www.ijarp.com ISSN 2456-9992 Page: 01-13

International Journal Advanced Research Publications

Research Article

Volume: 02
Issue: 10

A COST-EFFICIENT AI-BASED VIRTUAL TRY-ON FRAMEWORK FOR MULTIPLE CLOTHING CATEGORIES USING OPTIMIZED IMAGE PROCESSING

Selvaraj A.*[1], Dhanush Babu L. B.[2], Dhanusu Rasu L.[2]

[1]Assistant Professor(Sr.Gr), Department of Information Technology, K.L.N College of Engineering, Sivagangai, Tamil Nadu, India.

[2]Student, Department of Information Technology, K.L.N College Of Engineering, Sivagangai, Tamil Nadu, India.

Article Received: 15 September 2026, Article Revised: 05 October 2026, Published on: 25 October 2026

***Corresponding Author: Selvaraj A.**

Assistant Professor(Sr.Gr), Department of Information Technology, K.L.N College of Engineering, Sivagangai, Tamil Nadu, India.

Doi: https://doi-doi.org/101555/ijarp.9634

ABSTRACT

The increasing adoption of artificial intelligence in e-commerce has created opportunities for virtual try-on systems that enable users to visualize different clothing items on their own photographs. However, integrating generative artificial intelligence through application programming interfaces (APIs) can introduce significant operational costs due to image input processing, token consumption, and repeated image-generation requests. This paper proposes a cost-efficient AI-based virtual try-on framework designed to support multiple clothing categories while reducing API usage costs without unnecessarily compromising visual quality. The proposed system accepts a user's photograph and a selected garment image and processes these inputs before forwarding them to a generative AI model for virtual garment visualization. Image resizing, compression, and prompt optimization are explored as cost-reduction strategies to minimize unnecessary input data and improve resource utilization. The system employs a React.js and Vite-based frontend and a FastAPI backend to manage user interactions, image preprocessing, and API communication. The framework aims to preserve important visual characteristics, including garment color, texture, patterns, and design details, while maintaining the user's facial identity and body appearance. Unlike approaches that focus exclusively on visual generation, the proposed framework emphasizes the relationship between image preprocessing, API consumption, generation efficiency, and output quality. Its

effectiveness can be evaluated by comparing API usage costs, input token consumption, image-generation latency, and visual fidelity before and after optimization across different clothing categories. The proposed work seeks to provide a practical foundation for developing economically sustainable, AI-powered virtual try-on applications for online fashion retail.

KEYWORDS: Virtual Try-On, Generative Artificial Intelligence, API Cost Optimization, Token Consumption, Image Compression, Image Preprocessing, Computational Efficiency, E-Commerce.

I. INTRODUCTION

The fashion and textile industry has undergone significant transformation with the development of digital technologies and online shopping platforms. Customers increasingly prefer convenient, personalized, and interactive shopping experiences that enable them to explore different clothing styles before making purchasing decisions. However, traditional e-commerce platforms primarily display static product images, making it difficult for customers to visualize how a particular garment may appear on their own body. This limitation can create uncertainty during clothing selection and affect customer satisfaction.

Artificial Intelligence (AI), particularly generative AI and image processing, provides new opportunities to address these challenges through virtual try-on technology. A virtual try-on system enables users to upload their photographs and select garments from a product catalogue to generate a visual representation of themselves wearing the selected clothing. Such systems can support multiple clothing categories, including shirts, T-shirts, trousers, dresses, traditional garments, and other apparel.

The proposed project, AI-Based Virtual Try-On System with API Cost Optimization, focuses on developing a web-based application that generates virtual clothing previews using a generative AI image-generation model. The system consists of a React.js and Vite-based frontend and a FastAPI backend responsible for image processing and communication with the AI service. Image resizing, compression, and prompt optimization are explored to reduce unnecessary input data and improve API resource utilization.

