Image Processing for Virtual Apparel Fitting

A Mini Project Report

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

in

Computer Science and Engineering

By

BOYILLA DHANIKA	**2456-21-733-134**
BUDHARAJU POOJITHA	**2456-21-733-140**
CHALNATI ANUSREE	**2456-21-733-142**

Under the guidance of

Mrs . K. Pranathi

Asst.Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Gokaraju Lailavathi Womens Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Nizampet, Hyderabad-500090

(2023-2024)

I

Department of Computer Science and Engineering

Gokaraju Lailavathi Womens Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Nizampet, Hyderabad-500090

(2023-2024)

CERTIFICATE

This is to Certify that A Mini Project report entitled **"Image Processing For Virtual Apparel Fitting**" is being submitted by Boyilla Dhanika **(**2456-21-733-134**),** Budharaju Poojitha **(**2456-21-733-140**),** Chalnati Anusree (2456-21-733-142) in partial fulfillment of the requirement of the award for the degree of Bachelor of Engineering in "Computer Science and Engineering" O.U., Hyderabad during the year 2023-2024 is a record of bonafide work carried out by them under my guidance. The results presented in this project have been verified and are found to be satisfactory.

Project Guide

Mrs . K. Pranathi

Asst.Professor,

Dept of CSE

H.O.D

Dr. Padmalaya Nayak

Professor & Head,

Dept. of CSE

External Examiner(s)

Department of Computer Science and Engineering

Gokaraju Lailavathi Womens Engineering College

Accredited by NBA & NAAC

(Affiliated to Osmania University, Approved by AICTE)

Nizampet, Hyderabad-500090

(2023-2024)

DECLARATION

We, Boyilla Dhanika bearing Ht.No. 2456-21-733-134, Budharaju Poojitha bearing Ht.No. 2456-21-733-140 and Chalnati Anusree bearing Ht.No. 2456-21-733-142 hereby certify that the minor project entitled "**Image Processing For Virtual Apparel Fitting**" is submitted in the partial fulfilment of the required for the award of the degree of Bachelor of Engineering in Computer Science and Engineering.

This is a record work carried out by us under the guidance of Mrs . K. Pranathi, Asst.Professor, CSE, Gokaraju Lailavathi Womens Engineering College, Nizampet. The results embodied in this report have not been reproduced/copied from any source. The results embodied in this report have not been submitted to any other university or institute for the award of any other degree or diploma.

Boyilla Dhanika	**2456-21-733-134**
Budharaju Poojitha	**2456-21-733-140**
Chalnati Anusree	**2456-21-733-142**

ACKNOWLEDGEMENT

It is our privilege and pleasure to express my profound sense of respect, gratitude and indebtedness to our guide Mrs . K. Pranathi, Asst.Professor, Department of Computer Science and Engineering, Gokaraju Lailavathi Womens Engineering College, for her inspiration, guidance, cogent discussion, constructive criticisms and encouragement throughout this dissertation work.

We express our sincere thanks to Project Coordinator Mr Dhanunjay Rao, Assistant Professor, Department of Computer Science and Engineering, Gokaraju Lailavathi Womens Engineering College, for his valuable suggestions and constant helping in completing the work.

We express our sincere thanks to Dr. A. Sai Hanuman, Principal, Gokaraju Lailavathi Womens Engineering College, Nizampet, Hyderabad, for his encouragement and constant help.

We extend our sincere thanks to all the teaching and non-teaching staff of Computer Science and Engineering Department for their support and encouragement.

Last but not least, we wish to acknowledge my friends and family members for giving moral strength moral strength and helping us to complete this dissertation.

ABSTRACT

The Virtual Apparel Try-On project offers an advanced virtual try-on experience, enabling users to overlay a clothing image onto a model image seamlessly. Leveraging OpenCV, the application processes the uploaded images by isolating the clothing from its background and resizing it to fit the model accurately. Users interact with a straightforward web form to upload their images, and the resulting composite image is displayed alongside the original uploads, providing a realistic and immediate visualization. With robust error handling, the project ensures smooth image upload and processing, addressing any potential issues promptly. This integration of sophisticated image processing techniques within a web framework showcases the potential of AI-driven solutions in the fashion industry, delivering a professional and engaging virtual try-on experience.

CONTENTS

LIST OF TABLES

CHAPTER-1

INTRODUCTION

1.1 MOTIVATION

The fashion industry has been rapidly evolving with the growth of e-commerce, but online shopping for clothing still faces a significant challenge: customers cannot try on garments before purchasing. This limitation often leads to hesitation in buying, increased return rates, and customer dissatisfaction. Our motivation for this project stems from the desire to bridge this gap between physical and online shopping experiences.

Virtual try-on technology has the potential to revolutionize how people shop for clothes online. By allowing customers to see how garments might look on them without physically wearing them, we can enhance the online shopping experience, reduce returns, and increase customer confidence in their purchases.

1.2 PROBLEM STATEMENT

The main problem this project addresses is the inability of online shoppers to visualize how clothing items will look on them before making a purchase. Traditional online shopping platforms provide static images of garments, often displayed on models who may not represent the diverse body types of all customers. This leads to several issues:

1. Uncertainty about fit and appearance
2. High return rates for online clothing retailers
3. Customer hesitation to purchase clothing online
4. Limited ability for customers to experiment with different styles

Our project aims to solve these problems by creating a virtual try-on system that allows users to upload their own photos and see how selected clothing items would look on them.

1.3 PROJECT OBJECTIVE

The primary objective of this project is to develop a web-based virtual clothing try-on system that seamlessly integrates computer vision techniques with web development.
Specifically, our objectives are:

1. To create a user-friendly web interface where users can upload two images: one of themselves (or a model) and one of a clothing item.
2. To implement image processing algorithms that can effectively remove the green screen background from the clothing image.
3. To develop a method for accurately overlaying the processed clothing image onto the user/model image, taking into account scaling and positioning.
4. To ensure the system processes images quickly and efficiently, providing users with near-real-time results.
5. To create a visually appealing output that realistically simulates how the clothing item would look when worn by the user/model.

By achieving these objectives, we aim to create a practical tool that enhances the online shopping experience, provides value to both consumers and retailers, and demonstrates the power of combining computer vision with web technologies in solving real-world problems.

CHAPTER 2
LITERATURE SURVEY

The advancement of multimedia technology has significantly enhanced the utilization of cameras and image/video processing across various applications, including facial recognition.

2.1 VIRTUAL TRY-ON TECHNOLOGY OVERVIEW

Virtual try-on technology has emerged as a transformative solution in the fashion industry, allowing users to visualize clothing on themselves without physical trials. By leveraging computer vision techniques, these systems enable users to upload images of themselves and overlay clothing images in real-time. This technology enhances the online shopping experience by providing a more interactive and personalized approach, reducing return rates, and increasing customer satisfaction.

2.2 IMAGE SEGMENTATION TECHNIQUES

Image segmentation is a critical component of virtual try-on systems, as it involves isolating clothing from its background to facilitate accurate overlays. Techniques such as color-based segmentation in the HSV color space are commonly used to create masks that differentiate clothing from backgrounds. These methods are essential for ensuring that the overlay appears natural and realistic on the model image.

