# AnimEditH: AI-Powered Anime Character Editing Web App *

## Capstone Design Project Proposal

Chingis Oinar[1 [2018315169]],
Park Soo Hun[2 [2016314364]],
Eunmin Kim[3 [2018315083]], and
Gong He[4 [2018313592]]

[1] Sungkyunkwan University
2066, Seobu-ro, Jangan-gu, Suwon-si, Gyeonggi-do, Republic of Korea
[2] chingisoinar@gmail.com
[3] soohun96@g.skku.edu
[4] eunmin88@gmail.com
[5] gh970818@hotmail.com

**Abstract.** With the rise of popularity of anime growing steadily, the demand for unique styles continue. Current anime styled face generation tools are not very user friendly, do not provide an end to end program and do not give much freedom of style to the user. Another problem is that majority of them only create female anime faces. There are also only female face datasets readily available online. Creating new anime styled faces without the use of AI power generation is a very time consuming and tedious. We plan to create a web app that gives the user much more freedom of creating their own anime styled face. We allow user to free draw their face, colorize to their liking, add facial features with the usage of tags and freely edit the photo with the user of a editing toolbar. We will also allow user to create male or female anime styled faces. In order to do this, we will create our own dataset of anime styled male faces due to the lack of available datasets. We will also have a history functionality that saves all past created images and stores them. After the user is satisfied with their anime styled image, they can download the image for their own personal use. For the machine learning part of our web app, we will use Generative Adversarial Networks (GAN).

**Keywords:** Anime · GAN · Photo Editor

## 1 Introduction

**Anime** is hand-drawn and computer animation that was first produced, with commercial purposes, in 1917, Japan. A distinct art style was developed in the 1960s, inspired by the works of eminent cartoonist Osamu Tezuka. Subsequently,

* Supported by SKKU.

it has been spreading widely, developing a colossal domestic as well as global audience. Moreover, it is now being produced by cartoonists outside of its country of origin. The global anime market size reached a value of approximately *USD 23.56 billion* in 2020 and is estimated to grow up to staggering *USD 25.46 billion* in 2021 [7], whereas the revenue forecast is expected to reach *USD 48.03 billion* in 2028 [7]. Thus, the compound annual growth rate constituted to be around 9.5% per year making. Furthermore, Sony Pictures acquired Crunchyroll, an online anime streaming service, for *USD 1.175 billion*, with plans to widespread anime to everyone at home [6]. According to Tony Vinciquerra, Chairman and CEO of Sony Pictures Entertainment Inc., Crunchyroll brings a tremendous value to Sony's current anime businesses, including Funimation as well as company's terrific partners at Aniplex and Sony Music Entertainment [6]. Recently, the Otaku Coin cryptocurrency, which has been running on the Ethereum platform since 2018, has announced its plans to produce an isekai anime with Non-Fungible Tokens (NFTs) [8]. Although the items to be produced possess no inherent attributes, nor do they interact with any broader game rules, the concept has become viral and attracted a huge audience around it, so participants started to produce their artwork and story concepts [8]. The combination of factors above makes Anime a large business area with huge future potentials.

However, it takes tremendous efforts and a huge amount of time to acquire the skills needed to draw decent anime characters. To bridge this gap, there have been many projects and works introduced that generate Anime characters for people. However, the vast majority of them generate mainly female characters, such as Make Girls Moe [9]. Moreover, the largest part of available datasets online contain data samples of female characters dominantly. Therefore, in order to provide a full control of anime character generation, we need to tackle this issue of the imbalance between female and male characters. In addition to that, most of the current solutions generate the end product only, hence they do not provide a freedom to edit them.
Finally, inspired by the demand and some issues about current solutions our contribution can be summarized as follows:

- We provide more freedom to control anime character generation by allowing users to edit and colorize images as they wish, so they could create their own characters in end to end manner.
- We tackle the problem of gender imbalance in current datasets available online and make our own dataset of male character in addition to them.