A major challenge in generative AI-based virtual try-on applications is the cost associated with processing images and generating outputs through paid APIs. Repeated requests, large input images, and inefficient prompts can increase token consumption and overall operating expenses. Therefore, this project emphasizes not only virtual clothing visualization but also

the optimization of API usage to make the system more economical and practical.

The proposed system aims to provide an accessible and interactive virtual try-on experience while investigating methods to reduce API consumption without unnecessarily compromising image quality, garment appearance, and user identity preservation..

The proposed system supports the following Sustainable Development Goals (SDGs):

- SDG 9 – Industry, Innovation and Infrastructure
- SDG 12 – Responsible Consumption and Production

II. LITERATURE REVIEW

Several existing studies have explored the development of virtual try-on systems using computer vision, deep learning, image synthesis, and generative artificial intelligence. These approaches aim to improve online clothing visualization, preserve garment characteristics, and provide personalized shopping experiences. Recent developments have also introduced image-generation APIs that simplify implementation but create challenges related to computational cost, image processing, and resource consumption.

[1] VITON: An Image-Based Virtual Try-On Network (2018),This work introduces VITON, an image-based virtual try-on framework that enables users to visualize clothing items on a target person using image synthesis techniques. The approach combines clothing-agnostic person representations with garment images to generate a new visual representation. It established an important foundation for image-based virtual try-on research. However, maintaining realistic garment details and handling complex clothing transformations remain challenging.

[2] CP-VTON: Toward a Realistic Virtual Try-On System (2018),This study presents a Characteristic-Preserving Virtual Try-On Network designed to improve the alignment between garment images and the target person's body. The framework uses geometric matching and image composition to produce more realistic clothing visualizations. It demonstrates the importance of garment alignment in virtual try-on applications. However, geometric transformations may distort clothing patterns and fine details, particularly when garments have complex shapes or decorative designs.

[3] ACGPN: Towards Photo-Realistic Virtual Try-On by Adaptively Generating-Preserving Image Content (2020),This work proposes an adaptive content-generation and preservation approach for virtual try-on. The method aims to preserve relevant parts of the original person image while generating the regions required for clothing replacement. It

improves the handling of clothing boundaries and image composition. However, complex poses, occlusions, and intricate garment structures may still affect the quality of generated results..

[4] VITON-HD: High-Resolution Virtual Try-On via Misalignment-Aware Normalization (2021),This study introduces a high-resolution virtual try-on framework designed to improve garment visualization at higher image resolutions. It addresses misalignment between clothing images and person representations through specialized normalization techniques. The approach demonstrates the importance of resolution and alignment in generating visually detailed outputs. However, high-resolution image processing can require additional computational resources, making efficiency an important consideration for practical applications.

[5] LaDI-VTON: Latent Diffusion Textual-Inversion Enhanced Virtual Try-On (2023),This work explores the use of latent diffusion models for virtual try-on image generation. By operating in a compressed latent representation, the approach aims to generate realistic clothing visualizations while leveraging pretrained generative models. It demonstrates the potential of diffusion-based methods for flexible garment synthesis. However, inference requirements, generation latency, and hardware availability may create challenges when deploying such systems in resource-constrained environments.

[6] StableVITON: Learning Semantic Correspondence with Latent Diffusion Model for Virtual Try-On (2024),This study investigates semantic correspondence between garment images and person representations using latent diffusion models. The method aims to improve the alignment of garment features with the target person's body while generating realistic results. It highlights the importance of preserving garment structure and semantic information during image synthesis. However, the computational requirements of diffusion-based models may limit their accessibility for applications without dedicated hardware or sufficient processing resources**.**

[7] CatVTON: Concatenation Is All You Need for Virtual Try-On with Diffusion Models (2024),This work presents a simplified virtual try-on approach based on concatenating person and garment representations for diffusion-based image generation. The method aims to reduce architectural complexity while maintaining competitive visual quality. It demonstrates the potential for more efficient virtual try-on model design. However, the resources required for local model inference and the quality of generated outputs remain important considerations when adapting such approaches to practical web applications.

