Fashion Try-out

Virtual Fitting Room

Application Development Report Submitted   
In partial fulfillment of the requirements for the award of the degree of

**Bachelor of Technology in**

**Computer Science and Engineering**

**Ravipudi Sai Susritha**  **22N31A05K7**

**Pochammala Navya Sri**  **22N31A05H7**

**Rayakanti Sai Vamshi**  **22N31A05K8**

Under the esteemed guidance of

**Mr. M. SANDEEP**

**Associate Professor**



**Department of Computer Science and Engineering**

**Malla Reddy College of Engineering & Technology**

(Autonomous Institution- UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE,NBA &NAAC with A Grade)

Maisammaguda, Kompally, Dhulapally, Secunderabad -500100 website: [www.mrcet.ac.in](http://www.mrcet.ac.in/)

2024-2025



**Malla Reddy College of Engineering & Technology**

(Autonomous Institution- UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA &NAAC with ‘A’ Grade)

Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100 website: [www.mrcet.ac.in](http://www.mrcet.ac.in/)

**CERTIFICATE**

This is to certify that this is the bonafide record of the Application development-2 entitled **“FASHION TRY-OUT”** submitted by **Ravipudi Sai Susritha (22N31A05K7), Pochammala Navya Sri (22N31A05H7) and Rayakanti Sai Vamshi (22N31A05K8)** of B.Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering, Department of CSE during the year 2024-2025.

**Internal Guide** **Head of the Department**

**Mr. M. Sandeep** **Dr. S. Shanthi**

**Associate Professor**  **Professor**

**External Examiner**

**DECLARATION**

## We hereby declare that the Application Development “Fashion Try-out” submitted to Malla Reddy College of Engineering and Technology (UGC Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a result of original research carried-out in this thesis. It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

**Ravipudi Sai Susritha - (22N31A05K7)**

**Pochammala Navya Sri - (22N31A05H7)**

**Rayakanti Sai Vamshi - (22N31A05K8)**

**ACKNOWLEDGEMENT**

We feel honored to place our warm salutation to our college Malla Reddy College of Engineering and Technology (UGC-Autonomous) for giving us opportunity to do this Application as part of our B. Tech Program. We are ever grateful to our Director **Dr. V.S.K Reddy** and Principal **Dr. S. Srinivasa Rao** who enabled us to have experience in engineering and gain profound technical knowledge.

We express our heartiest thanks to our HOD**, Dr. S. Shanthi** for encouraging us in every aspect of our course and helping us realize our full potential,

We would like to thank our Project Guide **Mr. M. Sandeep** for his regular guidance, suggestions, and constant encouragement. We are extremely grateful to our Project Coordinator **Ms. D.Sai Eswari** for her continuous monitoring and unflinching co-operation throughout project work.

We would also like to thank all the faculty members and supporting staff of the Department of CSE and all other departments who have been helpful directly or indirectly in making our project a success

We are extremely grateful to our parents for their blessings and prayers for the completion of our project that gave us strength to do our project.

**With regards and gratitude**

**Ravipudi Sai Susritha - 22N31A05K7**

**Pochammala Navya Sri - 22N31A05H7**

**Rayakanti Sai Vamshi- 22N31A05K8**

## ABSTRACT

Fashion Try-out is a mobile application designed to elevate the online shopping experience by enabling users to visualize how clothing items will look on them through realistic 3D simulations. Users can upload images of clothes they are interested in, and the app provides a detailed, interactive fit analysis by overlaying these garments onto their own images. The app utilizes advanced 3D rendering technology and precise body measurement algorithms to offer an accurate representation of how different clothes will fit and appear on each user. This allows users to see how well items will fit, identify whether they need size adjustments, and assess the overall look of various garments from multiple angles.

Fashion Try-out empowers users to make more informed purchasing decisions by providing a virtual fitting experience that mirrors real-life trials. By addressing common online shopping challenges related to fit and style, the app aims to enhance customer satisfaction, reduce return rates, and bridge the gap between virtual and physical fashion experience.

**Keywords**: Mobile Application, Virtual Fitting, Realistic 3D Simulations, Body Measurement Algorithms.

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PG.NO** |
|  |  |  |
| **1** | **INTRODUCTION** | **1** |
|  | 1.1 PURPOSE AND OBJECTIVES | **1** |
|  | 1.2 EXISTING AND PROPOSED SYSTEM | **2** |
|  | 1.3 SCOPE OF APPLICATION | **5** |
| **2** | **LITERATURE SURVEY** | **6** |
| **3** | **SYSTEM ANALYSIS** | **8** |
|  | 3.1 HARDWARE &SOFTWARE REQUIREMENTS | **8** |
| **4** | **SYSTEM DESIGN** | **9** |
|  | 4.2 ARCHITECTURE | **9** |
|  | 4.3 UML DIAGRAMS | **10** |
| **5** | **METHODOLOGY** | **14** |
|  | 5.1 TECHNOLOGIES USED | **14** |
|  | 5.2 MODULES DESCRIPTION | **16** |
| **6** | **IMPLEMENTATION** | **18** |
|  | 6.1 SAMPLE CODE | **18** |
|  | 6.2 OUTPUT SCREENS | **40** |
| **7** | **CONCLUSION & FUTURE SCOPE** | **44** |
| **8** | **BIBILOGRAPHY** | **45** |

**TABLE OF CONTENTS**

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE.NO** | **NAME** | **PG.NO** |
| **4.1.1** | **ARCHITECTURE DIAGRAM** | **11** |
| **4.2.1** | **CLASS DIAGRAM** | **12** |
| **4.2.2** | **USE CASE DIAGRAM** | **13** |
| **4.2.3** | **ACTIVITY DIAGRAM** | **14** |
| **4.2.4** | **SEQUENCE DIAGRAM** | **15** |
| **5.1.1** | **ARCHITECTURE OF PYTHON** | **16** |

**CHAPTER 1**

**INTRODUCTION**

Fashion Try-out is a mobile application designed to enhance the online shopping experience by allowing users to visualize how clothing items will look on them through realistic 3D simulations. Users can upload images of garments they are interested in, and the app provides a detailed, interactive fit analysis by seamlessly overlaying these items onto their own photos. Utilizing advanced 3D rendering technology and precise body measurement algorithms, the app offers an accurate representation of how different clothing will fit and appear on each user.

This functionality enables users to evaluate fit, identify necessary size adjustments, and assess the overall style of various garments from multiple angles. By empowering users to make more informed purchasing decisions, Fashion Try-out delivers a virtual fitting experience that closely mirrors real-life trials. By addressing common online shopping challenges related to fit and style, the app aims to enhance customer satisfaction, reduce return rates, and bridge the gap between the virtual and physical fashion experience.

**1.1PURPOSE AND OBJECTIVES**

|  |
| --- |
| Online shopping grew up very fast. People are getting more used to online shopping, online auctions, etc., to purchase their interested products. However, an issue for buying clothes online is that client cannot try the product before he/she gets that product. The feeling after dressing on affects the client decision about buying the clothes.Therefore, there is an increasing demand to develop virtual dressing room to simulate the visualization of dressing.  The objectives of this project are:   1. Realistic 3D Simulations: Provide detailed, lifelike 3D representations of how clothing looks on users. 2. Accurate Size Recommendations: Offer precise size suggestions based on advanced fitting algorithms. 3. True-to-Life Colour Accuracy: Ensure accurate color representation of garments in virtual simulations. 4. Quick Loading Times: Deliver fast and efficient loading speeds for a smooth user experience. 5. User Uploaded Clothing: Allow users to upload their own clothing images for virtual fitting. 6. Broad Compatibility: Support a wide range of devices and operating systems for seamless app access. 7. Style Recommendations: Provide personalized style suggestions based on user preferences and fit analysis   **1.2 EXISTING AND PROPOSED SYSTEM** **Existing System Limitations**  1. Static Views: Offers limited interaction, making it hard for users to visualize how clothing looks from different angles. 2. Limited Realism: Lack of realistic representations can affect customer confidence and satisfaction. 3. No Fit Analysis: Users can’t get personalized size recommendations, leading to potential returns. 4. Compatibility Limitations: May only work on certain devices or browsers, restricting access. 5. Inaccurate Colour Representation: Colors may look different in person, leading to disappointment upon delivery. 6. Slow Loading Times: Can frustrate users and drive them away. 7. No User Uploaded Clothing: Users can’t showcase their own style or integrate their wardrobe, limiting personalization.  **Proposed System Enhancements**  1. Realistic 3D Simulations: Users can interact with clothing in a 3D environment, viewing it from multiple angles and in different poses. 2. Accurate Size Recommendation**s**: Advanced algorithms provide tailored size suggestions based on user measurements or previous purchases, improving fit and satisfaction. 3. True-to-life Colour Accuracy: Improved color representation ensures users see exactly what they’ll receive. 4. Quick Loading Times: Optimized performance for faster access, enhancing user experience. 5. User Uploaded Clothing: Allows users to add their own items, creating a more personalized shopping experience. 6. Broad Compatibility: Designed to work seamlessly across a wide range of devices and platforms, increasing accessibility. 7. Style Recommendations: AI-driven suggestions based on user preferences and trends help users discover new items that match their style. |

**1.3 SCOPE OF THE PROJECT**

The scope of the Fashion Try-out project involves creating a mobile application that offers users a realistic virtual fitting experience through advanced 3D simulations. This includes developing the app for iOS and Android platforms, integrating sophisticated 3D rendering technology and body measurement algorithms to accurately display how clothing items will fit and appear on users' images. Key features will include user and garment image uploads, detailed fit analysis, and interactive visualizations that allow users to view garments from multiple angles. The app will also integrate with online retail platforms to provide a seamless shopping experience, and include user profile management and customization options for personalized fitting experiences.