## 2 Motivation & Objective

With the up rise of popularity of anime, we want to provide an AI-powered end to end Anime Styled Face Generation web app to creators or fans of the genre. Since creating anime styled faces and editing is usually a very time consuming task and is quite tedious, we want to develop an app that will shorten time needed and reduce effort required. We want to build it so that users can free draw a face or create a face using tags to generate an anime styled face and make it editable/color-able or completely randomize a design if they choose to do so. Other similar apps only generate anime faces and are not editable afterwards, only downloadable. There aren't many options of how they want it to look like before generating, which makes it difficult to generate a style of the user's liking. Majority of them are randomly generated, so they are not geared towards user satisfaction. Users may not be content with their randomly generated and may want to change specific features or colors but are unable to. the only way is to use another app where they can edit the generated image. We plan to make an app that allows the user to have complete control over how they want to customize their generated anime styled face using AI models.

## 3 Related Work

**Generative Adversarial Networks.** Generative Adversarial Networks (GANs), proposed by Goodfellow et al., have recently achieved impressive results in the field [3]. The core idea can be summarized as training *generator* and *discriminator* networks in minimax game manner, where the former tries to fool the other network by producing high quality output, whereas the latter network tries to classify each image correctly. Specifically, by incorporating two separate networks, generator and discriminator, GAN learns a loss function due to which it is able to produce highly realistic images. Given a training dataset, generative models synthesize new samples from the same distribution. Thus, Generator's objective is to generate data that is indistinguishable from the real data, whereas the Discriminator takes both real and generated data and tries to distinguish them correctly. Furthermore, considering that GAN learns an objective that adapts to the training data, they have been applied to a wide variety of tasks. Recently, GANs have been employed for Image Restoration, Text-to-Image Translation, Image-to-Image Translation, Face Frontal View Generation, Cartoon Characters Generation, Style Transfer, Face Aging and 3D Object Generation.

Many variants of GANs have been proposed for generating images. Mirza & Osindero et al. introduced *Conditional Generative Adversarial Networks* (CGAN) using which they managed to generate MNIST numbers of a particular class [1]. Later, Reed et al. demonstrated how an encoded text can be used to produce an image of interest [5]. Odena et al. proposed ACGAN where auxiliary classifier to predict the condition input was introduced in addition to the vanilla discriminator [2]. Although the training process is quite simple, optimizing such models is not a trivial task. One of many issues faced in the past is a generation of blurry images. Isola et al. introduced a novel architecture of CGANs and demonstrated that GANs can be efficiently used for Image Colourizing tasks [4], which implies filling in blank or empty spots in images with color relative to surroundings. The unique idea proposed by Isola et al. is in the discriminator, *Patch*GAN, that tries to classify if each $N \times N$ patch in an image is real or fake. Thus, it models the image as a Markov random field, assuming independence between pixels. This architecture helps overcome the issue of blurry images and can be regarded as a form of style loss.

## 4 Problem Statement & Proposed Solution

Animation is a field that uses computer graphics. All works are done by computer programs, and many technologies have been developed to make this easier. But in the end, it is the same that the user has to draw the character. However, if you are not an expert at drawing anime characters or if you are doing it as a hobby, drawing and coloring a new character every time can be a huge burden. If an user selects the options(sex, hair length, etc...) you want and an artificial intelligence model draws and paints it, it will be able to bring about an innovative change in the creation of animated characters.
Therefore we propose an animation character face creation automation web app. It will include the processes described below.

**Drawing** Users can draw the face of an animated character on the sketchpad. You can also select the option(sex, hair length, etc...) that the artificial intelligence model randomly generates instead of drawing it yourself. All images at this stage are black and white.

**Coloring** Coloring is also user selectable. A sketch pad is provided for users to color in the desired color. If you don't want to color it yourself, you can also leave it to the AI model[4] to color it at random.

**Editing** Users may not be satisfied with the result. If there are any parts that user wants to edit for coloring or drawing, the user can freely edit the final image.

**History** When sketching or coloring using an AI model, it is randomly generated, so you may try to find the best one after several attempts. For this, we provide a history gallery. The history gallery allows you to recall or save previously created or modified images.

## 5 Planning

### 5.1 Dataset

**Image Data Collection.** It is clear that image dataset in high quality is a must for the high-quality image generation. There is well-known saying that goes as "Garbage in, garbage out". Thus, we need to provide a good-quality data, if we want to expect decent results. Web services hosting images such as Safebooru[6]

[6] https://safebooru.org/

provide access to a large number of images of anime characters. In addition to that, there a many Anime Face Datasets available online and hosted on Kaggle[7]. However, we noticed that most of them provide female characters dominantly, hence most of the projects online, such as Make Girls Moe [9], generate female characters only. In contrast to these projects, we aim to provide users a full control and freedom to create any character of interest. Therefore, in addition to available datasets online, we are going to crawl images of male characters and address the issue of this imbalance. Following other works, we are going to use lbpcascade animeface [8], an open-source project, to detect anime faces and zoom out the bounding box by a rate of 1.5x in order to capture the full style.

