MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Build a Virtual Dressing Room using Deep Learning

Nguyen Trong Duy
Bui Tu Anh
Nguyen Huy Hoang

Instructor
M.S.E Le Dinh Huynh

Bachelor of Artificial Intelligence
Hoa Lac Campus – FPT University
- December 11th 2023 -

ACKNOWLEDGEMENT

This thesis is dedicated to the unwavering support, invaluable assistance, and generous funding provided by M.S.E Le Dinh Huynh. His continuous guidance and belief in this work have been instrumental in bringing it to fruition. We extend our gratitude to FPT University for providing essential courses and resources that greatly facilitated the completion of this thesis. We also express our heartfelt thanks to our dedicated teammates for their remarkable efforts and significant contributions throughout this journey. Their collaboration has truly enriched the quality of our work. Furthermore, we extend our appreciation to our families and friends for their constant encouragement. Without their support, this thesis would not have been completed.

We are deeply grateful for the support and resources that have played a pivotal role in the successful culmination of this endeavor.

DECLARATION

This thesis is the culmination of our dedicated efforts, representing original work and excluding any collaborative contributions unless explicitly acknowledged within the text. It has not been presented, either in part or in its entirety, for the fulfillment of any degree, diploma, or other qualification at any university or institution. The content herein is authentic, and any collaborative endeavors are transparently attributed in the following pages.

Signed: _____

Date: _____

Nguyen Trong Duy, Bui Tu Anh, Nguyen Huy Hoang, and full qualifications
FPT University

ABSTRACT

With the rapid development trend of the fashion e-commerce industry, people are purchasing most of their items online and are spending more on them, especially fashion items, since browsing different styles and categories of clothes is easy with just a few mouse clicks. However, with the convenience that online shopping provides, customers tend to worry about how the image of a particular fashion item on the website will fit them. Additionally, retailers tend to automate sales steps with AI technology gradually. Hiring models or photo studios to advertise products takes a lot of time and money for fashion retailers. To solve this problem, 3D virtual fitting technologies have been created, but the difficulty of measuring the depth of clothing and body shape takes more time than 2D images. In this project, we build a virtual fitting system using Deep Learning technology with input from 2D images of people trying on clothes. Our product undergoes two key phases: initially integrating the HR-VTON and GPVTON virtual try-on methods with DressCode and VITON HD datasets, aiming to present high-fidelity visual representations of garments. The meticulous preprocessing of user-provided data involves six structured steps followed by established methodologies. Addressing challenges in the Openpose step, including finger keypoint loss, requires a comprehensive reiteration of the Human Parse for the DressCode dataset. Experiments with low-resolution images prompted reassessment, substituting resolution enhancement with StableSR after SRGAN phrase generation. Efforts to balance realism and fidelity lead to meticulous crafting of user interaction aspects, integrating interface and backend functionalities for a virtual dressing room with super high-resolution image upscaling. These measures aim to enhance user experiences within the project's scope. The preprocessing phase refined Densepose and Agnostic steps for improved detail integration into the Try-on model. Challenges in the HR-VITON method, generating fixed-resolution and low-sharpness images, lead to the use of StableSR for resolution augmentation, surpassing input image resolution. The Virtual Dress Room offers "Virtual Dressing" and "Upscaling Resolution" features, allowing users to virtually dress models and flexibly adjust image resolution based on preferences.

Additional Keywords: Deep Learning, Virtual Try-on, Image Super Resolution.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 Motivation

Nowadays, people are purchasing most of their items online and are spending more on them, especially fashion items, since browsing different styles and categories of clothes is easy with just a few mouse clicks. Despite the convenience that online shopping provides, customers tend to worry about how a particular fashion item image on the website will fit them.

Figure 1. Illustration of the benefits of dressing for models without having a photoshoot studio

Research shows that shoppers who use dressing seven times more likely to make a purchase than those who buyer without. Not only that, but if the customer has a good experience with a sales assistant in the dressing rooms, they can buy up to three times as many items in one transaction. Recent surveys of Vue.ai have shown that retailers are 3x more likely to buy products when shown on models most representative to them, with a 23% uplift in average order value and an increase of 11.5% in returns. This tool is completely driven by AI and is taking us to a point where an average shopper considers a virtual dressing room to be roughly equivalent in quality to the physical experience, which is shown in Figure **1**. Unlike a regular eCommerce experience, the Virtual Dressing Room gives shoppers an opportunity to become their own stylists and put together different looks and visual outfits on models on a screen from anywhere in the world. Product imagery is a vital component of retail e-commerce. These images help shoppers visualize what they may look like in person and on them. 75% of online retailer rely on product photos when deciding on a potential purchase. However, hiring models to take photos is costly for sellers.

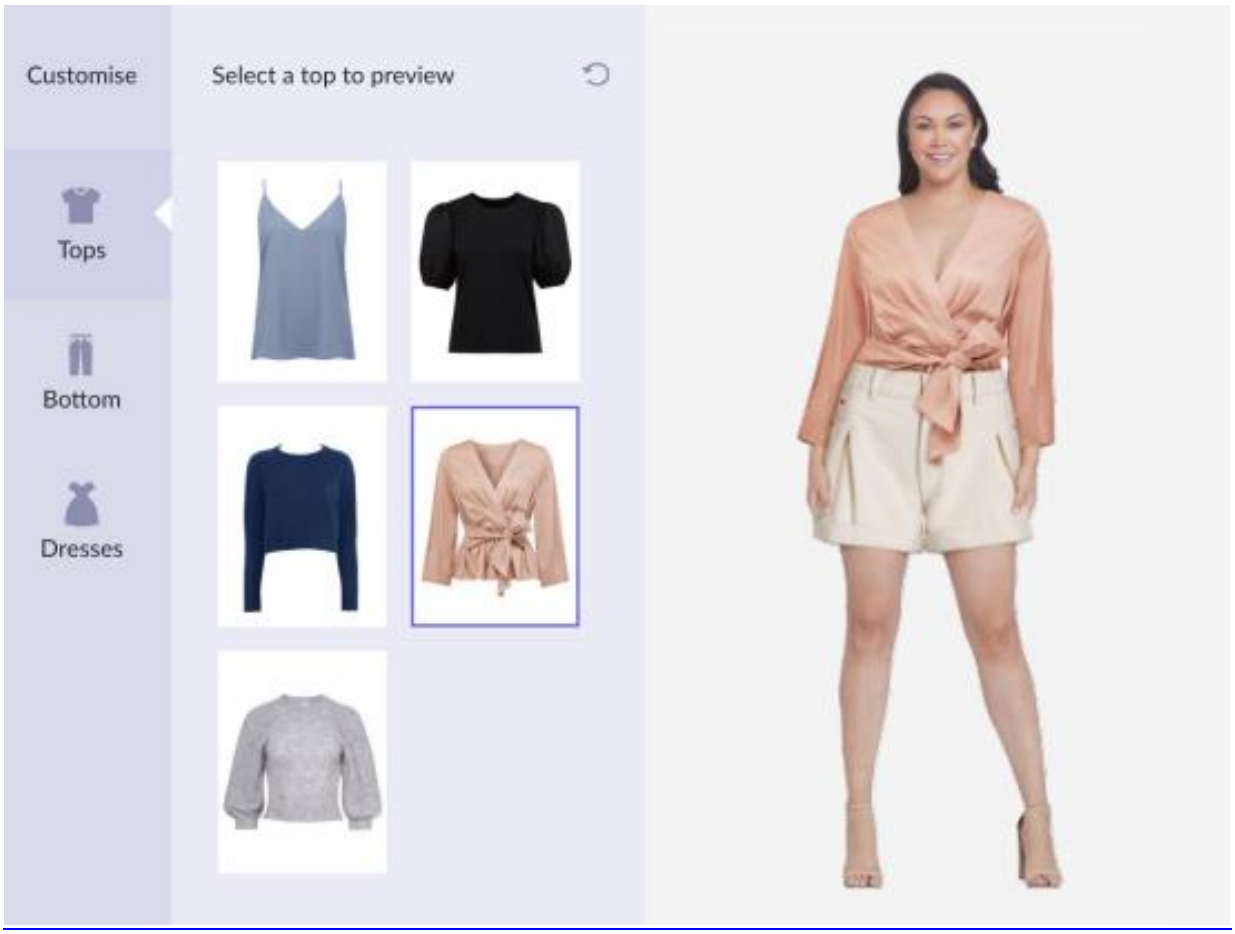

Figure 2. Example Virtual Dressing Room Platform (Vue.ai)

According to Vue.ai [1] in the US, product photos on models usually cost up to $500 for a single look, and in some cases, even more AI can save up to over 75% on photoshoot costs AI helps remove the need for many processes and resources required in traditional photoshoots. It removes a lot of the associated costs, making building product imagery a lot cheaper than what retailers have known till now AI can build product imagery five times faster than traditional processes. By reducing the number of processes and people involved, AI can build on-model product photos much faster than existing methods. This helps retailers go live with the products on their e-commerce site and start selling sooner. In short, virtual dressing rooms (Figure **3**) not only bring value to shoppers but also help sellers cut out the costly process of hiring models to take product photos.

Therefore, there is an urgent need to provide a quick and simple solution for virtual try-ons. Instead of using 3D information such as depth and structure for images, we believe simply relying on a regular 2D photo is the most convenient and quick way to satisfy this need.

1.2 Related works

1.2.1 Virtual Try-on

Recent advances in the field of visual modeling have shown significant potential to drive revolutionary changes across a variety of fields. In particular, Generative Adversarial Networks (GANs), specifically networks that exploit architecturally efficient ways such as StyleGAN or more recent versions such as Diffusion Models, have highlighted advances noteworthy set in image synthesis. Realistic image ,in the field of human synthesis, the methods considered ,popular mainly use frameworks based largely on the ,StyleGAN model [2] [3] [4] [5] to yield the synthesis results are remarkably accurate. For example, StyleGAN-Human [6] focuses on three key factors considered important to achieve superior human synthesis [6] [7] dataset intensity, dispersion measure of data and data uniformity. At the same time, approaches such as InsetGAN [7] combine perfectly with the output from various pre- and post-trained GANs, specifically targeted at generating disjoint anatomical segmentations. virtual environments are based on visual representations of the entire human body. Differentiated modeling dominates virtual testing methods, which are largely image-based [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21], following a strict two-phase generation framework. These methods automatically adjust the processing of garments from inventory to fit the desired shape, then it searches and aggregates the output by combining the detailed adjusted garment with a reference version. Due to the direct correlation between the changing and deforming quality of the garment and the degree of authenticity in the output products, another aspect that also contributes is important regarding the development of tissues. This deformation module can be accommodated within a comprehensive framework. Some previous methods [9] [11] [15] [19] [20] [21] make good use of neural networks to be able to infer sparse control points to provide conditions. Adjust clothing in the target's image. Thus, making it possible to allow correction of clothing deformations. In contrast, alternative methods choose to estimate [8] [10] [12] [13] [14] maps for buoyant flows [22] to model truly heterogeneous deformations, lightening Shows the relative pixel position in the source image to each corresponding pixel in the target image. This approach, somewhat unlike techniques adapted based on Thin Plate Splines (TPS), directly predicts dense correspondence for each pixel, capturing extremely complex deformations efficiently. results compared to higher clarity.

1.2.2 Image Super Resolution

Recent investigations have shifted the focus from explicit preferences to exploring preferences as implicit, finding enhanced performance in domains with marked differences [23] pioneered the use of semantic segmentation probability maps as a tuning mechanism and guide for Super Resolution (SR) in feature space. Subsequent research efforts [24] [25] [26] [27] [28] [29] [30] have made quite liberal and extensive use of pre-trained Generative Adversarial Networks (GANs) [23] [25] [27] [28] to explore relevant high-resolution latent spaces derived from low-resolution inputs. Although these previous methods and approaches have proven to be adequate, the amount of implications they use is often tailored to specific situations, such as categories. is believed to be limited or facial imagery,

limiting the ability to generalize to complex real-world retention-first SR tasks [24] [26] [30]. Potential priors that can be given as alternatives in the field of orthogonal SR include the combination of decomposers and Vector Quantized Generated Adversarial Networks (VQGANs) [31] [32] [33]. However, these methods have limitations due to incomplete prior representation or inaccurate feature alignment, thus yielding output results of unsatisfactory quality. However, profound advances in Super Resolution (SR) imaging [34] [35] [31] have yielded impressive results [32], that can effectively minimize these shortcomings. previously encountered in this field.

1.3 Objectives and Contributions

In this project, we underwent two distinct development phases. Initially, our focus revolved around integrating and deploying the HR-VTON virtual try-on method in conjunction with the DressCode 80G and VITON HD 12G datasets. This strategic endeavor aimed to facilitate the presentation of high-fidelity visual representations of garments. The preprocessing of user-provided data and clothing samples, preceding their input into the system's model, comprised a meticulously orchestrated sequence of six structured steps. These steps were meticulously delineated within the confines of the established methodology outlined in the respective literature. Additionally, our efforts extended to addressing challenges encountered during the Openpose step, particularly in instances where the keypoint on the finger was lost. This situation necessitated a comprehensive reiteration of the Human Parse for the DressCode dataset. Our exploration of the HR-VTON method encompassed experiments involving low-resolution images. This prompted a reassessment of the image enhancement stage subsequent to GAN-based phrase generation, resulting in the substitution of the resolution enhancement facet with StableSR. Moreover, to ensure a nuanced balance between realism and fidelity, we meticulously crafted the user interaction facet of our endeavor. This encompassed the design and integration of both interface and backend functionalities for the virtual dressing room, incorporating image upscaling capabilities utilizing super high resolution.
These measures served as a testament to our comprehensive approach aimed at enhancing user experiences within the scope of this project.

- In this project we are going to build the Virtual Dress Room application is equipped with two key features: "Virtual Dressing" allows users to dress the model virtually. The "Upscaling Resolution" feature enhances flexibility in resolution for both input and output images, catering to user preferences.

- Through the evaluation process, we selected two methods with high performance, HR-VITON and GP-VITON, for the application. The preprocessing phase of input

data, we further refined both the Dense Pose and Agnostic steps components to enhance their level of detail before integrating them into the Try-on model.

