FINAL PROJECT REPORT

Team MCS08
Member:
Yeoh Ming Wei (32205449)
Yew Yee Perng (32205481)
Toh Xi Heng (33200548)

Word Count:10513

Table of Contents

1 Introduction

1.1 Report Introduction

As our final year project has been concluded, this report aims to cover all the contents that we had done throughout 24 weeks of effort, including research, development and outcome for our project. Moreover, we will briefly explain our methodology and deliverables of software development and finally draw a conclusion to summarize our project.

1.2 Project Introduction

As mentioned in our project proposal previously, our project is titled “Singing Video Generation with Music Separation” that aims to develop an advanced model that is able to generate high quality human singing face videos with lively face expression and head movement with the provided music. This was a focus of research on developing a talking face generation due to the demand of realistic virtual avatars and interactive interfaces especially inside the entertainment industry. So, the objective of this project is to continue the research on finding and proposing a valid solution to solve the problem of generating human talking face video.

These are the specific aims of the project which includes:

- Perform an in-depth research on developing a model
- Brainstorm various approaches to tackle problems that are caused by the generation, such as lip synchronization, head movement and emotion expression.
- Create a robust application for improvise human talking face generation based on the solution proposed

2 Project Background

2.1 Introduction to Project

In this modern era, the demand to bring still images to life has arisen driven by the popularity of social media and video creations with the power of generative AI. There are factors contributing to the growth of these video generations. Firstly, users can personalize their content by adding realistic voices and moments to their images which allows them to share their experiences or memories in a more unique and dynamic way. Besides that, it also allows the users to share meaningful content such as bringing old family photos to life with their actual voice on top of the image, making it a realistic one for those who lost their loved ones.

The advancement in these technologies brings positive impact to different groups of people. Content creators and digital artists are able to produce digital content for entertainment, advertising or social media, providing them with creativity possibilities and more dynamic content with less effort. Furthermore, the media and entertainment industry are able to use image animation techniques to enhance visual effects, create virtual characters or revive old footage. This can reduce production costs and add new dimensions to storytelling.

Given our project as singing video generation with music separation, improvements on singing video generation is vital for providing a better user experience for the users. The growing popularity of generative AI has provided us an opportunity to address and dig deeper to explore the possible optimisations to achieve a higher video quality or a higher accurate lip synchronization method. Hence, our team has decided to take an approach on optimizing the video quality by decomposing the generated video into frames by performing raster to vector optimizations on the frames to improve the quality of the image and then convert these improved frames back into the video. These improvements serve as an objective we would pursue but to be implemented if possible given the time constraints and ability of the team.

To sum up, the main objective of the project is to develop a software application that allows users to input their desired audio and image then to be generated into a singing video as an output.

2.2 Literature review

2.2.1 Background

2.2.1.1 The Challenges of Music Separation

Music separation is one of the key challenges in music processes. In general, a music file contains broad categories which consists of audio mixture of the voice of a human and the background music which can be also known as the music instrumental. The objective of music separation is to correctly identify the sound events by isolating the source within a complicated mixed audio input. As music separation is popular and widely used by many applications such as the music industry, speech recognition and hearing aids, there are many researches being explored to improve the quality of separated audio or better efficiency. Some of the techniques include the earlier methods such as Fourier transform and Non-negative Matrix Factorisation(NMF) or the computer vision models which are Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

However, there is no doubt that a music file cannot be separated perfectly with the presence of a trained system dedicated to music separation. This is known as the cocktail party problem which is a disability to perfectly recognize a single audio source from overlapping signals. The difficulty increases tremendously when a trained system not only needs to identify the audio source, it also needs to extract or isolate the source from a mixed audio input to produce separated audio source files.

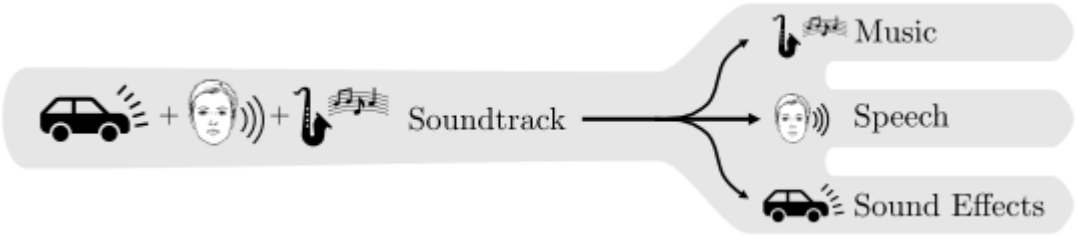


Figure 2.2.1.1: A process of Cocktail Fork problem

2.2.1.2 Limitation of Generating A Realistic Virtual Avatar

Virtual avatar is currently demanding for various applications, such as the gaming industry, virtual try-on and many more. The concept of virtual avatars is already known during the early development such that various applications started their own development as a visual representation of a human object. This includes the gaming industry which is widely used as a visual representation of players and Non-Playable Characters (NPCs), or the usage of virtual reality (VR) equipment which allows the user to control the virtual avatar using motion capture and many more.

However there are some limitations that restrict the virtual avatar being captured poorly that causes some factor of the avatar to remain unrealistic compared to a normal human behavior,

such as lip synchronization, head movement and emotion expression and it remains a challenge to solve real world problems. However, there are various approaches that help to resolve the issue by using Deep Learning methods such as Convolutional Neural Networks (CNN), Generative Adversarial Networks (GAN)s, Neural Rendering Fields (NeRF) for generating the dataset. More explanation will be provided at the related work section.

2.2.2 Rationale

2.2.2.1 How Do Both Backgrounds Relate to Each Other?

As a recap for the objective of our project, the project aims to generate a singing video which contains elements such as instrumental, vocal and a virtual avatar with movement. The generation can be done by providing input such as a music file for decomposing and an image for generation of virtual avatar. Additionally, we will improvise the system by having a software with better user interface and have a more effective generation.

With that said, separate backgrounds allow us to understand the key challenges of approach and identify different techniques to tackle the problem. Later, we will be able to analyze the best method to approach both backgrounds and combine the method to produce a robust software.

2.2.2.2 Comparison of Deep Learning and Traditional Method for Audio Separation

There are various approaches or methods that can be done to separate the audio. However, these methods have advantages and disadvantages that affect the result based on various aspects, such as the quality of the audio or the efficiency when separating the audio. Based on the comparison of both deep learning and traditional methods, it is mentioned that deep learning has more advanced techniques that surpassed the traditional method (Rincón-Trujillo & Córdova-Esparza, 2019).

In terms of flexibility and adaptability, deep learning methods introduced new sources that show a better improvement compared to traditional methods. A few examples are acoustic-only source and audiovisual source that are being proposed by many researchers and accomplished many results (Rincón-Trujillo & Córdova-Esparza, 2019).

Besides that as a continuation from the explanation above, having more sources provides advantages in stability of separation which also increases the quality of the audio, which results in a higher accuracy compared to traditional methods. From a proposal (Gao, Du, Dai, & Lee, 2017), their system managed to provide a faster performance by using a framework based on joint learning that allows managing both background noise and speech.

However, there are some drawbacks such as audiovisual sources that require a great amount of training data in order to achieve a commendable result. It can be seen on one of the research which requires 60 speakers with around 200 videos each in TCD-TIMIT Dataset (Gabbay, Ephrat, Halperin, & Peleg, 2018). As a result, it will increase the computational resources as well. Researchers (Liu & Wang, 2018) proposed a solution that produces better results which is an advantage, however their algorithm didn't manage to perform in real-time computation.

2.2.2.3 How Dynamic Scene Representations Relates Virtual Avatar?

Dynamic Scene Representations has been the popular research in computer vision. It allows capturing the motion of objects over time by ensuring all three properties which are permanency, amodal completeness and spatiotemporal continuity. To further explain the properties, permanency ensures that the object still exists after being blocked from the view. Amodal completeness ensures that the object is available as a 3D model even though the view of the model is partially seen. Lastly, spatiotemporal continuity will track each of the objects over space and time simultaneously. It relates to a virtual avatar as dynamic scene representation is allowed to provide changing over time for a given avatar face by changing the surface of a human's face using transformation.