**Tag Estimation.** In order to provide a user to control a generation process, we are going to apply Illustration2Vec [11], which is a CNN-based tool for estimating tags of anime illustrations for our purpose. Given an anime image it outputs a set of predictions each belonging to a certain tag out of 512 kinds of general attributes, for instance 'smile' or 'open mouth'. We are going to pick a set of suitable attributes, including hair color, eyes color, smile and open mouth, and use them for training. Following prior works, for set of tags with mutual exclusivity, such as hair color or eye color, we choose the one with maximum probability from the network as the estimated tag. Meanwhile, for the rest of the attributes, such as smile or open mouth, we will employ a threshold value of 0.25.

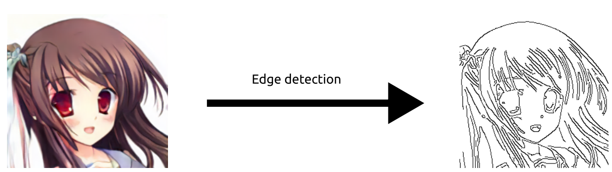


**Fig. 1.** Color image to Black&White image conversion.

[7] https://www.kaggle.com/
[8] https://github.com/nagadomi/lbpcascade_animeface

**Image Data Processing.** Obviously, we want to provide users a full control, hence we want to allow users to paint images themselves in case they do not want to use tags that are limited to a training set data distribution. Therefore, users can get Black & White (B&W) images as well. For this task, we can apply common image processing techniques, such as Canny edge detector algorithm, and produce corresponding B&W images, as shown in the Fig. 1. In this example, we applied a Bilateral Filter to reduce a noise before passing it to a Canny edge detector (provided by Opencv[9]), which is followed by a bitwise not operator to invert B&W.

### 5.2 Generative Adversarial Network

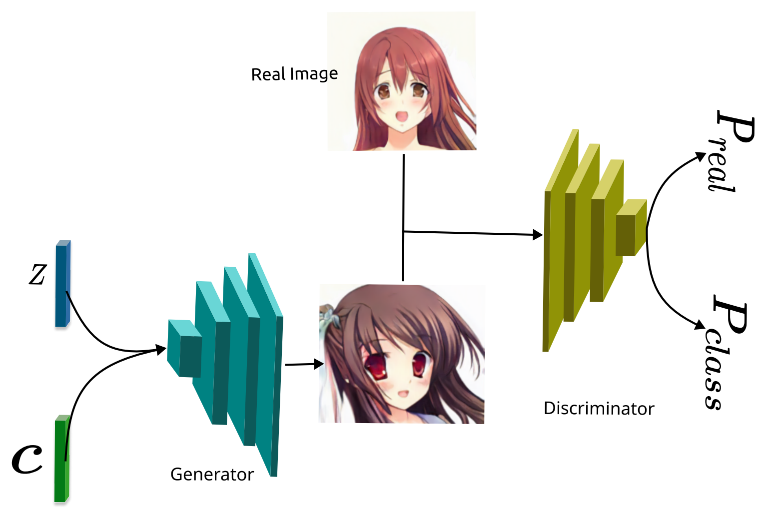


**Fig. 2.** Image generation flow.

**Generation.** Figure 2 summarizes the flow of image generation process. The different colors for the Generator and Discriminator networks emphasize their different architectures. As seen, in order to control image generation, we concatenate a random noise vector $z$ with one-hot encoded tag vector $c$, responsible for classes. Finally, the discriminator produces both the probability of being real and class probabilities to ensure consistency. There many architectures we have considered to tackle the task; however, we decided to use a slightly modified DRAGAN, originally proposed by by Kodali et al.[10], following [9], where a

[9] https://opencv.org/

multi-label classifier is added on the top of discriminator network. In contrast to Wasserstein GANs, DRAGAN can be trained with the simultaneous gradient descent, making training much faster. Finally, the implementation of DRAGAN is quite simple and flexible making it easy to replace DCGAN in any GAN related tasks.