In addition to development, the project encompasses rigorous testing and quality assurance to ensure high performance and accuracy across various devices. Ongoing maintenance will be crucial to address bugs, introduce new features, and enhance the app based on user feedback. The scope also covers marketing strategies to attract users, data privacy measures to protect personal information, and support systems to assist users with technical issues. Overall, the project aims to revolutionize online shopping by bridging the gap between virtual and physical fashion trials, enhancing customer satisfaction, and reducing return rates through a comprehensive and user-centric virtual fitting solution.

**CHAPTER 2**

**LITERATURE SURVEY**

[Sen He](https://paperswithcode.com/author/sen-he), [Yi-Zhe Song](https://paperswithcode.com/author/yi-zhe-song), [Tao Xiang](https://paperswithcode.com/author/tao-xiang) et al. [1] presented Image-based virtual try-on aims to fit an in-shop garment into a clothed person image. To achieve this, a key step is garment warping which spatially aligns the target garment with the corresponding body parts in the person image. Prior methods typically adopt a local appearance flow estimation model. They are thus intrinsically susceptible to difficult body poses/occlusions and large mis-alignments between person and garment images. To overcome this limitation, a novel global appearance flow estimation model is proposed in this work. For the first time, a StyleGAN based architecture is adopted for appearance flow estimation. This enables us to take advantage of a global style vector to encode a whole-image context to cope with the forementioned challenges. To guide the StyleGAN flow generator to pay more attention to local garment deformation, a flow refinement module is introduced to add local context. Experiment results on a popular virtual try-on benchmark show that our method achieves new state-of-the-art performance. It is particularly effective application scenario where the reference image is full-body resulting in a large mis-alignment with the garment image.

[Xintong Han](https://paperswithcode.com/author/xintong-han), [Xiaojun Hu](https://paperswithcode.com/author/xiaojun-hu-1), [Weilin Huang](https://paperswithcode.com/author/weilin-huang-1), [Matthew R. Scott](https://paperswithcode.com/author/matthew-r-scott-1) et al. [2] presented ClothFlow, an appearance-flow-based generative model to synthesize clothed person for posed-guided person image generation and virtual try-on. By estimating a dense flow between source and target clothing regions, ClothFlow effectively models the geometric changes and naturally transfers the appearance to synthesize novel images as shown in Figure 1. We achieve this with a three-stage framework: 1) Conditioned on a target pose, we first estimate a person semantic layout to provide richer guidance to the generation process. 2) Built on two feature pyramid networks, a cascaded flow estimation network then accurately estimates the appearance matching between corresponding clothing regions. The resulting dense flow warps the source image to flexibly account for deformations. 3) Finally, a generative network takes the warped clothing regions as inputs and renders the target view. We conduct extensive experiments on the DeepFashion dataset for pose-guided person image generation and on the VITON dataset for the virtual try-on task. Strong qualitative and quantitative results validate the effectiveness of our method.

[Haoye Dong](https://paperswithcode.com/author/haoye-dong), [Xiaodan Liang](https://paperswithcode.com/author/xiaodan-liang), [Bochao Wang](https://paperswithcode.com/author/bochao-wang), [Hanjiang Lai](https://paperswithcode.com/author/hanjiang-lai), [Jia Zhu](https://paperswithcode.com/author/jia-zhu), [Jian Yin](https://paperswithcode.com/author/jian-yin) et al. [3] presented Virtual try-on system under arbitrary human poses has huge application potential, yet raises quite a lot of challenges, e.g. self-occlusions, heavy misalignment among diverse poses, and diverse clothes textures. Existing methods aim at fitting new clothes into a person can only transfer clothes on the fixed human pose, but still show unsatisfactory performances which often fail to preserve the identity, lose the texture details, and decrease the diversity of poses. In this paper, we make the first attempt towards multi-pose guided virtual try-on system, which enables transfer clothes on a person image under diverse poses. Given an input person image, a desired clothes image, and a desired pose, the proposed Multi-pose Guided Virtual Try-on Network (MG-VTON) can generate a new person image after fitting the desired clothes into the input image and manipulating human poses. Our MG-VTON is constructed in three stages: 1) a desired human parsing map of the target image is synthesized to match both the desired pose and the desired clothes shape; 2) a deep Warping Generative Adversarial Network (Warp-GAN) warps the desired clothes appearance into the synthesized human parsing map and alleviates the misalignment problem between the input human pose and desired human pose; 3) a refinement render utilizing multi-pose composition masks recovers the texture details of clothes and removes some artifacts. Extensive experiments on well-known datasets and our newly collected largest virtual try-on benchmark demonstrate that our MG-VTON significantly outperforms all state-of-the-art methods both qualitatively and quantitatively with promising multi-pose virtual try-on performances.

[Amit Raj](https://paperswithcode.com/author/amit-raj), [Patsorn Sangkloy](https://paperswithcode.com/author/patsorn-sangkloy), [Huiwen Chang](https://paperswithcode.com/author/huiwen-chang), [Jingwan Lu](https://paperswithcode.com/author/jingwan-lu), [Duygu Ceylan](https://paperswithcode.com/author/duygu-ceylan), [James Hays](https://paperswithcode.com/author/james-hays) et al. [4] presented Swap Net, a framework to transfer garments across images of people with arbitrary body pose, shape, and clothing. Garment transfer is a challenging task that requires (i) disentangling the features of the clothing from the body pose and shape and (ii) realistic synthesis of the garment texture on the new body. We present a neural network architecture that tackles these sub-problems with two task-specific sub-networks. Since acquiring pairs of images showing the same clothing on different bodies is difficult, we propose a novel weakly-supervised approach that generates training pairs from a single image via data augmentation. We present the first fully automatic method for garment transfer in unconstrained images without solving the difficult 3D reconstruction problem. We demonstrate a variety of transfer results and highlight our advantages over traditional image-to-image and analogy pipelines.

[Yuhao Xu](https://paperswithcode.com/author/yuhao-xu), [Tao Gu](https://paperswithcode.com/author/tao-gu), [Weifeng Chen](https://paperswithcode.com/author/weifeng-chen), [Chengcai Chen](https://paperswithcode.com/author/chengcai-chen)  et al. [5] presented OOTDiffusion, a novel network architecture for realistic and controllable image-based virtual try-on (VTON). We leverage the power of pretrained latent diffusion models, designing an outfitting UNet to learn the garment detail features. Without a redundant warping process, the garment features are precisely aligned with the target human body via the proposed outfitting fusion in the self-attention layers of the denoising UNet. In order to further enhance the controllability, we introduce outfitting dropout to the training process, which enables us to adjust the strength of the garment features through classifier-free guidance. Our comprehensive experiments on the VITON-HD and Dress Code datasets demonstrate that OOTDiffusion efficiently generates high-quality try-on results for arbitrary human and garment images, which outperforms other VTON methods in both realism and controllability, indicating an impressive breakthrough in virtual try-on. Our source code is available at https://github.com/levihsu/OOTDiffusion.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 HARDWARE AND SOFTWARE REQUIREMENTS**

System requirements are the functionality that is needed by a system in order to satisfy the customer's requirements. System requirements are abroad and a narrow subject that could be implemented to many items. The requirements document allows the project team to have a clear picture of what the software solution must do before selecting a vendor.

Hardware:

User Device

* Minimum RAM: 3 GB
* Minimum Storage: 200 MB , additional storage for user-uploaded images
* Minimum Processor: Quad-core CPU
* Camera: Front-facing camera

Development

* GPU
* RAM: Minimum 8 GB, recommended 16 GB for optimal performance

Software:

* Python
* Numpy
* Tensorflow-gpu
* Opencv-python
* MongoDB
* F
* Mobile OS:

Ios (13.0 or later)

Android (8.0 or later)

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**

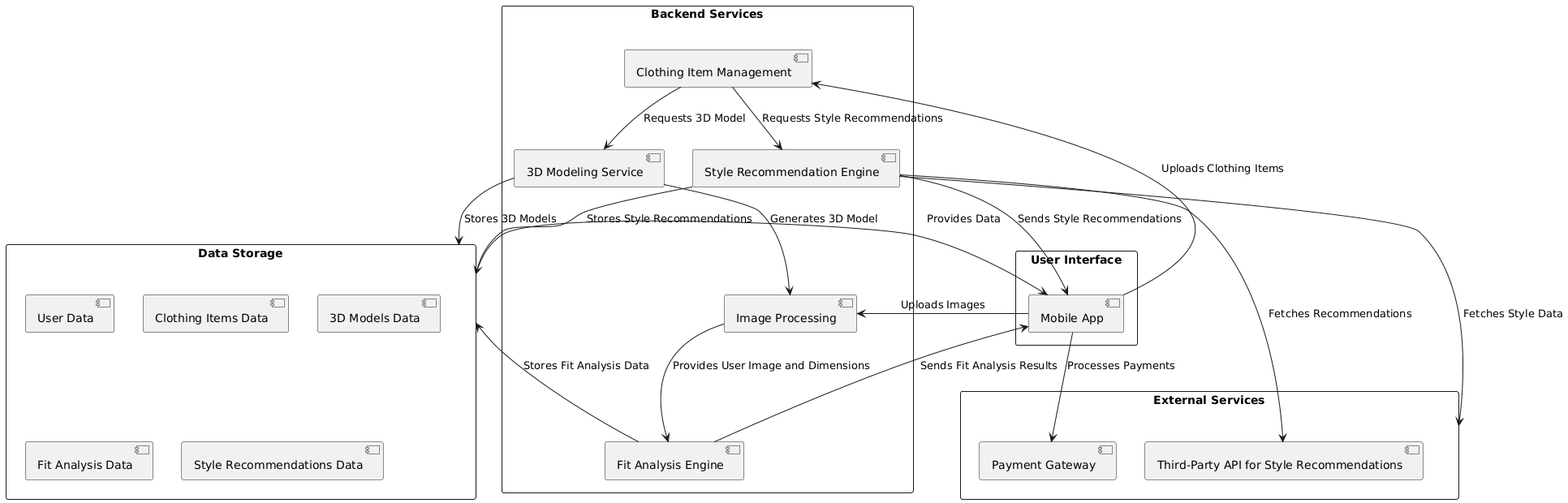
System requirements are the functionality that is needed by a system in order to satisfy the customer's requirements. System requirements are abroad and a narrow subject that could be implemented to many items. The requirements document allows the project team to have a clear picture of what the software solution must do before selecting a vendor. Without an optimized set of future state requirements, the project team has no effective basis to choose the best system for your organization.