[8] Generative Artificial Intelligence for Digital Fashion and Virtual Try-On (Recent Research),Recent research into generative AI has explored the use of pretrained image-generation models for creating personalized clothing visualizations. These approaches can reduce the need to develop an image-generation model entirely from scratch and can support flexible garment transformation through image conditioning and prompt-based instructions. However, the quality of the generated output depends on the model's capabilities, input-image quality, and prompt design. General-purpose image-generation models may also alter garment patterns or other visual details.

[9] Image Compression and Preprocessing for Efficient AI-Based Image Generation (Related Research Area),Image preprocessing techniques, including resizing, compression, and format conversion, are widely used to reduce image storage requirements and data-transfer overhead. In AI-based virtual try-on applications, these techniques can help prepare images for submission to cloud-hosted generation services. Appropriate preprocessing may reduce image-related processing costs when the API's billing mechanism depends on image dimensions or image tokens. However, excessive compression may remove fine garment details and negatively affect output quality. Therefore, the preprocessing configuration must balance resource efficiency with visual fidelity.

[10] Prompt Optimization and Cost-Efficient Generative AI Applications (Related Research Area),Prompt optimization focuses on removing redundant instructions and retaining the essential information needed to guide a generative AI model. In API-based applications, shorter prompts can reduce text-token consumption when text input is billed by token usage. Combined with image preprocessing and efficient request management, prompt optimization can contribute to lower operating costs. However, reducing prompt length does not necessarily reduce image-generation costs, and overly simplified instructions may negatively affect the generated result. The relationship between prompt length, API cost, and image quality therefore requires systematic evaluation.

III. PROPOSED SYSTEM AND METHODOLOGY

A. System Overview

The proposed AI-Based Virtual Try-On System with API Cost Optimization is a web-based application that enables users to visualize different clothing items on their own photographs using generative artificial intelligence. The system allows users to upload a personal photograph, select a garment from a product catalogue or upload a clothing reference image, and generate a virtual representation of the selected garment. It supports multiple clothing

categories, including shirts, T-shirts, trousers, dresses, sarees, and other apparel, depending on the capabilities of the selected AI model.

The primary objective of the proposed system is to provide an interactive virtual clothing visualization experience while reducing the operational costs associated with generative AI API usage. Conventional API-based image-generation applications may incur additional costs when processing large input images, sending lengthy prompts, or performing unnecessary repeated requests. To address these challenges, the proposed system incorporates image resizing, compression, prompt optimization, and efficient API request management.
The application uses React.js and Vite for the frontend interface and Python with FastAPI for backend processing. The backend validates uploaded images, performs preprocessing, prepares optimized prompts, and communicates with the configured generative AI image-generation API. The generated image is returned to the frontend for display. The system's effectiveness can be evaluated by comparing API consumption, generation time, estimated cost, and visual quality before and after optimization.

B. System Modules

The system follows a modular architecture consisting of a user interface, an image preprocessing pipeline, a prompt optimization component, a backend API service, and a generative AI image-generation service.

The React.js frontend collects the user's photograph and the selected garment image. These inputs are transmitted to the FastAPI backend, where validation and preprocessing are performed. The images are resized and compressed according to the configured optimization settings, and a concise prompt is prepared to communicate the intended clothing transformation.

The backend then submits the prepared inputs to the configured AI image-generation API. After processing the request, the AI service returns the generated virtual try-on image. The backend handles the response and sends the resulting image to the frontend for display.

The proposed methodology emphasizes balancing API cost reduction with output quality. Different image resolutions and prompt configurations can be evaluated using the same test inputs to determine their effect on API consumption, processing time, and garment-detail

preservation. API credentials are maintained on the backend to reduce the risk of exposing sensitive keys through client-side code.

i. Modules Overview

The proposed system is divided into the following modules to simplify development, improve maintainability, and support cost optimization.

The system uses React.js and Vite for the frontend interface and Python with FastAPI for backend processing. The backend manages image validation, preprocessing, prompt optimization, and communication with the generative AI image-generation API. The system also incorporates output evaluation to compare API usage, generation time, and visual quality under different optimization configurations.

