A Haircut Recommender System with EfficientNetV2S for Face Shape Classification

Anastasiia Chaikovska

Abstract: The way individuals present themselves has a significant impact on their mental health and self-esteem. However, there is still considerable scope for exploring the potential of personalised styling guidance. This article outlines the development of a face shape-based haircut recommender system, which offers tailored hairstyle suggestions. The system employs a dataset of 4,998 celebrity images categorised by face shape, utilising transfer learning with EfficientNetV2S to achieve a classification accuracy of 97% through optimised pre-processing and training strategies. A minimal viable product was developed as a user-friendly web application using Streamlit, which enables users to upload or capture photos and receive customised haircut recommendations. The application incorporates backend processing for face detection and classification and frontend features for delivering personalised style guidance, supported by a curated database of haircut examples. The solution was well received at the KIRO2024 Exhibition, showcasing its practicality and appeal. Future enhancements will include expanding recommendations to accommodate hair types, integrating male styling options, and offering augmented previews of suggested styles directly on user images.

1 Introduction

It is evident that the issue of how to look is not a significant concern in comparison to other global challenges such as unemployment, armed conflicts and natural disasters. However, the way an individual presents themselves can have a profound impact on their mental health and well-being. It can lead to low self-esteem, negative body image, eating disorders and depression [1].

It is important to recognise that each individual has their own unique body shape, face shape, skin colour, hair type and colour. Consequently, the same style may appear differently on different people, with varying degrees of impact.

The term "style" is broad and encompasses haircuts, makeup, clothing, and accessories. An ideal style recommender provides guidance on haircuts that complement individual face shapes, the best colors for makeup and clothing based on skin, eye, and hair colors, and clothing styles that flatter specific body shapes.

While not the sole component of an ideal style recommender, a haircut recommender system can assist in reducing negative body perception.

2 Related Work

There have been a number of attempts to develop a recommender system of this nature. One of the aforementioned systems [2] was implemented as a mobile AI-based hairstyle app, which provided users with haircut recommendations based on face shape and occupation. The incorporation of occupational data resulted in a slight reduction in the system's user-friendliness. This app is currently unavailable.

There are still mobile apps [3, 4, 5] that enable users to edit faces, add makeup or change hairstyles. However, these apps are not recommender systems; they simply provide editing capabilities.

In [6] was developed a prototype of a recommender system that delivered three haircut recommendations. The Azure Face API detected faces in images and extracted face-related attributes needed for their recommender system. The API of Beta Face was used to retrieve information on the face shapes detected in the dataset's images

Regarding the face shape classification, there is a significant amount of research conducted using a variety of algorithms. In [7], the author utilises K-nearest neighbour (KNN) and a random forest classifier. All of the models produced exhibited signs of overfitting. In [6], an MLP was selected for the classification task, achieving an accuracy of 58.8% for 10 recommendations. In [8], the authors employed the VGG-Very-Deep-16 network structure, achieving a validation error of 23.7%. The most successful approach was that described in [9], whereby the authors applied transfer learning with EfficientNetV2S, achieving an accuracy rate of 96.32%.

3 Face Shape Classifier

In developing my recommender system, I sought to utilise the most accurate method, and thus attempted to recreate the approach outlined in [9]. Despite attempting to reproduce the results using the same hyperparameters as the original, the number of trainable parameters differed, and the claimed accuracy rate was not achieved. It was therefore assumed that some of the model building blocks had not been incorporated into the paper. As a result, improvements were made to the data preprocessing and fine-tuning, leading to a final accuracy rate of 97%.

3.1 Dataset and Preprocessing

For training the neural network I used a Face Shape dataset [10], which consists of 4998 images of female celebrities categorized according to their face shape. There are five face shape classes: Heart (999 images), Oblong (999 images), Oval (1000 images), Round (1000 images) and Square (1000 images).