#### 1. Client-Side (Frontend)

* User Interface (UI):
  + Responsive Design: A mobile-first approach to ensure compatibility across devices.
  + 3D Visualization Module: Uses WebGL or similar technologies for realistic clothing simulations.
  + User Profile Management: Allows users to create accounts, manage preferences, and upload personal clothing.
* Interaction Layer:
  + Fit Analysis Tool: Integrates with body measurement input to recommend sizes.
  + Style Recommendation Engine: Suggests outfits based on user preferences and browsing history.

#### 2. Server-Side (Backend)

* Application Server:
  + RESTful API: Handles requests from the frontend for data retrieval, user management, and clothing uploads.
  + Authentication Service: Manages user accounts and secure logins (OAuth, JWT).
* Database:
  + User Database: Stores user profiles, preferences, and uploaded clothing.
  + Clothing Inventory Database: Manages details about clothing items, including sizes, colors, and images.
  + Recommendation Database: Stores data used for generating style recommendations.
* 3D Asset Management:
  + Asset Storage: Stores 3D models and textures for clothing, ensuring quick access.
  + Rendering Engine: Processes 3D models to provide realistic visualizations.

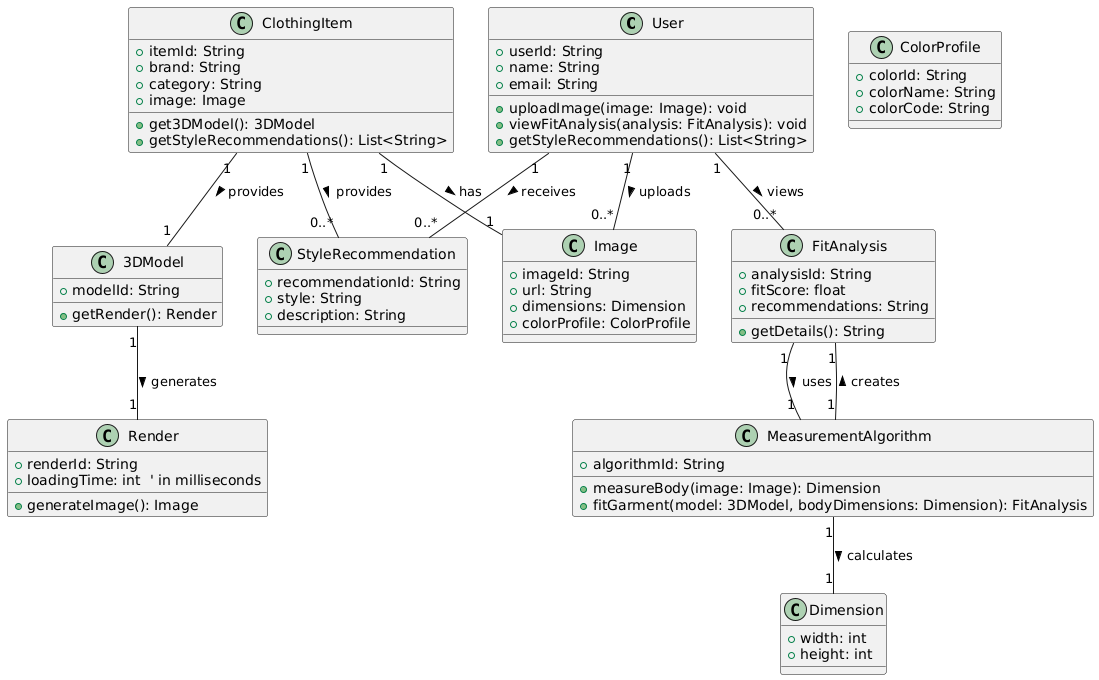


**Fig: 4.1.1 Architecture Diagram**

**4.2 UML DIAGRAMS**

**Class Diagram**

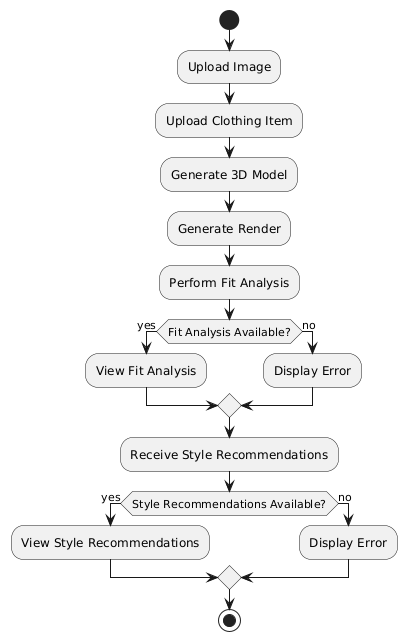
In software engineering, a class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and the relationships among objects. It is a representation of an object in many ways, it is simply a templet from which objects are created. Classes form the main building blocks of an object-oriented application. Although thousands of students attend the university, you would only model one class, called student, which would represent the entire collection of students. Classes are linked together to generate class diagrams in a complex system with numerous related classes. Various sorts of arrows represent different relationships between classes.



**Fig: 4.2.1 Class Diagram**

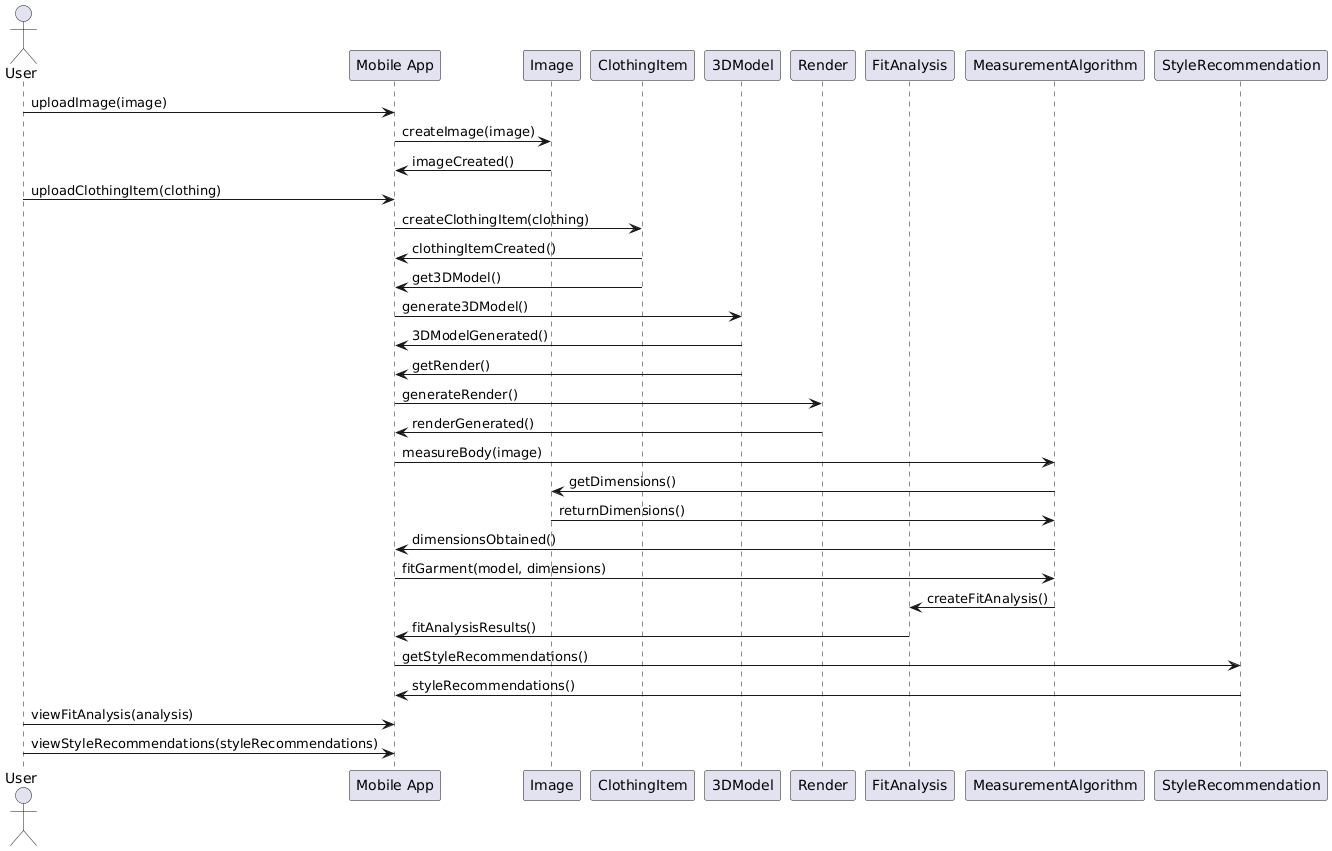
**Activity Diagram**

An activity diagram is a type of Unified Modeling Language (UML) flowchart that shows the flow from one activity to another in a system or process. It's used to describe the different dynamic aspects of a system and is referred to as a 'behavior diagram' because it describes what should happen in the modeled system. It represents the business and operational workflows of a system. It is a dynamic diagram that shows the activity and event that causes the object to be in the particular state.