One of the research (Gafni, Thies, Zollhofer, & Niesner, 2021) shows the usage of dynamic scene representation through input frames. By allowing tracking of facial movement, a 3D morphable model will enable reconstruction of facial expressions. Through input frames, it will provide estimation of the movement and provide a list of transformations. Lastly, volumetric rendering will be used to synthesize the image of the face.

The issue of dynamic scene representation that involves multiple moving objects or the occlusion of the object from different viewing angles. One of the approaches that is proposed is using Neural Rendering Fields (NeRF) however it is not a wise approach as it needs a vast amount of queries to render an image.

For the next section, we will discuss various approaches or techniques available from the researchers to tackle the relevant problems that are mentioned in the background section. Later, we will review the technique available to ensure that the approach chosen is better through comparison and apply it for the design of the system.

2.2.3 Related Work

2.2.3.1 Audio Separation

2.2.3.1.1 Non-negative Matrix Factorization (NMF)

Non-negative Matrix Factorization algorithm was firstly introduced by (Lee & Seung, 2000) which proved to be a useful decomposition of multivariate data. The algorithm was later adapted for audio separation. One of the methodologies from Smaragdis and Brown (2003) shows the usage of Non-negative Matrix Factorization for Polyphonic Music Transcription which aims to recognize the musical instrument and pitch from music signals generated by multiple instruments, and later generate it into symbolic representation (Smaragdis & Brown, 2003).

In detail, the principle used nonnegative decomposition of the spectrogram of the music signals onto a dictionary of components that consists of different sound units, such as music note, chord, percussion and many more. In addition, the architecture obtains the factorization and optimization of a cost function by using NMF models. Later, the process continues by reconstructing the components based on the nonnegative factorization (Makino, 2019).

As a result of the research above, it is mentioned that the process was successful without extensive computational resources and complicated system design. Hence, NMF is still being vastly researched in a situation where the data needed is small which is a better solution compared to Deep Neural Based techniques that require more data for training (Makino, 2019). However, this may be only useful for simple types of application and more data may be reliable for a better quality of audio separation.

2.2.3.1.2 U-Net architecture

A U-Net architecture is a pre-trained model or an encoder/decoder Convolutional Neural Network (CNN) architecture with skip connections (Hennequin, Khlif, Voituret, & Moussallam, 2020). This architecture allows stem or track separation from mixed audio source with the implementation of 1D and 2D convolution (Henning, Choudhry, & Ma, 2021). One of the deep learning tools called Spleeter which is a new tool for audio separation with a pre-trained model which is U-Net architecture that contains vocal separation, 4 stems separation and 5 stems separation. Splitter used a total of 12 layers for U-net, 6 layers for encoding and 6 layers for decoder.

It works by providing vocal and non-vocal components into the neural network system, comparing output inside the neural system, and performing adjustment to reduce comparison, allowing the system to estimate both of the components. The system aims to improve the estimation of identifying vocal and non-vocal components after the training of a neural network system. (Singing voice separation with deep U-net convolutional networks, 2024)

The usage of Spleeter for music separation has proven that the output speed shows better efficiency due to implementation in Tensorflow, which allows code execution on the Central Processing Unit (CPU) and Graphic Processing Unit (GPU). As a result, the computation will be parallelized which reduces the running time of processing. Additionally, the model is based on CNN. An experiment shows that Spleeter can process 100 seconds of stereo audio in less than a second. As a better efficiency of processing, Spleeter can be useful for processing large datasets.

2.2.3.2 Virtual Avatar Generation

2.2.3.2.1 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) is a deep learning based generation that assists in generation of realistic images with enhancement through training of two different neural networks modeled against each other. It is called a Generator that generates samples through input data. Another neural network which is called a Discriminator, that receives the samples from Generator. These two trained each other so that the Discriminator is able to identify data that are realistic or artificial. The final generated data can be achieved when it cannot differentiate between real data.

There are also improved GANs that are related to virtual avatar generation such as the StyleGANs. StyleGAN is an extension of the vanilla GANs that has an ability to generate controllable images of the human face over different aspects such as the facial features, hairstyles and accessories. There are alternative GANs methods such as CycleGAN which are appropriate for Image-to-Image translation.

GANs generation provides many advantages such as the high quality realistic image generation by using StyleGAN. However, there will be drawbacks which require large data datasets and high computational resources. The time spent for model training using CycleGAN is approximately 520 hours (García, 2023).

2.2.3.2.2 SadTalker

SadTalker is a cutting-edge framework designed to generate animated talking head videos by combining an audio input with a reference image of a human face. Unlike traditional methods like GANs or Neural Radiance Field (NeRF), SadTalker focuses on achieving natural, expressive animations that accurately reflect human expressions and lip movements.

SadTalker's generation process involves several steps. First, it uses 3D Morphable Models (3DMM) and audio-driven motion modules to extract facial geometry and head pose based on audio cues. This allows SadTalker to create facial landmarks that guide lip synchronization and expression. The framework then synthesizes these landmarks into a

video output, where subtle head movements and detailed lip syncing combine to create a more human-like animated presence.

SadTalker's results demonstrate smooth, synchronized lip movements and realistic head motions that respond naturally to audio input. While the animations can occasionally exhibit minor artifacts, they generally avoid the uncanny valley effect that other frameworks face. However, the process can be computationally demanding, especially given the need for detailed 3D facial representation and motion synthesis, which may impact scalability for larger applications.

2.2.3.2.3 Raster-to-Vector Image Conversion

An additional enhancement in SadTalker involves raster-to-vector image conversion, which improves animation quality and scalability by converting pixel-based raster images into scalable vector graphics (SVG). This technique allows for higher fidelity in facial expressions and smoother motion across varying screen resolutions. Raster-to-vector conversion optimizes the generated animations, ensuring they maintain clarity without pixelation, which is especially valuable when generating high-definition videos or resizing images for different devices.

By incorporating vector-based processing, SadTalker can produce crisper, resolution-independent animated avatars, which enhances the overall quality of the output and extends compatibility across platforms. This improvement also contributes to the framework's flexibility, reducing the computational load when scaling animations and minimizing the storage requirements for high-quality video frames.

Raster Image	Vector Image
	

2.2.4 Conclusion

This literature review highlights the challenges and progress in both music separation and virtual avatar generation providing a foundation for understanding the transition and refinement in our project's focus. Initially aimed at creating a singing video featuring elements such as audio separation and a virtual avatar, the project has now shifted its emphasis to enhancing avatar realism rather than separating music components. The limitations of traditional methods for music separation, like Non-negative Matrix Factorization (NMF) and even deep learning-based techniques such as U-Net architectures, reveal persistent challenges in achieving perfect audio isolation due to factors like the cocktail party problem. While these methods are valuable for understanding audio processing, they are no longer the main focus of our project.

Instead, the focus has shifted towards solving challenges in virtual avatar generation, such as realistic lip synchronization, head movements, expressions and video quality. The adoption of SadTalker represents a significant improvement in our approach, providing a more sophisticated solution than traditional frameworks like Generative Adversarial Networks (GANs) or Neural Radiance Field (NeRF). SadTalker achieves more natural, expressive animations by leveraging audio-driven facial motion modeling and 3D Morphable Models to generate synchronized lip movements and head gestures. Additionally, the incorporation of raster-to-vector image conversion enhances the quality and scalability of the generated avatars, ensuring high fidelity across different screen resolutions.

By shifting our approach from audio separation to focusing on avatar generation, we address a more pressing challenge in producing high quality videos with accurate lip synchronization. This change in directions reflects the project's evolution from audio-based processing to advanced techniques in virtual avatar generation, with SadTalker providing the backbone for this transformation. As a result, our project aims to deliver an application that outputs a higher video quality singing video generation with the use of raster to vector images, making a significant enhancement over the initial proposal.