2.3 GREEN SCREEN REMOVAL METHODS

Green screen removal techniques are often employed in virtual try-on applications to isolate clothing items. By utilizing color segmentation, these methods allow for the effective extraction of clothing from images with uniform backgrounds. This process is crucial for achieving high-quality overlays, as it ensures that only the clothing is displayed on the model image, enhancing the overall visual appeal.

2.4 IMAGE OVERLAY AND BLENDING ALGORITHMS

Image overlay and blending algorithms play a vital role in combining the clothing and model images seamlessly. Techniques such as bitwise operations and alpha blending are used to ensure that the clothing is accurately placed on the model while maintaining the natural appearance of both images. These algorithms enhance the realism of the virtual try-on experience, providing users with a clear visualization of how the clothing will look when worn.

2.5 WEB-BASED IMAGE PROCESSING SYSTEMS

Web-based image processing systems have gained popularity due to their accessibility and ease of use. By leveraging frameworks like Flask, developers can create applications that allow users to upload images and receive processed results in real-time. These systems utilize libraries such as OpenCV for image manipulation, enabling efficient processing and rendering of virtual try-on experiences directly in the browser.

2.6 COMPARISON OF VIRTUAL TRY-ON APPROACHES

Various approaches to virtual try-on technology exist, each with its strengths and weaknesses. Some systems rely on 2D image overlays, while others explore 3D modeling for more accurate representations. Comparative studies indicate that while 3D modeling offers enhanced realism, 2D overlay methods are often more accessible and easier to implement, making them suitable for a wider range of applications.

2.7 CURRENT CHALLENGES IN VIRTUAL CLOTHING TRY-ON

Despite advancements in virtual try-on technology, several challenges remain. These include ensuring accurate fit and alignment of clothing on diverse body shapes, handling variations in lighting conditions, and optimizing image processing for real-time applications. Addressing these challenges is crucial for enhancing the user experience and achieving high levels of accuracy in virtual try-on systems.

2.8 CONCLUSION

The literature highlights the importance of developing robust virtual try-on systems that leverage advanced image processing techniques to enhance the online shopping experience. By focusing on effective image segmentation, overlay algorithms, and web-based processing, this project aims to contribute to the existing body of knowledge in virtual try-on technology. The findings from this literature survey will guide the development of the Virtual Cloth Assistant, addressing current challenges and optimizing the overall performance of the system.

CHAPTER-3

SOFTWARE REQUIREMENT SPECIFICATION

This chapter gives an overview of the software and hardware components required for our project.

3.1 SOFTWARE REQUIREMENTS

Operating System : Windows 10

Coding Language : Python 3.10.7

3.2 HARDWARE REQUIREMENTS

System : intel i5 or above

Storage : Sufficient storage

3.3 FUNCTIONAL REQUIREMENTS

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

- Image Upload: Allow users to upload model and clothing images.
- Green Screen Removal: Remove green background from clothing image.
- Image Resizing: Resize clothing to match model dimensions.
- Image Overlay: Seamlessly overlay clothing onto model image.
- Result Display: Show original and processed images to user.
- Web Interface: Provide user-friendly interface for interaction.
- Error Handling: Manage and communicate processing errors.

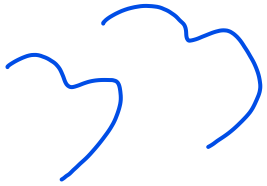
- Real-time Processing: Process images quickly for immediate results.

3.4 NON-FUNCTIONAL REQUIREMENTS

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability: Accessible across devices and browsers.
- Security: Protect user data and uploads.
- Maintainability: Well-documented, modular code.
- Reliability: Handle various inputs without crashing.
- Scalability: Support multiple concurrent users.
- Performance: Process images within 5 seconds.
- Usability: Intuitive interface requiring minimal training.
- Compatibility: Work with major web browsers.
- Efficiency: Optimize resource usage.
- Accessibility: Adhere to basic web accessibility guidelines.



CHAPTER-4
SYSTEM DESIGN

4.1 System Design

In this phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture.

There are two kinds of design documents developed in this phase:

High-Level Design (HLD)

- Brief description and name of each module
- An outline about the functionality of every module
- Interface relationship and dependencies between modules
- Database tables identified along with their key elements
- Complete architecture diagrams along with technology details

Low-Level Design(LLD)

- Functional logic of the modules
- Database tables, which include type and size
- Complete detail of the interface
- Addresses all types of dependency issues
- Listing of error messages

4.2 PROPOSED METHODOLOGY:

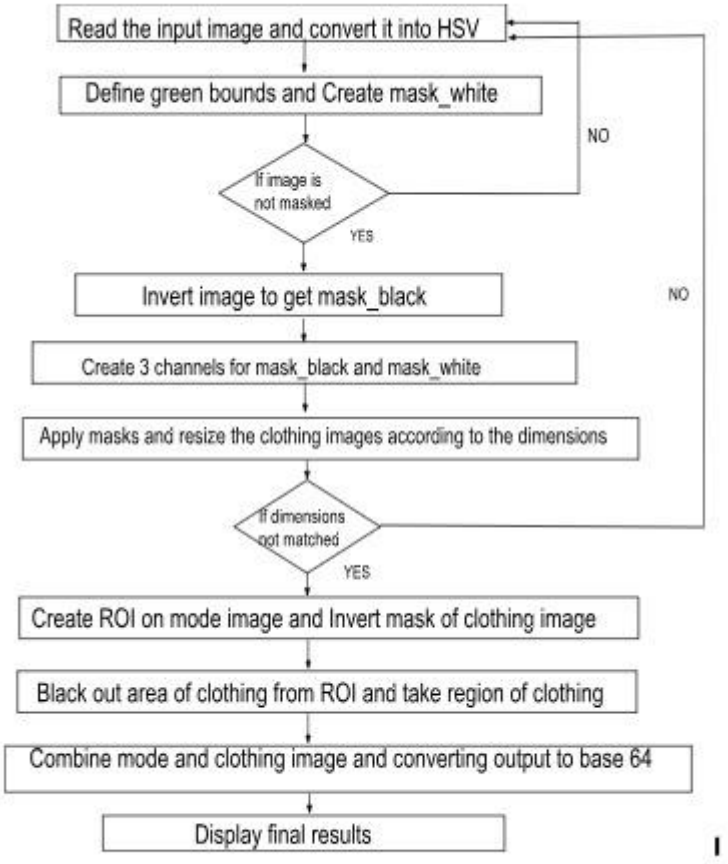


Figure-4.2.1 Proposed Methodology

84

4.3 UML Design:

Unified Modeling Language (UML) is a general purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed.

UML is not a programming language; it is rather a visual language. We use UML diagrams to portray the behavior and structure of a system, UML helps software engineers, businessmen and system architects with modeling, design and analysis. The Object Management Group (OMG) adopted Unified Modeling Language as a standard in 1997. It's been managed by OMG ever since. International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodically.