**Fig: 4.2.2 Activity Diagram**

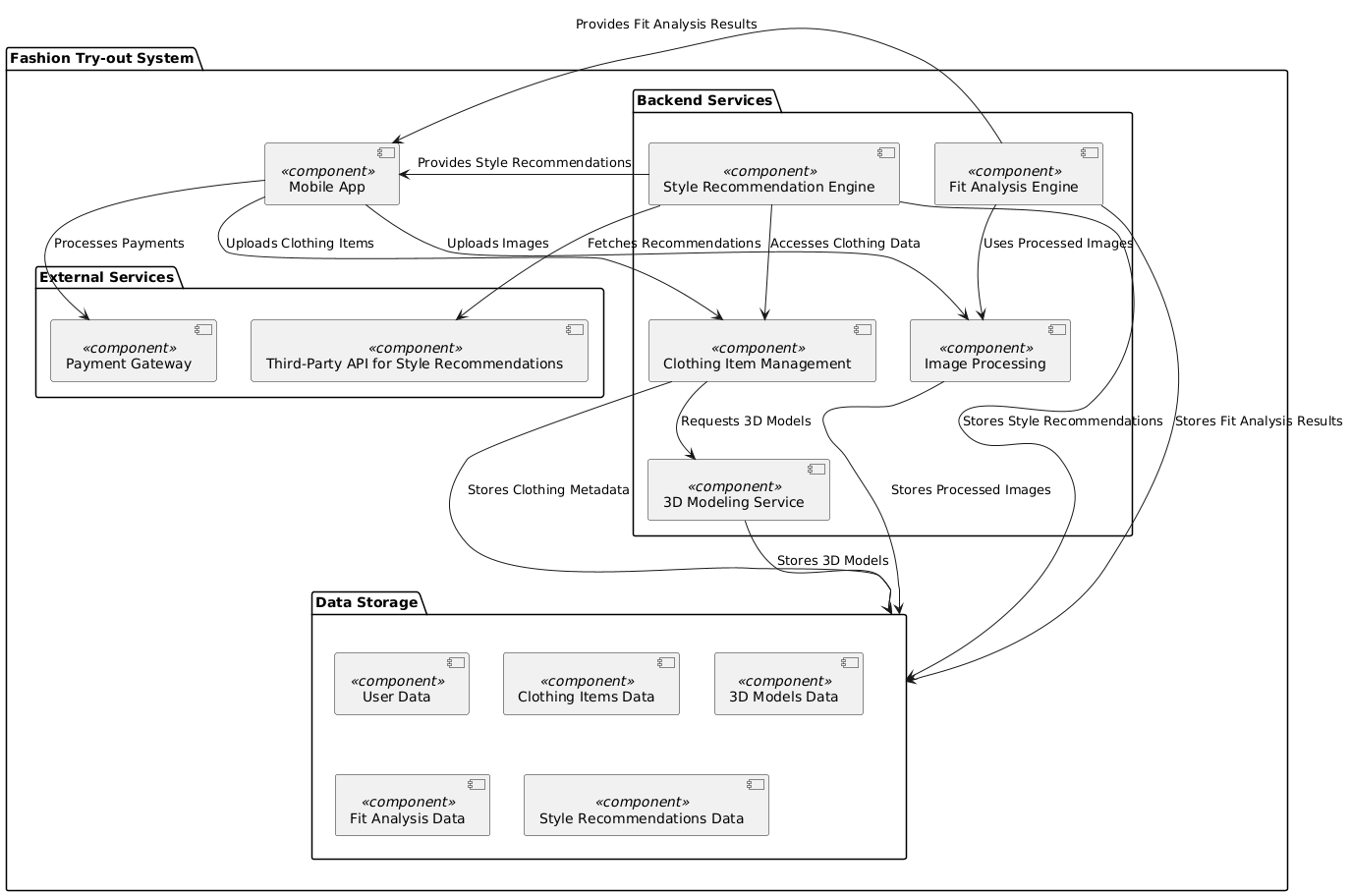
**Sequence Diagram**

A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines and the messages that they exchange over time during the interaction. A sequence diagram shows the sequence of messages passed between objects. Sequence diagrams can also show the control structures between objects. It is used to represent the flow of messages, events and actions between the objects or components of the system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

**Fig: 4.2.3 Sequence Diagram**

**Component Diagram**

UML Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering.



**Fig: 4.2.4 Component Diagram**

**CHAPTER 5**

**METHODOLOGY**

**5.1 TECHNOLOGIES USED**

**Python**

Python serves as a versatile programming language ideal for backend development and data manipulation. In this application, Python manages server-side logic, handles data interactions, and integrates machine learning models. Its extensive libraries and frameworks simplify tasks such as user authentication, data processing, and API development. With Python’s clear syntax, developers can easily maintain and update complex functionalities, ensuring that the application can evolve with user needs and incorporate new features efficiently.

**NumPy**

NumPy is a powerful library for numerical computing in Python. In this application, NumPy facilitates the handling of large arrays and matrices, which is crucial for image processing and data analysis. It enables efficient mathematical operations, making it easier to manipulate pixel values in images. For instance, when preparing input data for machine learning models, NumPy helps standardize and normalize image data, ensuring that the models perform optimally. This capability is essential for providing accurate virtual try-on results that align with user expectations.

**TensorFlow-GPU**

TensorFlow-GPU is an open-source machine learning framework that leverages GPU acceleration. This technology is pivotal for building and training deep learning models, especially convolutional neural networks (CNNs) that excel in image recognition tasks. In this application, TensorFlow-GPU powers algorithms that analyze user images to determine how garments will fit and look on them. By utilizing GPU acceleration, the application can process images rapidly, delivering real-time feedback to users as they try on different outfits virtually, thus enhancing their overall experience.

### **OpenCV-Python**

### OpenCV-Python is a comprehensive computer vision library. In the context of this application, OpenCV is essential for various image processing tasks, such as detecting body features, extracting garment contours, and overlaying clothing on user images. It provides tools for face detection, edge detection, and feature extraction, which are crucial for accurately fitting clothes on users' images. This allows for a seamless virtual try-on experience, where clothing is realistically rendered on the user's body, ensuring that the application feels intuitive and engaging.

### **MongoDB**

### MongoDB is a NoSQL database designed for storing unstructured data. For this application, MongoDB efficiently stores user profiles, garment details, and images in a way that allows for flexible and scalable data management. Its document-oriented structure means that data can be stored in a more natural format, making it easier to accommodate varying garment attributes and user preferences. This flexibility is vital as the application grows, allowing for the addition of new features without disrupting existing data structures or user experiences.

**5.2 MODULES DESCRIPTION**

**User Interface (UI) Module**

The User Interface (UI) Module of the Fashion Try-out application is designed to provide an intuitive and engaging user experience. It includes navigation elements, buttons, and layout designs that facilitate easy access to various app features. Users can seamlessly navigate through an interactive main menu, upload garments, and customize their profile settings, all within a visually appealing interface.

The Fashion Try-out application offers a rich set of features that work together to deliver a seamless and engaging experience for users interested in virtual fashion fitting. At the heart of the application is the User Interface (UI) Module, which is designed to provide an intuitive and visually appealing experience. This module focuses on simplicity, ensuring that users can easily navigate through the app. It includes an interactive main menu that grants access to key features such as garment uploading, profile settings, and the virtual fitting room. The interface is customizable, adapting to user preferences and previous interactions, while its modern aesthetic with carefully selected color schemes and typography enhances visual appeal.

**Image Processing Module**

The Image Processing Module is responsible for handling user-uploaded images. It processes and prepares these photos for analysis by applying image enhancement techniques to ensure high-quality rendering. This module includes features like image cropping, resizing tools, background removal functionality, and support for various image formats such as JPEG and PNG, ensuring that users can easily upload their photos.

The Image Processing Module handles user-uploaded images, applying enhancement techniques like brightness adjustment and contrast sharpening to ensure high-quality output. It includes cropping and resizing tools that let users adjust their images for better rendering results. One standout feature is background removal, which isolates the user and places emphasis on the garment, resulting in a more realistic try-on experience. This module supports popular image formats like JPEG and PNG, ensuring compatibility across devices.

**3D Rendering Module**

Central to the app's functionality, the 3D Rendering Module utilizes advanced rendering technology to generate realistic visualizations of clothing items on user-uploaded images. It allows users to view garments from multiple angles, offering rotation and zoom capabilities for detailed inspection. With high-fidelity rendering and environment simulation for lighting and shadows, users can enjoy a lifelike virtual fitting experience.