To enhance classification performance, background and other image noise were eliminated. There are three distinct phases to the preprocessing stage (Figure 1):

- **face detection** - get a face area bounding box
- **face cropping** - the face area is cropped into a square face image
- **image scaling** - all the cropped images are resized to 150px x 150px, which is the input size for the neural network (in my implementation this stage goes during loading of the dataset)

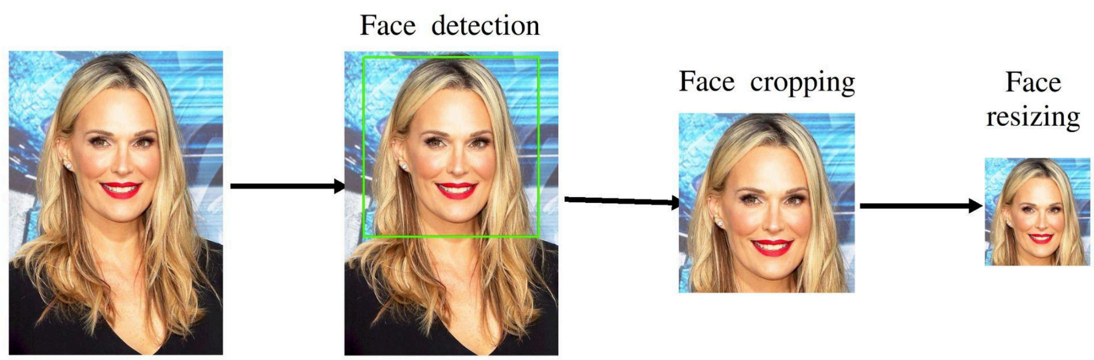

Figure 1. Three phases of the preprocessing

I improved face detection accuracy by 18% by replacing a HOG-based detector described in [9] with a MtCNN [11], which resulted in successful face detection across all dataset images.

The dataset was split into a train (80%) and test (20%) set. The train set was further split into a train (80%) and validation (20%) set.

3.2 Data augmentation

The augmentation method increases the number of images in the dataset while making it more difficult for the network to learn, because none of the images are completely standard. With OpenCV I implemented all variants from the [9] (Figure 2):

- image rotation for a value between -50◦ and 30◦
- adding Gaussian noise to an image
- horizontal image mirroring
- changing the contrast of an image for gamma value of 2

Additionally I added augmentation during training:

- random zoom with preserved aspect ratio in the range (-10%, 10%)
- random change of the contrast in the range (-20%, 20%)
- random change of the brightness in the range (-40%, 40%)

Figure 2. Data augmentation with OpenCV and Keras augmentors

3.3 Model and Transfer Learning

Following the recreation of the model from [9], which did not yield the same results as originally claimed, an effort was made to identify the optimal hyperparameters through a trial-and-error approach.

Stage	Operator	Stride	Channels	Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

Table 1. EfficientNetV2S model

Ultimately, my model is as follows:

- **base model** - I downloaded EfficientNetV2S (Table 1) with pre-trained Imagenet weights as a base model but without the top layer and with average pooling on the top. The base model contains batchnorm layers, so to keep weights from going wild the base_model is running in inference mode. For initial fit the base model is frozen
- **top layer** - on the top I added dense layer with 5 neurons (number of face shape classes) and softmax activation function
- **optimizer** - Adam optimizer with 0.0001 learning rate
- **loss** - Sparse Categorical Cross Entropy Loss
- **batch-size** - 32
- **dropout** - 0.2

The model was trained in three stages (Figure 3):

- **initial fit** - the base model was frozen, only the top dense layer was trained for 100 epochs and the best weights were saved
- **fine tune the 7th stage** - unfreezed the 7th stage of the EfficientNetV2S and trained it for 169 epochs
- **fine tune the 6th and the 7th stages** - unfreezed the 6th and the 7th stages of the EfficientNetV2S and trained it for 43 epochs