The major modules include User Interaction, Image Upload and Validation, Image Preprocessing, Garment Selection, Prompt Optimization, AI-Based Virtual Try-On, API Cost Optimization, and Output Evaluation. Together, these modules provide a structured workflow for generating virtual clothing previews across multiple garment categories while investigating cost-efficient image-generation techniques.

ii. Types of Modules

1. Garment Catalogue Module

This module displays garments available for virtual try-on. Users can browse categories such as T-shirts, shirts, dresses, jackets, sarees, and jeans. They can also search the catalogue and filter items by attributes such as gender and color. Garment information is loaded from the project's image folders and fashion dataset.

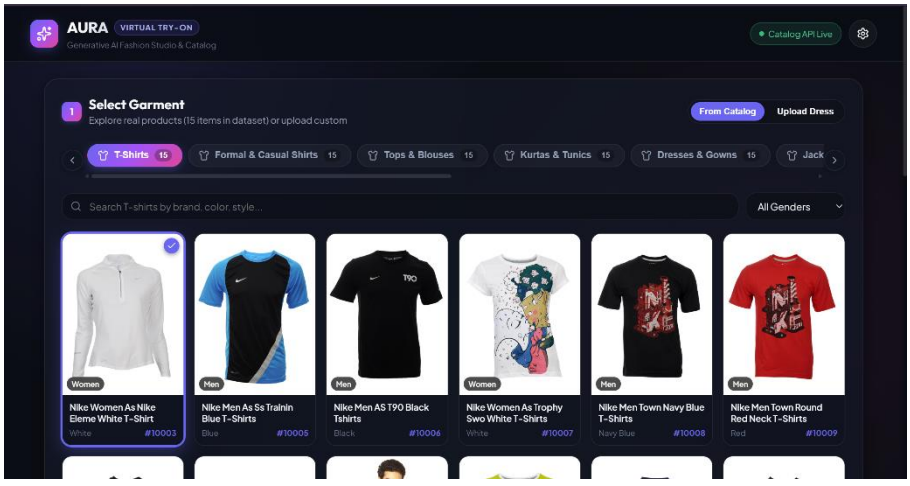


Fig (1) Garment Catalogue Module.

2. Person Image Input Module

This module obtains the person image used for the virtual try-on. Users can upload a photo, capture one using the device camera, or choose a preset model image. The selected image is previewed before generation so the user can confirm their input.

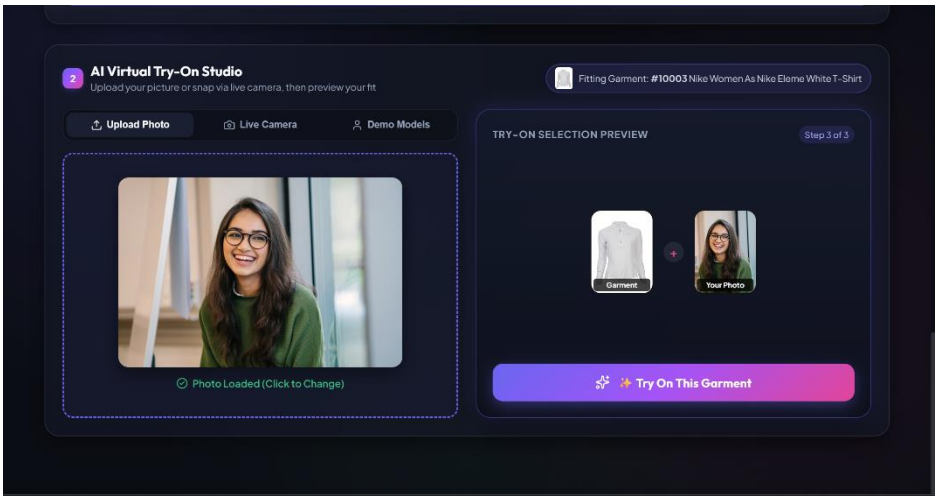


Fig (2) Person Image Input Module.