**Colorization.** The task of Anime Colorization can be viewed as Image-to-Image translation, meaning we want to translate our B&W image into a color image. However, it is important to provide a full control of this process by allowing users to leave some color marks using which a model can colorize a given image. Pix2Pix GAN is a special type of Conditional GANs, proposed by Isola et al., that has been widely adopted for a range of Image-to-Image translation tasks [4]. The generator is an encoder-decoder model utilizing a U-Net architecture, whereas the discriminator is the PatchGAN discussed earlier. Thus, having a source color image we can obtain training data by computing the edge map leaving some color hints, the example is illustrated by Fig. 3. To recap, the input to the Conditional GAN is a color-marked image, whereas the desired output is the corresponding color image.

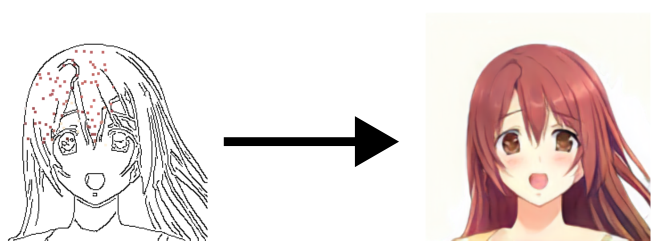

**Fig. 3.** Marked image to color image conversion.

### 5.3 Frontend

**Creation and Design of Splash Screen.** Many apps have a splash screen that show off their company's logo or themes before the web app loads. When our web app loads, we want to implement a splash screen to represent our group EDITH and our group member's name. Creation and design of our splash screen will be the first task as it will be the first thing users will see when opening our web app.

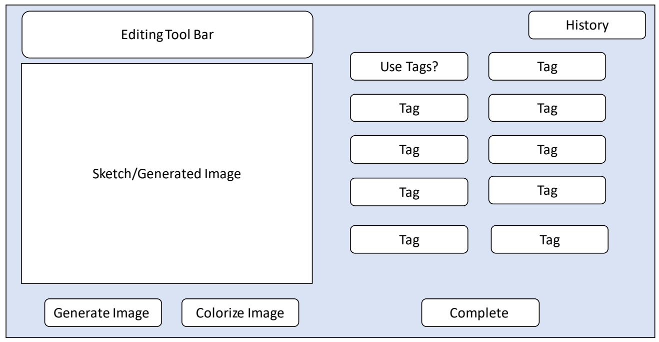


**Fig. 4.** Screen 1.

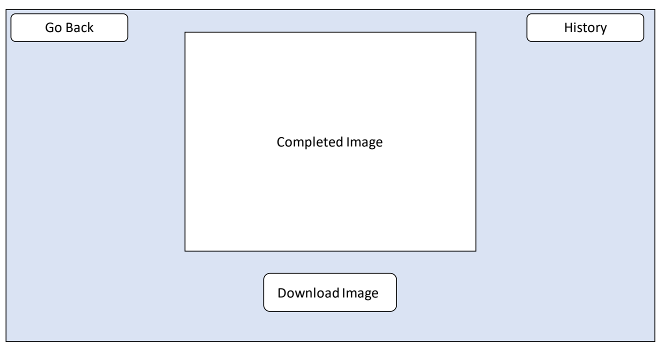


**Fig. 5.** Screen 2.

**Creation and Design of Screens.** We will be using React to create the front end of our web app. Using React, we will create and design different screens that will make up the whole web app. The first screen will be a photo editing screen like Fig. 4. The user has the option to either completely randomize their design or create their own design. There will be tags (drop down selectable menus) that represent key features of a person's face like hair style, eyes, nose, etc. There will also be a sketchpad where users can free draw a face or edit an already generated anime styled image. There will also be a button to generate image either randomly or from chosen tags and a button that colors the image. there will also be a button where the user can complete their image. There will also be an editing tool bar that lets user edit their image freely. The second screen will be a image saving screen like Fig.5. Here the user can download their image and their image will automatically be saved. There will be an image of the image created in the previous step along a button that lets the user download the image. They can also navigate to the history screen or previous screen. The third screen will be the image history like Fig.6. Every image saved is stored and is shown here. Here the user can download or delete or edit the image.

