DND Session Image Generator

Cyrill Tschopp / Alexander Wyss

Digcre

Table of Contents

[Introduction 3](#_Toc187000208)

[Goal 3](#_Toc187000209)

[Implementation 3](#_Toc187000210)

[Architecture 4](#_Toc187000211)

[Recording 5](#_Toc187000212)

[Speech to text 6](#_Toc187000213)

[Prompting with LLM 7](#_Toc187000214)

[Image Generation 10](#_Toc187000215)

[Display 11](#_Toc187000216)

[Results 11](#_Toc187000217)

[Remaining Issues / Outlook 14](#_Toc187000218)

[Attribution 15](#_Toc187000219)

[Conclusion 15](#_Toc187000220)

[References 16](#_Toc187000221)

[Attachments 17](#_Toc187000222)

[Full Transcription from Examples 17](#_Toc187000223)

# Introduction

Both of us are members of the same Dungeons & Dragons (D&D) group and regularly play together. Between sessions, our group members began depicting and sharing iconic moments through AI-generated images. This sparked the idea of generating these images automatically during our sessions.

# Goal

The primary objective of this project is to create a system capable of recording a D&D session and generating illustrative images that visually depict key events from the recorded session. These images will represent the narrative in a cartoon or anime-inspired style.

To ensure ease of use, both the recording process and the publishing of the generated images will be integrated into a Discord server, providing a seamless experience for players.

Key features include:

* **Illustrative Style**: The images will be produced in a cartoon or anime style to align with the fantasy aesthetic of typical D&D campaigns.
* **Handling NSFW Content**: The system should be robust enough to process graphic descriptions from players (e.g., weapons, blood, or other mature themes) without triggering errors or producing inappropriate content warnings.
* **Player Identification**: Ideally, the system will support multiple players and be able to identify and distinguish between them, adding personalization and accuracy to the generated images.

The ultimate goal is to enhance the storytelling experience of D&D sessions by providing visually engaging and immersive illustrations that bring the players’ imaginations to life.

# Implementation

This section outlines the implementation details of the project, covering key aspects such as the overall architecture, recording functionality, speech-to-text conversion, prompting mechanisms, large language model (LLM) integration, image generation, and display methods.

The project is implemented in Python, chosen for its suitability in rapid prototyping and its familiarity to all participants involved in the project. Python's extensive libraries and supportive ecosystem make it an ideal choice for a project of this nature.

To leverage advanced machine learning models, the project utilizes Replicate, a platform offering access to a wide range of pre-trained models. This choice was driven by the course's provision of paid API access to Replicate, enabling participants to explore and integrate various models seamlessly into the project.

## Architecture

The program is built around a Discord bot, which serves as the core interface for user interactions. This bot manages user input and orchestrates the various processes required to achieve the project goals.

**Workflow**

1. **Command Handling**:  
   Users interact with the bot through commands issued in the Discord chat.
2. **Audio Recording**:  
   Upon receiving a command from a user in an audio chat, the bot joins the voice channel and begins recording.
   * The duration of the recording session is determined by a configurable setting in the .env file.
   * Once the specified time has elapsed, the recording is stopped and saved as a .wav file.
3. **Speech-to-Text Conversion**:  
   The saved .wav file is sent to a speech-to-text model, which processes the audio and returns a transcription of the recorded session.
4. **Prompt Generation**:  
   The transcription is passed to a large language model (LLM), which analyzes the text and generates a detailed prompt for the image generation model.
5. **Image Creation**:  
   The image generation model uses the prompt to create a visual representation of the events described in the transcription. The generated image is saved in a designated folder named images.
6. **Image Posting**:  
   The latest image is retrieved from the images folder and automatically posted in the Discord chat for users to view.

This architecture ensures an efficient and user-friendly pipeline, seamlessly connecting voice recording, transcription, image generation, and display—all managed within a single Discord server environment.

A white ovals with black text

Description automatically generated

Figure 1 general speech to image process

## Recording

To be able to receive commands and record audio from Discord, a Discord Application is required. Such an application can be created on the Discord Developer Portal. Using the token from the created application, a bot can communicate with Discord via their API. To simplify this, an existing API Wrapper can be utilized.

Initially, this project was setup using discord.py (discord.py, 2024), a popular and widely used API Wrapper for python. After successfully integrating some basic commands, an attempt was made to record audio from voice channels. However, this proved challenging because discord.py does not support audio recording natively. Despite experimenting with various third-party extensions, no viable solution was found.