3. Custom Garment Input Module

In addition to selecting an item from the catalogue, users can upload their own garment image. The application previews the uploaded garment and sends it to the backend along with the person image for processing

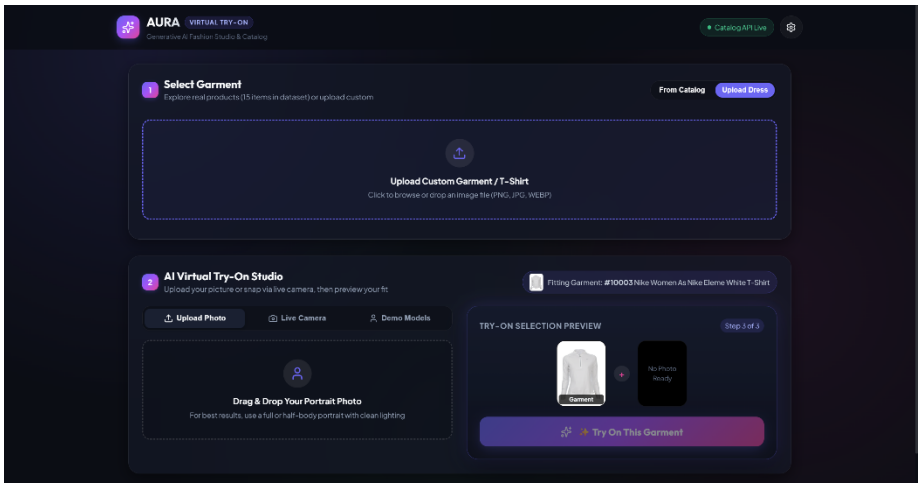


Fig (3) Custom Garment Input Module.

4. Virtual Try-On Generation Module

This module coordinates the try-on request. The FastAPI backend receives the person image and selected or uploaded garment, prepares the images, and sends them to the Google Gemini image-generation service with instructions to fit the garment onto the person. The generated image is saved by the backend and returned to the frontend.

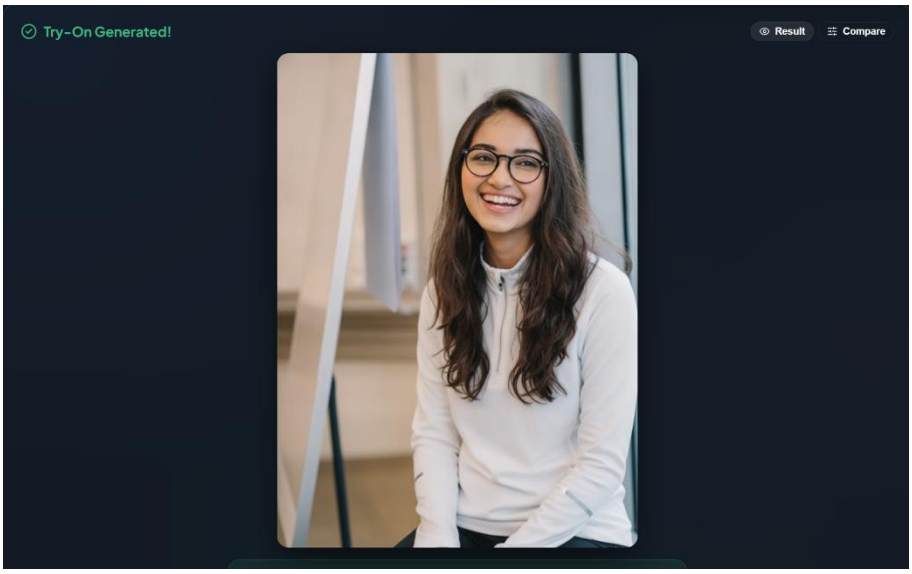
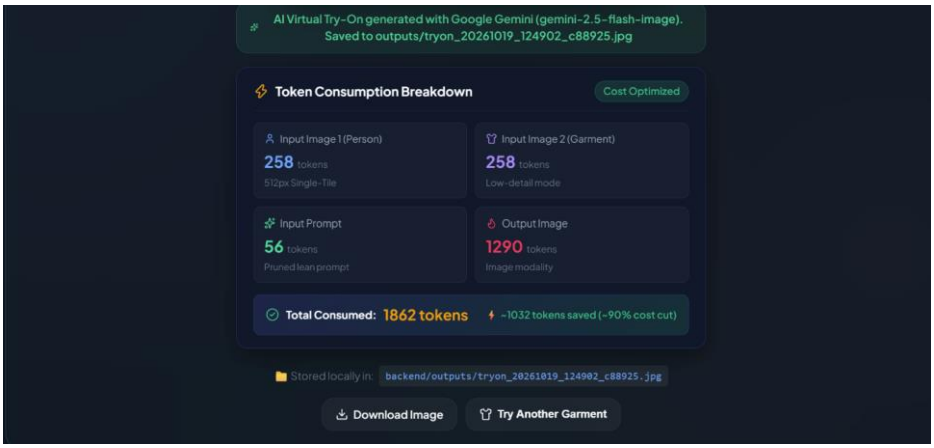


