiki/Doki_Doki_Literature_Club!

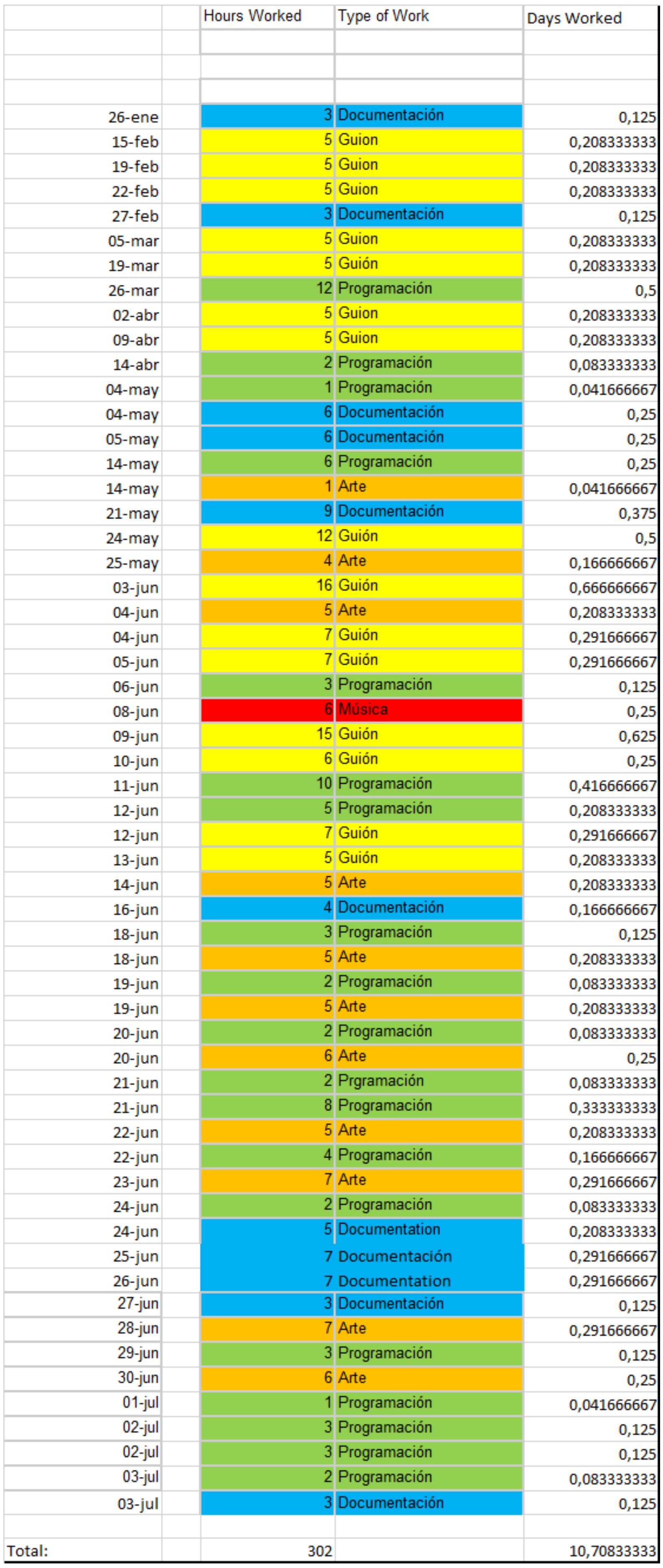
## A.2 Appendices

Developer's diary in Google Docs, detailing the contents of each progress session:

https://docs.google.com/spreadsheets/d/1Zpzj6enpZOnVkrHGUCMy_l2vbaImBtgMxDOjAfHzaCE/edit?usp=sharing