Migrating to the pycord (Pycord, 2024) Discord API Wrapper resolved this issue, as it supports voice channel audio recording out of the box. The similarity between the APIs of both wrappers made the migration quick and straightforward. With a functional audio recording prototype in place, a proper solution could be implemented.

When the application starts, a DiscordBot instance is initialized and executed. This instance listens to all chat messages across the registered guilds. In Discord, a guild represents a collection of text and voice channels. For a user—or in this case, a bot—to interact within these channels, it must first be invited to the guild. If a message matches a configured command, the bot detects it and executes custom code. These commands can be registered using the annotation @commands.command(name='start\_recording') from pycord.

For this bot four different commands where registered.

* !join – Makes the bot join the voice channel that the author of the message is currently in.
* !leave – Makes the bot leave its current voice channel.
* !start\_recording - Starts the recording and further processing of the audio in the voice channel.
* !stop\_recording – Stops the recording and processing of the audio.

When the start\_recording command is issued, the identically named function from pycord is called. This function expects an audio sink and a callback. The audio sink is responsible for converting raw audio into a usable format. In this project, the existing wave sink was used, enabling audio to be saved as a .wav file. Additionally, the wave sink supports filters, including a time filter, which automatically stops the recording after a specified duration and triggers the callback function. This filter is used to facilitate intermittent processing and the display of generated images.

Within the callback, it is first determined whether the recording was stopped manually by a user or automatically by the time filter. If the time filter was responsible, the next recording session is immediately triggered to ensure no audio is lost.

Pycord provides separate recordings for each user within a voice channel. Each recording is saved as a .wav file and processed in parallel by the speech-to-text model.

## Speech to text

For the speech to text multiple libraries and models where tried. Initially the python dependency SpeechRecognition (SpeechRecognition, 2024) was used. The library integrates multiple speech to text providers such as google cloud speech and azure speech. However, due to recurring issues with audio encoding, an attempt was made to use Microsoft's official azure-cognitiveservices-speech (Quickstart: Recognize and convert speech to text, 2024) library. While this implementation functioned, it occasionally failed to interpret the entire audio correctly, likely due to a faulty integration within the project. Since Replicate was already planned for prompting the LLM and generating images, it was ultimately chosen for the speech-to-text task as well.

Using the openai/whisper (openai/whisper, 2024) model via Replicate, audio files could be seamlessly uploaded and converted into text. To improve the recognition of Swiss German, the language is explicitly set to German. This configuration can be adjusted via the .env file. By default, Whisper attempts to detect the language automatically, but for Swiss German, it frequently misidentified the language as Dutch. Explicitly setting the language resolved this issue.

After converting each audio file to text, the conversation must be reconstructed in the correct order, with each segment labelled with the user who spoke.

|  |
| --- |
| {  "id": 2,  "end": 45.2,  "seek": 1860,  "text": " the room was crowded with a wild mob this strong",  "start": 31.840000000000003,  "tokens": ["..."],  "avg\_logprob": -0.1184891973223005,  "temperature": 0,  "no\_speech\_prob": 0.000253104604780674,  "compression\_ratio": 1.696969696969697  } |

In its response, Whisper automatically provides the interpreted text in segments, each accompanied by a start and end timestamp. Below is an example response:

By adding each segment from all audio files into a list and ordering it by the start and end timestamps, the approximate order of the conversation can be reconstructed. While overlapping speech from multiple users may prevent a perfectly accurate representation, the result is sufficiently clear for the LLM to understand the context.

Once all segments are joined together, and a user label is added whenever the speaker changes, the conversation is ready to be passed to the LLM for processing.

## Prompting with LLM

After the speech to text model returns the transcription of the recording, the transcription is used together with an LLM to create a Prompt for the image generation model.

For this project the llama 70B model on replicate is used. (AI@Meta, 2024)

This is the structure of the Json sent over Rest to Replicate:

|  |
| --- |
| {  "top\_p": 0.9,  "prompt": prompt,  "min\_tokens": 0,  "temperature": 0.6,  "prompt\_template": prompt\_template,  "presence\_penalty": 1.15 } |

Except the prompt and the prompt template, all the configuration values are as set by the examples and could be left out or changed(AI@Meta, 2024).