● The GAN model method is applied within the HR-VITON. However, the up-sampling stage in this method generates images with fixed resolution and lacks high sharpness. Because, the SRGAN method exhibits a weakness in preserving perceptual information during upscaling procedures. To address these issues, StableSR is employed to augment resolution, offering customization to a higher resolution than the input image.

2. APPLICATION ARCHITECTURE

2.1 System Architecture

2.1.1 Architecture Overview

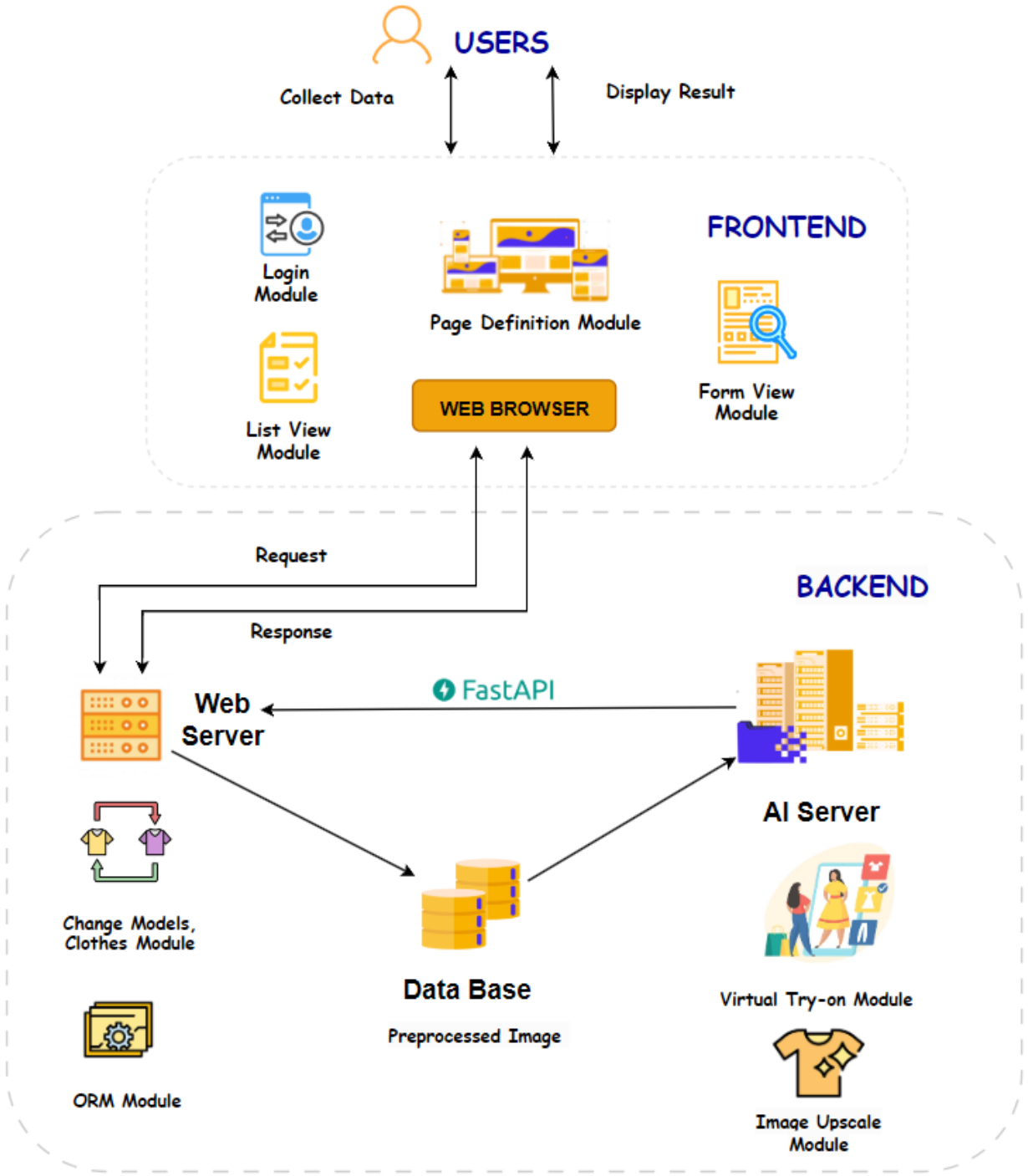

Figure 4. Proposed architecture

As shown in Figure **3**, the system architecture is divided into two main parts: the user interface (front-end) and the (back-end) side.

The user interface we designed includes integrated modules that make it easy to design user interface components and define display layout requirements:

- *Login Module*: allows the use of authentication and authorization features to review request authentication from user access.

- *The Page Definition Module* : is responsible for the specific interface, logic, and API configurations for the components considered in the Virtual Dressing Room website.

- *Form Viewer* : used to detail conventions for user interfaces that can facilitate viewing, creating, deleting, and changing information; is a CRUD operation.

- *The List Viewer* : takes care of displaying tabular data. And the view component templates list all HTML components and subsequent custom components.

Back-end handles all data operations and technical and AI server management.

- *ORM Module*: Helps map data records in the database management system to object form, serving to map API pre-process data for each model.

- *Virtual Try-on Module*: Generate images from try-on model on the AI server, then return the result as images.

- *Image Upscaling Module:* Upscaling image resolution from the super resolution model in the AI server

- *Change Models and Clothes Images Module*: Allows users to change input models and clothes images. Data uploaded has been processed and saved in the database serving the Virtual-Tryon module and upscaling module.

- *Virtual Try-on API :* Includes uploading model images, uploading clothing images, downloading try-on images, and converting input images to base64 format.

- *Image Upscaling API :* Includes uploading images, downloading images from super resolution model, and converting input images to base64 format.

- *Databases* : Are used for storing, maintaining, and accessing any sort of data. It can be relational, non-relational or customized.

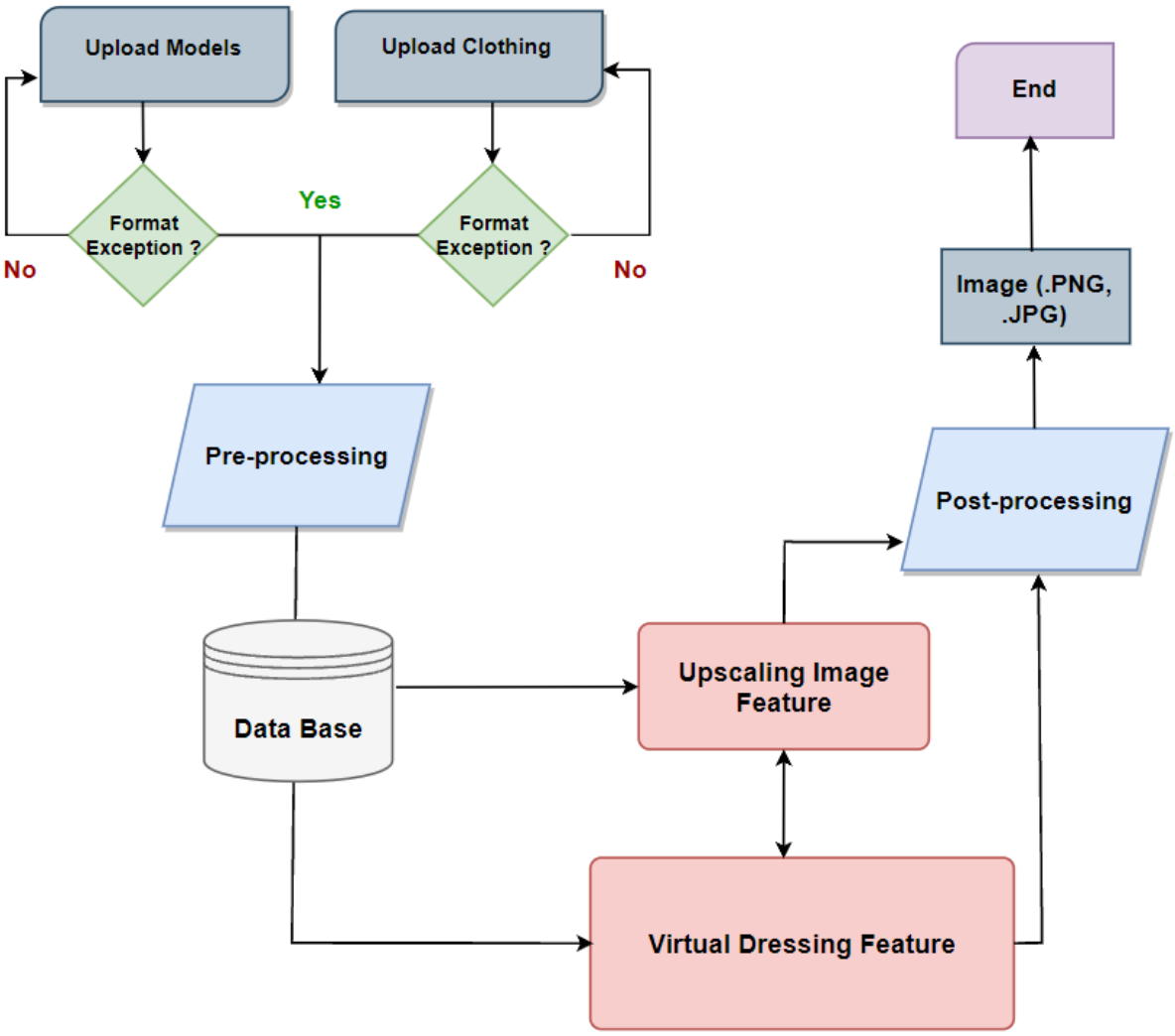

Figure 5. Pipeline of the Virtual Dressing Room application

The Virtual Dressing Room Application follows a defined pipeline in Figure **4**. Initially, users upload the model and clothing images. Subsequently, the exception module scrutinizes the input format; if it aligns with (.JPG or .PNG) image formats, the data proceeds to the Pre-processing module. Within this module, the data undergoes a six-step preprocessing phase before being archived in the database. The Virtual Dressing Feature retrieves the processed data from the database, leveraging a deep learning model to generate and display image results on the screen. At this juncture, users are presented with three options: first, if content with the image, they can progress to the post-processing step, converting it to optional (.JPG or .PNG) formats for download. Secondly, if unsatisfied

with the image's resolution, users may utilize the Upscaling Image Feature to enhance resolution before downloading the image. The final option enables users to upload low-resolution images; the system increases the resolution, integrates it into the virtual dressing feature, and subsequently provides a downloadable final image after post-processing.

2.1.2 Data Processing Module

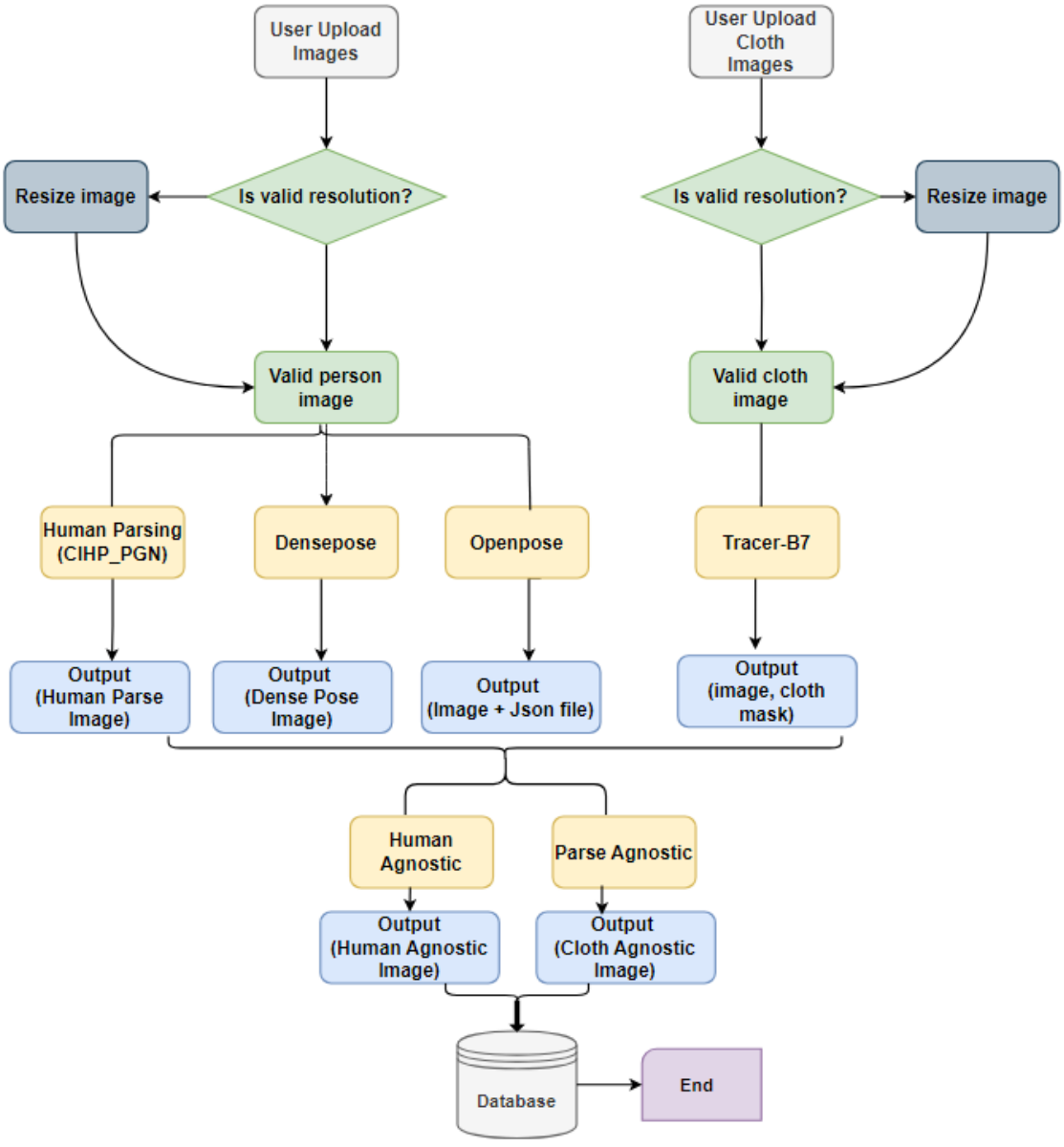

Figure 6. Data Processing Flow

The processing of user uploaded data is described in Figure **5**. Initially, the user's photo will be checked to see if it is the valid resolution, if not, the photo will be adjusted to the

appropriate size. After that, human images will go through three methods to process the data: Human Parse, Densepose and Human Pose Detection. These three methods of processing person images are independent of each other at this step. Also in this step, the garment image will pass through the Tracer-B7 model to generate an image called "cloth-mask", which separates the garment from the background of the image. Then, the results of the above are gathered for the next step including two methods Human Agnostic and Garment Agnostic, returning images showing the areas of the body identified to wrap new clothes when the image goes through the try-on model. After the entire above process, the original image along with the results obtained will be pushed into the database, to make it more convenient for use.