Do we really need UML?

- Complex applications need collaboration and planning from multiple teams and hence require a clear and concise way to communicate amongst them.
- Businessmen do not understand code. So UML becomes essential to communicate with non programmer's essential requirements, functionalities and processes of the system.
- UML is linked with object oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

Types of UML Diagrams:

Structural Diagrams:

Capture static aspects or structure of a system. Structural Diagrams include: Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

Behavior Diagrams:

Capture dynamic aspects or behavior of the system. Behavior diagrams include: Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

The image below shows the hierarchy of diagrams according to UML

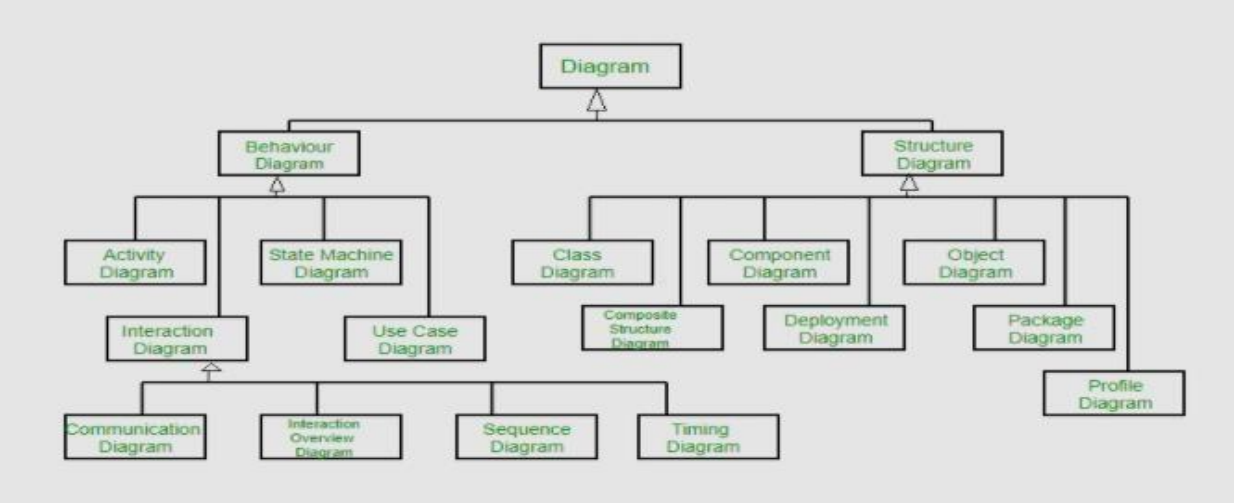


Figure-4.3.1 UML Hierarchy diagrams

4.3.1 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

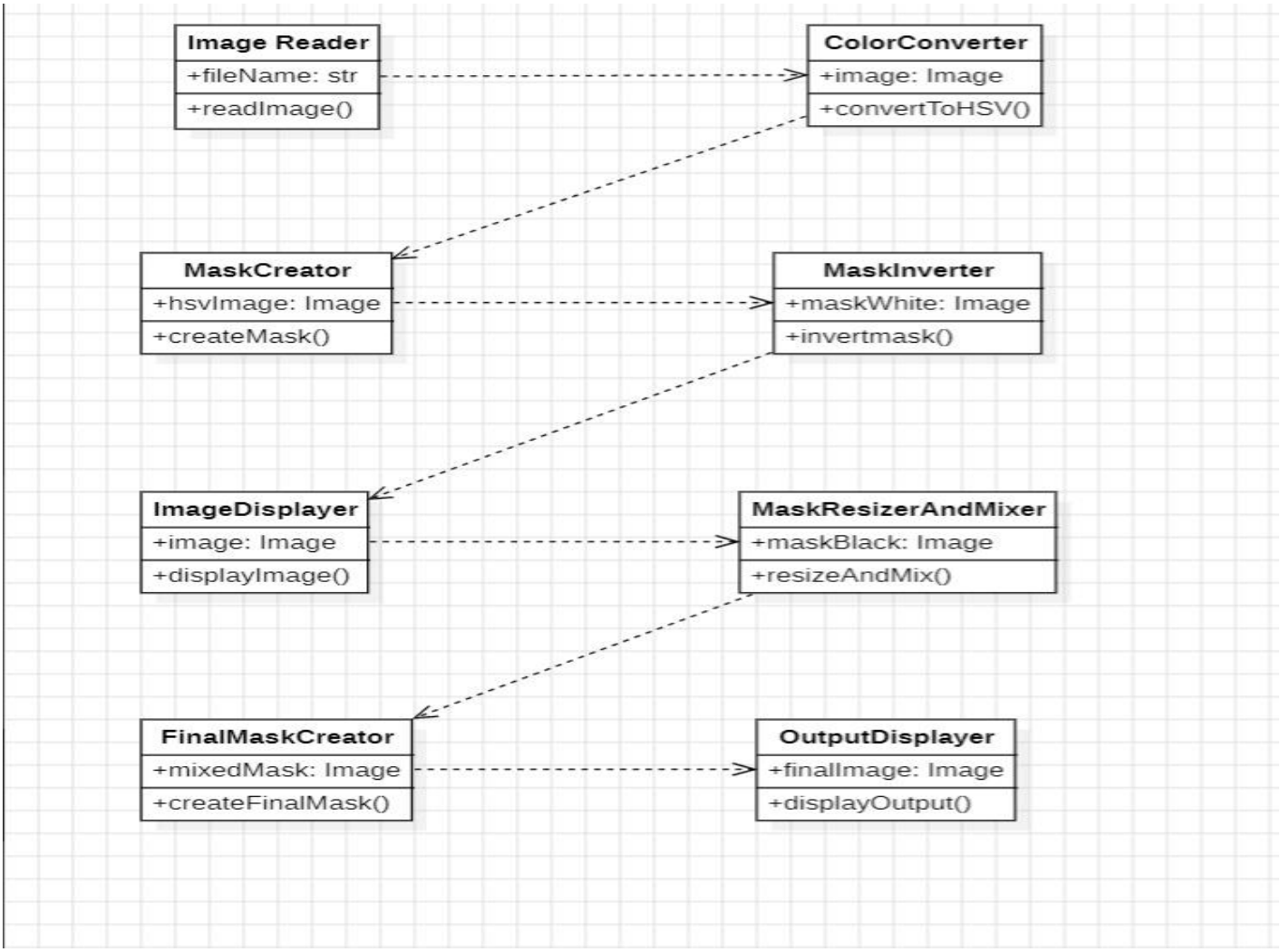


Figure-4.3.1.1 Class Diagram

4.3.2 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

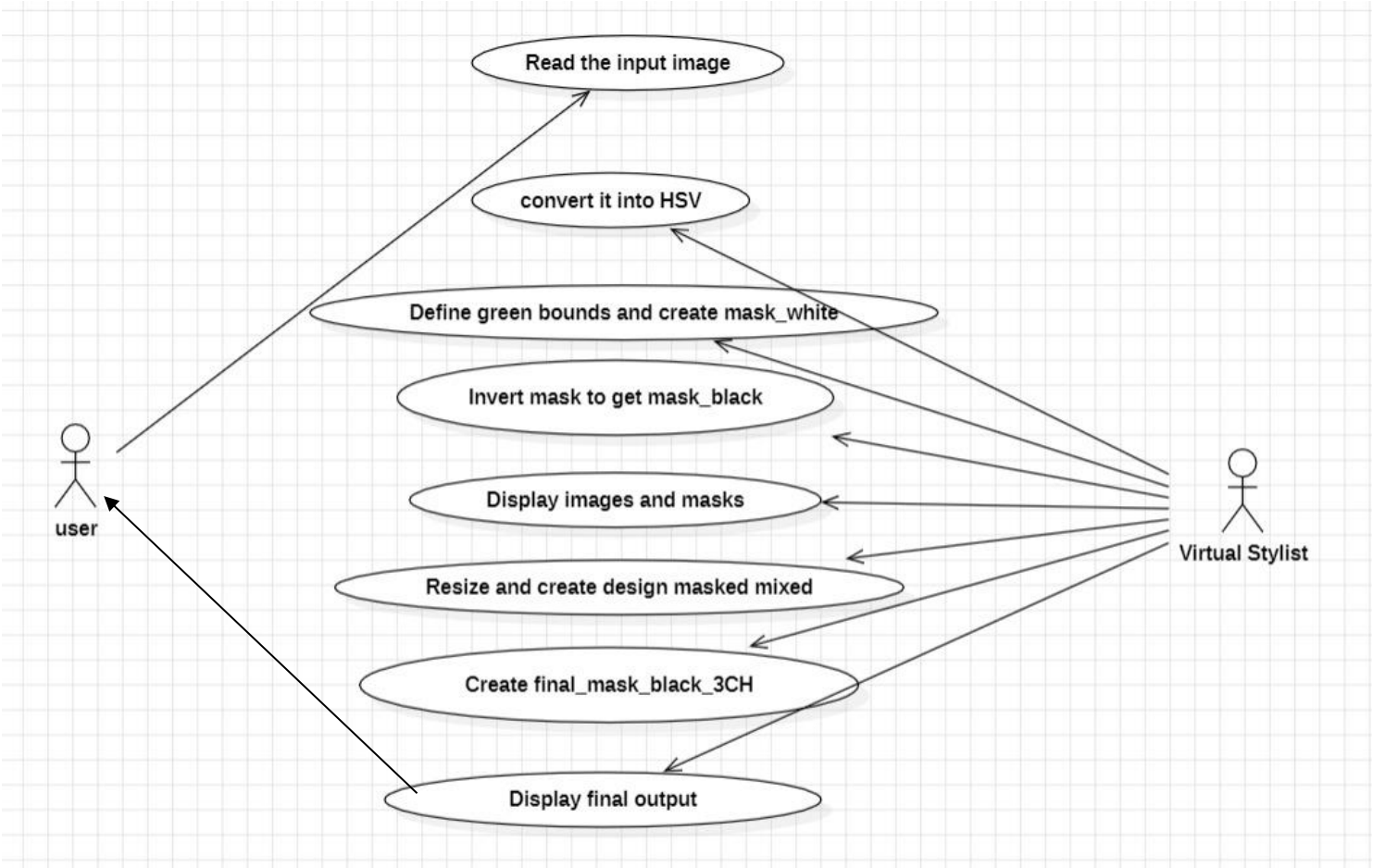


Figure-4.3.2.1 Use Case Diagram

4.3.3 COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams areoften drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.

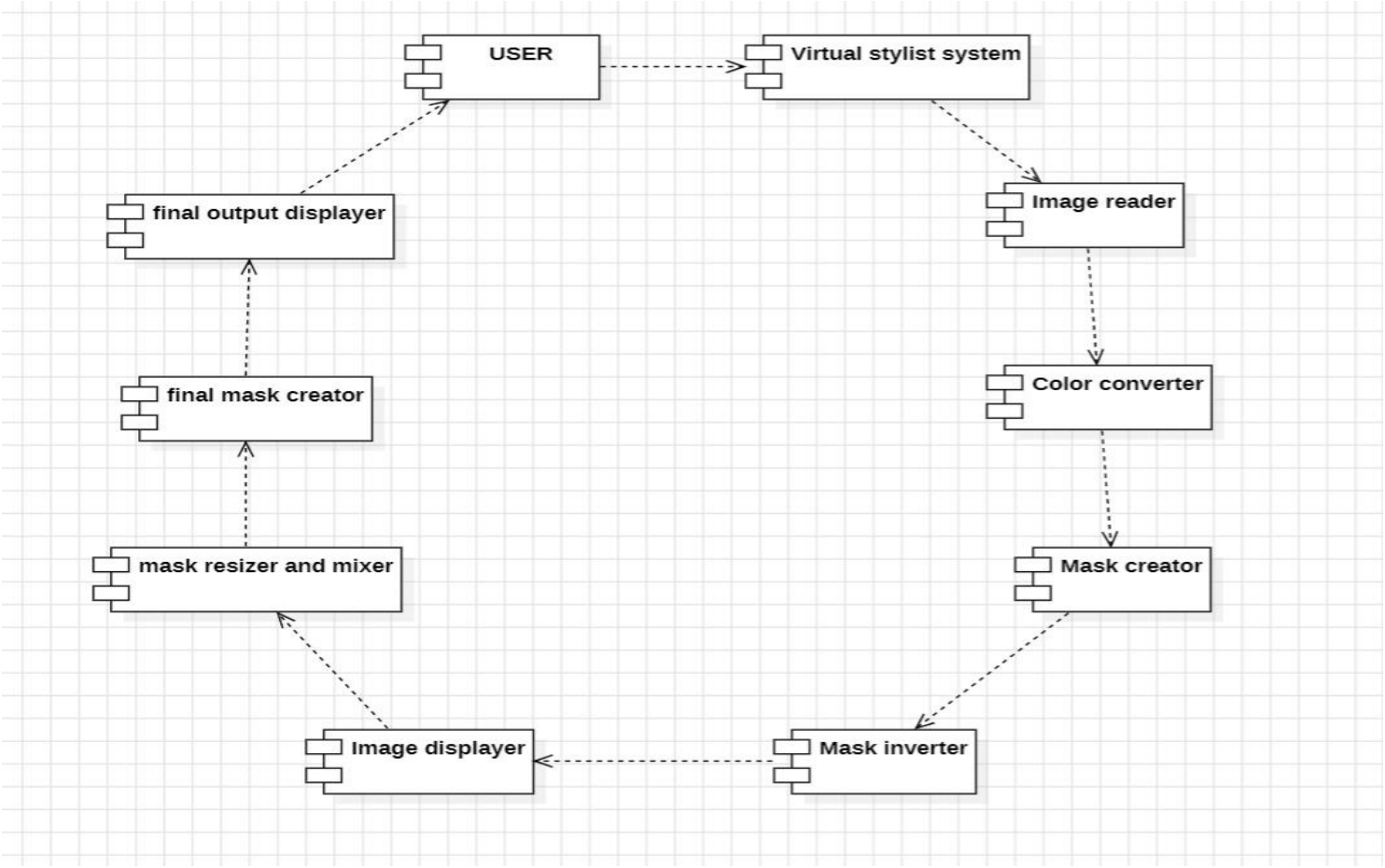


Figure-4.3.3.1 Component Diagram

4.3.4 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, eventscenarios, and timing diagrams.