3 Outcome

3.1 Initial Design

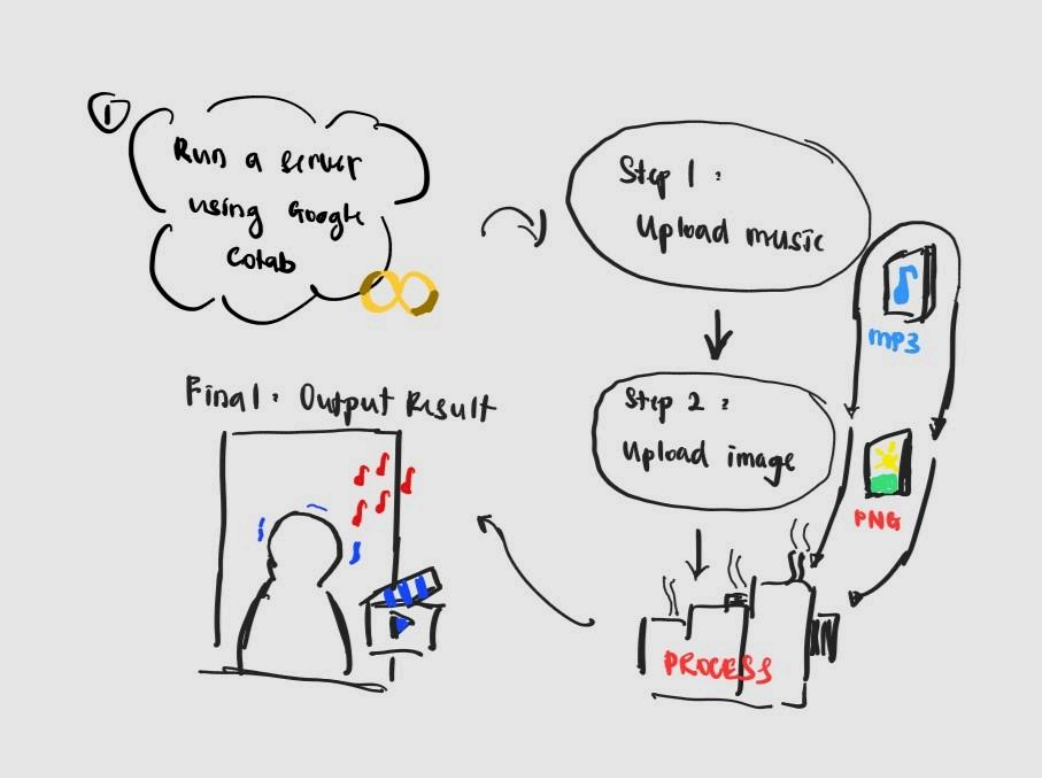


Figure 3.1.1: The process of our application design

To recap the main goal of our project, before the development of our project, we did some research on how to develop a model which is the main purpose of the project. After that, we had drawn a basic process on how we plan to develop our application.

The image shown is the project outcome that is expected to be developed when we start our application development. As we know, this kind of model works extensively and requires very good hardware. We decided to run our application through Google Colab so that we can make use of the better and powerful Graphic Processing Unit (GPU) provided through premium subscription. So for the server, we will use Google Colab to run as well. Then we will continue the process through 3 different steps. The first step involves uploading a music file and the second step will be uploading an image file. Then the model will use these files to generate a music video. The last step will be outputting the result and displaying the video to the user.

Through this process if it is successful, we will provide testing approaches to ensure the quality of the implementation. This includes testing different music and images, adding different file types and many more.

3.2 Implementation

For our current software deliverables, we basically followed the idea of our previous design. Firstly we coded our frontend and backend code in our local machine. These codes are linked to a repository in Github. Then we used Google Colab and wrote a line of codes inside the notebook (also known as Python Notebook, .ipynb) to clone all the codes that we wrote into its UNIX directory, which is available when Google Colab's GPU is connected. The code will be run through Google Colab's as we want to host our web application using their server and process through their provided GPU.

In our web application which is called "Avatar Model Generator", the front-end will have users upload their music and image file as an input for the video generator and a page that will show the generated video. The back-end will contain our model that processes the video with the given input files and outputs a video which is part of our main requirement of our project.

To further explain our front-end, our application contains 3 pages:

- Upload music page
- Upload image page
- Result page

3.2.1 Upload Music Page

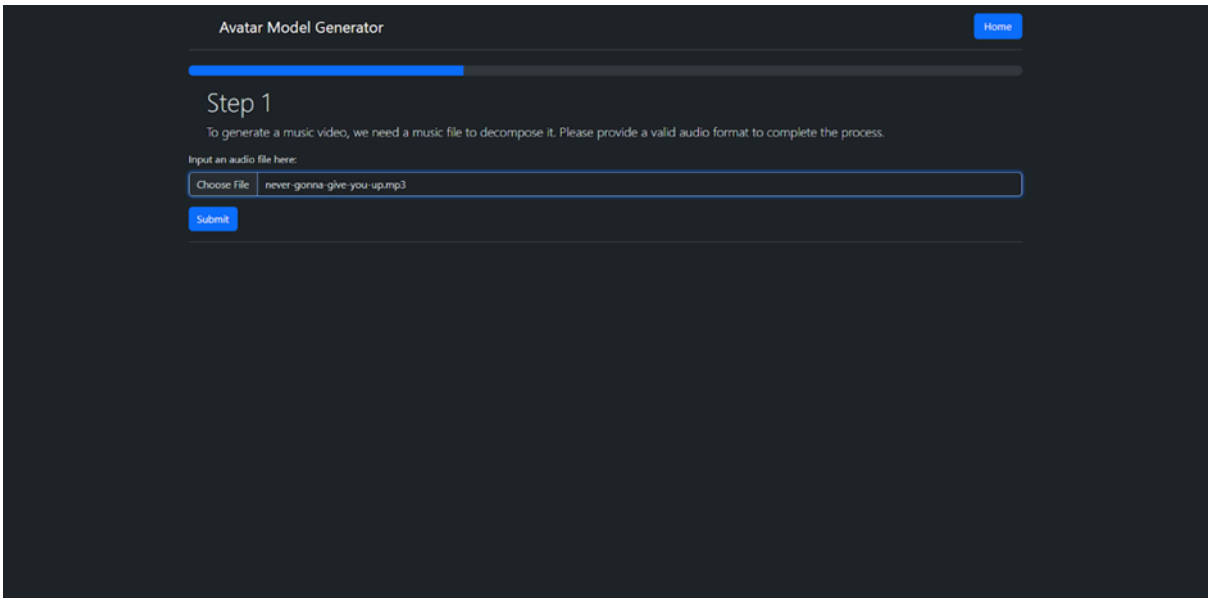


Figure 3.2.1.1: Interface Before Uploading a Music

The image above (Figure 3.2.1.1) shows the first page of our application or our home page which is uploading the music file. To ensure that our user understands the process, we provide a description that is located before the upload box. Users may upload their desired

music file by clicking the “Choose File” button. After uploading the file, the user can press the submit button to proceed to the next step.

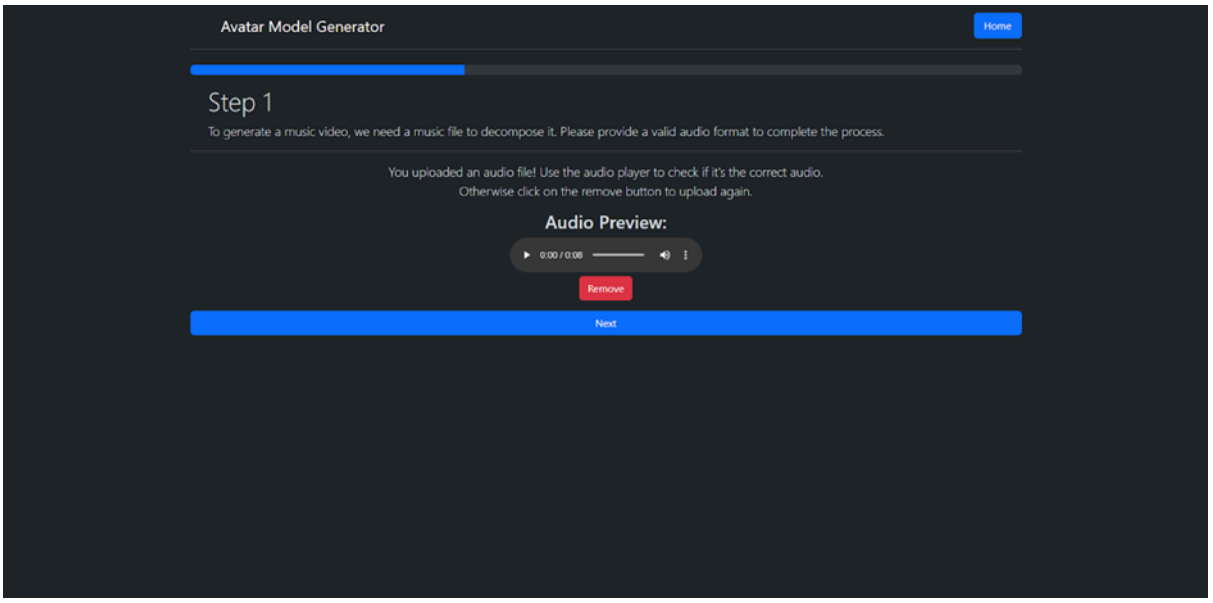


Figure 3.2.1.2: Interface After Uploading - Audio Preview

In this page (Figure 3.2.1.2), users can listen to audio to make sure that correct music is uploaded. A remove button is available for users to recover their mistake and upload a new music file. It will be redirected to the last page which is shown at Figure 3.2.1.1. Users can proceed to the next step by pressing the next button.