2.1.3 Virtual Dressing Room Feature

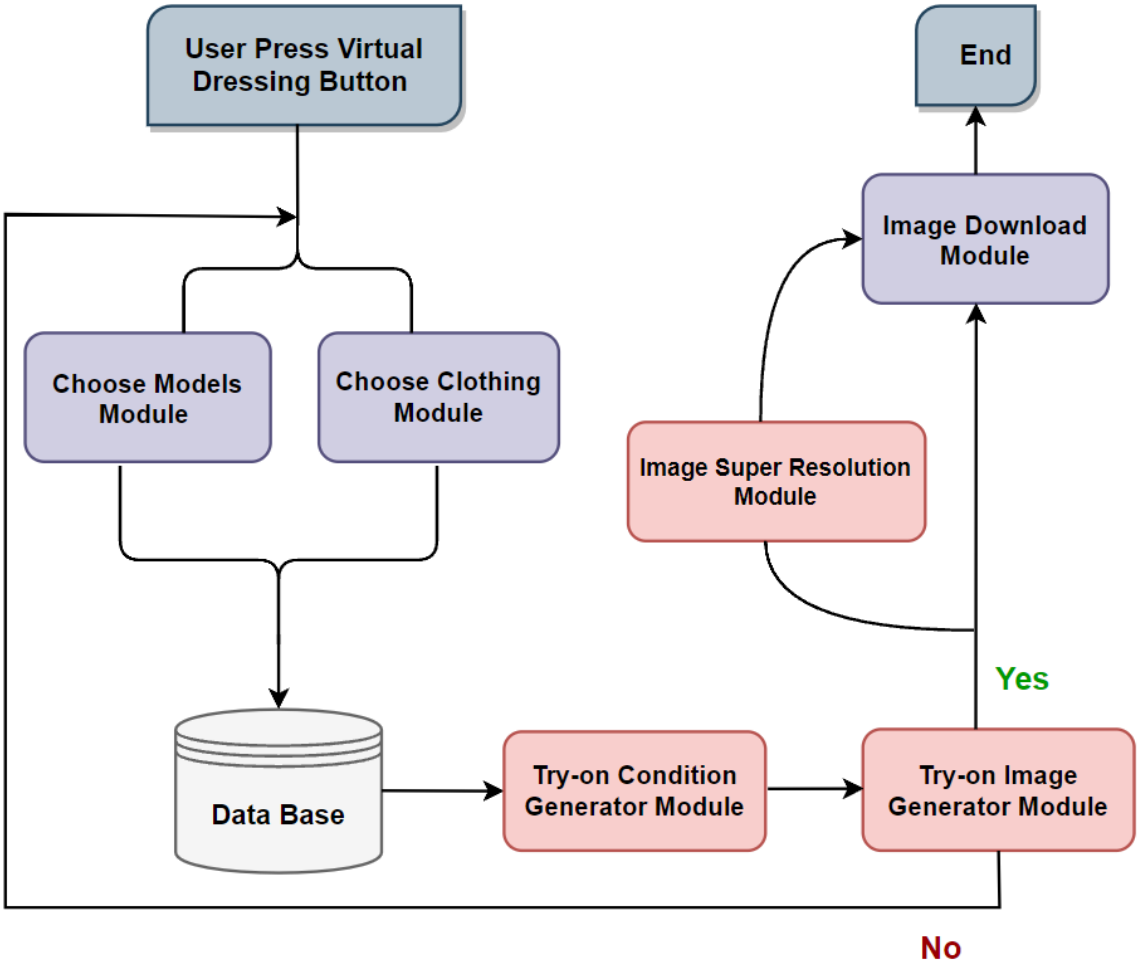

Figure 7. Virtual Dressing Room Feature Flow

Initially, when the user presses the virtual dressing button, the virtual dressing feature

retrieves the processed data from the database, leveraging a deep learning model to generate and display image results on the screen. Then the images in the database will go through the Try-on Condition Generator Module to generate model images, and the clothes will be processed according to the conditions. The image is then passed through the Try-on Image Generator Module to attach warping clothing images to the model. Then, users are presented with two options: first, if they are content with the image, they can progress to the post-processing step, converting it to optional (.JPG or .PNG) formats for download, or they can return to the image selection step. shirts and models to continue to choose from. Secondly, if unsatisfied with the image's resolution, users may utilize the Image Super Resolution Module to enhance resolution before downloading the image or may return to the step of selecting clothes and models to continue try-on show in Figure **6**.

2.1.4 Upscaling Image Feature

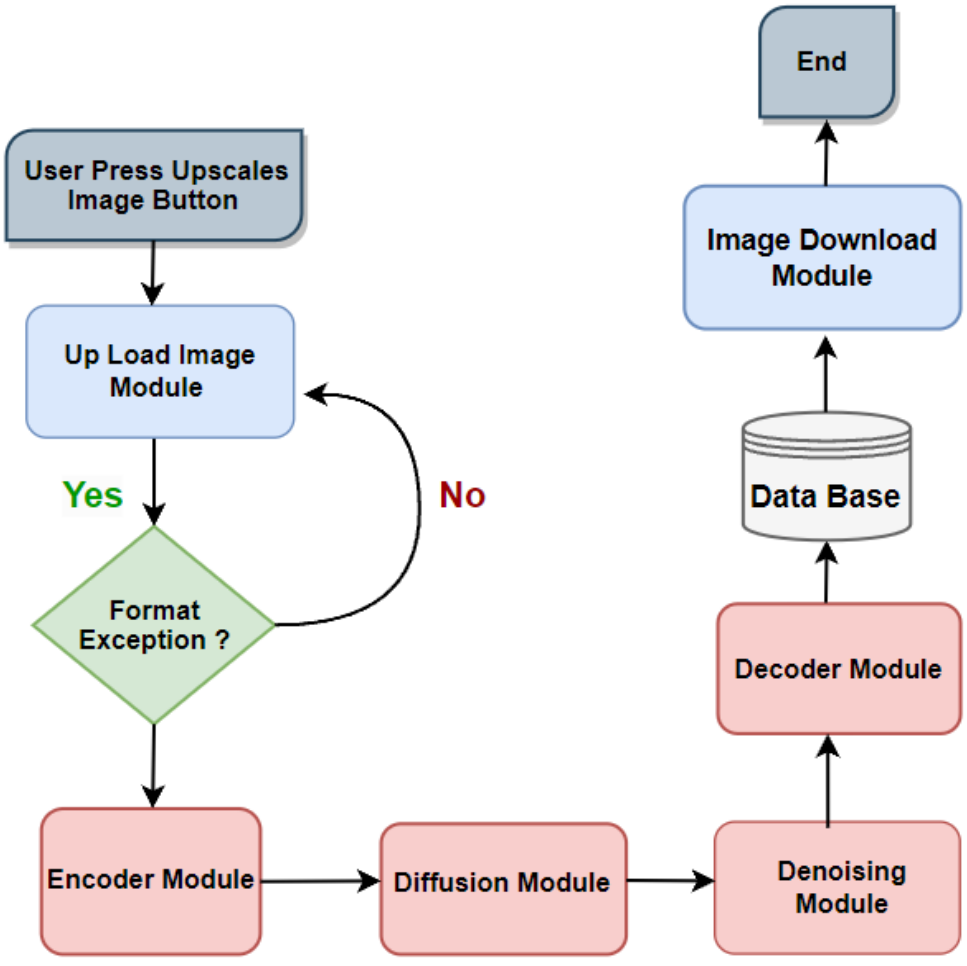

Figure 8. Upscaling Image Feature Flow

At first, the user presses the virtual dressing button, and then the data is uploaded from the input image. The image undergoes a check for format exceptions. If it complies with the

format (.JPG or .PNG), it proceeds to the encoder module for processing. The image is transformed into vector encoder format. Subsequently, in the diffusion step, the image module undergoes spatial feature transformation (SFT) and noise addition as per the diffusion mechanism. Following this, in the denoising module, the image is denoised using the U-Net diffusion mechanism, enhancing its resolution while decoding and retaining crucial photo information. The Decoder Module computes the decoder and generates the resultant image. Afterwards, the image is saved in the database before potential download for virtual dressing purposes, if required by the user. Finally, the higher resolution image can be downloaded in two size options: x2 and x4, compared to the original image, based on the user's preferences from the image download module show in Figure **7**.

2.2 Model Application Overview

2.2.1 Data Processing Implementation

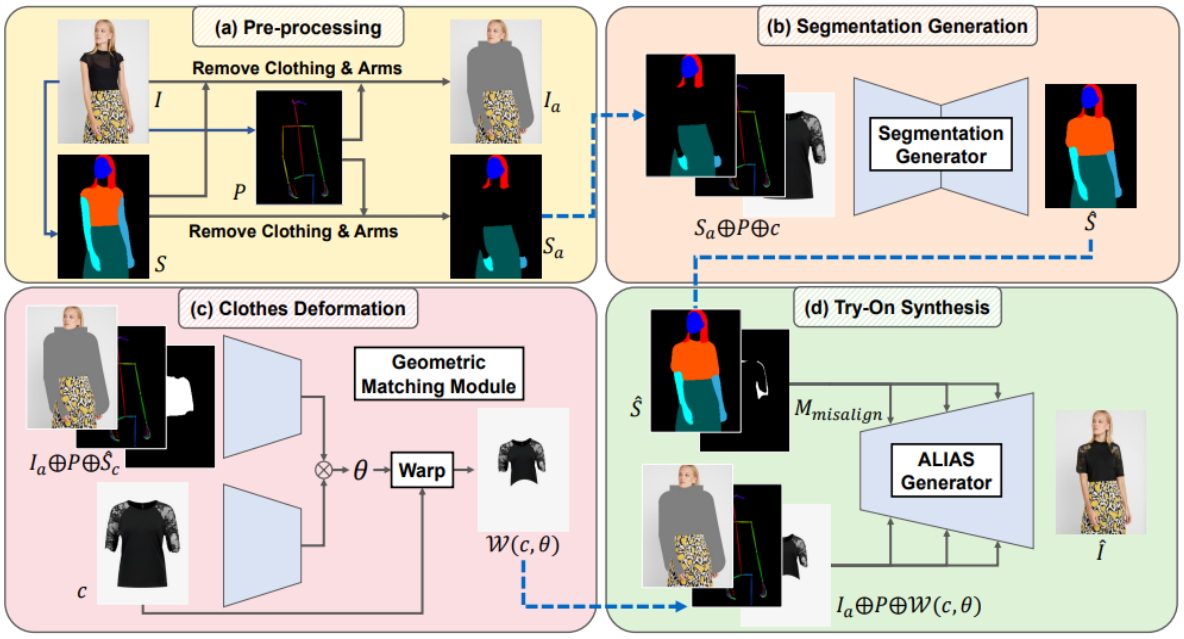

Figure 9. Overview of a VITON-HD [10]

The primary objective is to enable our system to efficiently process and produce high-quality photos, irrespective of their source. To accomplish this, we have developed a multi-step preprocessing procedure. We advise that photos ideally exhibit a well-lit background with distinct edges delineating the background from the subject (human or garment).

In order to achieve the goal that create a synthetic image of the same model wearing target garment, where their details are preserved, human pose estimation accounts for the largest proportion in the entire procedure.

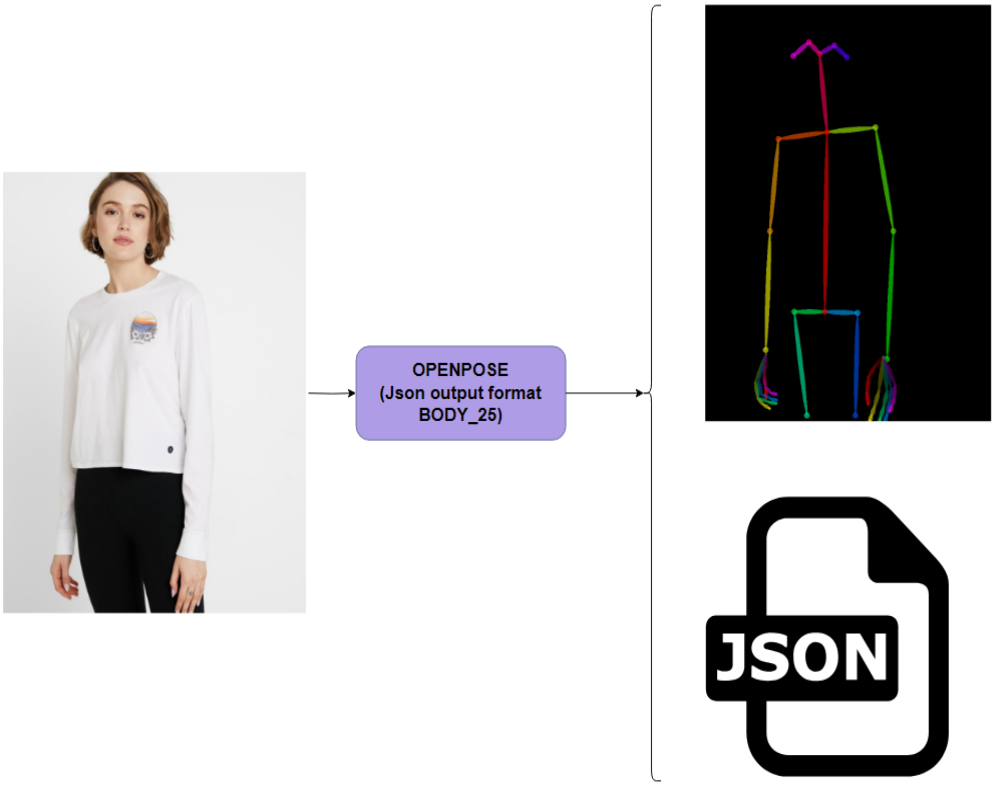

Figure 10. Input and output of detecting human key points

We use Openpose [36] to detect human key points on the input image, which will be returned as a JSON file consisting of a set of key points whose ordering is related to the UI output as the image accompanies in Figure **9**. Variations in human poses lead to different deformations of clothing, and the pose format output we chose is BODY_25, so the computed pose of a person is represented as coordinates of 25 key points.