Central to the virtual try-on experience is the 3D Rendering Module, which provides highly realistic representations of garments. This module uses advanced algorithms to create detailed, photorealistic renderings, showcasing fabric texture, folds, and stitching. Users can rotate the avatar or garment for a 360-degree view, inspect details up close with zoom functionality, and enjoy realistic lighting and shadow effects that make the garments appear lifelike.

**Fit Analysis Module**

The Fit Analysis Module employs precise body measurement algorithms to evaluate how clothing fits the user. By extracting body measurements from user images, this module provides personalized fit recommendations and identifies any necessary size adjustments. It features a fit scoring system based on garment dimensions, giving feedback on fit comfort and mobility, which helps users make informed choices.

The Fit Analysis Module complements this by offering personalized recommendations based on the user’s body measurements. Through sophisticated algorithms, this module extracts measurements from images and assigns a fit score to each garment, considering comfort and mobility. It suggests specific sizes and, when needed, provides recommendations for adjustments or alterations.

**Garment Database Module**

A comprehensive repository is maintained by the Garment Database Module, which allows users to explore a wide range of clothing items. This module provides detailed information about each garment, including images and descriptions, while also offering filter and search options for easy navigation. Users can benefit from ratings and reviews, as well as integration with external retailers for live inventory updates.

**User Profile Module**

The **User Profile Module** manages individual accounts and preferences, storing vital information such as measurements, upload history, and personalized recommendations. Users can create secure accounts, customize their settings according to size preferences and style interests, and access a history of previously uploaded garments, enhancing their overall experience.

**Analytics and Feedback Module**

To improve the application continually, the Analytics and Feedback Module focuses on gathering user feedback and app performance data. It analyzes usage patterns to enhance overall functionality and user satisfaction. This module features a user feedback submission interface, performance tracking of fit accuracy, and data analysis tools that help identify trends, ensuring the app evolves according to user needs.

**Integration Module**

Finally, the Integration Module facilitates connections with third-party services, including social media sharing, payment gateways, and retail partners. Users can easily share their outfit visuals on social platforms, engage in secure payment processing, and receive real-time inventory and shipping updates through API connections. This module ensures that Fashion Try-out remains connected and functional in a fast-paced digital shopping environment.

**CHAPTER 6**

**IMPLEMENTATION**

**6.1 SAMPLE CODE**

from \_\_future\_\_ import absolute\_import

# from \_\_future\_\_ import division

from \_\_future\_\_ import print\_function

import tensorflow as tf

import numpy as np

# np.set\_printoptions(threshold=np.nan)

import scipy.io as sio

def tps\_stn(U, nx, ny, cp, out\_size, is\_points=False, points=None):

  """Thin Plate Spline Spatial Transformer Layer

  TPS control points are arranged in a regular grid.

  U : float Tensor

      shape [num\_batch, height, width, num\_channels].

  nx : int

      The number of control points on x-axis

  ny : int

      The number of control points on y-axis

  cp : float Tensor

      control points. shape [num\_batch, nx\*ny, 2].

  out\_size: tuple of two ints

      The size of the output of the network (height, width)

  is\_points: true if transform points instead of image.

  points: keypoints coordinates.

  ---------

  def \_repeat(x, n\_repeats):

    rep = tf.transpose(

        tf.expand\_dims(tf.ones(shape=tf.stack([n\_repeats, ])), 1), [1, 0])

    rep = tf.cast(rep, 'int32')

    x = tf.matmul(tf.reshape(x, (-1, 1)), rep)

    return tf.reshape(x, [-1])

  def \_interpolate(im, x, y, out\_size):

    # constants

    num\_batch = tf.shape(im)[0]

    height = tf.shape(im)[1]

    width = tf.shape(im)[2]

    chanels = tf.shape(im)[3]

    x = tf.cast(x, 'float32')

    y = tf.cast(y, 'float32')

    height\_f = tf.cast(height, 'float32')

    width\_f = tf.cast(width, 'float32')

    out\_height = out\_size[0]

    out\_width = out\_size[1]

    # clip coordinates to [-1, 1]

    x = tf.clip\_by\_value(x, -1, 1)

    y = tf.clip\_by\_value(y, -1, 1)

    # scale coordinates from [-1, 1] to [0, width/height-1]

    x = (x + 1) / 2 \* (width\_f - 1)

    y = (y + 1) / 2 \* (height\_f - 1)

    # do sampling

    x0\_f = tf.floor(x)

    y0\_f = tf.floor(y)

    x1\_f = x0\_f + 1

    y1\_f = y0\_f + 1

    x0 = tf.cast(x0\_f, 'int32')

    y0 = tf.cast(y0\_f, 'int32')

    x1 = tf.cast(tf.minimum(x1\_f, width\_f - 1), 'int32')

    y1 = tf.cast(tf.minimum(y1\_f, height\_f - 1), 'int32')

    dim2 = width

    dim1 = width\*height

    base = \_repeat(tf.range(num\_batch)\*dim1, out\_height\*out\_width)

    base\_y0 = base + y0\*dim2

    base\_y1 = base + y1\*dim2

    idx\_a = base\_y0 + x0

    idx\_b = base\_y1 + x0

    idx\_c = base\_y0 + x1

    idx\_d = base\_y1 + x1

    # use indices to lookup pixels in the flat image and restore

    # channels dim

    im\_flat = tf.reshape(im, tf.stack([-1, channels]))

    Ia = tf.gather(im\_flat, idx\_a)

    Ib = tf.gather(im\_flat, idx\_b)

    Ic = tf.gather(im\_flat, idx\_c)

    Id = tf.gather(im\_flat, idx\_d)

    # and finally calculate interpolated values

    wa = tf.expand\_dims(((x1\_f-x) \* (y1\_f-y)), 1)

    wb = tf.expand\_dims(((x1\_f-x) \* (y-y0\_f)), 1)

    wc = tf.expand\_dims(((x-x0\_f) \* (y1\_f-y)), 1)

    wd = tf.expand\_dims(((x-x0\_f) \* (y-y0\_f)), 1)

    output = tf.add\_n([wa\*Ia, wb\*Ib, wc\*Ic, wd\*Id])

    return output

  def \_meshgrid(height, width, fp):

    x\_t = tf.matmul(

        tf.ones(shape=tf.stack([height, 1])),

        tf.transpose(tf.expand\_dims(tf.linspace(-1.0, 1.0, width), 1), [1, 0]))

    y\_t = tf.matmul(

        tf.expand\_dims(tf.linspace(-1.0, 1.0, height), 1),

        tf.ones(shape=tf.stack([1, width])))

    x\_t\_flat = tf.reshape(x\_t, (1, -1))

    y\_t\_flat = tf.reshape(y\_t, (1, -1))

    x\_t\_flat\_b = tf.expand\_dims(x\_t\_flat, 0)  # [1, 1, h\*w]

    y\_t\_flat\_b = tf.expand\_dims(y\_t\_flat, 0)  # [1, 1, h\*w]

    num\_batch = tf.shape(fp)[0]

    px = tf.expand\_dims(fp[:, :, 0], 2)  # [n, nx\*ny, 1]

    py = tf.expand\_dims(fp[:, :, 1], 2)  # [n, nx\*ny, 1]

    d = tf.pow(x\_t\_flat\_b - px, 2.) + tf.pow(y\_t\_flat\_b - py, 2.)

    r = d \* tf.log(d + 1e-12)  # [n, nx\*ny, h\*w]

    x\_t\_flat\_g = tf.tile(x\_t\_flat\_b, tf.stack([num\_batch, 1, 1]))  # [n, 1, h\*w]

    y\_t\_flat\_g = tf.tile(y\_t\_flat\_b, tf.stack([num\_batch, 1, 1]))  # [n, 1, h\*w]

    ones = tf.ones\_like(x\_t\_flat\_g)  # [n, 1, h\*w]

    grid = tf.concat([ones, x\_t\_flat\_g, y\_t\_flat\_g, r], 1)  # [n, nx\*ny+3, h\*w]

    return grid

  def \_transform(T, fp, input\_dim, out\_size):

    num\_batch = input\_dim.get\_shape()[0]

    height = tf.shape(input\_dim)[1]

    width = tf.shape(input\_dim)[2]

    num\_channels = input\_dim.get\_shape()[3]

    # grid of (x\_t, y\_t, 1), eq (1) in ref [1]

    height\_f = tf.cast(height, 'float32')

    width\_f = tf.cast(width, 'float32')

    out\_height = out\_size[0]

    out\_width = out\_size[1]

    grid = \_meshgrid(out\_height, out\_width, fp)  # [2, h\*w]

    # transform A x (1, x\_t, y\_t, r1, r2, ..., rn) -> (x\_s, y\_s)

    T\_g = tf.matmul(T, grid)  # MARK

    x\_s = tf.slice(T\_g, [0, 0, 0], [-1, 1, -1])

    y\_s = tf.slice(T\_g, [0, 1, 0], [-1, 1, -1])

    x\_s\_flat = tf.reshape(x\_s, [-1])

    y\_s\_flat = tf.reshape(y\_s, [-1])

    input\_transformed = \_interpolate(

        input\_dim, x\_s\_flat, y\_s\_flat, out\_size)

    output = tf.reshape(

        input\_transformed,

        tf.stack([num\_batch, out\_height, out\_width, num\_channels]))

    return output

  def \_point\_transform(T, fp, points, out\_size):

    # grid of (x\_t, y\_t, 1), eq (1) in ref [1]

    height = out\_size[0]

    width = out\_size[1]