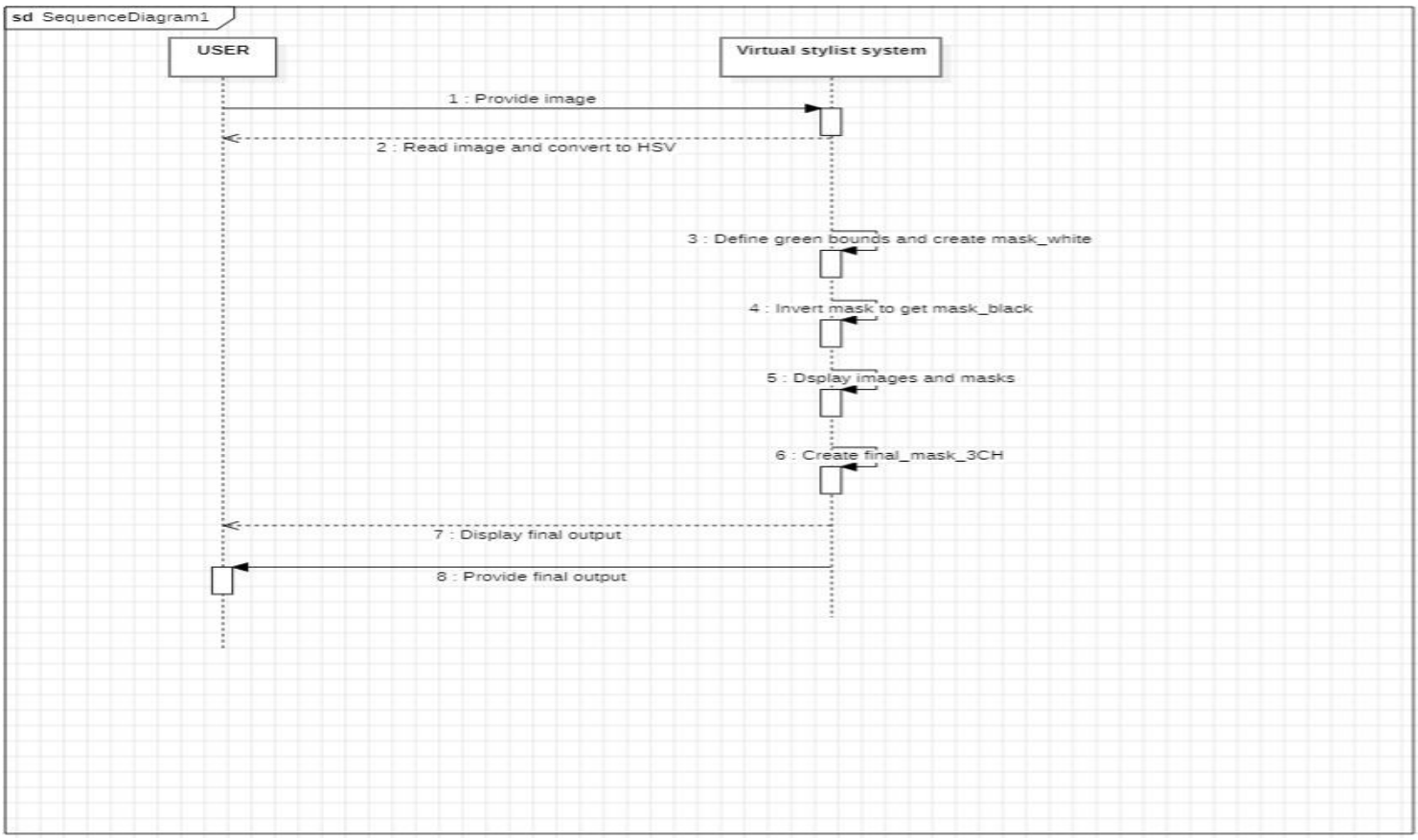


Figure-4.3.4.1 Sequence Diagram

4.3.5 ACTIVITY DIAGRAM:

In UML, an activity diagram is used to display the sequence of activities. Activity diagrams show the workflow from a start point to the finish point detailing the many decision paths that exist in the progression of events contained in the activity.