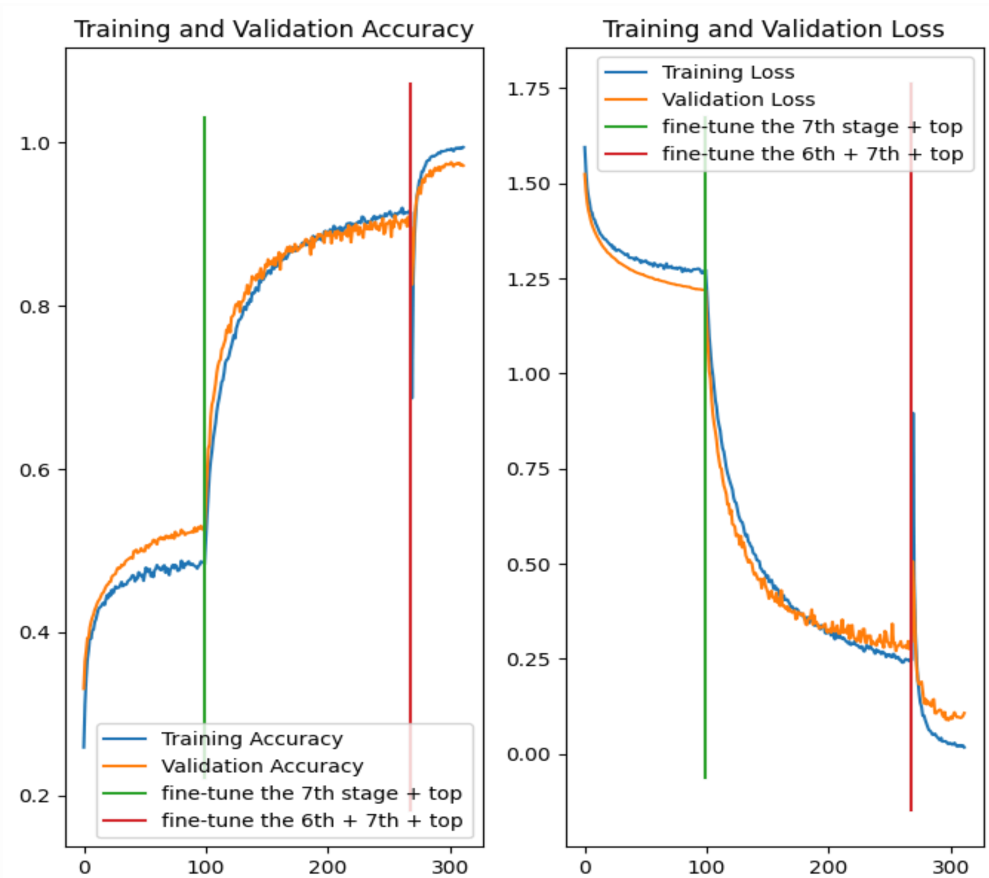

Figure 3. Training and validation accuracies and losses

The test batch was evaluated, and the results showed a significant improvement in accuracy, with the rate increasing from 0.5282 to 0.9736. Additionally, the loss was reduced considerably, from 1.2078 to 0.0973. The prediction outcomes on the test images are presented in Figure 4 for reference.