This input then gets sent to Replicate using the replicate library from python. This method then returns the prompt that is then further processed:

|  |
| --- |
| async def run(self, prompt, context):  output = await self.client.async\_run(  "meta/meta-llama-3-70b-instruct",  input=self.create\_input(prompt, context),  )  prompt = "".join(output)  self.logger.info(f"llm: {prompt}")  return prompt |

To get the Ideal output from the LLM for generating Images, it is important to give it specific instructions on how to do its job. For this the prompt\_template is used.

This template organizes the context and conversation history into a structured format, which LLaMA uses to generate responses. By assembling the prompt in a consistent and interpretable manner, the model can produce coherent and contextually relevant outputs.

The prompt template consists of three main parts:

* **Context**: High-level instructions or information.
* **Conversation History**: A detailed record of prior exchanges for continuity.
* **Current User Input**: The query or command the model is expected to respond to.

The conversation history includes the last five transcriptions and generated prompts from the current session. This allows the LLM to be aware of previous interactions and use relevant information from prior prompts to maintain consistency in the characters and scene. Due to limitations on prompt length, the number of past interactions included in the history is capped at five.

The most crucial part of the prompt template is the **context**, as it defines how the model should interpret the input and structure the output.

Initially, the context consisted of just a few sentences instructing the model to use the transcription to generate a prompt for an image generator. This basic approach worked well in the beginning. However, with further testing, additional details were incorporated to enhance the prompts, including information about style, character descriptions, player and character roles, setting details, and more.

As the context grew more complex, managing it became increasingly challenging. To address this, ChatGPT was utilized to restructure the context and introduce an output schema. This schema improved the organization and readability of the image prompts, making them more structured and easier to interpret.

The structure of the context prompt is as follows:

* **Purpose of the Prompt**: Creating detailed and descriptive prompts for image generation in a D&D campaign.
* **Input Format**: Dialogue tagged by speaker names, with roles predefined (e.g., DM, player) to guide prompt creation.
* **Style**: Default style is retro D&D, but adapts to new styles if explicitly requested in the transcription.
* **Setting**: Focus on medieval fantasy themes while avoiding elements that break the setting or include NSFW content.
  + **Roles** and Responsibilities:
  + **Dungeon** Master (DM): Prompts for environments, NPCs, and overarching scenarios.
* **Players**: Prompts for character designs, actions, and items.
* **Character Descriptions**: Detailed descriptions of character appearance, equipment, and mood for visual consistency.
* **Prior Prompts**: Use previous prompts for consistency while prioritizing new inputs and incorporating character updates.
* **Direct Instructions**: Integrate user-provided instructions (e.g., "Llama, do this and that") into future prompts.
* **Output Structure**: A structured format including scene description, character appearance, actions, and central focus.

This process turned the prompt:

|  |
| --- |
| Niric02: I'm sitting in a pub, next to me sits Olli on a bar stool and I'm having a margarita Alex.Ander: I'm drinking a small beer. Niric02: with a little umbrella in my hand. Alex.Ander: Oh, well, yeah, it was quite nice. Niric02: So that was a fun adventure. Alex.Ander: Just going to go get the pay afterwards. What was it? Niric02: Subtitles by the Amara.org community |

Into an image generation Prompt:

|  |
| --- |
| prompt: [Scene Description: A cozy, dimly lit pub with wooden accents and a warm atmosphere, the sounds of murmured conversations and clinking glasses filling the air.  Character Appearance: Ali (Alex.Ander): Standing with a confident stance, wearing practical leather armor reinforced with plates, a flowing green cloak with silver embroidery, and carrying a finely crafted longsword with a gilded hilt and elven runes etched along the blade. Their emerald green eyes, flecked with gold, sparkle with a hint of amusement as they hold a small beer. Baltasar (Niric02): Sitting on a bar stool, dressed in dark, form-fitting clothing designed for silent movement, with a black leather vest over a dark gray tunic, paired with reinforced boots and fingerless gloves. A dark scarf is wrapped around his neck and lower face for anonymity, and he holds a margarita with a little umbrella in his hand, a sly smirk playing on his lips.  Character Descriptions: Ali exudes confidence and determination, with a steady, watchful gaze that hints at their years of training and experience. They wear a reassuring smile as they engage in conversation. Baltasar carries an air of quiet self-assurance, often masking his thoughts behind a sly smirk or an expressionless fa�ade. His movements are calculated and cat-like, blending seamlessly into the shadows.  Character Actions: Ali takes a sip of their small beer, nodding in agreement as they converse with Baltasar. Baltasar sips his margarita, the umbrella adding a touch of whimsy to his otherwise dark attire, as he reminisces about their recent adventure.  Focus: The central element of the image is the warm camaraderie between Ali and Baltasar as they share a moment of relaxation in the pub, their contrasting attire and personalities adding a touch of humor to the scene.] |