Fig (4) Virtual Try-On Generation Module.

5. Generated Result and Token Usage Module

After the virtual try-on process is completed, the system returns the generated image to the frontend for display and download. The backend saves the image in the backend/outputs folder and returns its filename, image URL, generation status, and model information. The result screen also presents a token consumption breakdown for the person image, garment image, input prompt, and output image, along with the total token count and estimated tokens saved. Users can download the generated image or try another garment.



C. System Workflow

The proposed AI-Based Virtual Try-On System with API Cost Optimization follows a structured workflow to generate virtual clothing previews while minimizing unnecessary API consumption.

1. The user uploads a personal photograph and selects a garment image from the product catalogue or uploads a clothing reference image.
2. The system validates the uploaded images and checks whether they satisfy the required input conditions.
3. The image preprocessing module resizes and compresses the images to reduce unnecessary data while retaining important garment details.
4. The prompt optimization module prepares concise instructions for the generative AI model, reducing redundant text while preserving essential requirements.
5. The FastAPI backend submits the preprocessed images and optimized prompt to the configured AI image-generation API.
6. The AI model generates a virtual try-on image based on the supplied person photograph and garment reference image.
7. The backend processes the generated response and returns the resulting image to the frontend for display.
8. The system records or measures available API usage information, estimated generation cost, processing time, and output quality for evaluation.
9. The generated image is displayed to the user, completing the virtual try-on process.

The workflow integrates image preprocessing, prompt optimization, and AI-based image generation into a single application. It provides an interactive virtual clothing visualization experience while enabling the evaluation of API cost reduction and generated-image quality across multiple clothing categories.

IV. IMPLEMENTATION

A. User Interface

The system provides a web interface through which users can browse garments and create virtual try-on images. Users can select garments from categories such as T-shirts, shirts, dresses, jackets, sarees, and jeans. They can search and filter catalogue items, or upload a custom garment image.

To provide an image of the person, users can upload a photo, capture one using the device camera, or select a preset model. The interface previews the selected images and displays the generated try-on result, with options to download it or try another garment.

B. Garment Catalogue Management

The system maintains its garment catalogue using image files organized by category and product information from a fashion dataset. The catalogue service reads this information and provides garment categories and product details to the application.

Users can browse the available items and filter them by attributes such as gender and color. The catalogue images are served by the backend to the frontend. The project uses these organized files and dataset records rather than a MySQL database.

C. Server-Side Processing

The backend is developed using Python and FastAPI. It provides API endpoints for retrieving garment categories, products, and preset model images, and for receiving virtual try-on requests from the frontend.

When a request is submitted, the backend receives the person image and the selected or uploaded garment image. It prepares the images and sends them to the Google Gemini image-generation service with instructions to generate a virtual try-on result. The backend then saves the generated image in its output folder and returns result information to the frontend.