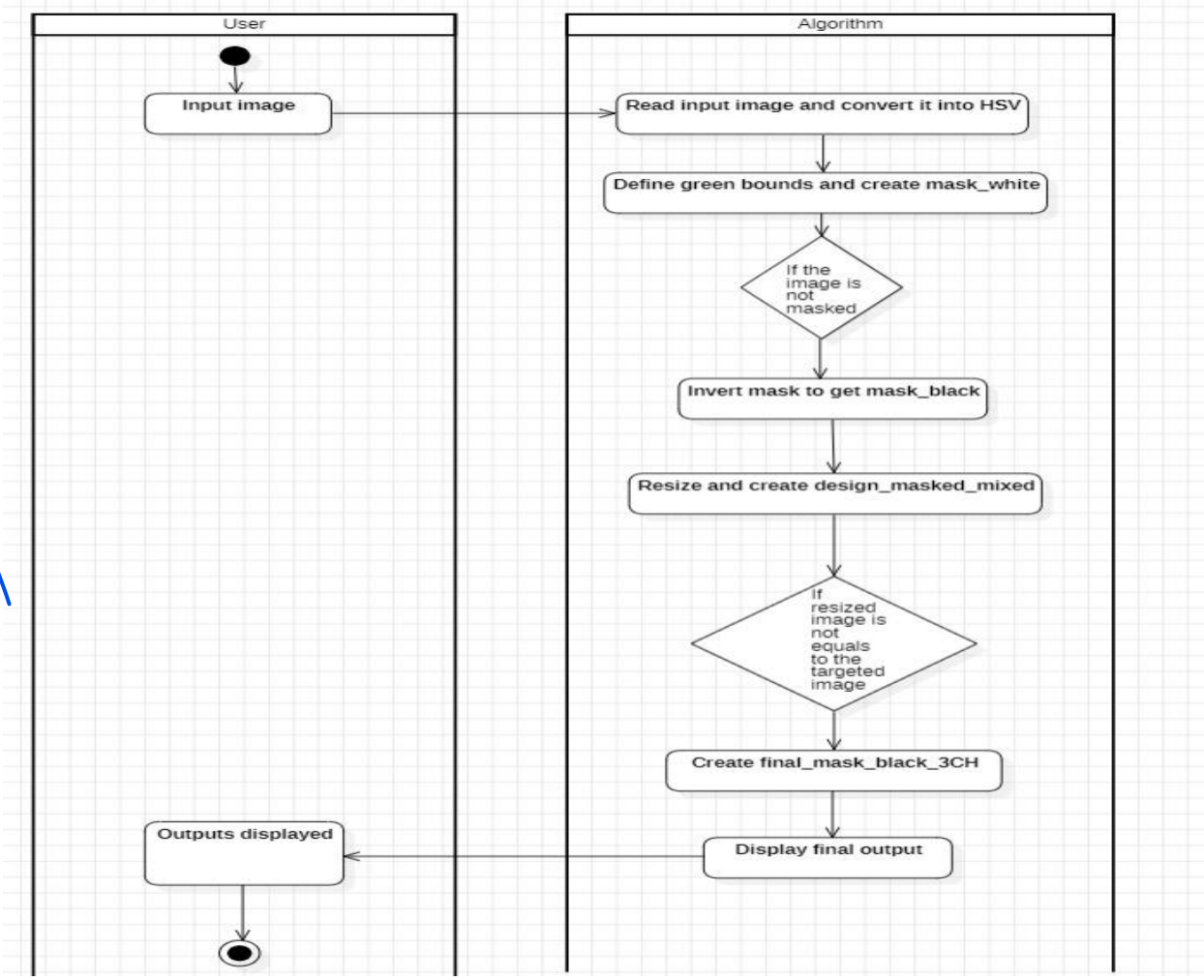


Figure-4.3.5.1 Activity Diagram

4.4 TECHNOLOGY DESCRIPTION

OpenCV (Open Source Computer Vision Library)

OpenCV is a powerful open-source library designed for computer vision and image processing tasks. It offers a wide array of functionalities for real-time image and video analysis, including facial recognition, object detection, and feature extraction. OpenCV supports multiple programming languages such as Python, C++, and Java, making it versatile and widely used in both academic and industrial applications. The library provides tools for implementing complex computer vision algorithms, streamlining the development of facial recognition systems, and enhancing image processing capabilities.

Image Preprocessing Techniques

The project employs several image preprocessing techniques to prepare the uploaded images for effective analysis. This includes reading the images using OpenCV, converting the clothing image from BGR to HSV color space for better segmentation, and creating binary masks to isolate the clothing from its background. The clothing image is resized to match the model's dimensions, ensuring proper alignment during the overlay process. These preprocessing steps are crucial for optimizing the images for the overlay process and ensuring high-quality results.

Mask Creation and Bitwise Operations

The application uses mask creation techniques to isolate the clothing from its background. By applying color-based segmentation through the HSV color space, the project generates binary masks that differentiate the clothing from the background. Bitwise operations are then used to combine the masked clothing with the model image, allowing for a seamless overlay that enhances the user experience.

CHAPTER 5

5.1 IMPLEMENTATION

Development Environment Setup

The implementation process begins with setting up the development environment, which involves installing essential software and libraries such as Python and OpenCV. OpenCV is a vital library for computer vision tasks, providing a range of functions for image processing and manipulation. To support the development and testing of the Virtual Cloth Assistant application, it is crucial to ensure that all dependencies, including Flask for web development, are correctly installed and configured. This setup lays the groundwork for developing the image processing algorithms that facilitate the virtual try-on experience.

Image Preprocessing

Image preprocessing is a critical step in preparing input data for effective clothing overlay. This process includes converting the uploaded clothing images and model images to a format suitable for processing. The images are resized to a standard dimension to ensure uniformity, which is essential for accurate overlay and blending. Additionally, normalization of pixel values may be performed to enhance the performance of the image processing algorithms. These preprocessing steps are vital for improving the quality and accuracy of the virtual try-on results.

Clothing Overlay Using OpenCV

The core functionality of the Virtual Cloth Assistant revolves around overlaying the clothing image onto the model image using OpenCV. The application utilizes techniques such as color space conversion to HSV, mask creation to isolate the clothing from its background, and bitwise operations to blend the images seamlessly. The processing pipeline includes:

1. Mask Creation: The application generates masks to isolate the clothing based on its color. This involves creating

binary masks that differentiate between the clothing and the background.

2. Image Resizing: The clothing image is resized to match the dimensions of the model image, ensuring a proper fit.
3. Overlaying the Clothing: The masked clothing is then combined with the model image, allowing for a realistic representation of how the clothing would appear on the user.

This method is efficient and robust, capable of handling various clothing styles and colors while maintaining the integrity of the model image.

Distance Optimization

To achieve optimal results during the virtual try-on process, it is essential to consider the distance between the camera and the model during image capture. Although this project does not involve real-time distance measurement, testing with images taken at various distances can help determine the optimal range for capturing images that yield the best overlay results. This optimization ensures that the application performs effectively across different user scenarios.

Integration and Testing

The integration phase combines the image processing algorithms with the web application's front-end interface. The system is tested using uploaded images to validate its functionality. This testing phase includes evaluating the accuracy of the clothing overlay, the effectiveness of the masking techniques, and the overall performance of the application under different conditions. Any issues identified during testing are addressed to ensure the system operates as intended.

Deployment

Deployment involves setting up the application in a practical environment, which includes hosting the Flask application on a server accessible to users. The application is configured to handle image uploads and process them efficiently. Ensuring that the server environment is optimized for image processing tasks is crucial for

maintaining performance and responsiveness.

Evaluation and Fine-Tuning

Following deployment, the application undergoes a thorough evaluation to assess its performance and accuracy. This involves testing the system with various clothing and model images to identify potential weaknesses or areas for improvement. Fine-tuning may include adjusting parameters within the image processing algorithms and enhancing preprocessing techniques to improve the overall reliability and efficiency of the application. This iterative process ensures that the Virtual Cloth Assistant meets the desired performance standards and operates effectively in real-world applications.

5.2 MODULES:

Image Capture Module

The Image Capture Module is essential for obtaining images through user uploads. This module allows users to upload clothing and model images, which are then processed for the virtual try-on experience. High-quality images are crucial for accurate overlay and visualization, making this module a foundational component of the overall system.

Image Preprocessing Module

The Virtual Cloth Assistant is a web application that allows users to virtually try on clothing by overlaying a clothing image onto a model image. The application uses OpenCV for image processing, including color space conversion, mask creation, and image resizing, to seamlessly integrate the clothing onto the model. Users can upload their own clothing and model images, and the application displays the resulting composite image alongside the original uploads. The project demonstrates the potential of AI-driven solutions in enhancing online shopping experiences in the fashion industry.

Clothing Overlay Module

The Clothing Overlay Module executes the core functionality of the application by utilizing OpenCV techniques. It creates masks to isolate clothing from its background, resizes images to ensure proper alignment, and blends the clothing onto the model image. This module ensures that the clothing is accurately placed and visually appealing, enhancing the overall user experience.

Integration and Testing Module

The Integration and Testing Module combines all system components, including image capture, preprocessing, and overlay functionalities. It validates the system's performance by testing various image combinations to ensure seamless interaction and accuracy. Any identified issues are addressed to guarantee the system operates as intended.

Deployment Module

The Deployment Module focuses on setting up the application in a real-world environment. This includes hosting the Flask application on a server and configuring it for optimal performance. The module ensures that the application is operational and effectively performs clothing overlays in the intended environment.