Along with Openpose, PGN [37] is a deep learning method for semantic part segmentation, instance-aware edge detection, and instance-level human parsing built on top of Tensorflow. The PGN is trained and evaluated on the Crowd Instance-Level Human Parsing (CIHP) Dataset for instance-level human parsing. We applied that CIHP_PGN model, which is shown in Figure **10**, in our preprocessing procedure for the purpose of recognizing each semantic part (e.g., arms, legs, clothes).

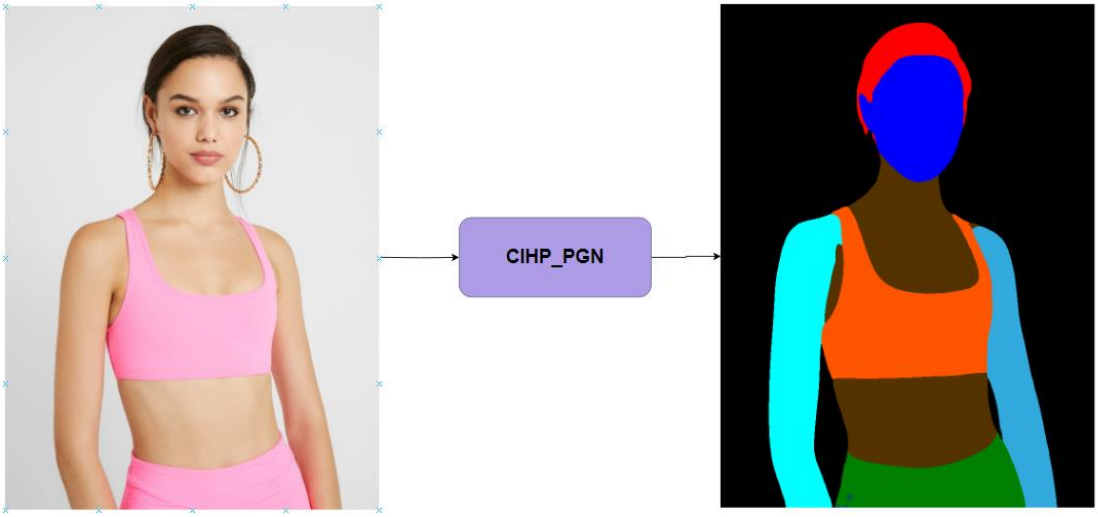

Figure 11. Input and output of recognizing semantic parts

For the pose map, we used Densepose [38] to generate dense pose estimation, which maps all pixels of the person's regions in the RGB image to the 3D surface of the person's body, which is shown in Figure **11**.

For getting the same image as VITON-HD dataset, we changed "alpha" from 0.7 to 1 and "inplace" is set to False in "DensePoseResultsFineSegmentationVisualizer" class from file **densepose_results.py** when initialized that class. Garment processing is also a critical part in the procedure, and we use the Tracer-B7.

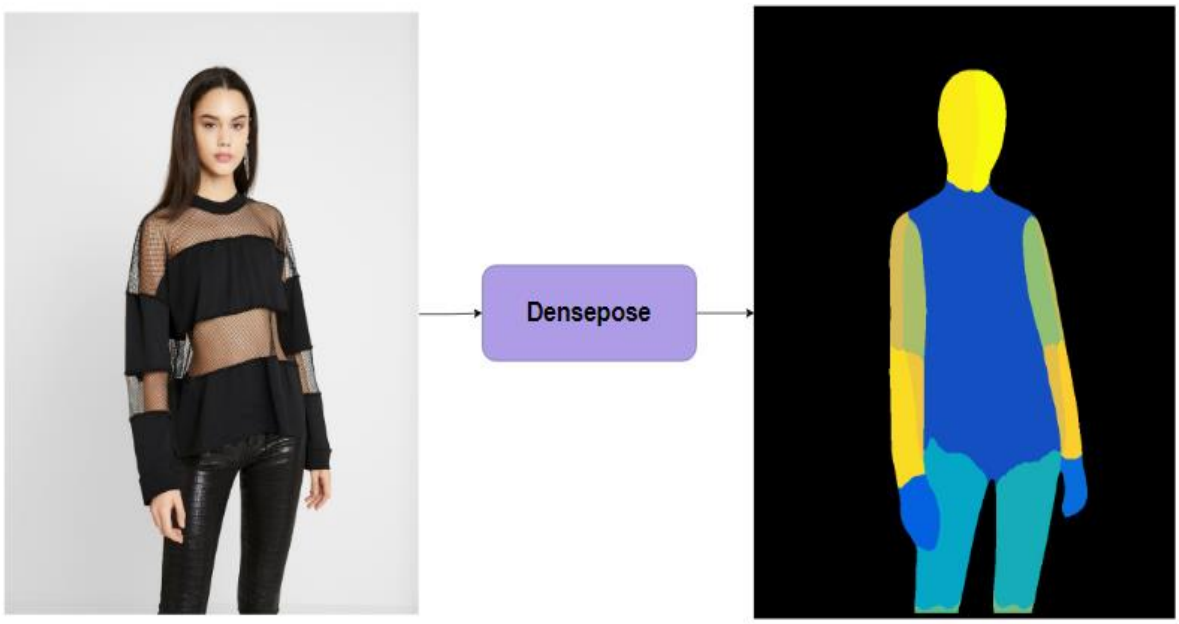

Figure 12. Input and output of dense pose estimation

segmentation network, which is contained in the CarveKit framework, for high-quality background removal.

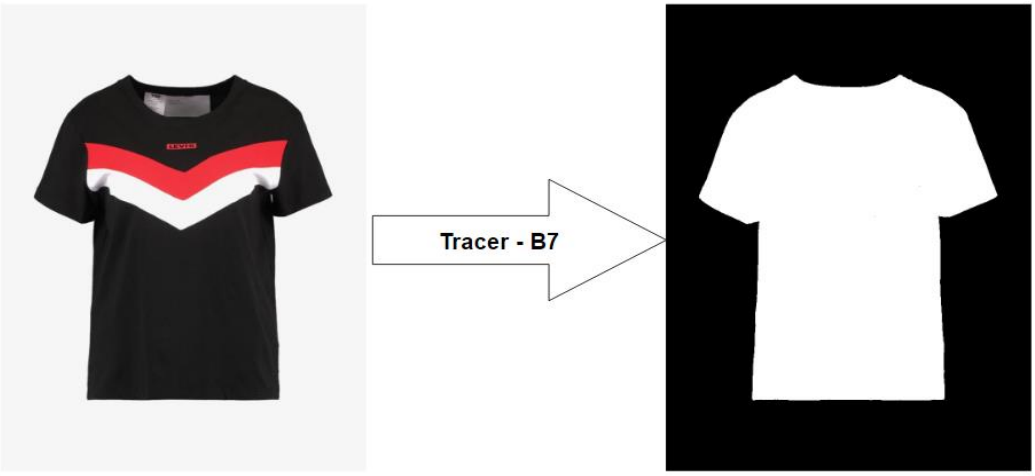

Figure 13. Input and output of cloth mask generation

At the end, as shown in Figure **13**, we use parsing images and JSON files with key points to compute Parse Agnostic and Human Agnostic according to parse labels and corresponding body parts.

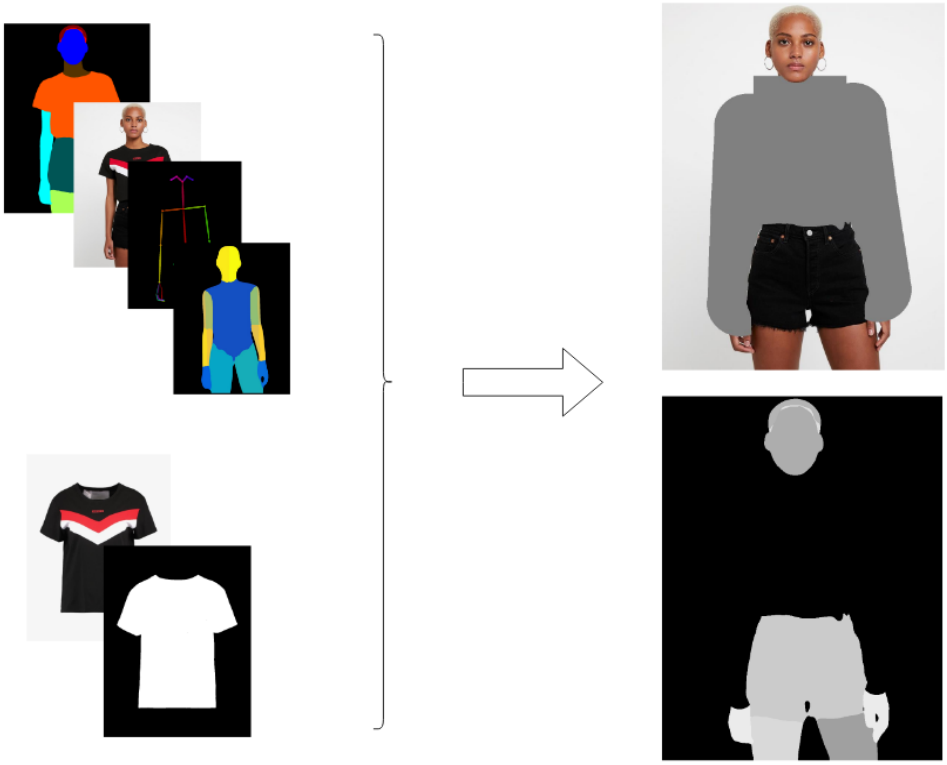

Figure 14. Qualitative result of data processing with upper-clothes

Each garment requires a different set of labels; for example, the upper-cloth requires the head, hair, and lower body to be left intact. Leveraging insights from the above steps, we have extended our capabilities by developing two specialized pipelines dedicated to the processing of lower-clothes and dresses data. Our new processing pipelines' results can be found in Appendix.

2.2.2 Virtual Try-on Model Inference

With the Virtual Dressing Room Feature module Virtual Try-on, we apply the HR Viton method composed of two stages: Try-on Condition Generator and Try-on Image Generator which is shown in Figure 14 .The Try-on Condition Generator aims to produce a segmentation map serving as conditions for the Try-on Image Generator. Extracted features are directed into the decoder's feature fusion blocks, where feature maps from distinct pyramids merge to predict the segmentation map and the flow required for garment image warping.

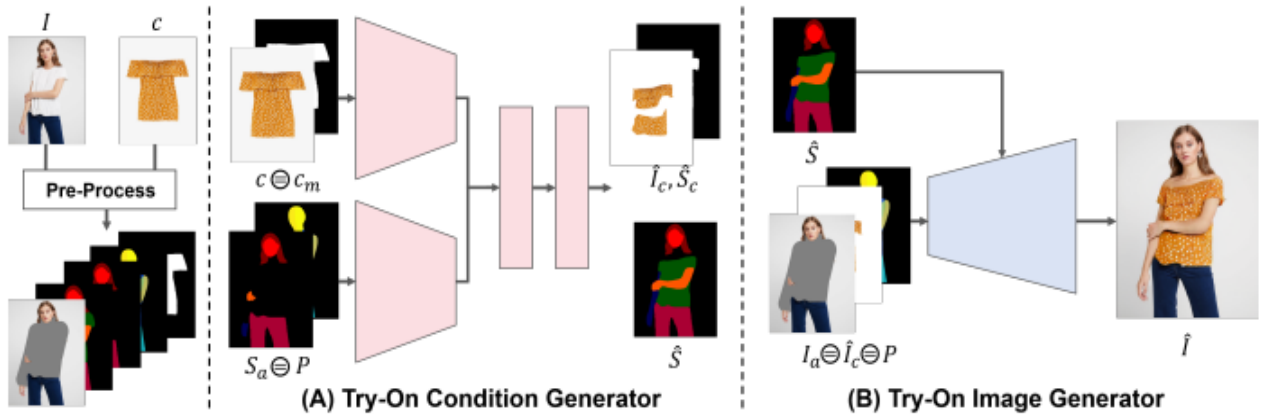

Figure 15. Overview of the proposed method (HR-VITON)

Following this phase, the HR Viton technique's Try-on Image Generator functions to produce the ultimate test image, designated as *I*. This generation process involves amalgamating the clothing-agnostic image I_a, the deformed clothing image $\hat{I}_c$, and the pose map P, guided by $\hat{S}$. The test image generator comprises a sequence of residual blocks complemented by up-sampling layers. These blocks incorporate SPADE [39] as a normalization layer, modulating parameters derived from $\widehat{S}$. The authors adopt identical loss functions as those utilized in SPADE [39] and pix2pixHD [40]. Additionally, the authors introduce a discriminative technique aimed at sieving out the low-quality segmentation map generated by the test condition generator during test instances. The acceptance probability for input x is delineated as:

$$P\ accept_{(x)} = \frac{P_d(x)}{L_{Pd}(x)} \quad (1)$$

Where Pd and Pg are the data distribution and the implicit distribution given and considered by the reconstructor with L being the normalization constant. When the author uses the loss of least squares GAN, this discriminator is considered as the optimal treatment which is derived as follows:

$$D^*_{(x)} = \frac{P_d(x)}{P_d(x)+P_g(x)} \quad (2)$$

The probability of acceptance can be expressed using the discriminator D(x):

$$P\ accept_{(x)} = \frac{D(x)}{L(1-D(x))} \quad (3)$$

This equality only shows that it is satisfied if D = D∗. L is written as follows (4) In practice, the author constructed x from the feature of the segment map and the input consideration conditions (i.e. P, Sa, c and cm) and obtained L by use the entire data set.