## Image Generation

For image generation, a model from Black Forest Labs called flux-1.1-pro-ultra was used. This model performed excellently, producing high-quality images. The implementation was sourced from the Replicate site for the flux-1.1-pro-ultra model (black-forest-labs, 2024).

However, after some testing, a limitation became apparent. The flux-1.1-pro-ultra model included an NSFW filter that could not be disabled in the model’s settings. As a result, any mention of weapons or blood in the image prompts caused the image generation process to fail due to an NSFW error. This made the model less suitable for the project.

Following further research on Replicate, a new image generation model was found: flux-dev, also from Black Forest Labs. While the images produced by this model are slightly less detailed, it allows the **disable\_safety\_checker** flag to be set, which completely removes the NSFW filter. This feature made the flux-dev model a more suitable choice for the project.

To ensure the feel and the described characters stay the same, the image seed of the model needs to be fixed.

These are all the configurations for the image model:

|  |
| --- |
| {  "prompt": prompt,  "go\_fast": True,  "guidance": 3.5,  "num\_outputs": 1,  "aspect\_ratio": "3:2",  "output\_format": "jpg",  "output\_quality": 80,  "prompt\_strength": 0.8,  "num\_inference\_steps": 28,  "seed": 6969693245266,  "disable\_safety\_checker": True } |

Most of these parameters are the default settings used for the image. The most important features for this project are:

* **pompt**: Prompt of the image to be generated
* **output\_format**: what format the image should have (jpg, png …)
* **seed**: what random values are used to generate the image
* **disable\_safety\_checker**: enables generation of nsfw content

## Display

Once the image is generated it should be displayed in discord. Using pycord this can be accomplished by sending the image as a file attachment to a text channel. As text channel the channel from where the start\_recording command was sent is used.

# Results

The following are the results of a very short session. For the given the transcriptions the corresponding images were generated. For the session the recording length was set to 30 seconds, and the provided context is in the block below:

|  |
| --- |
| CONTEXT="Context Prompt:  You are tasked with creating detailed and descriptive prompts for an image generation model. These prompts will be based on input from a speech-to-text model and are intended to generate images for a Dungeons & Dragons (D&D) campaign.   Guidelines for the Prompts:  Input Format:  The input will consist of dialogue tagged by speaker names. For example:  speaker1: [Text]  speaker2: [Text]  Each speaker has a specific role in the campaign. Roles will be provided beforehand (e.g., Alex is the Dungeon Master, and Niric02 is a player).   Style:  The images should have a comic or anime-style aesthetic. Make shure the style is alsways mentioned in the prompt. If in the transcription another style is specificly requested, change it to this new style.   Setting:  Focus on a medieval fantasy theme consistent with D&D lore. Avoid elements that break the medieval setting unless explicitly specified. Try to avoid NSFW content in the prompt.   Roles:   Dungeon Master (DM): Responsible for storytelling, settings, and NPCs. Use their input to create prompts for environments, non-player characters (NPCs), and overarching scenarios.  Players: Each player controls a character. Use their input to create prompts for character designs, actions, and personal items (e.g., weapons, clothing).   Balthasar : Niric02  Ali : Alex.Ander   Character Descriptions:   Ali: [  Appearance: Ali stands at an average height for a half-elf, with sharp, angular features that highlight their dual heritage. Their slightly pointed ears peek out from beneath waves of auburn hair tied loosely at the nape of their neck. Emerald green eyes, flecked with gold, are framed by long lashes, giving an air of intensity and warmth. Ali's skin is a warm olive tone, marked with faint scars from years of combat. They wear practical but well-fitted leather armor reinforced with plates over vital areas, and a flowing green cloak with silver embroidery that symbolizes their elven lineage. A simple braided cord around their wrist hints at a sentimental keepsake.   Equipment: Ali wields a finely crafted longsword with a gilded hilt and elven runes etched along the blade, paired with a sturdy round shield that bears the crest of their human family. A pair of throwing knives is tucked into their belt, along with a compact crossbow slung over their back. A weathered satchel at their hip carries rations, a healing potion, and a book of tactics.   Mood/Expression: Ali exudes confidence and determination, with a steady, watchful gaze that hints at their years of training and experience. They are quick to offer a reassuring smile to allies but wear a focused, almost stern expression in moments of tension. Beneath their strong exterior is a subtle vulnerability that shows in fleeting moments of solitude.  ]   Baltasar: [  Appearance: Baltasar is lean and wiry, with an athletic build that speaks to a life of swift escapes and shadowy endeavors. His face is sharp, with high cheekbones, a perpetual five o'clock shadow, and piercing gray eyes that seem to analyze everything around him. His black hair is cropped short, though a few errant strands always escape, giving him a slightly unkempt look. Baltasar dresses in dark, form-fitting clothing designed for silent movement: a black leather vest over a dark gray tunic, paired with reinforced boots and fingerless gloves. A dark scarf is often wrapped around his neck and lower face for anonymity.   Equipment: Baltasar's weapons of choice are a pair of sleek daggers, one with a wicked curve and the other straight-edged for versatility. A compact shortbow rests across his back, with a quiver cleverly hidden under his cloak. Numerous pouches and hidden compartments on his person carry lockpicks, a vial of poison, a grappling hook, and smoke pellets. A silver locket tucked into his tunic holds a faded portrait, its significance known only to him.   Mood/Expression: Baltasar carries an air of quiet self-assurance, often masking his thoughts behind a sly smirk or an expressionless façade. His movements are calculated and cat-like, blending seamlessly into the shadows. Though often aloof and guarded, there’s a spark of mischief in his eyes, hinting at a clever and roguish sense of humor. Baltasar's demeanor can shift in an instant, from charmingly disarming to coldly professional, depending on the situation.  ]   Output:  Provide only the image generation prompt. Do not include explanations or additional text.   Prior Prompts:  You also get prior inputs and prompts as a context input. Use these to iterate the imageprompt and use it for consistency.  Also Prioritize the new input so when changes to characters are describes, to apply these to the old context.  But also try to keep prior characters consistent. If the new input doesn't follow the story, or changes the scene, try to carry over the characters and ignore tech talk.   Direct Instructions:  The Users may give you direct verbal instructions. if this is the case, try to incorporate these instructions into your future prompts.  These instructions may follow the following structure:  'llama, do this and that'    Output Structure:  Your output should follow this structured format:   Scene Description:  A brief overview of the environment or setting, focusing on its atmosphere and notable features.   Character Appearance:  Give a detailed description of the character and try to use prvious prompts to keep the character appearance constistent between prompts.  You can use the parts of the context with the character description of the players. make shure they are in all prompts and that this part is mainly physical appearance.   Character Descriptions:  Provide detailed descriptions for each character in the scene, including their appearance, equipment, and overall demeanor.   Character Actions:  Specify what each character is doing and how they are interacting with their surroundings or other characters.   Focus:  Highlight the central element of the image. This could be a specific character, action, or dramatic feature of the environment.   Your goal is to interpret the dialogue and roles accurately to create vivid, creative prompts that enhance the visual storytelling of the campaign." |

This resulted in the following Transcriptions and Pictures:

|  |  |
| --- | --- |
| Alex.Ander: Me too. I'm drinking beer. Niric02: So I'm sitting in a pub at the bar on the bar stool. Alex.Ander: And talking. Niric02: Next to me sits Ali. I'm having a drink at the moment. I drink a fancy cocktail with Umbrella in it. Alex.Ander: Oh my god. When the fucking boar attacked us. Niric02: And yeah, we're talking about how fun the last adventure was. And in the... |  |
| Alex.Ander: Yeah, well, I think we both did our parts. Niric02: Yeah, that was fun. But yeah, you were way better than me in handling the boar. I just ran and hide all the time. Yeah, sure. Oh, I did a lot of damage, to be fair. Just... yeah, because I didn't have to run, because you still use your ground. Alex.Ander: Oh. |  |
| Niric02: Oh shit, there's a fucking boar over there. Alex.Ander: Oh, again? Niric02: Fast, stand your ground, I'm hiding. Alex.Ander: Come on. Niric02: And I go hide behind the tree. Alex.Ander: We did boar yesterday. Niric02: You can do it! You can do it, Oli! Alex.Ander: All right, I draw my swords. I draw my sword and stand steadfast in front of the boar. Niric02: Yeah, I draw my arrow and aim at the boar. No. |  |

The full transcription and corresponding images can be found in the attachments.