Evaluation and Fine-Tuning Module

The Evaluation and Fine-Tuning Module assesses the application's performance after deployment. It tests the system under various conditions to identify areas for improvement. Fine-tuning involves adjusting parameters in the overlay algorithms and enhancing preprocessing techniques to ensure the application meets desired performance standards. This iterative process is essential for maintaining the application's effectiveness in real-world scenarios.

5.3 EXECUTABLE CODE

app.py

```
from flask import Flask, request, render_template
import base64
import numpy as np
import cv2

app = Flask(__name__)

def process_image(model_image, clothing_image):
    try:
        # Read the images using OpenCV
        model = cv2.imdecode(np.frombuffer(
            model_image, np.uint8), cv2.IMREAD_UNCHANGED)
        clothing = cv2.imdecode(np.frombuffer(
            clothing_image, np.uint8), cv2.IMREAD_UNCHANGED)

        # Convert the clothing image to HSV
        hsv = cv2.cvtColor(clothing, cv2.COLOR_BGR2HSV)
        lower_green = np.array([25, 52, 72])
        upper_green = np.array([102, 255, 255])

        # Create masks to isolate the green screen
        mask_white = cv2.inRange(hsv, lower_green, upper_green)
```

```
    mask_black = cv2.bitwise_not(mask_white)

    # Prepare the masks for bitwise operations
    mask_black_3CH = cv2.merge([mask_black, mask_black, mask_black])
    mask_white_3CH = cv2.merge([mask_white, mask_white, mask_white])

    # Apply the masks to the clothing image
    dst3 = cv2.bitwise_and(clothing, mask_black_3CH)

    # Resize the clothing image to match the model's dimensions
    model_h, model_w = model.shape[:2]
    clothing_resized = cv2.resize(dst3, (model_w, model_h))

    # Create a region of interest on the model image where the clothing will be placed
    roi = model[0:model_h, 0:model_w]

    # Create a mask of the clothing and its inverse mask
    clothing_gray = cv2.cvtColor(clothing_resized, cv2.COLOR_BGR2GRAY)
    _, mask = cv2.threshold(clothing_gray, 1, 255, cv2.THRESH_BINARY)
    mask_inv = cv2.bitwise_not(mask)

    # Black-out the area of the clothing in the ROI
    model_bg = cv2.bitwise_and(roi, roi, mask=mask_inv)

    # Take only the region of the clothing from the clothing image
    clothing_fg = cv2.bitwise_and(
        clothing_resized, clothing_resized, mask=mask)

    # Put the clothing in the ROI and modify the main image
    dst = cv2.add(model_bg, clothing_fg)
    model[0:model_h, 0:model_w] = dst

    # Convert the final output to a format suitable for displaying in HTML
    _, buffer = cv2.imencode('.png', model)
    result_image = base64.b64encode(buffer).decode('ascii')

    return result_image
  except Exception as e:
    print(f"Error processing image: {e}")
    return None
```

```
@app.route('/', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        try:
            model_file = request.files['model'].read()
            clothing_file = request.files['clothing'].read()

            # Convert the images to base64 for displaying
            model_image = base64.b64encode(model_file).decode('ascii')
            clothing_image = base64.b64encode(clothing_file).decode('ascii')

            result_image = process_image(model_file, clothing_file)

            if result_image:
                return render_template('index.html', model_image=model_image, clothing_image=clothing_image,
result_image=result_image)
            else:
                return "Error processing images", 500
        except Exception as e:
            print(f"Error handling file upload: {e}")
            return "Error handling file upload", 500
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Virtual Cloth Assistant</title>
    <link
      rel="stylesheet"
```

```
    href="{{ url_for('static', filename='css/styles.css') }}"
  />
</head>
<body>
  <div class="container">
    <div class="header">
      <h1>Virtual Cloth Assistant</h1>
      <p>
        Wanna try out how that cloth suits you? Upgrade your shopping
        experience with an intelligent trial room.
      </p>
    </div>

    <form method="post" enctype="multipart/form-data">
      <div class="upload-container">
        <div class="upload-section">
          <label for="model">Upload Clothing Image:</label>
          <p>Please upload your Clothing image</p>
          <input
            type="file"
            id="model"
            name="model"
            accept="image/*"
            required
          />
          <div class="image-preview">
            <h3>Clothing Image:</h3>
            <img
              src="{{ model_image }}"
              alt="Model Image"
            />
          </div>
        </div>

        <div class="upload-section">
          <label for="clothing">Upload Model Image:</label>
          <p>Please upload your Model image</p>
          <input
            type="file"
            id="clothing"
```

```
            name="clothing"
            accept="image/*"
            required
          />
          <div class="image-preview">
            <h3>Model Image:</h3>
            <img
              src="{{ clothing_image }}"
              alt="Clothing Image"
            />
          </div>
        </div>
      </div>

      <div>
        <button type="submit" class="button">Try On</button>
      </div>
    </form>

    <div class="result-section">
      <h2>Result:</h2>
      <img
        src="{{ result_image }}"
        alt="Result Image"
        class="result-image"
      />
    </div>
  </div>
 </body>
</html>
```

styles.css

```
body {
  background-color: #181818;
  color: #ffffff;
  font-family: Arial, sans-serif;
  text-align: center;
```

```
}

.container {
  padding: 50px;
}

.header {
  margin-bottom: 40px;
}

.upload-container {
  display: flex;
  justify-content: center;
  gap: 20px;
  margin: 30px 0;
}

.upload-section {
  flex: 1;
  border: 1px solid #007bff;
  padding: 20px;
  border-radius: 10px;
}

.button {
  background-color: #007bff;
  color: white;
  padding: 10px 20px;
  border: none;
  cursor: pointer;
  margin-top: 10px;
}

.button:hover {
  background-color: #0056b3;
}

.result-section {
  margin-top: 20px;
  border: 1px solid #007bff;
```

```
  padding: 20px;
  border-radius: 10px;
  max-width: 600px;
  margin-left: auto;
  margin-right: auto;
}

.result-image {
  margin-top: 20px;
  max-width: 100%;
}

.image-preview img {
  max-width: 100%;
  height: auto;
}
```

CHAPTER - 6

TESTING

6.1 TESTING DEFINITION:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Unit tests were conducted manually, targeting specific functions and methods to validate their behavior.

Test objectives

- Confirm that each function accurately processes images.
- Ensure that image encoding and decoding functionalities operate without issues.
- Validate that exceptions are handled appropriately.

Features to be tested

- Verify that the images are of the correct format.
- Check for green bounds and background color.

Integration Testing

Integration testing evaluates the interaction between different modules to ensure they collaborate effectively.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

User Acceptance Testing

User Acceptance Testing is conducted to ensure the system aligns with the functional requirements of end users.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Test Cases:

S No	Test Case Description	Expected Outcome	Actual Outcome	Status
1.	Users upload images for virtual fitting	Users should successfully upload images and view results	Users uploaded images and saw results	Pass
2.	Assess image processing response time	Application should process images promptly	Images processed quickly	Pass
3.	Evaluate user interface clarity and usability	Interface should be user-friendly and straightforward	Interface was clear and easy to navigate	Pass

Figure-6.1 User acceptance test cases table

Performance Testing

Performance testing assesses the application's efficiency and its ability to handle load.