3.2.1 Upload Image Page

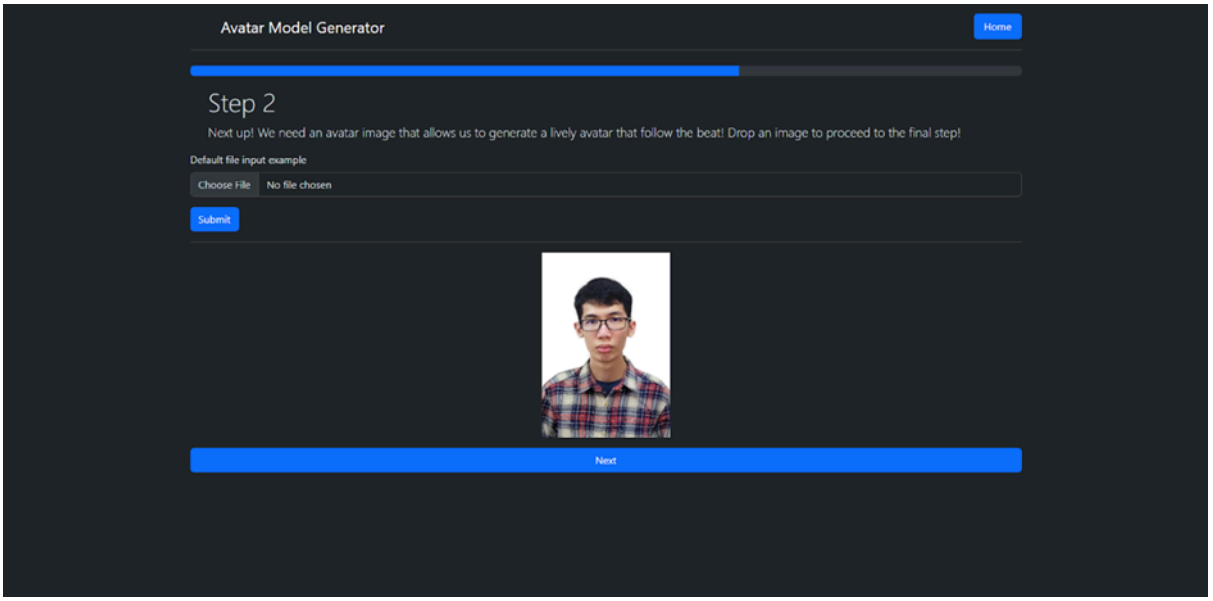


Figure 3.2.1.2: Upload And Preview Image Interface

The second step involves uploading an image file (Figure 3.2.1.2), the interface is quite similar to our previous page except that users are required to upload an image file instead. When an image is uploaded, the image will be shown below the upload box. The preview is to ensure that users upload the correct image. Users can upload a new one through the upload box again. When the user pressed the next button, it will start processing the video by the given two inputs.

3.2.1 Output Page

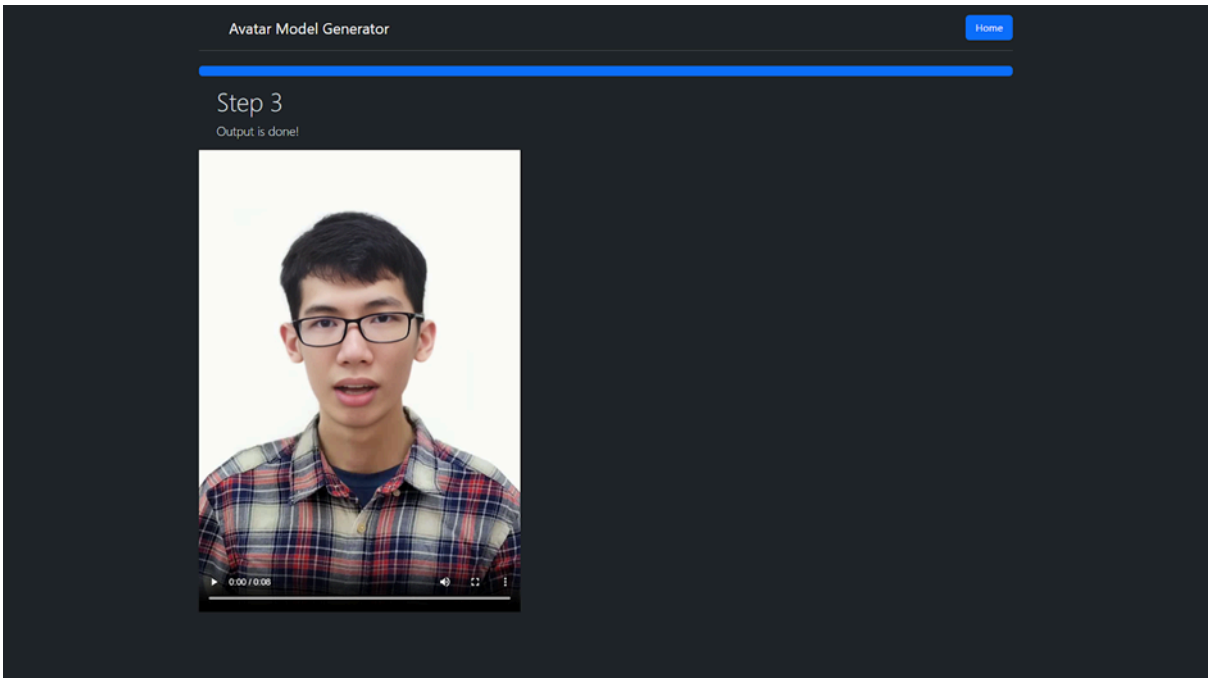


Figure 3.2.1.3: Output Page

After the model has finished processing the video, it will direct the user to the output page which is shown in Figure 3.2.1.3. In this page, users are able to watch the output video and download the video. If they want to generate a new video, users can press either the “Avatar Model Generation” button at the top left or the home button at the top right to redirect back to step 1 page.

3.3 Results

As for our current deliverables, we would say that we actually followed our project outcomes. It successfully generates a video with an avatar singing based on the music given. Our results, which is the output video, shows an accurate lip synchronization of the avatar and responsive head movement. However, the time to process the video depends on the length of the music which is the limitation of the application. Based on the figure below (Figure 3.3.1), an 8 second video will take an estimation of 3 minutes processing time to generate a video even with a powerful GPU which is quite long. It might take even longer if it was the actual length

of the music. In conclusion, although our results meet the requirements of the project, it requires a lot of time to generate even with better hardware.

```
127.0.0.1 - - [24/Oct/2024 11:07:01] "GET /favicon.ico?authuser=3 HTTP/1.1" 404 -
compile
Processing
landmark Det:: 100% 1/1 [00:00<00:00, 14.37it/s]
3DMM Extraction In Video:: 100% 1/1 [00:00<00:00, 48.34it/s]
mel:: 100% 213/213 [00:00<00:00, 38847.97it/s]
audio2exp:: 100% 22/22 [00:00<00:00, 185.25it/s]
Face Renderer:: 100% 107/107 [00:05<00:00, 18.58it/s]
OpenCV: FFMPEG: tag 0x5634504d/'MP4V' is not supported with codec id 12 and format 'mp4 / MP4 (MPEG-4 Part 14)'
OpenCV: FFMPEG: fallback to use tag 0x7634706d/'mp4v'
seamlessClone:: 100% 213/213 [00:31<00:00,  6.70it/s]
Face Enhancer::   0% 0/213 [00:00<?, ?it/s]IMAGEIO FFMPEG_WRITER WARNING: input image is not divisible by macro_
Face Enhancer::   0% 1/213 [00:00<01:51,  1.91it/s][swscaler @ 0x677da00] Warning: data is not aligned! This car
Face Enhancer:: 100% 213/213 [01:26<00:00,  2.47it/s]
127.0.0.1 - - [24/Oct/2024 11:09:52] "GET /step3?authuser=3 HTTP/1.1" 200 -
result
127.0.0.1 - - [24/Oct/2024 11:09:53] "GET /result/2024_10_24_11.07.32.mp4?authuser=3 HTTP/1.1" 206 -
```