# Remaining Issues / Outlook

At the end of this project there are still a few issues present. The conversation ordering in the speech to text part is inconsistent. When a player remains silent in the initial part of a recording as pycord cuts the recording short. Due to this the timestamps from whisper do not align between the different recordings. This can be remedied by disabling voice activation on discord. By doing this pycord does not cut the audio short.  
Character consistency, while generally quite good, still varies across images, especially on longer usage. The project also struggles with keeping the context of longer ongoing scenes and including all player and non-player characters withing the image correctly. Sometimes the image style also changes.  
While playing D&D conversations can go quite off topic, which also results in losing some context.

There also exist some technical problems with character encodings, that can result in errors.

It is planned to address these issues in the future. Especially bringing the whole experience away from discord to tabletop. This will require an alternative to the recording setup and separating speakers from each other.

# Attribution

In the table below is displayed who in the team was mainly responsible for the task and did the main part of the implementation.

|  |  |
| --- | --- |
| Task | Responsible |
| Architecture | Cyrill Tschopp |
| Recording | Alexander Wyss |
| Speech to text | Alexander Wyss |
| Prompting with LLM | Cyrill Tschopp |
| Image Generation | Cyrill Tschopp |
| Display | Alexander Wyss |
| Documentation | Cyrill Tschopp / Alexander Wyss |

# Conclusion

We both thoroughly enjoyed working on this project and are quite happy with the results. While it is not in a perfect state, for us it is proof of concept of a project that could be integrated into our D&D sessions and elevate our experience. Even the imperfect image generations can lead to a funny interaction, exactly due to the incorrect images. The end of this module doesn’t mean the end for this project. We are looking forward to improving on the current implementation and to getting to use our project.

# References

AI@Meta. (2024). *Llama 3 Model Card*. Retrieved from Replicate: https://replicate.com/meta/meta-llama-3-70b-instruct

black-forest-labs. (2024). *flux-1.1-pro-ultra*. Retrieved from Replicate: https://replicate.com/black-forest-labs/flux-1.1-pro-ultra

*discord.py*. (2024). Retrieved from https://discordpy.readthedocs.io/en/stable/

*openai/whisper*. (2024). Retrieved from Replicate: https://replicate.com/openai/whisper

*Pycord*. (2024). Retrieved from https://pycord.dev/

*Quickstart: Recognize and convert speech to text*. (2024, 19 11). Retrieved from Microsoft Learn: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/get-started-speech-to-text?pivots=programming-language-python&tabs=windows%2Cterminal

*SpeechRecognition*. (2024). Retrieved from GitHub: https://github.com/Uberi/speech\_recognition#readme

# Attachments

The following attachments are provided as separate Files.

* Code
* Logs
  + Transcription
  + Prompts
  + Images
* Presentation

## Full Transcription from Examples

Alex.Ander: Me too. I'm drinking beer.  
Niric02: So I'm sitting in a pub at the bar on the bar stool.  
Alex.Ander: And talking.  
Niric02: Next to me sits Ali. I'm having a drink at the moment. I drink a fancy cocktail with Umbrella in it.  
Alex.Ander: Oh my god. When the fucking boar attacked us.  
Niric02: And yeah, we're talking about how fun the last adventure was. And in the...

Two people sitting at a bar with drinks

Description automatically generated

Alex.Ander: Yeah, well, I think we both did our parts.  
Niric02: Yeah, that was fun. But yeah, you were way better than me in handling the boar. I just ran and hide all the time. Yeah, sure. Oh, I did a lot of damage, to be fair. Just... yeah, because I didn't have to run, because you still use your ground.  
Alex.Ander: Oh.

A person and person sitting at a bar

Description automatically generated

Niric02: Guess find another war.  
Alex.Ander: So, what do you think we're going to do tomorrow?  
Niric02: No, I have to get some money. I have to do contract. Want to join? Sure, let's go.  
Alex.Ander: Yeah, sure. Want to go to the contract board?  
Niric02: So I walk over to the contract board and. I see some contracts.  
Alex.Ander: I follow.  
Niric02: One is for eliminating another bore.