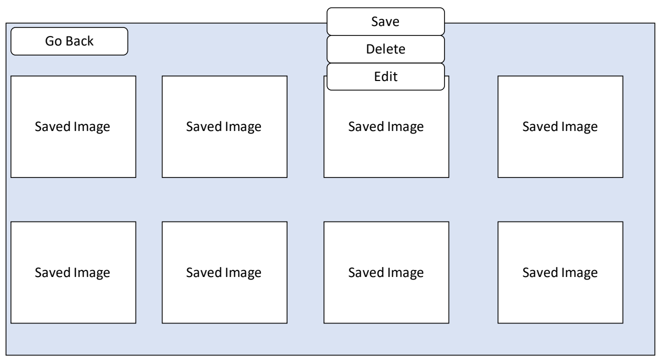


**Fig. 6.** Screen 3.

**Adding Editing Tools.** For the first screen, the photo editing screen. Each tag and editing tools will need to be implemented to work as intended. Each individual editing tool and tag can be subtasked and worked on throughout the semester as this is the main functionality.

### 5.4 Backend

The API server will be developed in Python django, and deployment will be through the AWS instance.

**GenerateSketch API** It is an API that receives the option tags selected by the user as a request, creates a character image through the AI model with the appropriate option tags, and sends the result image as the response.

**Colorize API** If the user gives a black-and-white sketch or an image randomly generated by the AI model as a request, it is an API that colors it using the AI model[4] and sends the result image as a response.

**History API** When the user clicks the history button, it is an API that caches the images that have been created and colored so far, and sends a response in the form of an image list.

## 6 Milestone & Team Cooperation

Table 1 depicts an approximate plan of our work. The roles are divided and shown under the *Person* column, whereas the latter column reveals all the subtasks described above. Finally, we are following Scrum Methodology, which is an agile

| Month | Subtask | Person |
|---|---|---|
| Oct. | Image Data Collection | Chingis, Soohun |
| | Processing | Chingis, Soohun |
| | Creation and Design of Splash Screen | Eunmin |
| | GenerateSketch API | Gone He |
| Nov. | Train Generation Model | Chingis |
| | Colorization Model | Chingis |
| | Creation and Design of Screens | Eunmin, Gong He |
| | Adding Editing Tools | Eunmin, Gong He |
| | Colorize API | Soohun |
| | History API | Soohun |
| Dec. | Testing | All |
| | Bug Fixing | All |

**Table 1.** An approximate plan of our team.

development methodology based on an iterative and incremental processes. Thus, we are going to have weekly meetings and discuss the progress as well as other issues, including subtasks, ideas and difficulties, via Google Meet[10].

## 7 Conclusion

Thus, with the development of technology, anime is still growing massively as a business area, and there is a huge demand and potential. There are many related projects but they are limited in terms of control provided to users. Also, current datasets have an imbalance problem in terms of female and male characters, making models biased towards female character generation. Inspired by the observations, our team, EDITH, wants to provide a full control of anime character generation process by allowing users to sketch and create their own characters in end-to-end manner. We will create a web app with a user-friendly interface, where users can freely edit images, and provide a set of useful GAN-based features, namely conditional character generation and colorization.

[10] https://meet.google.com/

## References

1. Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.
2. Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016.
3. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *In Advances in neural information processing systems*, pages 2672–2680, 2014.
4. Phillip Isola, Jun-Yan Zhu, Tinghui Zhou and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. *arXiv preprint arXiv:1611.07004*, 2018.
5. Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.
6. Mark Serrels, CNET Crunchyroll and Funimation merger: What does it mean and what happens next?, https://www.cnet.com/news/crunchyroll-and-funimation-merger-what-does-it-mean-and-what-happens-next/. Last accessed 29 Sep 2021
7. Report Overview, https://www.grandviewresearch.com/industry-analysis/anime-market. Last accessed 29 Sep 2021
8. Otaku Coin Cryptocurrency Wants to Create an Isekai Anime with NFTs https://www.animenewsnetwork.com/interest/2021-09-10/otaku-coin-cryptocurrency-wants-to-create-an-isekai-anime-with-nfts/.177170. Last accessed 29 Sep 2021
9. Make Girls Moe Homepage, https://make.girls.moe/. Last accessed 29 Sep 2021
10. Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. How to train your dragan. arXiv preprint arXiv:1705.07215, 2017.
11. Masaki Saito and Yusuke Matsui. Illustration2vec: a semantic vector representation of illustrations. In SIGGRAPH Asia 2015 Technical Briefs, page 5. ACM, 2015.