    # grid of (x\_t, y\_t, 1), eq (1) in ref [1]

    num\_batch = tf.shape(fp)[0]

    x\_t = points[:,::2] # [n, num\_points]

    y\_t = points[:,1::2] # [n, num\_points]

    x\_t\_b = tf.expand\_dims(x\_t, 1)  # [n, 1, num\_points]

    y\_t\_b = tf.expand\_dims(y\_t, 1)  # [n, 1, num\_points]

    px = tf.expand\_dims(fp[:, :, 0], 2)  # [n, nx\*ny, 1]

    py = tf.expand\_dims(fp[:, :, 1], 2)  # [n, nx\*ny, 1]

    d = tf.pow(x\_t\_b - px, 2.) + tf.pow(y\_t\_b - py, 2.) # [n, nx\*ny, num\_points]

    r = d \* tf.log(d + 1e-12)  # [n, nx\*ny, num\_points]

    ones = tf.ones\_like(x\_t\_b)  # [n, 1, num\_points]

    grid = tf.concat([ones, x\_t\_b, y\_t\_b, r], 1)  # [n, nx\*ny+3, num\_points]

    x\_t = tf.matmul(

        tf.ones(shape=tf.stack([height, 1])),

        tf.transpose(tf.expand\_dims(tf.linspace(-1.0, 1.0, width), 1), [1, 0]))

    y\_t = tf.matmul(

        tf.expand\_dims(tf.linspace(-1.0, 1.0, height), 1),

        tf.ones(shape=tf.stack([1, width]))

    x\_t\_flat = tf.reshape(x\_t, (1, -1))

    y\_t\_flat = tf.reshape(y\_t, (1, -1))

    x\_t\_flat\_b = tf.expand\_dims(x\_t\_flat, 0)  # [1, 1, h\*w]

    y\_t\_flat\_b = tf.expand\_dims(y\_t\_flat, 0)  # [1, 1, h\*w]

    num\_batch = tf.shape(fp)[0]

    px = tf.expand\_dims(fp[:, :, 0], 2)  # [n, nx\*ny, 1]

    py = tf.expand\_dims(fp[:, :, 1], 2)  # [n, nx\*ny, 1]

    d = tf.pow(x\_t\_flat\_b - px, 2.) + tf.pow(y\_t\_flat\_b - py, 2.)

    r = d \* tf.log(d + 1e-12)  # [n, nx\*ny, h\*w]

    x\_t\_flat\_g = tf.tile(x\_t\_flat\_b, tf.stack([num\_batch, 1, 1]))  # [n, 1, h\*w]

    y\_t\_flat\_g = tf.tile(y\_t\_flat\_b, tf.stack([num\_batch, 1, 1]))  # [n, 1, h\*w]

    ones = tf.ones\_like(x\_t\_flat\_g)  # [n, 1, h\*w]

    grid = tf.concat([ones, x\_t\_flat\_g, y\_t\_flat\_g, r], 1)  # [n, nx\*ny+3, h\*w]

    # transform A x (1, x\_t, y\_t, r1, r2, ..., rn) -> (x\_s, y\_s)

    T\_g = tf.matmul(T, grid)  # MARK

    x\_s = tf.slice(T\_g, [0, 0, 0], [-1, 1, -1])

    y\_s = tf.slice(T\_g, [0, 1, 0], [-1, 1, -1])

    T\_g = tf.concat([x\_s, y\_s], 1)

    T\_g = tf.transpose(T\_g, [0, 2, 1])

    T\_g = tf.reshape(T\_g, [num\_batch, -1])  # MARK

  def \_solve\_system(cp, nx, ny):

    gx = 2. / (nx - 1)  # grid x size

    gy = 2. / (ny - 1) # grid y size

    cx = -1. # x coordinate

    cy = -1. # y coordinate

    p\_ = np.empty([nx\*ny, 3], dtype='float32')

    i = 0

    for \_ in range(ny):

      for \_ in range(nx):

        p\_[i, :] = 1, cx, cy

        i += 1

        cx += gx

      cx = -1.

      cy += gy

    p\_1 = p\_.reshape([nx\*ny, 1, 3])

    p\_2 = p\_.reshape([1, nx\*ny, 3])

    d = np.sqrt(np.sum((p\_1-p\_2)\*\*2, 2))  # [nx\*ny, nx\*ny]

    r = d\*d\*np.log(d\*d+1e-12)

    W = np.zeros([nx\*ny+3, nx\*ny+3], dtype='float32')

    W[:nx\*ny, 3:] = r

    W[:nx\*ny, :3] = p\_

    W[nx\*ny:, 3:] = p\_.T

    num\_batch = tf.shape(cp)[0]

    fp = tf.constant(p\_[:, 1:], dtype='float32')  # [nx\*ny, 2]

    fp = tf.expand\_dims(fp, 0)  # [1, nx\*ny, 2]

    fp = tf.tile(fp, tf.stack([num\_batch, 1, 1]))  # [n, nx\*ny, 2]

    W\_inv = np.linalg.inv(W)

    W\_inv\_t = tf.constant(W\_inv, dtype='float32')  # [nx\*ny+3, nx\*ny+3]

    W\_inv\_t = tf.expand\_dims(W\_inv\_t, 0)          # [1, nx\*ny+3, nx\*ny+3]

    W\_inv\_t = tf.tile(W\_inv\_t, tf.stack([num\_batch, 1, 1]))

    cp\_pad = tf.pad(cp, [[0, 0], [0, 3], [0, 0]], "CONSTANT")

    T = tf.matmul(W\_inv\_t, cp\_pad)

    T = tf.transpose(T, [0, 2, 1])

    return T, fp

  T, fp = \_solve\_system(cp, nx, ny)

  if is\_points:

    output = \_point\_transform(T, fp, U, out\_size)

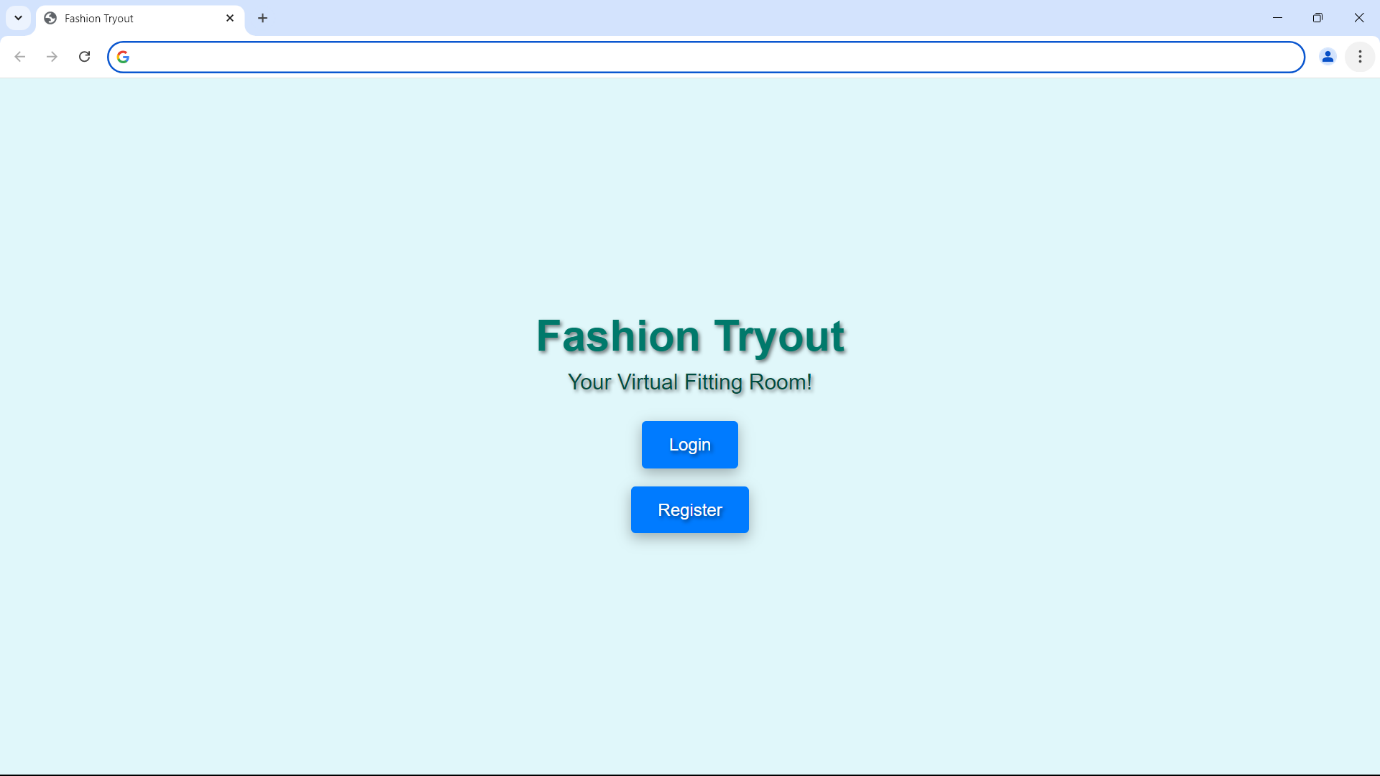
  else:

    output = \_transform(T, fp, U, out\_size)