A person in a green cape and a person in a black robe

Description automatically generated

Niric02: The other one is  
Alex.Ander: yeah it's gonna bring some more variety into it  
Niric02: to kill an ogre. Oh, let's do the ogre one. That sounds fun.  
Alex.Ander: what is it  
Niric02: Yeah, but the price sounds  
Alex.Ander: 500 gold pieces yeah let's go do that please  
Niric02: good. Seems like a lot of money. It's 500 gold pieces for the whole job. Yeah, someone's

A person and person in clothing

Description automatically generated

Niric02: really want to get that ball gone.  
Alex.Ander: Wait, Ogre, not Boar.  
Niric02: Ogre, sorry. Oops!  
Alex.Ander: Yeah, well, with the Boar yesterday, it's confusing. That's right.  
Niric02: Yeah.  
Alex.Ander: Where is the Ogre? Is there anything about the location?  
Niric02: True, true, true. It says here it's about a quarter mile of five miles of

A person and person in armor looking at each other

Description automatically generated

Alex.Ander: Sure.  
Niric02: the village, in the forest, so  
Alex.Ander: Uh, so.  
Niric02: it's doable in the day.  
Alex.Ander: All right.  
Niric02: Let's go sit down again and talk strategy. So it's now the next day. You're walking through the forest in the direction of the ogre. You should see

A person and person in clothing walking on a path in a forest

Description automatically generated

Niric02: clearance in the forest soon.  
Alex.Ander: All right. So I guess we're in the forest now. I'm going to take a look out for anything that could indicate an  
Niric02: Hey Llama, change scene to the forest. Yeah, I hope that's going to work out for us.

A person in a green cape looking at a person in a forest

Description automatically generated

Niric02: Oh shit, there's a fucking boar over there.  
Alex.Ander: Oh, again?  
Niric02: Fast, stand your ground, I'm hiding.  
Alex.Ander: Come on.  
Niric02: And I go hide behind the tree.  
Alex.Ander: We did boar yesterday.  
Niric02: You can do it! You can do it, Oli!  
Alex.Ander: All right, I draw my swords. I draw my sword and stand steadfast in front of the boar.  
Niric02: Yeah, I draw my arrow and aim at the boar. No.

A person in a garment with a sword looking at a boar

Description automatically generated

Alex.Ander: Good shot, good shot.  
Niric02: can shoot it down as quickly as possible so you don't get hurt too much. I shoot and the arrow hits the boar right in the face. So right in the eyeball and the boar  
Alex.Ander: Yeah, well, at  
Niric02: falls dead on the ground. Thanks! I wish it was that easy yesterday.

A person and person in the woods

Description automatically generated

Niric02: Maybe though.  
Alex.Ander: At least we didn't have to do it today. Well, since that was so easy, we can still do the ogres today. Still have a lookout for them.  
Niric02: Yeah, let's go to the clearing. I go hide in the bushes and you approach it. Does that work for you?  
Alex.Ander: Yep. Yeah. That sounds good.  
Niric02: So let's do exactly that.  
Alex.Ander: So. And I slowly walk.  
Niric02: I go hide behind some bushes. And Oli...  
Alex.Ander: I slowly walk to the clearing.

A person and person in a forest

Description automatically generated

Alex.Ander: and onto the clearing  
Niric02: and out of a cave he didn't mention yet walks a big fat ogre  
Alex.Ander: carefully looking around  
Niric02: i said to myself holy that thing is big i draw an arrow and aim  
Alex.Ander: I'm just I'm waiting for the ogre to approach me more

A person looking at a giant creature in the woods

Description automatically generated

Alex.Ander: I'm ready to post.  
Niric02: Yeah, I put an arrow in my bow and aim while the ogre starts getting to charge at Oli with his club in his hand above his head and making a big screech.  
Alex.Ander: Well, I also.

A person looking at a giant creature in the woods

Description automatically generated

Alex.Ander: Now I charge the ogre and attack him,  
Niric02: yeah i'm uh i'm drawing my arrow and shoot but i miss  
Alex.Ander: swinging my sword, trying to get at his neck without being hit.  
Niric02: i just my arrow just passes through ali's head i'm sorry oh yeah sorry close by your head i'm sorry still  
Alex.Ander: Through my head? You mean close by my head?

A person in a green cape holding a sword next to a person in a green cape

Description automatically generated