Figure 3.3.1: The Process of Generating A Video

Figure 3.3.1: The Result

3.4 Requirements Met By Software Deliverables

To recap on the project requirements that we need to be done for our project, the main objective of this project is to develop an application that can generate a high quality human singing face video with the provided song. These objectives had been splitted into different

requirements such as developing a model to decompose input music into 2 different outputs which are human voice and background music, generate an avatar that synchronize with the provided song, followed by ensuring accurate lip synchronization between the human singing face and the vocals and system testing through variety of songs and also evaluate the application's performance.

Firstly, the decomposition of input music into 2 different music files requirement was actually not done as per request from our supervisor to focus more on avatar generation instead, this requirement is given to another team which is doing the same project as us. However, since our video generation still requires an audio file, we did allow our application to input the music directly instead as we do not have any model to split the audio.

We already satisfied the requirement of generating an avatar that is able to synchronize with the provided song by allowing the user input a music file and image file that represents an avatar. Through the provided input, we had a model that processed the video using the input given by the user and able to show the output from our web application. From countless observations of our output, we can conclude that our model can generate videos that have high accuracy of lip synchronization between the human singing face and the vocal.

Lastly, we also tested our application by using different languages of songs and also different length of songs as well to perform system testing and evaluate the application's performance when processing the video.

3.5 Decision Made

Since our model is using the Pytorch deep learning model which is relevant to Python, we decided to use Flask as our web framework to match the programming language for both. Besides that, Flask enables us to link our code to Google Colab easily and is able to separate Python code with web codes such as HTML and CSS. With the combination of Flask, we integrate our model by calling the function when it is needed to process.

For the front-end, HTML will be the best choice for our front-end that is displayed in the webpage. Additionally, we use Bootstrap Flask as our framework which can be used with Flask. The decision is a wise choice as bootstrap provides minimalistic design that will attract the user to use our application.

As mentioned above, we wanted to link our code to Google Colab. So, Google Colab will be used to pull all the required code and make use of its functionality such as running the web server and processing the video using its GPU.

As a result, we actually followed our own decisions of implementation and we were quite lucky that it went well. There are no decision changes throughout the whole process of development.

3.6 Discussion of Project Results

3.6.1 Comparison Between Previous and Current Products

Based on our previous results or previous thorough research, we managed to have a model that is able to generate a video which contains an avatar that is well synchronized with the music given. However, it is not user friendly enough as it is code-based that most of the users are not familiar with how it works as it is complicated, even with a user manual. Besides that, the output of the file cannot be shown directly to the user, hence it is tedious as users need to find the output directory. Moreover, the input of the file cannot be double checked by the user as well and cannot be modified if the user really made a mistake. In conclusion, our previous products are not suitable to be used by users on a daily basis as it is very frustrating in terms of usability.

For our current deliverables, we have managed to further improve based on our previous products by creating a web-based application with user interface which is more user-friendly for the user to use our application. In our application, we provide step-by-step instructions to guide users on how to input the music and image files for generation, which is better compared to our previous product. We also managed to show a preview of the specific music and image files for the user to check, additionally the user is allowed to upload a new file if the input file is wrong. For the final step, we will show the preview output as well so that users are able to watch and download the video directly from the web application.

As a comparison of our current deliverables and previous products, it is clear that our current deliverables are more user-friendly than the previous one and it is better for the user to understand how to use the application. Besides that, it has a better user experience and recovery error tolerance as sometimes users tend to make small mistakes accidentally. It also saves a lot of time for the user as users do not need to find the input and output directory, instead our application can show it directly to the user.

3.6.2 How Result Extend The State of The Art

Since our main focus of our project is avatar generation and music separation, these have extended the state of art in terms of deep learning towards avatar video generation. Our deliverables has successfully delivered a real-time synchronization between audio and avatar which is a novel approach that has an avatar that responds to musical beats and vocals in real time to show a low-latency animation. In result, the avatar will have a smoother animation and on sync based on the vocal and background music with an expressive emotion based on the musical tone which pushes the limits of audio and visual experiences.

Besides that, our project had improved more on avatar personalisation which shows different expressions based on the music moods. From the music given, the avatar might have a different singing style that shows different head movement. Also, it will enhance the avatar

through better lip synchronization with the vocal of the music. This creates a more complex avatar in terms of having more awareness and responsive avatar modeling.

As a whole our project introduces a better technique in terms of model optimization and adaptive resolution control that might allow our avatar generation to be used across a wide variety of devices such as Virtual Reality or Augmented Reality. These devices require a real time synchronization between audio and avatar generation which is suitable for our solution to solve these problems with better techniques.

3.6.2 How Result Solve Problems or Meet A Need

Since the main focus for our project is generating a music video, it will be beneficial for the entertainment industries that want to create music videos as it will solve various problems. In terms of entertainment industries, which includes Youtubers that are interested in creating music videos or music video teams that create music videos for artist singers. There might be problems such as a Youtuber that wants to create a new music video for his or her new song but they are inexperienced at creating it. There is also a possibility that there are a lot of new songs pending in a specific music video team as they lack members to produce music videos faster for the singer.

Hence, our application will benefit them from generating a music video that has its own face and is able to show that he is singing his own song through a few clicks. It can be created easily by inputting an image containing an avatar and a song file, the rest will be dependent on the video generation model. As having an animation video that has quite similar quality with hands-on animation video, it saves a lot of time and effort for video production and possibly being able to produce more music videos at a faster rate. Besides that, our application contains step by step instructions which are more user friendly for users to use the application even though they do not have any prior experience.

3.7 Limitation

Although our project is successful as it meets most of the requirements and we had a web-based application that can be run without any problems, our application is only the bare minimum and contains limitations that will restrain the usage for the user.

As for our limitation for the testing process, we did not manage to perform multiple requests by processing multiple videos at once. Since some users might try to process multiple videos through multiple submissions, it might affect the system performance due to heavy traffic. If we perform testing and evaluate the performance, we will try to limit the user submission to ensure balanced performance when generating multiple videos and also avoid application crashes.

Besides that, due to hardware limitations of our local machine, we did not test our machine through the local machine and only ran it by using Google Colab's virtual GPU. As our local machine does not have powerful hardware, it is not recommended to run the application through local runtime as it might damage the hardware while extensively rendering video. However, there might be a possibility that running locally would be a better choice because it is easier or faster to run locally instead of Google Colab. Some users might not be familiar with Google Colab as well. So, a better solution to solve this limitation will be further research on recommended hardware requirements to avoid hardware damage. Hence users are able to deploy our application locally.

There are also limitations such that file extension is too limited for uploading music and image files. Currently our application only supports MP3 and PNG files when uploading. However, there are quite a few commonly used file extensions which are not included. For example, in music files, MP3, WAV, FLAC are the commonly used extensions while PNG, JPG, JPEG are the common ones for image files. This enables more supported files for the user to upload which saves the hassle of converting music files or image files into the correct file extension.

Lastly, our application can only process one video at a time which is a limitation for users that want to produce multiple videos at once. As mentioned before above, due to hardware limitation we were unable to process multiple videos as it will damage the hardware. However, we did consider doing it as an improvement for our future work.

3.8 Potential improvements and future work

As for now, we have a working application that is able to generate a human singing video, but there is more room for improvement that allows us to provide more impact for the industry. This includes video quality optimization that generates a better video resolution and higher refresh rate. This is a good recommendation as modifying the state of the video through processing more or less affects the quality of the video, hence merging a new model that can generate a better resolution and provide better refresh rate video would be better. From our previous research, we had a thought of slicing into a sequence of image frames, enhancing each of the images and later combining everything together to form a better video.