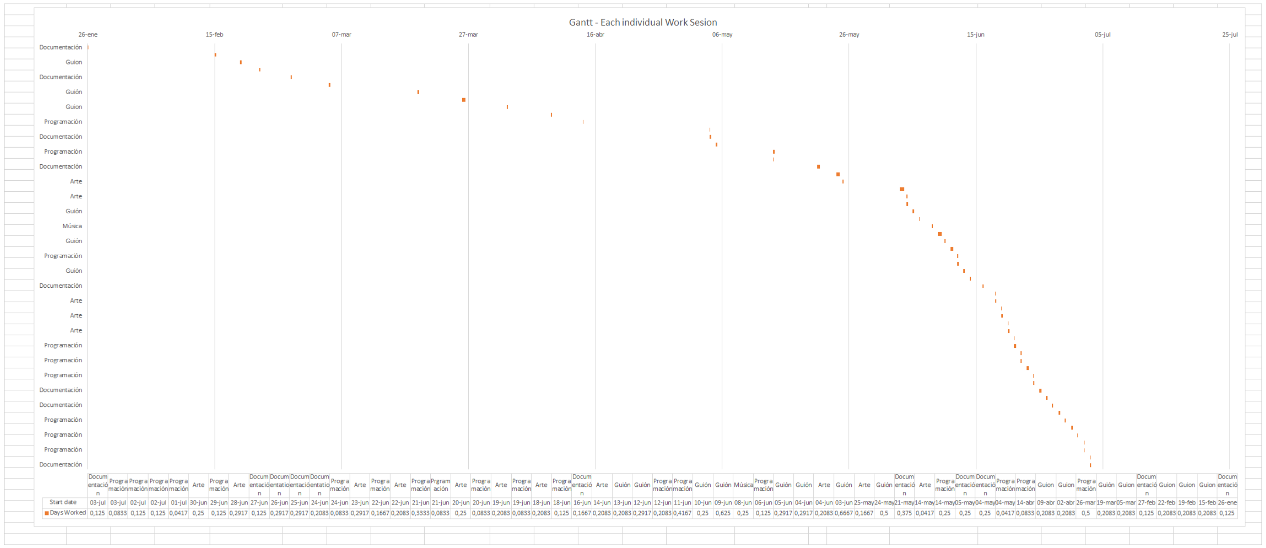
Github repository containing the source code:
https://github.com/sergitim/Dark-Sorcerer-VN/commits/main

A build of the game can be downloaded at the following link:
https://drive.google.com/file/d/1yKSc07MmbJGzAXw8s3B2VQTYgWJ9d1Ds/view?usp=sharing



| | Hours Worked | Type of Work | Days Worked |
|---|---|---|---|
| 26-ene | 3 | Documentación | 0,125 |
| 15-feb | 5 | Guion | 0,208333333 |
| 19-feb | 5 | Guion | 0,208333333 |
| 22-feb | 5 | Guion | 0,208333333 |
| 27-feb | 3 | Documentación | 0,125 |
| 05-mar | 5 | Guion | 0,208333333 |
| 19-mar | 5 | Guión | 0,208333333 |
| 26-mar | 12 | Programación | 0,5 |
| 02-abr | 5 | Guion | 0,208333333 |
| 09-abr | 5 | Guion | 0,208333333 |
| 14-abr | 2 | Programación | 0,083333333 |
| 04-may | 1 | Programación | 0,041666667 |
| 04-may | 6 | Documentación | 0,25 |
| 05-may | 6 | Documentación | 0,25 |
| 14-may | 6 | Programación | 0,25 |
| 14-may | 1 | Arte | 0,041666667 |
| 21-may | 9 | Documentación | 0,375 |
| 24-may | 12 | Guión | 0,5 |
| 25-may | 4 | Arte | 0,166666667 |
| 03-jun | 16 | Guión | 0,666666667 |
| 04-jun | 5 | Arte | 0,208333333 |
| 04-jun | 7 | Guión | 0,291666667 |
| 05-jun | 7 | Guión | 0,291666667 |
| 06-jun | 3 | Programación | 0,125 |
| 08-jun | 6 | Música | 0,25 |
| 09-jun | 15 | Guión | 0,625 |
| 10-jun | 6 | Guión | 0,25 |
| 11-jun | 10 | Programación | 0,416666667 |
| 12-jun | 5 | Programación | 0,208333333 |
| 12-jun | 7 | Guión | 0,291666667 |
| 13-jun | 5 | Guión | 0,208333333 |
| 14-jun | 5 | Arte | 0,208333333 |
| 16-jun | 4 | Documentación | 0,166666667 |
| 18-jun | 3 | Programación | 0,125 |
| 18-jun | 5 | Arte | 0,208333333 |
| 19-jun | 2 | Programación | 0,083333333 |
| 19-jun | 5 | Arte | 0,208333333 |
| 20-jun | 2 | Programación | 0,083333333 |
| 20-jun | 6 | Arte | 0,25 |
| 21-jun | 2 | Prgramación | 0,083333333 |
| 21-jun | 8 | Programación | 0,333333333 |
| 22-jun | 5 | Arte | 0,208333333 |
| 22-jun | 4 | Programación | 0,166666667 |
| 23-jun | 7 | Arte | 0,291666667 |
| 24-jun | 2 | Programación | 0,083333333 |
| 24-jun | 5 | Documentation | 0,208333333 |
| 25-jun | 7 | Documentación | 0,291666667 |
| 26-jun | 7 | Documentation | 0,291666667 |
| 27-jun | 3 | Documentación | 0,125 |
| 28-jun | 7 | Arte | 0,291666667 |
| 29-jun | 3 | Programación | 0,125 |
| 30-jun | 6 | Arte | 0,25 |
| 01-jul | 1 | Programación | 0,041666667 |
| 02-jul | 3 | Programación | 0,125 |
| 02-jul | 3 | Programación | 0,125 |
| 03-jul | 2 | Programación | 0,083333333 |
| 03-jul | 3 | Documentación | 0,125 |
| Total: | 302 | | 10,70833333 |