$$L = \max_x \frac{D(x)}{(1-D(x))} \quad (4)$$

2.2.3 Method of Upscaling Image for Try-on

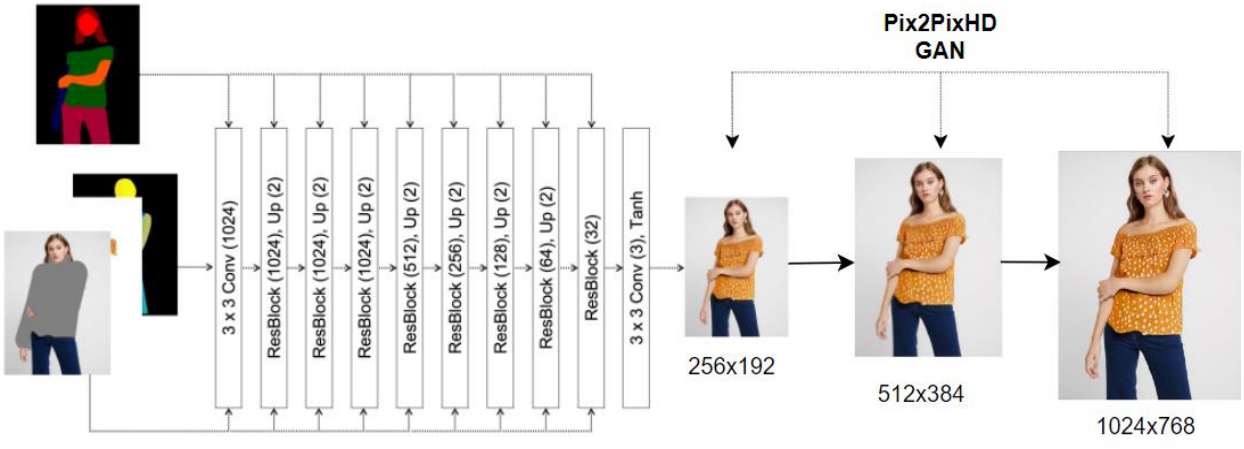

Figure 16. The detailed of Upscaling Stage of the Try-on Image Generator (HR-VITON)

The generator consists of a series of residual blocks adjusted with upsampling layers which

are then divided into two multi-scale discriminators used for the considered conditional adversarial loss. Spectral normalization [19] is applied and adjusted to accommodate all relevant convolutional layers. The method used is similar to the loss applied and adapted by the author in the SPADE [39] and pix2pixHD [40] methods. Specifically, their super-complete objective functions mentioned by the author include adversarial loss with specific conditions, perceptual loss, and loss with appropriate characteristics, which is shown in Figure 15. However, during inference, we found that pix2pixHD GAN does not perform well in generating Image Upscaling results. Therefore, we export the Try image upgrade method using StableSR, which is shown in Figure 16.

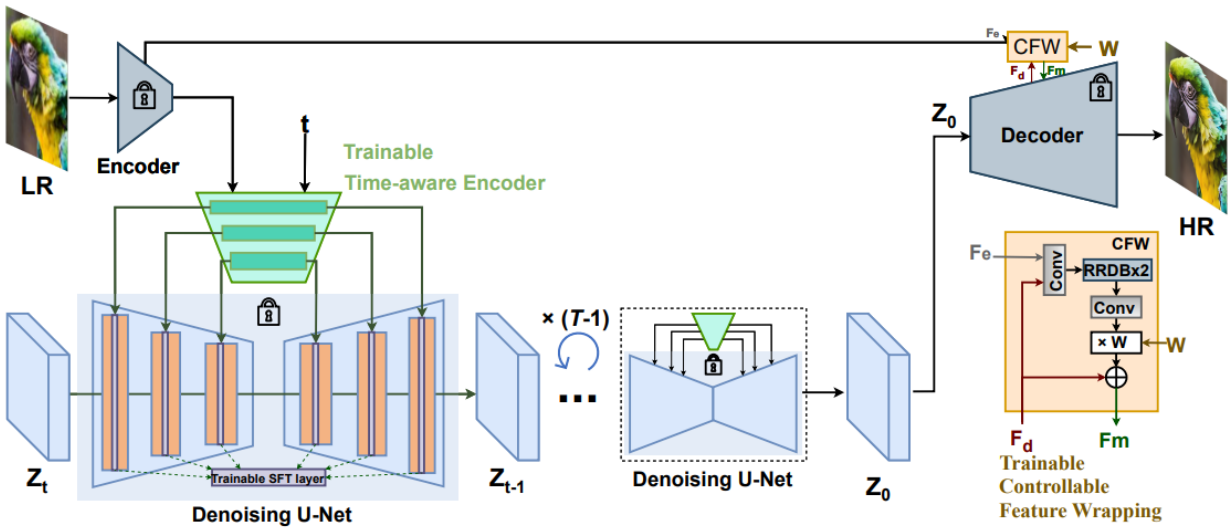

Figure 17. Framework of StableSR

The StableSR method requires fine-tuning a time-aware encoding module that appears to have been integrated into the existing Stable Diffusion Model (depicted in Figure 16). This simple yet powerful design takes advantage of the ultra-wide diffusion inherent to super-resolution (SR) imaging. To more precisely control and tune the generation process, an additional encoder is considered and integrated to extract multi-scale features $\{F^N\}_{n=1}^{N}$ from the features recognized from the image LR is impaired. This refinement process covers a wide range of issues including the use of features in combination with Spatial Feature Transfer (SFT) layers, leading to the establishment of a trainable model. These extracted features were used to completely match the intermediate feature maps $\{F_{dif}^n\}_{n=1}^{N}$ of the redundant blocks in Steady The diffusion framework requires the use of transformations that have in [23]:

$$\hat{F}_n^d = (1 + \alpha^n) \odot F_d^n + \beta^n; \alpha^n, \beta^n = \mathcal{M}_0^n(F^n) \quad (5)$$

In this framework, αn and βn represent affine parameters, while M_θ^nrepresents a compact

network consisting of several convolutional layers. The subscript 'n' describes the spatial scale attributed from U-Net architecture [41] nested within the Steady Diffusion framework. After that, the diffusion pattern remains unchanged.

$$F_m = F_d + \mathcal{C}(F_e; F_d; \theta) \times w \quad (6)$$

The author introduces the Controllable Feature-Wise (CFW) module, which changes the Fm feature modulation by combining additional Fe information from LR features and F_d features from the established decoder. Through an adjustable 'w' factor, the CFW module facilitates a balance between quality and fidelity. Taking advantage of the autoencoder's latent space, using encoder features to adjust corresponding decoder features in Steady Diffusion enhances fidelity.

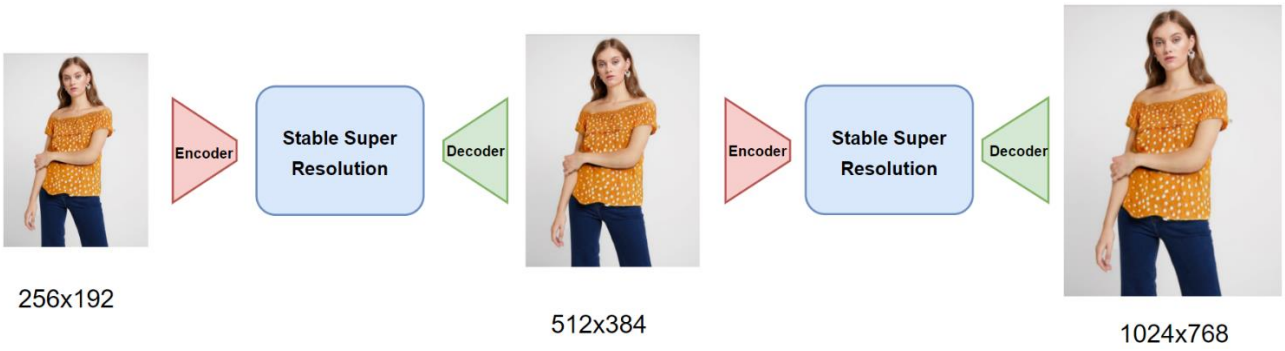

Figure 18. The proposed super resolution method with StableSR for Try-on Image Generator

The use of a time-aware encoder allows to achieve promising recovery results without having to modify the pre-established synthesis model, thus proactively reducing training costs. Minimizing and predicting training costs becomes possible. Additionally, a progressive ensemble sampling strategy has been built to overcome fixed size limitations in pre-trained diffusion models, allowing adaptability to solutions of varying sizes. Addressing the loss of fidelity that arises from the inherent randomness of the diffusion model involves the use of a controllable feature pack module, allowing users to adjust quality in real time versus fidelity by adjusting scalar values during inference. different. A comprehensive evaluation of our approach, using both synthetic and real benchmarks, highlights its superiority over existing state-of-the-art methods.

3. SYSTEM DESIGN & IMPLEMENTATION

The Odoo Framework [42] stands out as a robust and flexible platform tailored for application development, offering a wide array of tools and functionalities necessary for creating complete business solutions. Its prowess particularly shines in e-commerce, providing robust capabilities that empower businesses to establish and manage online

stores. Python is the primary language within the Odoo framework for backend development and is extensively employed in defining models, implementing business logic, and crafting customizations. With its lucid syntax and extensive standard library, Python is an optimal choice for constructing intricate and scalable e-commerce applications.

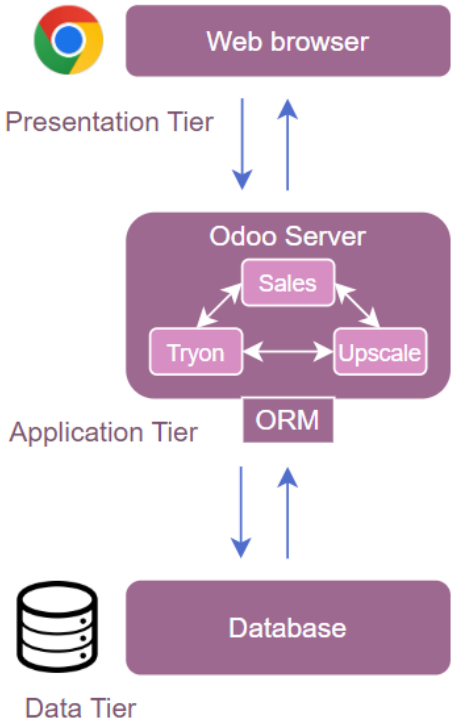

Figure 19. System deployment architecture

Various enterprises, including Faber-Castell, Danone, and Luxe Decor, harness the potent capabilities of Odoo-ecommerce to fashion and oversee their online retail platforms. Within your specific design, the Odoo server seamlessly integrates PostgreSQL [43] as its foundational database, incorporating it seamlessly into the overarching architecture. Moreover, the Odoo service establishes seamless communication with an AI model, generating responses that are subsequently transmitted to the frontend. Developed using HTML, CSS, and JavaScript, the frontend interacts with the Odoo service through a direct RESTful API integration. This API acts as a conduit, facilitating smooth and efficient communication between the frontend and the Odoo service. Such an approach enhances the system's functionality and user experience by enabling seamless data retrieval, manipulation, and other interactions between the two components.

3.1 Backend

Odoo employs the Model-View-Controller (MVC) architectural pattern to manage requests and interact with data, leveraging the Python programming language for its construction. The backend communicates with an AI model service, facilitating the integration of artificial intelligence into the system. This integration is enabled through a rapid API that streamlines communication with the AI model. Responsible for processing logic, data management, and offering APIs to the frontend, the backend also manages the storage and preprocessing of image data for AI models. This effort aims to decrease processing time by executing preprocessing tasks on the image data before passing it to the primary AI model for processing. Explore the functionality of the Try-On API .

Table 1. API Web Application

API	Method	URL	Description
Dashboard try-on	Post	/try-on	Returns list products, models, and test function dressings.
Open upscale	Post	/upscale-image	Opens the interface for users to upload photos and upscales image

Upon a user's selection of a model image and a product, the system triggers the Try-On API. This action involves retrieving the preprocessed images of both the model and the product, subsequently invoking the AI Service API. The preprocessed images serve as inputs for the AI Service API, which then returns the result—a visualization of the try-on effect—displayed on the interface for the user's evaluation.

Table 2. API Model AI

API	Method	URL	Description
Dressing clothes	Post	/process_image	Receives input images of models and outfits and returns fitting results
Upscale image	Post	/upscaleImage	It takes as input an image and an upscale parameter, and the returned data is the image after being upscaled

This seamless fusion of the Try-On API and the AI Service API empowers users to make

informed decisions about their desired products and visualize how they would appear when worn. Similarly, the system engages the Upscale API when a user chooses an image. Before presenting the result to the user, the system utilizes the AI Service's Upscale API. its resolution. The selected image undergoes processing via AI algorithms, enhancing its quality or increasing .The system retrieves the processed data from the Upscale API, presenting an improved version of the chosen image to the user.