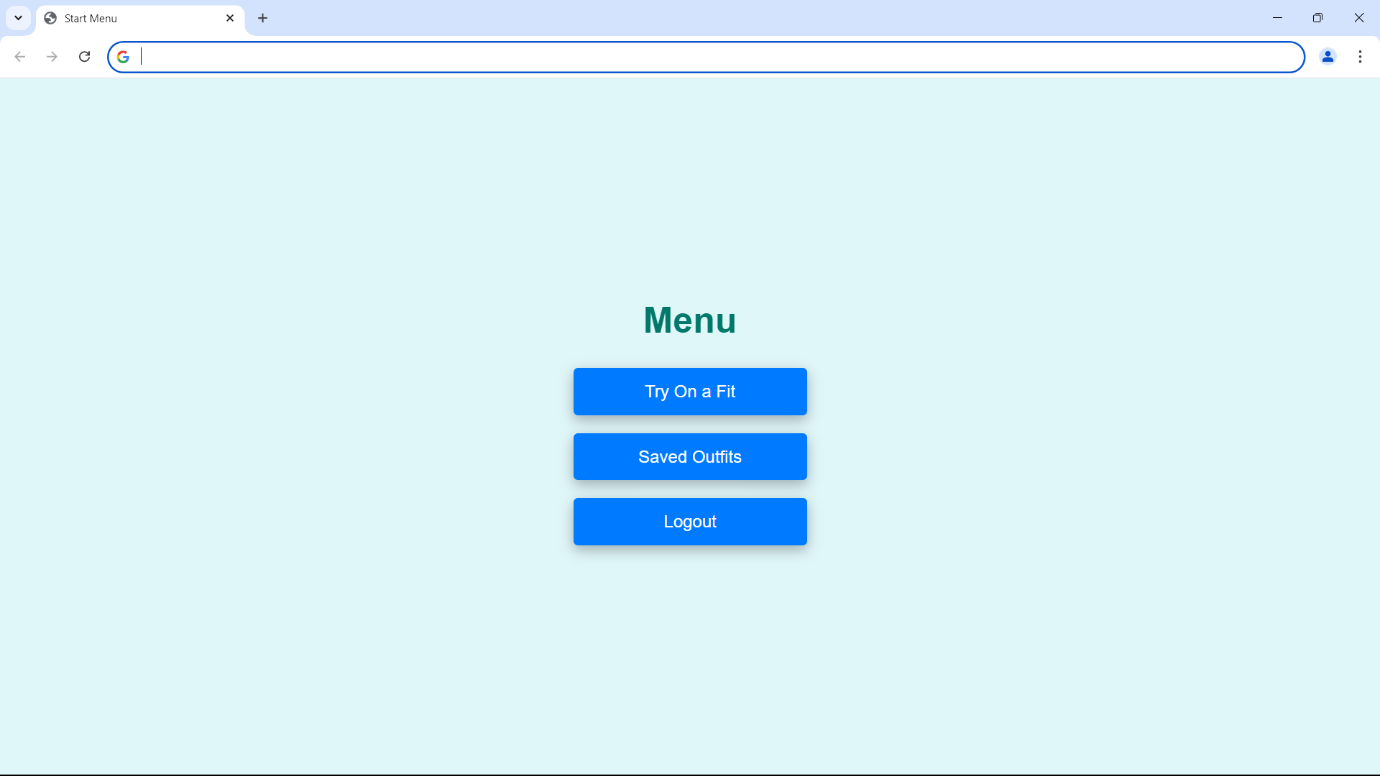
  return output

**6.2 OUTPUT SCREENS**

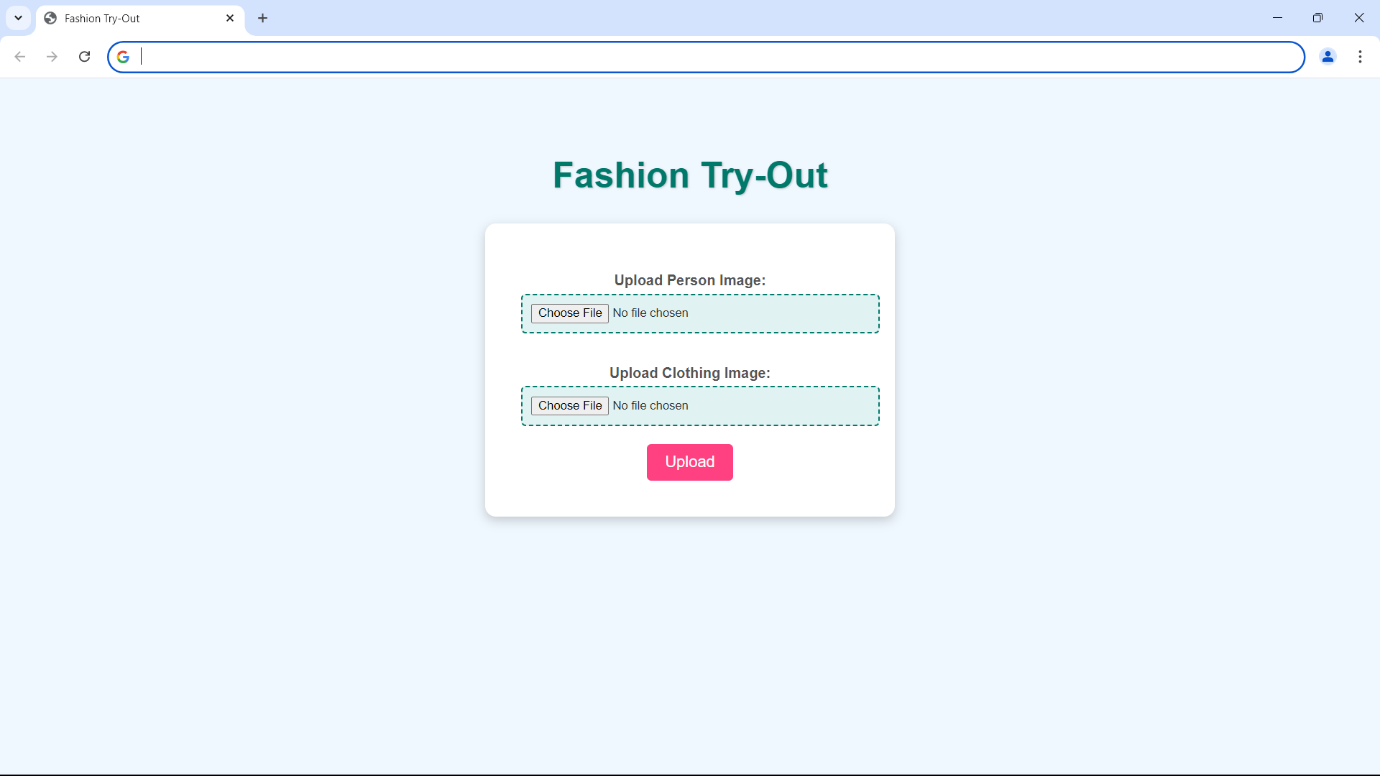
**Login Screen:**



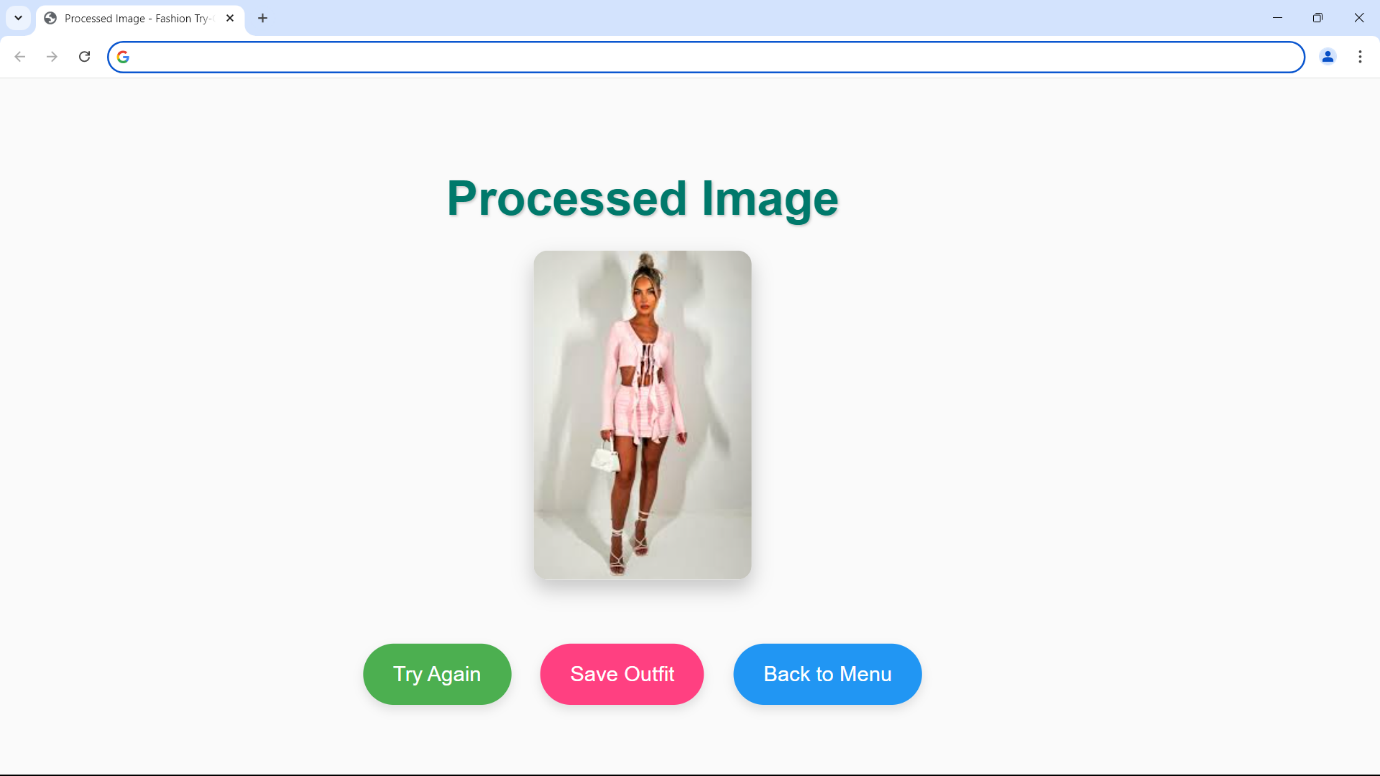
**Home Screen:**



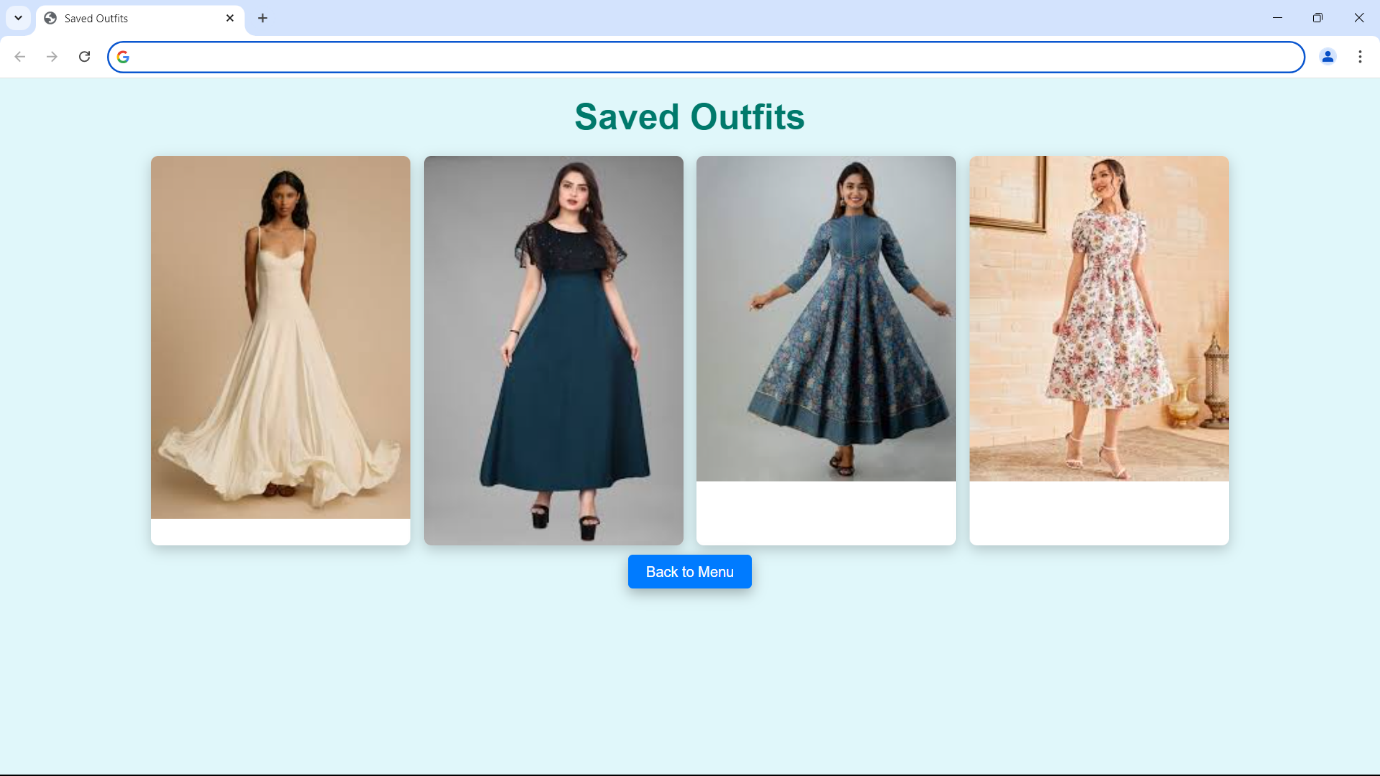
**Image Processing:**

****

**Result:**

****

**Saved Outfits:**

****

**CHAPTER 7**

**CONCLUSION AND FUTURE SCOPE**

**Conclusion**

Fashion try out brings benefits to both users and businesses. It offers convenience to buyers in terms of saving time and money and offering transparency while making an online purchase. At the same time, it reduces the number of products returned due to numerous reasons, saving businesses a lot of money. It also helps businesses develop a deeper and long-term relationship with their customers that ensures success and profitability.

Virtual reality design is a rapidly growing field that offers endless possibilities for creative and technical individuals. They allow customers to visualize how different items will look on them without physically trying them on, making the shopping experience more convenient and efficient. The technology has also expanded into other areas such as makeup and furniture, providing even greater opportunities for virtual try-ons. As this technology continues to evolve, it is likely that we will see even more innovative applications in the future.

**Future Scope**

 We will explore the integration of stoked reality( AR) and virtual reality(VR) technologies to produce indeed more immersive and engaging shopping gests . In conclusion, our design represents a significant step towards the future of online fashion retail by bridging the gap between physical and digital shopping gests . We\'re agitated about the eventuality of our virtual pass- on system to empower consumers, enhance their online shopping trip, and contribute to the elaboration of the fashion assiduity. As we continue to introduce and upgrade our system, we anticipate indeed lesser strides in the field of virtual pass- on technology, eventually serving both consumers and the fashion assiduity as a whole.

The future of fashion try-out is incredibly exciting, especially with the integration of AI**-**powered experiences. AI can fine-tune virtual try-on technology to ensure perfect fit and precise measurements for each customer. Imagine shopping for clothes and being confident that what you see on the screen will look just as good in real life.

Fashion try-out is set to be revolutionized by technologies like virtual reality (VR) and augmented reality (AR), enabling customers to try on clothes digitally from anywhere. Personalized AI recommendations will enhance the shopping experience, while smart mirrors in stores will streamline in-person trials. As sustainability becomes a priority, brands may focus on virtual options to reduce waste and returns. Additionally, subscription services for curated outfits and social media integration will foster community engagement, making try-outs more interactive and tailored to individual preferences. Overall, these innovations promise to create a more efficient and enjoyable fashion shopping experience.