D. Result and Token Usage Reporting

After image generation, the system displays the try-on result and provides options to download the image or try another garment. Generated images are saved in the backend's output folder, and the response includes information such as the image filename, image URL, generation status, and model used.

The result screen also presents token usage information for the input images, prompt, and generated output, together with the total token count and an estimated token saving. This helps users review the generation details alongside the result

V. RESULTS AND DISCUSSION

A. RESULT

The virtual try-on system was developed to let users preview how a selected **[garment/accessory]** may look on an uploaded image or live camera view. The user selects an item, provides an image or activates the camera, and views the generated preview. The system displays the selected item on the user's image, helping them assess its appearance without physically trying it on.

The project demonstrates the main virtual try-on workflow: **[uploading an image / capturing a camera image]**, selecting an item, generating the preview, and displaying the result. During testing, the system **[successfully generated previews for ___ items / worked under ___ conditions]**. The output quality depended on factors such as image clarity, lighting, pose, and how well the item aligned with the user's body.

B. DISCUSSION

The project provides a convenient way to preview products and may help users make more informed choices before purchase. It also demonstrates how image processing and **[the technology or model used in your project]** can be applied to an interactive shopping experience.

The preview is an approximation and may not perfectly represent the item's actual fit, size, color, or appearance. Results can vary with the input image and the accuracy of item placement. Further improvements could include more precise body or facial landmark detection, better alignment and scaling, support for more item types, and testing with a wider range of users and images.

VI. CONCLUSION

The virtual try-on mini project demonstrates how digital technology can help users preview selected items on their own images before making a purchase. The system brings together image input, item selection, and preview generation in a simple workflow, offering a convenient alternative to relying only on product photos.

The quality of the preview depends on the input image and how accurately the selected item is positioned. Although the result may not perfectly represent real-world fit or appearance, the project provides a useful foundation for improving online shopping experiences. Future enhancements could include more accurate item alignment, support for additional products, faster processing, and testing with a wider range of images and users.

VII. REFERENCES

1. B. Wang, H. Zheng, X. Liang, Y. Chen, L. Lin, and M. Yang, "Toward Characteristic-Preserving Image-Based Virtual Try-On Network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.
2. A. H. Raffiee and M. Sollami, "Adversarial Semantic Image Synthesis for Virtual Try-On," *arXiv preprint arXiv:1807.07688*, 2018.
3. H. Yang, R. Zhang, X. Guo, W. Liu, W. Zuo, and P. Luo, "Towards Photo-Realistic

Virtual Try-On by Adaptively Generating-Preserving Image Content," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

4. S. Choi, S. Park, M. Lee, and J. Choo, "VITON-HD: High-Resolution Virtual Try-On via Misalignment-Aware Normalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
5. D. Morelli, A. Fincato, M. Cornia, F. Landi, F. Cesari, and R. Cucchiara, "Dress Code: High-Resolution Multi-Category Virtual Try-On," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2022.
6. J. Xie et al., "StableVITON: Learning Semantic Correspondence with Latent Diffusion Model for Virtual Try-On," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
7. Z. Zhu et al., "CatVTON: Concatenation Is All You Need for Virtual Try-On with Diffusion Models," *arXiv preprint*, 2024.
8. Google, "Gemini API Documentation," Google AI for Developers. Available: https://ai.google.dev/gemini-api/docs
9. Google, "Gemini API Pricing," Google AI for Developers. Available: https://ai.google.dev/gemini-api/docs/pricing
10. Google, "Image Understanding and Image Generation with the Gemini API," Google AI for Developers. Available: https://ai.google.dev/gemini-api/docs/image-understanding
11. FastAPI, "FastAPI Documentation." Available: https://fastapi.tiangolo.com/
12. React, "React Documentation." Available: https://react.dev/
13. Vite, "Vite Documentation." Available: https://vite.dev/guide/
14. OpenCV, "OpenCV Documentation." Available: https://docs.opencv.org/
15. Python Software Foundation, "Python Documentation." Available: https://docs.python.org/3/