Test case

S No	Test Case Description	Expected Outcome	Actual Outcome	Status
1.	Check responsiveness of the user interface during processing	UI should remain responsive while images are being processed	User interface remained responsive throughout	Pass
2.	Confirm proper handling of invalid image formats	Application should reject invalid formats and provide feedback	Invalid formats were correctly rejected with feedback	Pass

Figure-6.2 Performance testing test cases table

CHAPTER - 7

RESULTS

- After running the program we can see the following results

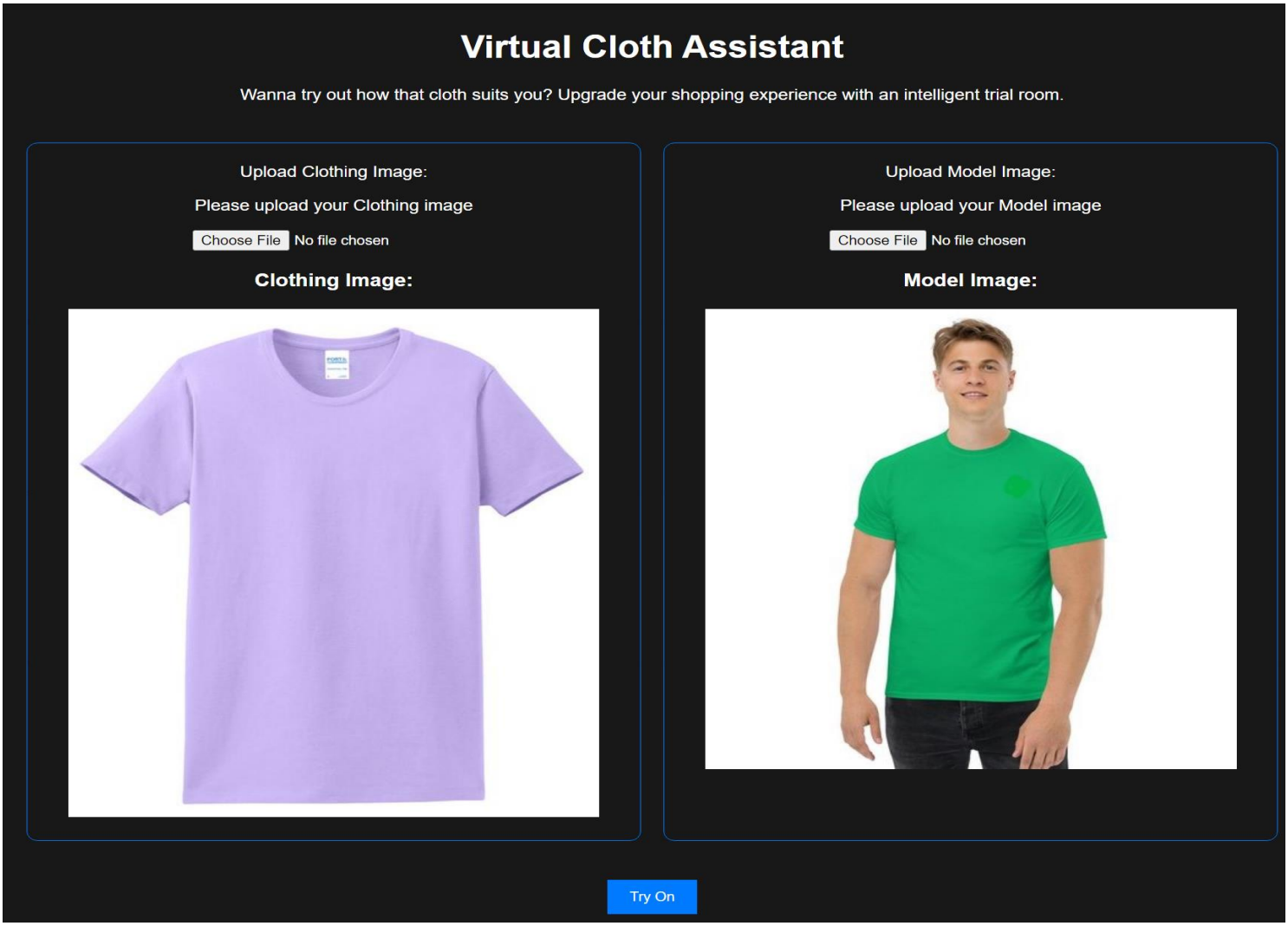


Figure-7.1 Input image

Figure-7.2 Result image

CHAPTER 8

8.1 CONCLUSION

The Virtual Apparel Try-On project exemplifies the transformative impact of AI technology in the fashion sector. By allowing users to seamlessly combine clothing images with model images, this application offers an innovative solution for virtual fitting. Utilizing OpenCV for sophisticated image processing, the project effectively isolates clothing from backgrounds and resizes it to fit the model accurately. The intuitive web interface facilitates easy image uploads, providing users with instant visual feedback alongside their original images. Additionally, the application's strong error management capabilities ensure a smooth user experience, promptly addressing any issues that arise. Overall, this project highlights the potential of advanced image processing techniques to enhance online shopping experiences in the fashion industry.

8.2 FUTURE SCOPE

While the current iteration of the Virtual Apparel Try-On project is impressive, there are numerous avenues for future enhancement and expansion:

1. Realistic Fabric Dynamics: Future developments could include algorithms that simulate the behavior of different fabrics, allowing for a more authentic representation of how clothing moves and drapes on the body.
2. Augmented Reality Features: Incorporating augmented reality (AR) functionalities would enable users to visualize clothing on themselves in real-time through their mobile devices, creating a more engaging and interactive shopping experience.
3. Tailored Recommendations: Implementing machine learning techniques to analyze user preferences and body types could lead to personalized clothing suggestions, enhancing the shopping experience by helping users find items that best suit their style and fit.
4. Broader Clothing Range: Expanding the application to include various clothing types, such as footwear,

accessories, and complete outfits, would increase its versatility and attractiveness to a wider audience.

5. Mobile Application Development: Developing a dedicated mobile application for the Virtual Apparel Try-On project could enhance accessibility, allowing users to try on clothes conveniently from their smartphones.

By pursuing these enhancements, the Virtual Apparel Try-On project can evolve into a more comprehensive and sophisticated tool for online fashion retail

REFERENCES

- S. Hauswiesner, M. Straka and G. Reitmayr, "Virtual Try-On through Image-Based Rendering" vol. 19, no. 9, Sept. 2013.
- F. Yu et al., "VTON-MP: Multi-Pose Virtual Try-On via Appearance Flow and Feature Filtering" , vol. 69, no. 4, Nov. 2023.
- M. Cao et al., "Educational Virtual-Wear Trial: More Than a Virtual Try-On Experience" , vol. 35, no. 6 Nov.-Dec. 2015.