3.2 Frontend

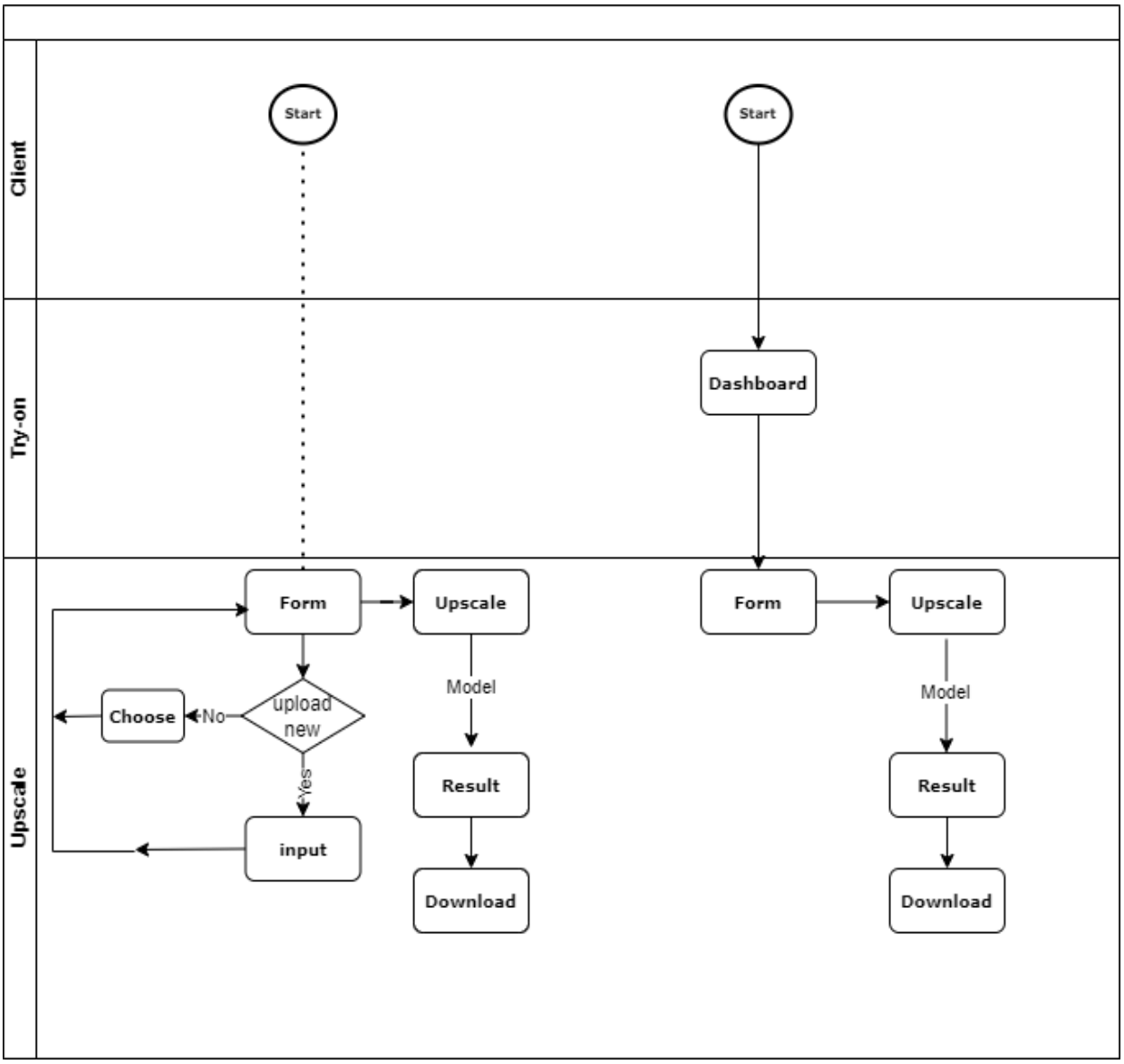

Figure 20. Flow chart in the application

The front-end of the system is developed using the core technologies of HTML, CSS, and JavaScript (JS). These technologies form the foundation for building interactive and user-

friendly interfaces. In particular, JavaScript plays a significant role in handling user actions, and the front-end relies extensively on the Document Object Model (DOM) API for manipulating elements and responding to user interactions. The front-end built on the core layout of Odoo, with the inheritance of user management and product management functionalities, brings convenience and ease of use to system administrators and users alike. CSS is used in front-end development in Odoo to control the visual aspects of the user interface. Responsive design techniques are employed to create interfaces that adapt to different screen sizes, ensuring a consistent experience across mobile devices in Figure 19.

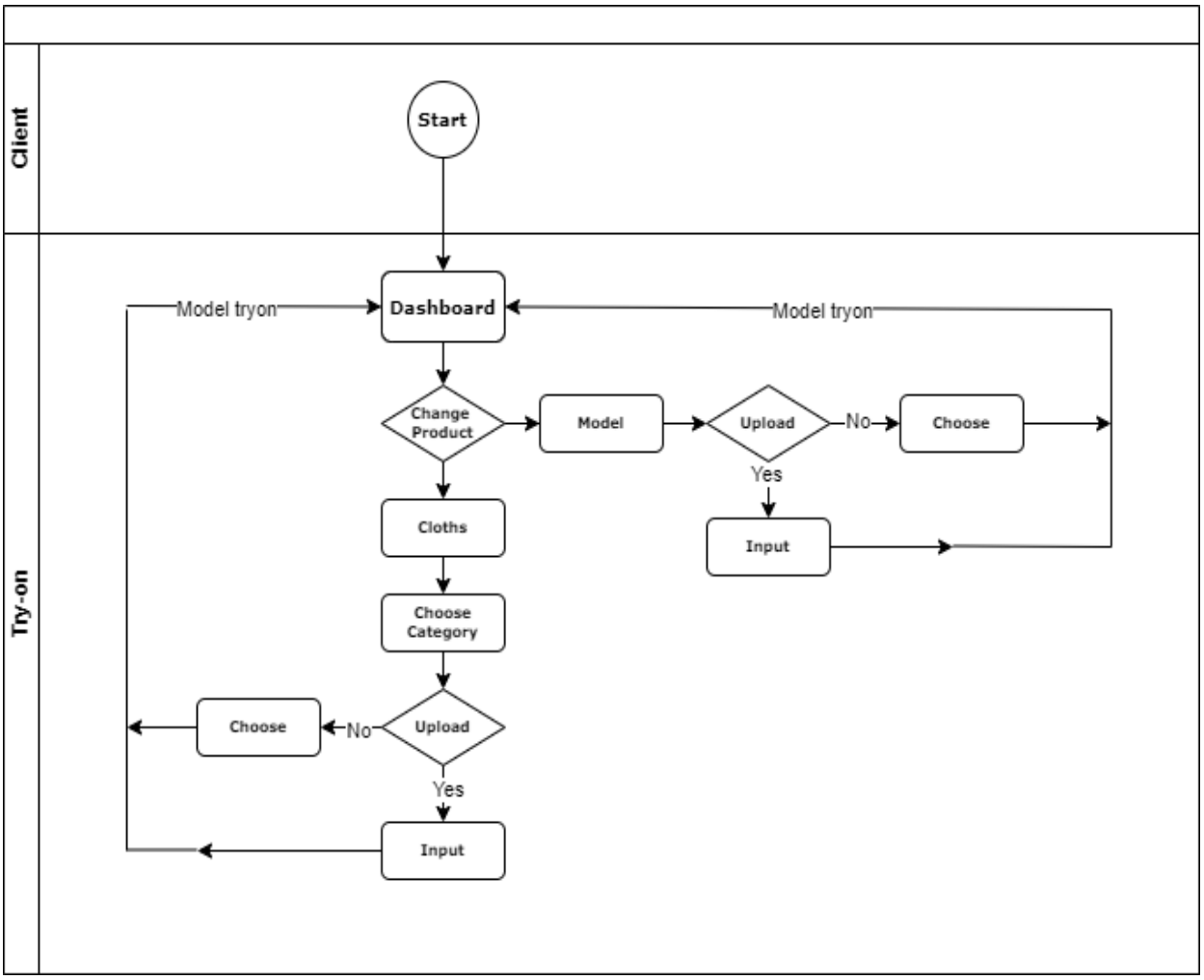

Figure 21. Flow chart in try-on function

When accessing the system, users can directly access the upscale function, where they can select photos in an existing folder or upload new photos and then perform the upscale function. And users can download the returned results image. When accessing the system, users can access the fitting room. After selecting the outfit and model and trying on the outfit, the user can continue to perform the upscale function from the resulting image. And

the result after upscaling allows downloading. In the virtual fitting room function, as shown in Figure 20, users can view models' photos and outfits. It is allowed to change photo models and outfits based on photo data available on the system and photos uploaded by users. After each outfit selection, the fitting results are returned immediately.

Below is the actual user interface in Figure 21 and 22 :

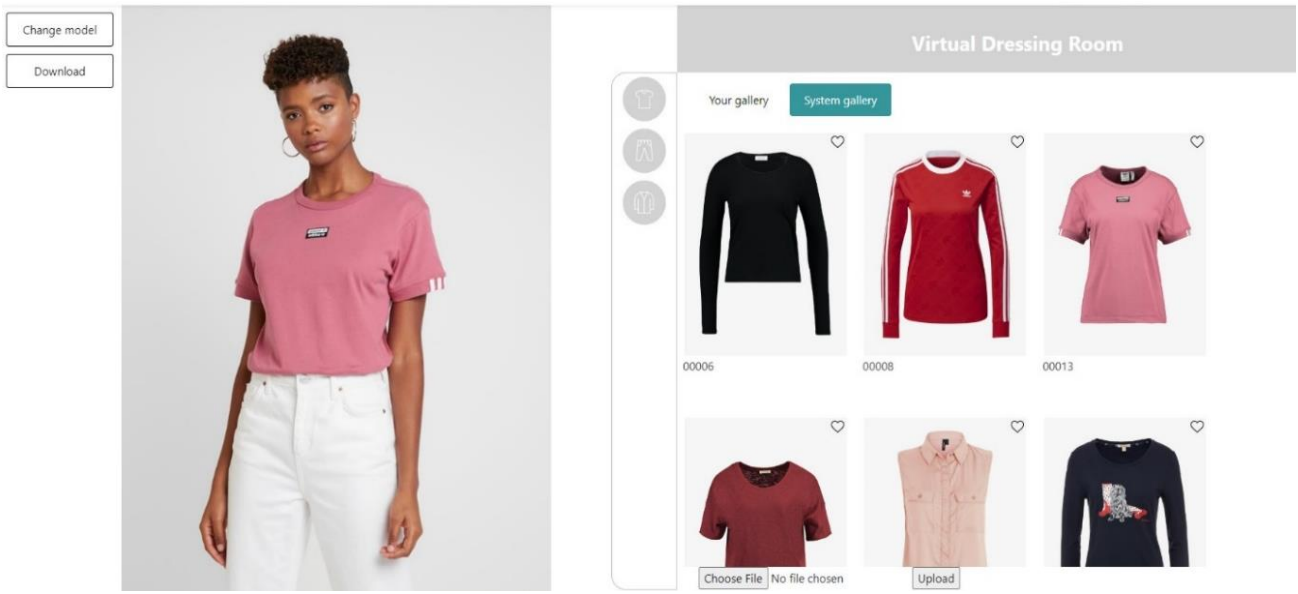

Figure 22. User interface of Virtual Dressing Room

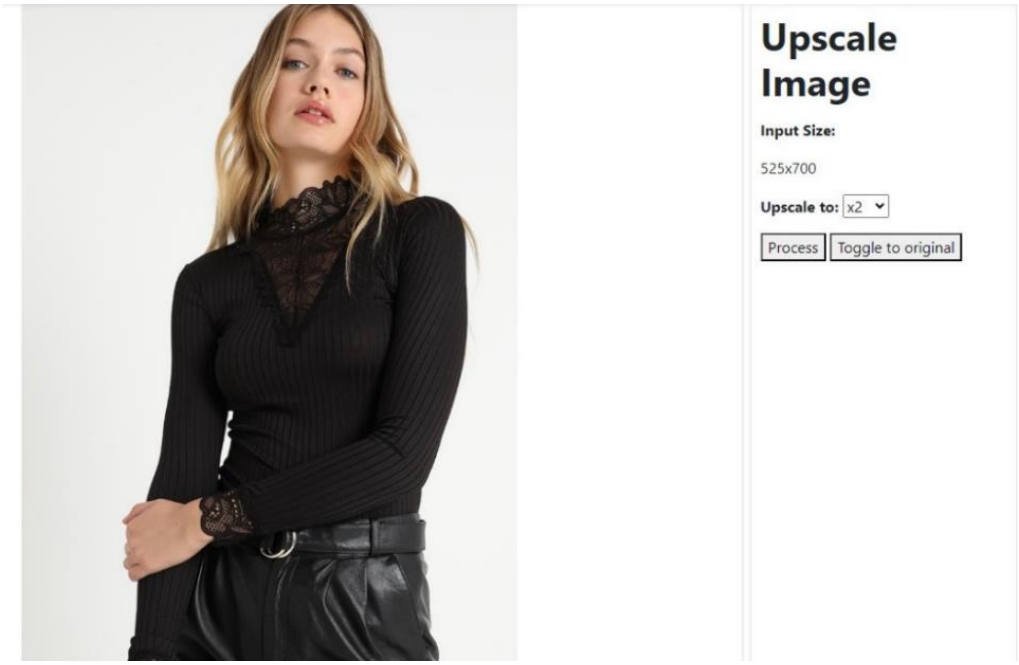

Figure 23. User interface of Upscale Image

3.4 Communication

Table 3. Communication types

Types	Communication	Description
Client-server	RESTful HTTP request and response	Almost all client-server requests and responses
Odoo server - AI server	Fast API	All requests from the backend call to the AI service.

There are two types of communication, client-server, and Odoo server - AI server. In Table 3, we attempted to summarize the used technologies as follows.

4. EXPERIMENT AND EVALUATION

4.1 Experiment Settings

With the goal of gaining an objective view to compare the capabilities that match the resources and efficiency we desire, we have collected and installed models as well as data sets, details of which can be consulted. Refer to tables 4, 5 and 6.

Table 4. Information of datasets that we collected and used to test

Source	Dataset	Number of Images	Image Size (pixels)	Image Type
VITON-HD [9]	VITON-HD 1024	2032	1024 x 768	JPG
DressCode [38]	Lower-clothes	~900	512 x 384	PNG
	Upper-clothes	~800		
	Dresses	~900		

To ensure smooth functionality, our application incorporates over 2800 model photos and an equivalent number of upper-clothes. Approximately 2000 images are sourced from the VITON HD dataset (1024 x 768), covering various upper-clothing types, while the rest come from the DressCode dataset (512 x 384). All images, including persons and garments,

are in .JPG format for streamlined processing and data consistency. Additionally, we've gathered around 900 images each for dresses and lower-clothes from DressCode (512 x 384) to enhance application diversity. In selecting a model for Feature Upscaling, we tested two approaches—StableSR for Stable Diffusion and BSRGAN for GANs. Our objective is to upscale images from our dataset (512 x 384) by a factor of 2, optimizing for limited resources and application requirements.