The future of fashion try-outs is set for transformative changes driven by technology and evolving consumer preferences. Virtual and augmented reality will enable customers to try on clothes digitally from home, enhancing the online shopping experience and reducing return rates. AI will play a crucial role in offering personalized recommendations tailored to individual body types and style preferences, making the try-out process more relevant and enjoyable. Sustainability will become increasingly important, with virtual try-outs minimizing environmental impact by reducing waste and returns. In physical stores, innovations like smart mirrors will streamline the in-store experience, allowing for quick outfit changes without the hassle. Additionally, social shopping features will facilitate sharing and feedback among friends, creating a more engaging environment. Subscription models for curated outfits will further allow consumers to experiment with styles at home. Overall, the future of fashion try-outs will combine convenience, personalization, and sustainability, fundamentally reshaping consumer interactions with fashion.

**CHAPTER 8**

**BIBILOGRAPHY**

 [1] [Sen He](https://paperswithcode.com/author/sen-he), [Yi-Zhe Song](https://paperswithcode.com/author/yi-zhe-song), [Tao Xiang](https://paperswithcode.com/author/tao-xiang) “Image-based virtual try-on aims to fit an in-shop garment into a clothed person image.” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[2][Xintong Han](https://paperswithcode.com/author/xintong-han), [Xiaojun Hu](https://paperswithcode.com/author/xiaojun-hu-1), [Weilin Huang](https://paperswithcode.com/author/weilin-huang-1), [Matthew R. Scott](https://paperswithcode.com/author/matthew-r-scott-1) ” ClothFlow, an appearance-flow-based generative model to synthesize clothed person for posed-guided person image generation and virtual try-on.” European Conference on Computer Vision (ECCV), 2020.

[3] H. Duan and X. Wang “Virtual Try-on Network (MG-VTON) can generate a new person image after fitting the desired clothes into the input image and manipulating human poses.” IEEE International Conference on Computer Vision (ICCV), 2017.

[4][Amit Raj](https://paperswithcode.com/author/amit-raj), [Patsorn Sangkloy](https://paperswithcode.com/author/patsorn-sangkloy), [Huiwen Chang](https://paperswithcode.com/author/huiwen-chang), [Jingwan Lu](https://paperswithcode.com/author/jingwan-lu), [Duygu Ceylan](https://paperswithcode.com/author/duygu-ceylan), [James Hays](https://paperswithcode.com/author/james-hays)

“Swap Net, a framework to transfer garments across images of people with arbitrary body pose, shape, and clothing.” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[5] [Yuhao Xu](https://paperswithcode.com/author/yuhao-xu), [Tao Gu](https://paperswithcode.com/author/tao-gu), [Weifeng Chen](https://paperswithcode.com/author/weifeng-chen), [Chengcai Chen](https://paperswithcode.com/author/chengcai-chen) “OOTDiffusion, a novel network architecture for realistic and controllable image-based virtual try-on (VTON).” IEEE/CVF International Conference on Computer Vision (ICCV), 2021.