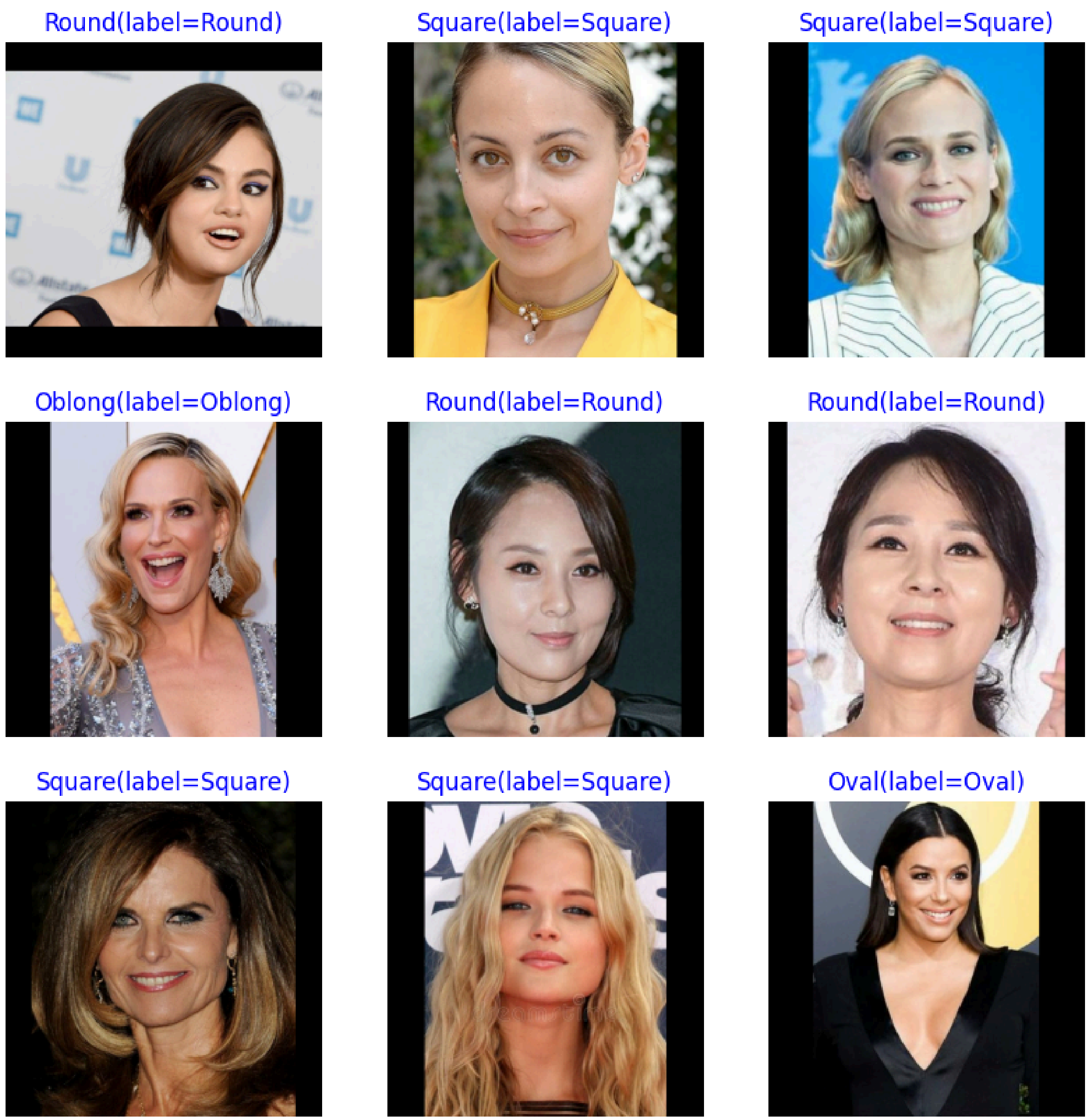

Figure 4. Training and validation accuracies and losses

4 Recommender

The objective was to create a minimal viable product, namely a user-friendly web application that enables users to upload or take a photo and receive haircut recommendations. To create this WebApp, I had to find a solution that met the following criteria:

- The ability to rapidly develop a fully functional app, ideally in Python
- Easy deployment and maintenance
- No cost

Streamlit [12] met all of my requirements:

- It is simple to create UI elements using Python code, eliminating the need for additional implementation steps
- Deployment is straightforward. Simply connect your GitHub account and point to the repository and branch. Each push or commit will invoke a webhook to update the code
- Streamlit apps can be deployed anywhere, but the Streamlit Community Cloud is available as a free option, offering:
    - The CPU requirement is 0.078 cores, with a maximum of 2 cores
    - The memory requirement is 690MB, with a maximum of 2.7GB
    - The storage requirement is unlimited

A comprehensive collection of haircut recommendations for each face shape was collated through an extensive review of fashion and style articles. Images of hairstyles for various lengths were created with the help of Stability AI [13].

The WebApp system is a client-server structure with both the frontend and the backend written in Python. Data flow can be seen in Figure 5.

Backend

During the initial run, the server loads the weights for the face shape classifier and caches the haircut recommendations in JSON format, along with example pictures. This optimises the app's performance. The uploaded image is processed by a face detector, the face is cropped and resized to 150px x 150px, and it is then sent to the face shape classifier. The resulting data is then transmitted to the front end.