Table 5. Experiment settings of upscaling models

Model	Approach	Framework	GPU	Platform	Image Size (Input)	Upscaling Factor
StableSR [44]	Stable Diffusion	FastAPI	A100	Google Colab	512 x 384	2
			RTX 4090	Vast.ai		
Real – ESRGAN [45]	GANs	-	T4	Google Colab		
BSRGAN [46]						

When conducting experiments with try-on models, our goal is to ensure stable output while optimizing the available resources. Details regarding the configurations we employ are listed in Table 6.

Table 6. Experiment settings of try-on models

Model	Framework	GPU	Platform
VITON HD	-	T4	Google Colab
HR VITON	Flask API	RTX 3090	Vast.ai
		RTX 4090	
GP VTON	Fast API	RTX 3090	
		RTX 4090	

4.2 Experimental Comparison

4.2.1 Upscaling Feature Metrics Comparison

We compare models based on Structural Similarity (SSIM) in (7) and Learned Perceptual Image Patch Similarity (LPIPS) aims to evaluate the ability to retain image structure and assess the degree of perceptual similarity between two images. The SSIM index is calculated on various windows of an image. The measure between two windows x and y of common size N x N is:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (7)$$

With:

- μ_x : the pixel sample mean of x
- μ_y : the pixel sample mean of y
- σ^2_x : the variance of x
- σ^2_y : the variance of y
- σ_{xy} : the covariance of x and y
- $c_1 = (k_1L)^2$, $c2 = (k_2L)^2$ two variables to stabilize the division with weak denominator
- L is the dynamic range of the pixel-values (typically this is $2^{\#\text{bits per pixel}} - 1$)
- $k_1 = 0.01$ and $k_2 = 0.03$ by default

The Learned Perceptual Image Patch Similarity (LPIPS) metric is a method used to assess the perceptual similarity between two images. It is designed to capture the human visual perception of image quality and is particularly useful in evaluating the performance of image processing algorithms, such as image compression or image generation models.

Table 7. Qualitative results when comparing models on common benchmarks

Dataset	Metrics	Real – ESRGAN	BSRGAN	StableSR
VITON HD 512	SSIM↑	0.6422	0.6120	0.8173
	LPIPS↓	0.3397	0.3652	0.1907
DressCode	SSIM↑	0.7683	0.7840	0.8609
	LPIPS↓	0.2043	0.2173	0.1367

Table 8. Quantitative results when comparing Models on other benchmarks *[44] [45] [46]*

Dataset	Metrics	BSRGAN	Real – ESRGAN+	Stable SR
DIV2K Valid	CLIP-IQA ↑	0.5246	0.5276	0.6771
	MUSIQ ↑	61.19	61.05	65.92
RealSR	CLIP-IQA ↑	0.5114	0.4495	0.6234
	MUSIQ ↑	63.28	60.36	65.88
DRealSRSS	CLIP-IQA ↑	0.5091	0.4515	0.6357
	MUSIQ ↑	57.16	54.26	58.51

The LPIPS metric is often used in computer vision and image processing research to evaluate the visual quality of generated images or to compare the performance of different algorithms. It provides a quantitative measure of how well the perceptual qualities of two images match, taking into account factors like color, texture, and structure. We measured two parameters SSIM and LPIPS on images using StableSR, our chosen method, and we got the results **SSIM ≈ 0.8173 and LPIPS ≈ 0.1907** when using RTX 4070 on Vast.ai, consumed over 18GB of VRAM and 23GB of RAM and took us nearly 38s to process one image each. Besides SSIM and LPIPS, we would like to incorporate additional metrics to compare the capabilities of the models, which are CLIP-IQA [47] and MUSIQ [48].

4.2.2 Upscaling Feature Models Cost Comparison

With Vast.ai, we rented the two most commonly used GPUs in many Deep Learning problems, which are RTX-3090 and RTX-4090. They all have 24GB of VRAM and we always set the memory (storage) used during use on Vast.ai to 80GB, all with CUDA 12.2 installed. Due to security policies and installation difficulties, BSRGAN and Real - ESRGAN are not currently being measured on Vast.ai GPUs.

Table 9. Synthesize the running time of the models

	Time cost per image (seconds)		
Hardware	**BSRGAN**	**StableSR**	**Real-ESRGAN**
CPU (Hugging Face)	137	-	-
T4 (Google Colab)	1,02	-	5,7
RTX-3090 (Vast.ai)	-	64	-
RTX-4090 (Vast.ai)	-	38	-
A100 (Google Colab)	0,82	62	1,1

With Google Colab, we tried using their free GPUs, which are T4 and V100, but the VRAM of both did not meet the applicability of StableSR model as well as BSRGAN (CUDA out of memory error). So, we turned to their high-end GPU, the A100 with 40GB of VRAM and also 80GB of storage.

4.2.3 Try-on Feature Models Comparison

We use the same metrics we used in Upscaling Feature to evaluate to evaluate the ability to retain image structure and assess the degree of perceptual similarity between two images.

Table 10. Quantitative results when comparing models on benchmarks *[9] [16] [49]***.**

Dataset	Metrics	VITON HD	HR VITON	GP VTON
VITON-HD 512	SSIM↑	0.843	0.878	**0.8939**
	LPIPS↓	0.076	**0.061**	0.0799
VITON-HD 1024	SSIM↑	0.873	0.892	**0.8946**
	LPIPS↓	0.077	**0.065**	0.173
DressCode Upper	SSIM↑	-	0.8642	**0.8866**
	LPIPS↓	-	0.1132	**0.0729**

We used the published data of HR-VITON [16] and GP-VTON [49] to evaluate try-on models in order to show correlation comparisons based on scales for different datasets, which are VITON-HD 512 x 386, VITON-HD 1024 x 768 and DressCode Upper-clothes type. In the case of VITON-HD, we have not yet found appropriate DressCode data preprocessing for the model, so we do not have VITON-HD results on this data set. In addition, in the case of GP-VTON with the VITON-HD dataset of size 1024 x 768, since the information published in the paper is not sufficient to conclude, we have measured the SSIM and LPIPS of GP-VTON on VITON-HD 1024 dataset with results listed in Table 10.

The evaluation of the HR-VITON model's running time on the VITON-HD 1024 x 768 dataset yielded a noteworthy observation. Unlike GP-VTON and VITON-HD models, HR-VITON exhibited a 5 - 6 seconds extension in execution time. This discrepancy is attributed to the chosen evaluation method involving API calls. The model's prolonged runtime is primarily due to additional time spent reading data from requests and modifying outputs, making it crucial to consider these nuances when comparing execution times across different models. The results can be found at Table 11.

Table 11. Synthesize the running time of the models

	Time cost per execution (seconds)		
Hardware	**VITON HD**	**HR VITON**	**GP VTON**
RTX-3090 (Vast.ai)	6 ± 2	18 ± 5	8 ± 5
RTX-4090 (Vast.ai)	4 ± 2	13 ± 5	5 ± 3

4.3 Results

Figure 24. Qualitative comparison of Upscaling models (1)

Figure 25. Qualitative comparison of Upscaling models (2)

Figure 26. Qualitative comparison of Upscaling models (3)

Figure 27. Qualitative comparison of Upscaling models (4)

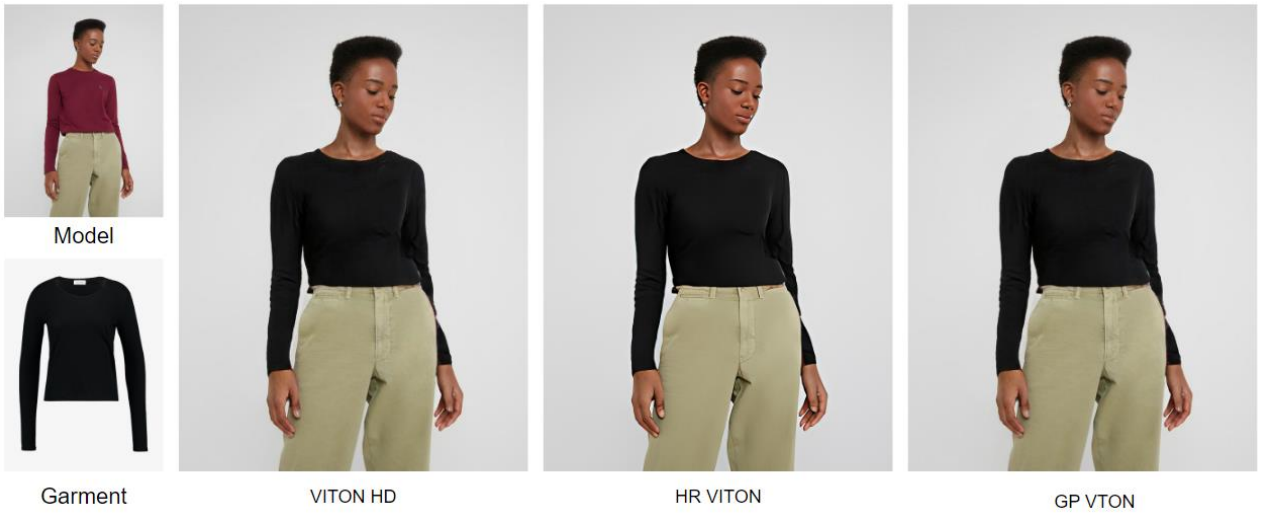

Figure 28. Qualitative comparison of Try-on models on VITON-HD dataset

Figure 29. Qualitative comparison between before and after using StableSR for VITON HD result on VITON-HD 1024 dataset

Figure 30. Qualitative comparison between before and after using StableSR for HR VITON result on VITON-HD 1024 dataset

Figure 31. Qualitative comparison between before and after using StableSR for GP-VTON result on DressCode dataset

StableSR, despite its extended execution time, stands out for producing images of unparalleled sharpness and realism. This remarkable visual fidelity comes at the cost of computational efficiency, setting it apart from the swifter but less visually striking GAN models show in Figure 23-26. Our experiments reveal a compelling interplay between execution time and image quality, underscoring the diverse applications each model may be best suited for. Within the realm of Try-on models, the dynamics between GP-VTON and HR-VITON unveil intriguing insights. Despite GP-VTON's meticulous attention to clothing details during the "warping" process, the emphasis on garment processing unintentionally reduces the ability to retain person's details . In addition, with high-resolution (Ex: over scale 1024) images GP-VTON does not produce satisfactory results shown in Figure 27. This contrasts with HR-VITON, which adeptly strikes a harmonious balance between clothing and model image processing. The result is a synthesis of detailed garments and faithfully preserved person's details, contributing to a more comprehensive and aesthetically pleasing outcome.

Our findings prompt a deeper consideration of model attributes beyond mere execution time, urging practitioners to weigh the trade-offs inherent in each design. While StableSR excels in producing visually stunning images, its prolonged runtime may necessitate strategic implementation. Similarly, the nuanced differences between GP-VTON and HR-VITON prompt a thoughtful selection based on the specific requirements of the application at hand. When combining the Try-On feature and the Upscaling feature, the result is high-resolution photos that still retain maximum details of clothes and people, which is shown in Figure 29-30. This combination will help users' experience be raised to a new level every time they want to see themselves in new clothes.

5. CONCLUSION

This project involves the construction of a virtual fitting system employing Deep Learning technology and utilizing 2D images depicting individuals trying on attire. The developmental process of our product encompasses two pivotal phases. Initially, it integrates the HR-VTON and GPVTON virtual try-on methodologies with the DressCode and VITON HD datasets, aiming to present highly detailed visual representations of garments. The methodical preprocessing of user-provided data comprises six structured steps adhering to established methodologies. Addressing challenges encountered during the Openpose step, notably the loss of finger keypoints, necessitates a comprehensive reiteration of the Human Parse for the DressCode dataset.

Experiments conducted with low-resolution images prompted a reassessment, replacing resolution enhancement with StableSR subsequent to SRGAN phrase generation. The endeavor to strike a balance between realism and fidelity led to the meticulous crafting of user interaction facets, culminating in the seamless integration of interface and backend functionalities to create a virtual dressing room capable of super high-resolution image upscaling. These measures are aimed at enhancing user experiences within the scope of this project. The preprocessing phase has refined both Densepose and Agnostic steps to ensure improved integration of detail into the try-on model. Challenges encountered within the HR-VITON method, particularly in the generation of fixed-resolution and low-sharpness images, prompted the adoption of StableSR for resolution augmentation, thereby surpassing input image resolution. Our evaluation comparing StableSR and SRGAN demonstrates the superiority of the StableSR method in the super resolution stage for try-on images. The Virtual Dress Room provides "Virtual Dressing" and "Upscaling Resolution" features, empowering users to virtually dress models and flexibly adjust image resolution to suit individual preferences.

However, concerning the future trajectory of this work, challenges persist in the runtime of StableSR models when upscaling large images, resulting in considerable time consumption. Additionally, the considerable GPU hardware requirements for all models necessitate further optimization for practical application. Further preprocessing methods are deemed necessary to align input images with the model's specifications. Furthermore, there is an intention to enhance the preprocessing component of the GP-VTON method within the virtual try-on module, with the objective of augmenting the performance pertaining to the virtual try-on problem. Consequently, future research endeavors will be directed towards addressing these aforementioned issues to enable the handling of such challenges.

6. REFERENCES

[1] "Vue.ai" [Online]. Available: https://vue.ai/. [Accessed 13 12 2023].

[2] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen and Timo Aila, "Training generative adversarial networks with limited data" *In NeurIPS,* p. 12104–12114, 2020.

[3] Tero Karras, Samuli Laine and Timo Aila, "A style-based generator architecture for generative adversarial network" *ICCV,* pp. 4401-4410, 2019.

[4] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen and Timo Aila, "Analyzing and improving the image quality of stylegan" *CVPR,* p. 8110–8119, 2020.

[5] Erik Härkönen, Tero Karras, Miika Aittala, Samuli Laine, Janne Hellsten, Jaakko Lehtinen and Timo Aila, "Alias-free generative adversarial networks" *In NeurIPS,* pp. 852-863, 2021.