Besides that, we can further improve the model by not only providing movement for the face expression but also covering the body movement as well when a full body image is given. As for the current model, it is possible to input a full body image but the model did not show body movement which seems stiffer as there is only face expression but the body did not move. So, improvement can be made by having body movement included in our model will show a more expressive animation when both face and body have movement.

The most frustrating part of our application by the users after testing is generating another output that requires the application to redirect back to the first step everytime. Additionally, it is quite annoying for a user to input a different image but upload the same music file again which is repetitive. For a more quality of life improvement, our application should allow multiple images and music to be uploaded and stored inside a database. So, a user may play around by choosing the same image with different music or other way around. As a result, users do not need to upload the same file over and over again and are able to access the database again after the output is generated. This will save a lot of time and effort of reducing repetitive steps from our application.

As for the last improvement, we had proposed above which will support a wide variety of image and music file extension when uploading. Since we only have a fairly new application, we decided to allow MP3 extension for music file and PNG for image file to reduce error of our application. However, users might have different file extensions which most of the time are commonly used. Hence, it is also better to ensure more support for different file extensions to save users more time from converting their files.

4 Methodology

4.1 Final Project Design

For our final project design, we decided to create it based on a step-by-step workflow. Given each step, the user will be required to upload an audio or an image to proceed to the next step. This linear flow helps guide users through the necessary steps in an orderly fashion, making it simple and clear what action is required at each step. In addition, we chose to use a minimalistic layout, allowing it to be clean and straightforward. Having minimal elements on the screen and only essential information such as instructions, file input and buttons for submissions reduces distractions and the difficulty of using the application. Besides that, we added instructional text to provide a brief description to explain what the users are required to do for the following step. For example, step 1 provides a description guiding users to provide a valid audio file, whereas step 2 explains the need for an image for the video generation. These design aids in providing concise instructions to help users understand the process and reduce confusion when using the application. Furthermore, we added a progress bar to visually display the current progress of the user, giving them an approximation of their progress and the remaining steps to be completed. This enhances the user experience by giving them a sense of progression. Lastly, we have a consistent button style, making the interface intuitive and easy to navigate through. With the background of the interface being dark gray, the chosen color, blue, attracts attention to the available actions the users can perform. Examples of the images can be referred to at figure 3.2.1.1, figure 3.2.1.2 and figure 3.2.1.3.

These choices of design provide several major benefits to the users. Firstly, the step-by-step approach simplifies navigation for users, providing assistance and reducing the difficulty faced for first time users, thus providing a User-Friendly interface. Secondly, the progress bar and instructions stated at each step provides guidance for the users. By doing so, they are able to obtain a smooth user experience and will be more likely to successfully complete the tasks given the structured flow. Thirdly, our design focuses more on the core functionality rather than on sophisticated user interfaces as we believe that the users of our application would be more interested in our output than the visuals we provide. Lastly, this design provides scalability, where we are able to extend the functionality by adding more steps if needed, without compromising the user experience. Additional functionality could be included while maintaining the same sequential flow.

In summary, the chosen design promotes an accessible, straightforward, and guided user experience, making the process of generating a singing video easier and more efficient.

4.2 Deviations from initial design proposed

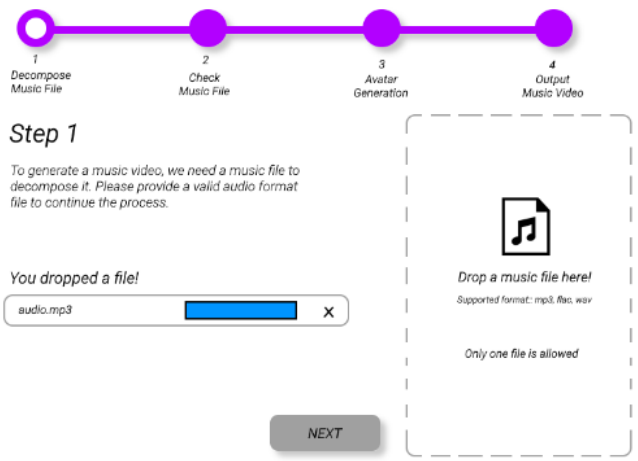


Figure 4.2.1: Example of Initial design

To begin with, our older design features a step by step progress tracker at the top with distinct circles and labeled steps to indicate the current step and provides a clear visual flow of the entire process. Our initial design also tries to include more detailed guidance on each step, with drag and drop areas and explicit feedback when the users uploaded their file. As for the file upload area, it features a drag and drop zone and file format details which makes it easier for users to understand how to upload files. The initial design emphasizes a more structured and guided experience, but we believe that it might complicate the users and end up taking more time to advance for each step. Refer to appendix for more images of initial design images.

Whereas for our new design, this design aims to achieve a more minimalistic and straightforward design for the users, which highly reduces the time and difficulty of using it. Our new design uses a simple linear progress bar to show our step labels, which follows the minimalistic approach we decided on to show the current progress of the users. The new design provides essential information for the users to understand the steps they are required to do to advance to the next step, without having excess details to complicate the flow. Lastly, we decided to remove the drag and drop feature and instead proceed with the button for choosing files to provide a more basic upload interface. This also helps us limit the user's choices to only the file types our application accepts. Examples of the images can be referred to at figure 3.2.1.1, figure 3.2.1.2 and figure 3.2.1.3.

4.3 Design implementation

4.3.1 Implementation

Our design follows a minimalistic approach, prioritizing functionality and user guidance over visual complexity. As our project focuses more on deep learning, we believe that the backend processes are more critical than the frontend design. Thus, the interface is kept simple to allow uses to interact seamlessly with the application's core functionality.

4.3.2 Software and tools used

The completion in creating our design relies on several tools and software. We used canva for our initial design mockups. Its user-friendly interface and versatile design elements allowed us to create a basic layout and visualize the step-by-step process flow before implementation. For developing the actual interface, we used standard web technologies such as HTML for structuring the content, CSS for styling the layout, and JavaScript for adding interactivity.

4.3.3 Activities carried out

We also carried out several activities which helped us in our design. We started off by creating wireframes in Canva to outline the basic structure and user flow of the interface. This includes defining the key steps, such as uploading an audio file and generating the avatar. Then, we started with our frontend development to set up and create our proposed interface. We also ensured that our application's interface could interact with our backend services such as music decomposition and video generation, to ensure the application ran smoothly. We then proceeded with testing to ensure that the interface was working accurately and were able to guide users through each step easily. Lastly, user testing was performed by our peers to gather feedback on the design's usability and made interactive adjustments to improve the user experience.

5 Software Deliverables

5.1 Brief description of software deliverables

As for our software deliverables of our project, we had implemented a web-based application. Our application allows users to input a music file and an image of an avatar regardless of a full body picture or only face picture. By using these files uploaded by the user, the model will be able to process within a few minutes and output a video that contains a picture of the avatar given by the user which is able to move and synchronize with the music. To run the application, we have written a Jupyter notebook in Google Colab which can be run just by one click. The web application server will be run using Google's Colab GPU hardware as our personal computer might not be able to handle heavy processes. Using Google Colab enables for better performance when generating the video. We had implemented the user interface of the web application using Bootstrap CSS which shows minimalistic design that suits most of the users preferences.