Frontend

Once a photo has been uploaded or taken, a face shape classification is determined. Based on this classification, text recommendations are provided on how to flatter the specific face shape, along with example images of haircuts suitable for various hair lengths, including short, medium, and long.

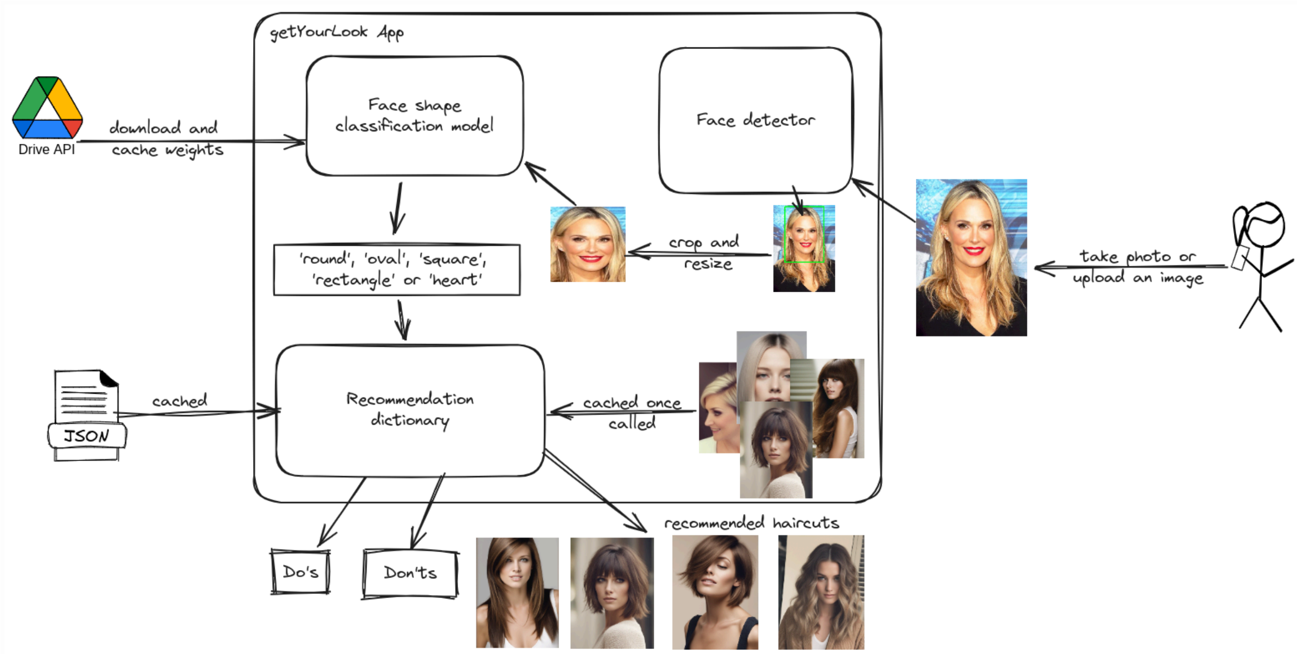

Figure 5. App data flow

6 Conclusion

At the KIRO2024 conference exhibition, the product was showcased at the "Women in AI&Robotics" booth. User feedback was positive, with respondents noting the product's ease of use and enjoyment factor. WebApp (Figure 6) is available at getyourlook.streamlit.app.