[6] Jianglin Fu, Shikai Li, Yuming Jiang, Kwan-Yee Lin, Chen Qian, Chen Change Loy, Wayne Wu and Ziwei Liu, "Stylegan-human: A data-centric odyssey of human generation" *In ECCV,* 2022.

[7] Anna Frühstück, Jingwan LuAnna Frühstück, Krishna Kumar Singh, Eli Shechtman, Niloy J. Mitra and Peter Wonka, " Insetgan for full-body image generation" *In CVPR,* pp. 7723-7732, 2022.

[8] Hongxia Yang, Shuai Bai, Huiling Zhou, Zhikang Li and Chang Zhou, "Single stage virtual try-on via deformable" *In ECCV,* 2022.

[9] Seunghwan Choi, Sunghyun Park, Minsoo Lee and Jaegul Choo, " Viton-hd: High-resolution virtual try-on via misalignment-aware normalization" *In CVPR,* pp. 14131-14140, 2021.

[10] Ayush Chopra, Rishabh Jain, Mayur Hemani and Balaji Krishnamurthy, "Zflow: Gated appearance flow-based virtual try-on with 3d priors" *In ICCV,* p. 5433–5442, 2021.

[11] Haoye Dong, Xiaodan Liang, Xiaohui Shen, Bochao Wang, Hanjiang Lai, Jia Zhu, Zhiting Hu and Jian Yin, "Towards multi-pose guided virtual try-on network" *In ICCV,* p. 9026–9035, 2019.

[12] Yuying Ge, Yibing Song, Ruimao Zhang, Chongjian Ge, Wei Liu and Ping Luo, "Parser-free virtual try-on via distilling appearance flows" *In CVPR,* p. 8485–8493, 2021.

[13] Xintong Han, Xiaojun Hu, Weilin Huang and Matthew R.Scott, " Clothflow: A flow-based model for clothed person generation" *In ICCV,* p. 10471–10480, 2019.

[14] Sen He, Yi-Zhe Song and Tao Xiang, "Style-based global appearance flow for virtual try-on" *In CVPR,* pp. 3470-3479, 2022.

[15] Thibaut Issenhuth, Jérémie Mary and Clément Calauzènes, "Do not mask what you do not need to mask: A parser-free virtual try-on" *In ECCV,* p. 619–635, 2020.

[16] Sangyun Lee, Gyojung Gu, Sunghyun Park, Seunghwan Choi and Jaegul Choo, " High-resolution virtual try-on with misalignment and occlusion-handled conditions" *In ECCV,* 2022.

[17] Kedan Li, Min Jin Chong, Jeffrey Zhang and Jingen Liu, "Toward accurate and realistic outfits visualization with attention to details" *In CVPR,* p. 15546–15555, 2021.

[18] Davide Morelli, Matteo Fincato, Marcella Cornia, Federico Landi, Fabio Cesari and Rita Cucchiara, " Dress Code: High-Resolution Multi-Category Virtual Try-On" *In ECCV,* 2022.

[19] Bochao Wang, Huabin Zheng, Xiaodan Liang, Yimin Chen, Liang Lin and Meng Yang, "Toward characteristicpreserving image-based virtual try-on network" *In ECCV,* p. 589–604, 2018.

[20] Han Yang, Ruimao Zhang, Xiaobao Guo, Wei Liu, Wangmeng Zuo and Ping Luo, "Towards photo-realistic virtual try-on by adaptively generating-preserving image content" *In CVPR,* p. 7850–7859, 2020.

[21] Ruiyun Yu, Xiaoqi Wang and Xiaohui Xie, "Vtnfp: An image-based virtual try-on network with body and clothing feature preservation" *In ICCV,,* p. 10511–10520, 2019.

[22] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik and Alexei A. Efros, " View synthesis by appearance flow." *In ECCV,* 2016.

[23] K. Y. C. D. C. C. L. Xintao Wang, "Recovering Realistic Texture in Image Super-resolution by Deep Spatial Feature Transform" *CVPR,* 2018.

[24] Sachit Menon, Alexandru Damian, Shijia Hu, Cynthia Rudin and Nikhil Ravi, "Pulse: Self-supervised photo upsampling via latent space exploration of generative models." *In CVPR,* 2020.

[25] Bolei Zhou, Jinjin Gu and Yujun Shen, "Image processing using multi-code gan prior." *In CVPR,* 2020.

[26] Xintao Wang, Yu Li, Honglun Zhang and Ying Shan, "Towards real-world blind face restoration with generative facial prior" *In CVPR,* 2021.

[27] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy and Ping Luo, "Exploiting deep generative prior for versatile image restoration and manipulation." *TPAMI,* 2021.

[28] Chen Change Loy, Jinwei Gu, Kelvin C.K. Chan, Xintao Wang, Xiangyu Xu and Jinwei Gu, "GLEAN: Generative latent bank for large-factor image super-resolution." *CVPR,* 2021.

[29] Kelvin C.K. Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu and Chen Change Loy, "GLEAN: Generative latent bank for large-factor image super-resolution and beyond." *TPAMI,* 2022.

[30] Tao Yang, Peiran Ren, Xuansong Xie and Lei Zhang, "Gan prior embedded network for blind face restoration in the wild." *CVPR,* 2021.

[31] Yang Zhao, Yu-Chuan Su, Chun-Te Chu, Yandong Li, Marius Renn, Yukun Zhu, Changyou Chen and Xuhui Jia, "Rethinking deep face restoration." *CVPR,* 2022.

[32] Chaofeng Chen, Xinyu Shi, Yipeng Qin, Xiaoming Li, Xiaoguang Han, Tao Yang and Shihui Guo, "Real-world blind super-resolution via feature matching with implicit high-resolution priors" *ACM MM,* 2022.

[33] Shangchen Zhou, Kelvin C.K. Chan, Chongyi Li and Chen Change Loy, "Towards robust blind face restoration with codebook lookup transformer" *NeurIPS,* 2022.

[34] Ke Yu, Chao Dong, Liang Lin and Chen Change Loy, "Crafting a toolchain for image restoration by deep reinforcement learning" *CVPR2018.*

[35] Jie Liang, Hui Zeng and Lei Zhang, "Efficient and degradation-adaptive network for real-world image super-resolution" *ECCV,* 2022.

[36] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei and Yaser Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields" *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2019.

[37] Ke Gong, Xiaodan Liang, Yicheng Li, Yimin Chen, Ming Yang and Liang Lin, "Instance-level Human Parsing via Part Grouping Network" *ECCV*, 2018.

[38] Riza Alp Güler, Natalia Neverova and Iasonas Kokkinos, "DensePose: Dense Human Pose Estimation In The Wild" *CVPR*, 2018.

[39] Taesung Park, Ming-Yu Liu, Ting-Chun Wang and Jun-Yan Zhu, "Semantic image synthesis with spatially-adaptive normalization" *CVPR*, p. 2337–2346, 2019.

[40] Taesung Park, Ming-Yu Liu, Ting-Chun Wang and Jun-Yan Zhu, "Highresolution image synthesis and semantic manipulation with conditional gans" *CVPR*, 2018.

[41] Olaf Ronneberger, Philipp Fischer, Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation" *MICCAI*, 2015.

[42] "Odoo Framework (Version 16)" [Online]. Available: https://www.odoo.com/. [Accessed 13 12 2023].

[43] "PostgreSQL Database Management System" [Online]. Available: https://www.postgresql.org/docs/. [Accessed 13 12 2023].

[44] Wang, Jianyi, Yue, Zongsheng, Zhou, Shangchen, Chan, Kelvin CK and Loy, Chen Change, "Exploiting Diffusion Prior for Real-World Image Super-Resolution" 2023.

[45] Xintao Wang, Liangbin Xie, Chao Dong and Ying Shan, "Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data" *ICCVW*, 2021.

[46] Zhang, Kai, Liang, Jingyun, Van Gool, Luc and Timofte, Radu, "Designing a Practical Degradation Model for Deep Blind Image Super-Resolution" *IEEE International Conference on Computer Vision*, pp. 4791--4800, 2021.

[47] Wang, Jianyi, Chan, Kelvin CK and Loy, Chen Change, "Exploring CLIP for Assessing the Look and Feel of Images" *AAAI*, 2023.

[48] Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar and Feng Yang, "MUSIQ: Multi-scale Image Quality Transformer" *ICCV*, 2021.

[49] Xin, Dong, Zhenyu, Xie, Zaiyu, Huang, Fuwei, Zhao, Haoye, Dong, Xijin, Zhang, Feida, Zhu and Xiaodan, Liang, "GP-VTON: Towards General Purpose Virtual Try-on via Collaborative Local-Flow Global-Parsing Learning" *CVPR*, 2023.

[50] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville and Yoshua Bengio, " Generative adversarial nets" *In NeurIPS*, 2014.

[51] Lee, Sangyun, Gu, Gyojung, Park, Sunghyun, Choi, Seunghwan and Choo, Jaegul, "High-Resolution Virtual Try-On with Misalignment and Occlusion-Handled Conditions" *ECCV*, 2022.

[52] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever and Mark Chen, "Glide: Towards photorealistic image generation and editing with text-guided diffusion models." *IMCL*, 2022.

[53] Change Loy, Xintao Wang, Ke Yu and Chao Dong, "Recovering realistic texture in image super-resolution by deep spatial feature transform." *In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[54] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser and Bjorn Ommer, "High-resolution image synthesis with latent diffusion models" *CVPR*, 2022.

[55] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford,

Mark Chen and Ilya Sutskever, "Zero-shot textto-image generation" *ICML*, 2021.

APPENDIX

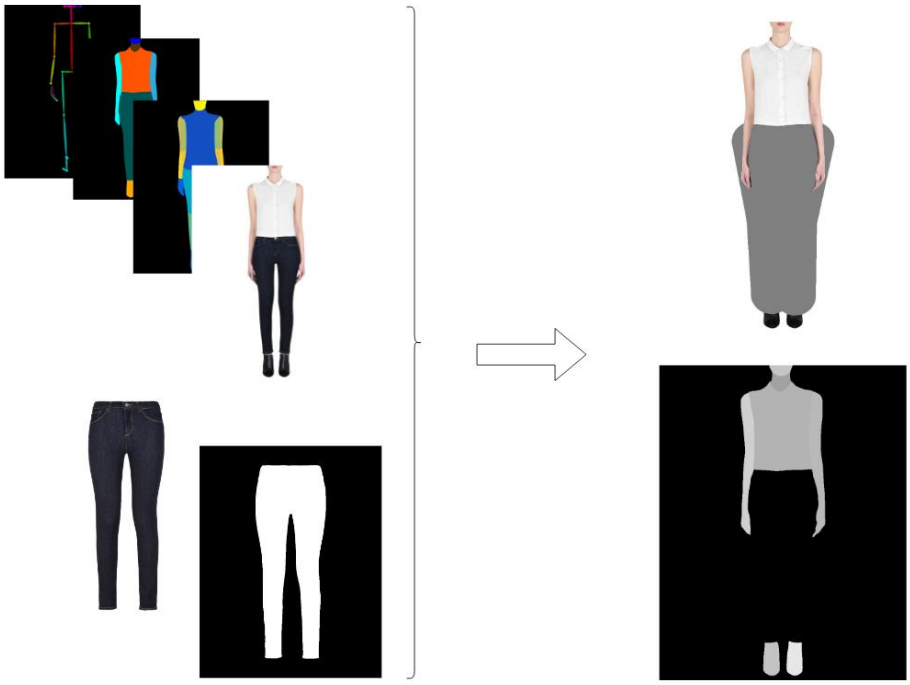

Figure 32. Qualitative results of preprocessing lower-clothes

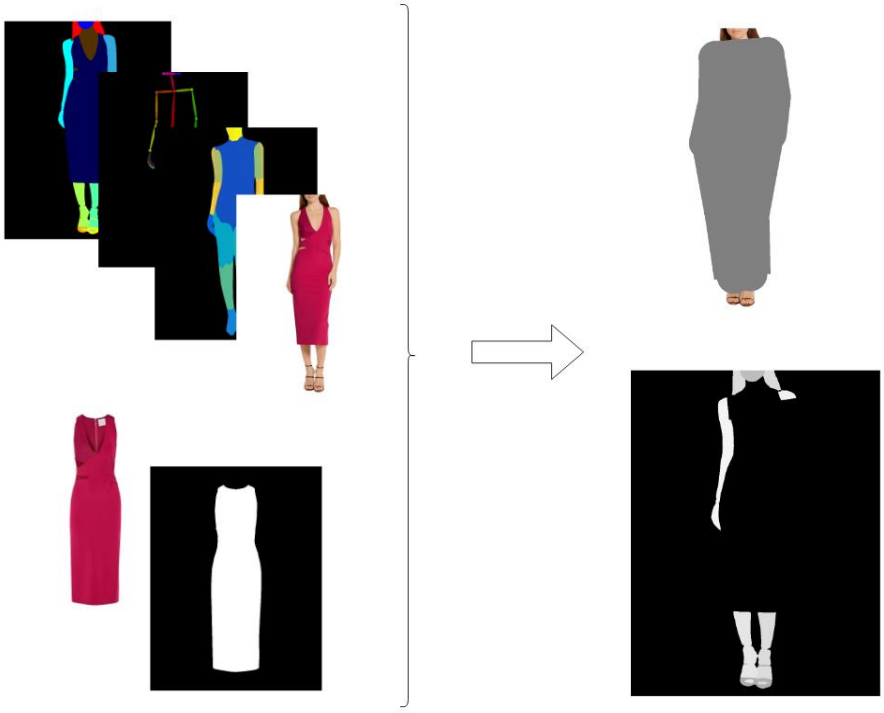

Figure 33. Qualitative results of preprocessing dresses

Table 12. List of labels that we used to preprocess new image

Label	Name
0	Background
1	Hat
2	Hair
3	Glove
4	Sunglasses
5	Upper-clothes
6	Dress
7	Coat
8	Socks
9	Pants
10	Tosor-skin
11	Scarf
12	Skirt
13	Face
14	Left-arm
15	Right-arm
16	Left-leg
17	Right-leg
18	Left-shoe
19	Right-shoe