Overview of The Software

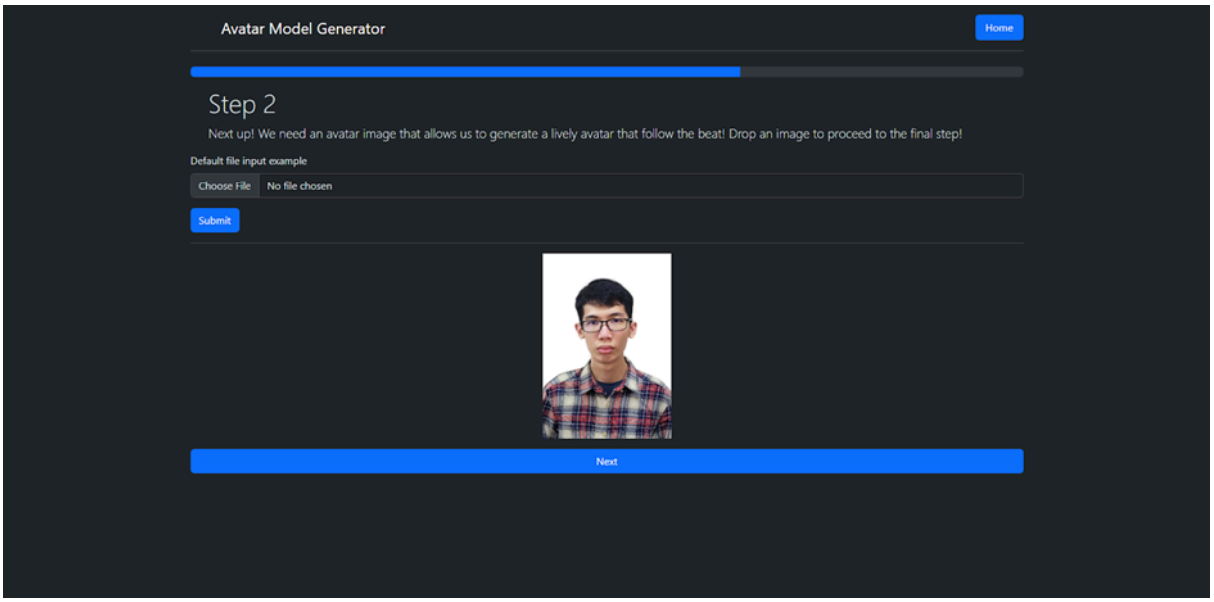


Figure 5.1.1: Step 2 - Upload Image and Image Preview

5.2 Software Quality

To ensure our application has the best experience for our user, we had to perform various testing and further improve our application to have a better software quality. It is an important factor as our long term users will keep using the application or new users will be interested with the application if our application has a positive review. As for now, we will summarize the software quality of our current release of our application by discussing the benefits and shortcoming based on various factors such as, robustness, security, usability, scalability, portability and usability. From the summary, we will also provide recommendations on how to further improve our application for each of the factors.

5.2.1 Robustness

In terms of robustness, our application is able to handle errors to ensure that our application does not crash unexpectedly. This can be seen such that having input validation for file extension for uploading the music and file is limited to reduce error for processing and previewing the files. These functions may have not supported other file extensions which might crash the application or doesn't even display if the file is not inputted correctly. We had also provided different steps for different files to ensure that users upload the correct file in sequence. Besides that, we limit only one file for each music and image file to cater our processing model input as multiple files might confuse the model on which file to use.

However, having robustness through limitation is not a good idea as it will restrict the usage of our application. For example, a user cannot upload more than one file or a user has a music file that has different file extensions and many more. Through testing, we also found that the user is able to call the model to process multiple videos at once through the back button which is not intended for our application. This will cause application crashes due to heavy workload and no fallback processes.

We will look forward to supporting more file extensions for music and image files. As for the multiple file upload, we will create a database that will store the files inside so that users are able to choose the files to process. Besides that, we will ensure users will not abuse the application through pressing the back button by canceling the generation process when the user redirects to the previous steps. These solutions not only increase the robustness of our application to handle errors effectively and at the same time provide more function for the users which ensure better user experience as well.

5.2.2 Security

Our application has security such that the user will be protected from data stealing and malicious attack. Since we are using Google Colab to run our application through its server, all the files uploaded will be stored inside a temporary storage which is available when connected to Google Colab's GPU. As a result, all the files will be deleted when the GPU

disconnected which leaves no trace of the user's personal information being stored. This shows data protection for our user to upload their user personal files with ease as the data will be removed after usage.

Moreover to ensure better security, we use a temporary URL in Google Colab that is able to proxy a local server running on a specific port to an accessible URL. As the generated is temporary and specific to the session, hackers will be less likely to be discovered randomly unless the URL is shared to the public. The URL will be inaccessible when the session terminates which limits exposure time and reduces the risk of persistent attacks.

The potential security issue is as mentioned above, a hacker can perform a malicious attack if a hacker discovered the URL. Since our application is only shared between our team, we did not take measures to prevent security risk. The file uploaded is not encrypted when uploading into the storage which is dangerous as hackers can access the file directly without any decryption. Other than that, a hacker is able to perform various malicious attacks that will harm the user of the application such as exploiting vulnerabilities through SQL injection, Denial of Service attacks to terminate the session and many more.

To mitigate security risks, we plan to validate user input to protect against injections, attacks and stealing data. We will also implement encryption when users upload new files inside our application so that the user's privacy will be secured as the content is private and not altered in any way.

5.2.3 Usability

Usability is an important factor for our application to ensure that users can easily understand how to use the application and interact with it. In this case, the user can accomplish the task in a short amount of time as well. Our team had discussed the design implementation of our application and created a prototype beforehand to make sure that all the importance of usability is contained in the application during development.

The first thing that we had for our application is to have a minimalist design through Bootstrap CSS library which is commonly used in other applications and easily attracts users due to its aesthetics. A minimalistic design shows simplicity that is able to eliminate distractions from other components so that users can complete their task as fast as possible as they will be more focused on the things that you want them to see.

In terms of efficiency, we provided step by step instructions that were able to guide the user on what to do on each step. This will enable the user to familiarize with our application faster through the description written. There is also a remove button after uploading the music so that users can perform an undo by removing the music file if they made a mistake which shows recovery from the mistake. As a result for our application, users can generate video

efficiency which meets the requirement of having better usability as users can accomplish tasks quickly.

As for the shortcoming of our application in terms of usability, we had not managed to provide a user guidance section in our application for our users to refer. Although we have descriptions for each of the steps, a user guide will help users understand the application better as a whole, such as the complete process of the application, how to handle errors and many more. For some cases such as error handling and some description to show that the video is processing should be shown as well so that the user can understand what is going on.

Since we had known these issues, we will plan to implement it for our next release to ensure overall satisfaction of our application to encourage users and attract them to continue using our application.

5.2.4 Scalability

Our application provides scalability in various ways. The frontend design follows a minimalistic step by step user interface and modular approach which allows for easy scaling. New features or steps can be added without requiring significant changes to the existing layout. Besides that, for every step, our application switches from 1 html page to another, thus if any changes are required we can simply add a new step such as enhancing the image in between steps and just make slight modifications to the existing code.

Our application runs on Google Colab, which makes it an individual application where if the number of users increases, there will be no effects on the existing users as they run their own applications instead of having a shared host. This allows our application to be scalable as we can take in more users without having to be worried about issues arising such as slower generation video time.

As for the shortcomings of our application in terms of scalability, there is a possibility of having performance bottlenecks since everyone is running on Google Colab, causing Google Colab to be heavily used thus causing delays in generating the video.

To resolve this issue, we could try optimizing the code and processing pipeline by reviewing the code and models for any inefficiencies. By doing so, this helps us to improve the performance to reduce the computational load, making it easier to scale the application.

5.2.5 Portability

Being a Cloud-Based implementation application provides portability to all users. By using Google Colab, the application leverages a cloud-based environment that is accessible from any device with an internet connection. Users can run the application on different operating systems such as Windows, macOS and android without worrying about compatibility issues.

Despite having the benefit of being portable, it brings obvious drawbacks as well. When the user uses our application, they are required to set up the application by running the provided code in Google Colab, which would take up some time. Therefore, despite being convenient, the setup time is unavoidable for users.

To counter this issue, we could implement our application as a web application and mobile application, where users would be able to use our application just by entering our web and mobile application.

5.2.6 Maintainability

Our team has taken into consideration the maintainability of our application in several ways. Firstly, our application code can be effectively modified without causing defects or bugs or decreasing the quality of our application. Secondly, we make use of version control and collaboration tools to track our changes, which enables us to troubleshoot easier and maintain a more systematic update on the existing code we have. If any error arises, we can just simply revert the changes.

As for the shortcomings of our application in terms of maintainability, lack of automated testing could make it hard to find the cause of bugs relating to the update. This could increase the risk of regressions when modifying the codebase, making the maintenance process more error-prone and time-consuming.

To avoid having such an issue, we can implement automated testing performing unit tests and integration tests to continuously catch issues early and ensure that changes do not introduce new bugs.