Figure A.1: Work Diary of the project



| | Documentación | Programación | Programación | Programación | Programación | Arte | Programación | Arte | Documentación | Documentation | Documentación | Documentation | Programación | Arte | Programación | Arte | Programación | Prgramación |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Start date | 03-jul | 03-jul | 02-jul | 02-jul | 01-jul | 30-jun | 29-jun | 28-jun | 27-jun | 26-jun | 25-jun | 24-jun | 24-jun | 23-jun | 22-jun | 22-jun | 21-jun | 21-jun |
| ■ Days Worked | 0,125 | 0,0833 | 0,125 | 0,125 | 0,0417 | 0,25 | 0,125 | 0,2917 | 0,125 | 0,2917 | 0,2917 | 0,2083 | 0,0833 | 0,2917 | 0,1667 | 0,2083 | 0,3333 | 0,0833 |

| | Arte | Programación | Arte | Programación | Arte | Programación | Documentación | Arte | Guión | Guión | Programación | Programación | Guión | Guión | Música | Programación | Guión | Guión |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Start date | 20-jun | 20-jun | 19-jun | 19-jun | 18-jun | 18-jun | 16-jun | 14-jun | 13-jun | 12-jun | 12-jun | 11-jun | 10-jun | 09-jun | 08-jun | 06-jun | 05-jun | 04-jun |
| ■ Days Worked | 0,25 | 0,0833 | 0,2083 | 0,0833 | 0,2083 | 0,125 | 0,1667 | 0,2083 | 0,2083 | 0,2917 | 0,2083 | 0,4167 | 0,25 | 0,625 | 0,25 | 0,125 | 0,2917 | 0,2917 |

| | Arte | Guión | Arte | Guión | Documentación | Arte | Programación | Documentación | Documentación | Programación | Programación | Guion | Guion | Programación | Guión | Guion | Documentación | Guion | Guion | Guion | Documentación |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Start date | 04-jun | 03-jun | 25-may | 24-may | 21-may | 14-may | 14-may | 05-may | 04-may | 04-may | 14-abr | 09-abr | 02-abr | 26-mar | 19-mar | 05-mar | 27-feb | 22-feb | 19-feb | 15-feb | 26-ene |
| ■ Days Worked | 0,2083 | 0,6667 | 0,1667 | 0,5 | 0,375 | 0,0417 | 0,25 | 0,25 | 0,25 | 0,0417 | 0,0833 | 0,2083 | 0,2083 | 0,5 | 0,2083 | 0,2083 | 0,125 | 0,2083 | 0,2083 | 0,2083 | 0,125 |

Figure A.2: Gantt Chart depicting the length of each work session