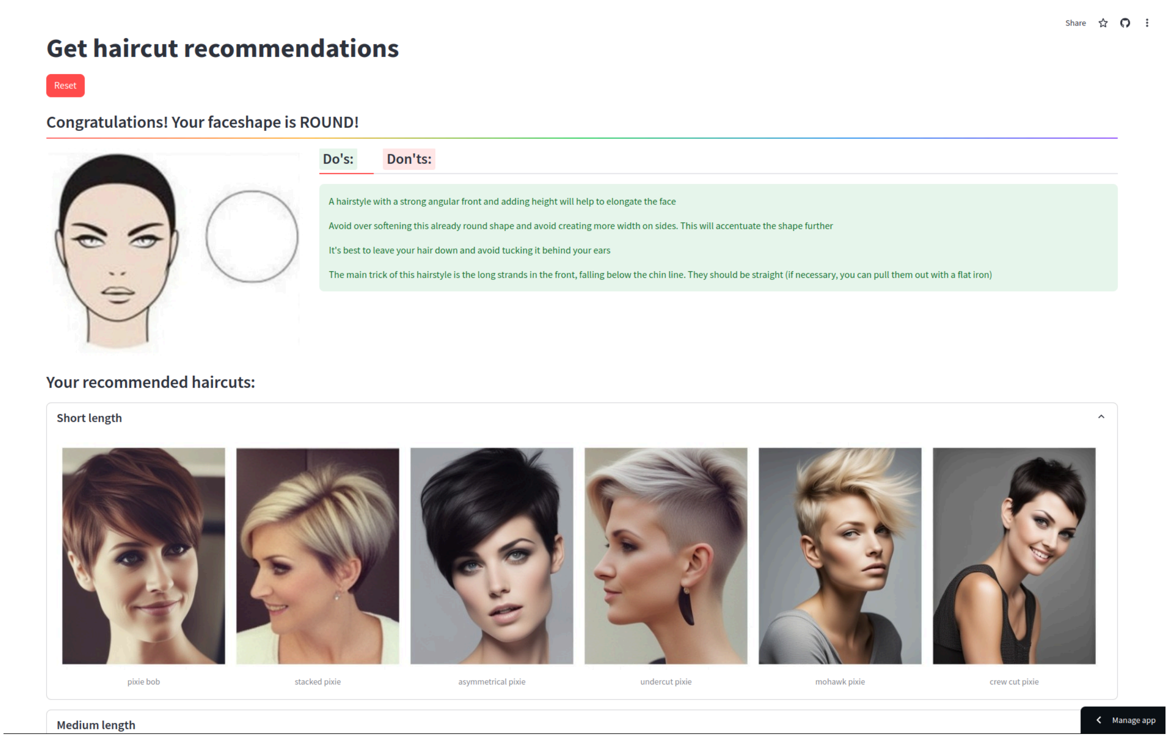

Figure 6. getYourLook WebApp

6.1 Future improvements

Apply style to the photo

To enhance the user experience, it would be beneficial to generate images from the input image and recommendation prompts. This would allow the user to view all recommended haircuts applied to their image, rather than a random selection.

Men haircuts

Currently, the model has been trained only to recognize and style female haircuts, and has been exposed only to images of female celebrities. To improve accuracy, the model should be retrained with images of male faces.

Hair type

It would be beneficial to consider hair type when making recommendations, as a user with curly hair may not benefit from a haircut with straight hair. Additionally, the appearance of certain haircuts may vary depending on hair thickness.

Sun-glasses shape

An individual's face shape can inform decisions regarding not only haircuts but also the shape of sunglasses and the application of makeup.

References

[1] Mandy French, “What is the connection between beauty standards and mental health?” - May. 2024
[2] Pallab Paul, Face It: The artificially intelligent hairstylist - 2017
[3] FaceApp: Perfect Face Editor
[4] Peachy - AI Face & Body Editor
[5] Facetune: Hair, Photo Editor
[6] Sark Xing, Ward de Groot, Yizhou Liu, Lara Leijtens, “hAIr: A Hairstyle Recommender System Using Machine Learning” - 2019
[7] Yogesh M. Kamble, Raj B. Kulkarni, Recommendation System for Hairstyle Based on Face Recognition Using AI and Machine Learning - Jan 2024
[8] Zhuocheng Liu , Yi Ji , Yuxi Hu and Tao Zhan, Design and Implementation of Hair Recommendation System Based on Face Recognition - 2019
[9] Grd P, Tomičić I, Barčić E, Transfer Learning with EfficientNetV2S for Automatic Face Shape Classification - 2024
[10] N. Lama, Face shape dataset - 2023
[11] Zhang, K., Zhang, Z., Li, Z., and Qiao, Y., Joint face detection and alignment using multitask cascaded convolutional networks - 2016
[12] Streamlit
[13] stability-ai