5.3 Sample Source Code

A sample of our source code can be referred to Appendix 8.1. The specific code shows how the function works after the next button in the step 2 page is pressed. The function will locate the input files and call the command to execute the process of generating the video. The page will direct the user to the output page after the output is generated.

6 Software and Project Critique

Our project aims to develop an application that can generate a high realistic human singing face video synchronized with the provided audio. Thus, we believe that our team has managed to execute the project successfully, as we managed to satisfy this main objective of the project. However, there is no doubt that the execution was not as smooth as planned. We had changes in our schedule, addition in objectives where we try to achieve more than we initially planned, which led to a higher time consumption, taking up time which was supposed to be allocated for implementation. Despite spending ample time in performing thorough research, our team did not go in detail for the integration of the model with the UI, which also took up a lot of time. Luckily, we managed to finish everything on time as planned. This excess time spent on fixing issues that arise could have been avoided if we took into account and allocated more time during our planning phase. Subsequently, we could have spent that time performing more testing, hence ensuring the quality of this project in terms of correctness and features. There were also some unmet objectives that we added midway throughout the project duration. Improvements on the existing model were tested however sacrificed due to time constraints and unexpected results where the images showed signs of stiffness and unnaturalistic.

Regardless, our team managed to achieve some set goals, as listed below:

- Development of the front-end with an minimalistic user interface
- Successfully integrate the user interface with our backend logic
- Performed sample testing to ensure the quality of the application
- Gained precious knowledge and experience in relation to using deep learning models

To sum up,there is no doubt that we had successfully met the key objectives of our project and would consider our project to be a decent success.

7 Conclusion

To conclude our project, we were able to create a web-based application that met the requirements of the project. The main requirement which is creating a model to generate a music video that contains an avatar that syncs with the provided music is successfully developed as it is the most important part of our project. We were also made possible by integrating our model into our web-based application and it can run successfully after a few rounds of testing.

As having a workable application, we planned to further our development of our application through various improvements such as improving the quality of video, supporting multiple processing, and a database that stores music and image files. We hope that our work contributes meaningfully to the field of AI-driven media applications, providing a robust foundation for future exploration into interactive content creation and audiovisual synthesis.

8 References

Gabbay, A., Ephrat, A., Halperin, T., & Peleg, S. (2018). Seeing through noise: Visually driven speaker separation and enhancement. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. doi:10.1109/icassp.2018.8462527

Gafni, G., Thies, J., Zollhofer, M., & Niesner, M. (2021). Dynamic neural radiance fields for monocular 4D facial avatar reconstruction. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. doi:10.1109/cvpr46437.2021.00854

Gao, T., Du, J., Dai, L.-R., & Lee, C.-H. (2017). A unified DNN approach to speaker-dependent simultaneous speech enhancement and speech separation in low SNR environments. *Speech Communication*, *95*, 28–39. doi:10.1016/j.specom.2017.10.003

García, L. G. (2023). Retrieved from https://repositori.uji.es/xmlui/bitstream/handle/10234/203633/TFG_Gonzalez_Garcia_Lucia.pdf?sequence=1&isAllowed=y

Hennequin, R., Khlif, A., Voituret, F., & Moussallam, M. (2020). Spleeter: A fast and efficient music source separation tool with pre-trained models. *Journal of Open Source Software*, *5*(50), 2154. doi:10.21105/joss.02154

Henning, R., Choudhry, A., & Ma, M. (2021). Deep Learning based music source separation. Retrieved from https://repository.stcloudstate.edu/cgi/viewcontent.cgi?article=1009&context=joss

Lee, D. D., & Seung, H. S. (2000). Algorithms for non-negative matrix factorization. Retrieved from https://proceedings.neurips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.pdf

Liu, Y., & Wang, D. (2018). A Casa Approach to deep learning based speaker-independent co-channel speech separation. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. doi:10.1109/icassp.2018.8461477

Makino, S. (2019). *Audio source separation*. SPRINGER.

Rincón-Trujillo, J., & Córdova-Esparza, D. M. (2019). Analysis of speech separation methods based on Deep Learning. *Research in Computing Science*, *148*(9), 21–29. doi:10.13053/rcs-148-9-2

Rouard, S., Massa, F., & Défossez, A. (2022). Hybrid transformers for music source separation. Retrieved from https://arxiv.org/abs/2211.08553

Schwartz, B. (2024). The risk management process in project management. Retrieved from https://www.projectmanager.com/blog/risk-management-process-steps

Singing voice separation with deep U-net convolutional networks. (2024/01/02/, 2024 Jan 02). *Targeted News Service* Retrieved from https://www.proquest.com/wire-feeds/singing-voice-separation-with-deep-u-net/docview/2909431230/se-2

Smaragdis, P., & Brown, J. C. (2003). Non-negative matrix factorization for Polyphonic Music transcription. *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*. doi:10.1109/aspaa.2003.1285860

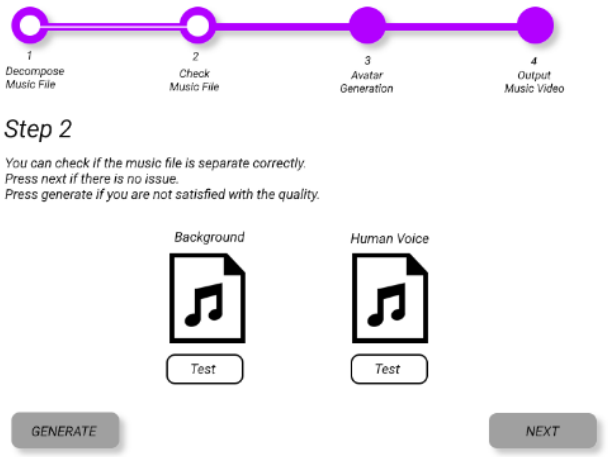
Wei, H., Yang, Z., & Wang, Z. (2024). *AniPortrait: Audio-Driven Synthesis of Photorealistic Portrait Animation*. doi:arXiv:2403.17694v1

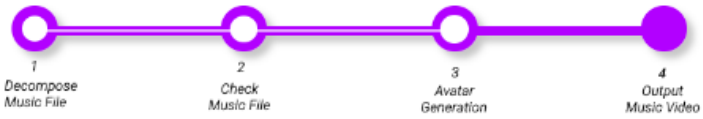
9 Appendix

Appendix 8.1

```python
@app.route('/step3')
def step3():
    path = os.path.join(os.path.dirname(__file__), 'result')
    if not os.path.exists('result'):
        os.mkdir(path)
    # selected audio from exmaple/driven_audio
    img = '../fit3162/uploads/avatar.png'
    audio = '../fit3162/uploads/track.mp3'

    os.chdir('..')
    os.chdir('SadTalker')

    cmd = f"python3.8 inference.py --driven_audio {audio} " \
          f"--source_image {img} --result_dir ../fit3162/result --still --preprocess full --enhancer gfpgan"

    popen = subprocess.Popen(cmd, shell=True, stdout=subprocess.PIPE, universal_newlines=True)

    print("compile")
    print("Processing")
    while not glob.glob('../fit3162/result/*.mp4'):
        pass

    os.chdir('..')
    os.chdir('fit3162')

    mp4_name = glob.glob('./result/*.mp4')[0].split('/')[-1]

    return render_template('step3.html', filename = mp4_name)
```

Appendix 8.2



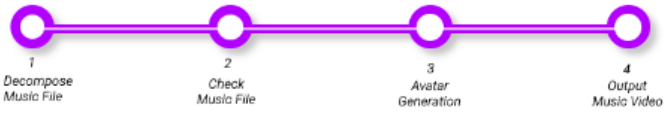


Step 3

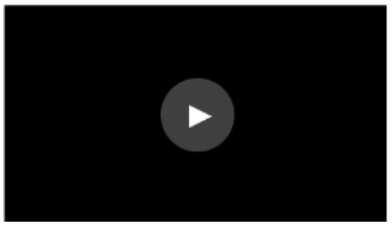
Next up! We need a avatar image that allow us to generate a lively avatar that follow the beat!
Drop an image to proceed to the final step!

You dropped an image!

NEXT



This is the generated music video!
You can watch and download the video if its great!



DOWNLOAD

FINISH