**Vision and Voice**

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

**Bachelor of Technology**

**in**

**Computer Science and Engineering**

**by**

**Ravipudi Sai Susritha 22N31A05K7**

**Pochammala Navya Sri 22N31A05H7**

**Rayakanti Sai Vamshi 22N31A05K8**

Under the esteemed guidance of

**Ms. B. SREE SHARANYA**

**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/)



**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 entitled **“VISION AND VOICE”** 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. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

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

**Ms. B. Sree Sharanya** **Dr. S. Shanthi**

**Associate Professor**  **Professor**

**External Examiner**

**DECLARATION**

## We hereby declare that the Application Development “Vision and Voice” 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 **Ms. B. Sree Sharanya** 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

“Vision and Voice” is an innovative technological solution for individuals with visual impairments. Object detection is an essential component in many applications such as security surveillance, robotics, autonomous vehicles and many more. Text to speech is a helpful technology for people with disabilities. We utilize the combination of these cutting-edge technologies to develop a system which acts as eyes to the visually impaired. This system enables its users to identify objects, once an object is detected the object is labelled and the object’s label is converted using TTS engine to provide vocal feedback. Using a pre-trained deep learning model such as YOLOv5, the system performs object detection on images or video feeds, identifying and classifying various objects with high accuracy. Using a pre-trained model like YOLOv5 allows the system to bypass the resource-intensive process of training from scratch. Pre-trained models like YOLOv5 have already been trained on large, diverse datasets (e.g., COCO or VOC), which ensures they have high accuracy in detecting and classifying a wide range of objects. This ensures fast processing, allowing the system to quickly identify objects in images or video feeds with minimal delay. This system works in real-time. It can detect objects and give vocal feedback instantly as soon as an object is recognized. “Vision and Voice” has the scope to be further customized and expand into various domains in the future.

**Keywords:** Object detection, Text to Speech, Realtime processing, YOLOv5, deep learning

|  |  |  |
| --- | --- | --- |
| **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 | **3** |
| **2** | **LITERATURE SURVEY** | **5** |
| **3** | **SYSTEM ANALYSIS** | **8** |
|  | 3.1 HARDWARE &SOFTWARE REQUIREMENTS | **8** |
| **4** | **SYSTEM DESIGN** | **9** |
|  | 4.2 ARCHITECTURE | **9** |
|  | 4.3 UML DIAGRAMS | **11** |
| **5** | **METHODOLOGY** | **15** |
|  | 5.1 TECHNOLOGIES USED | **15** |
|  | 5.2 MODULES DESCRIPTION | **16** |
| **6** | **IMPLEMENTATION** | **20** |
|  | 6.1 SAMPLE CODE | **20** |
|  | 6.2 OUTPUT SCREENS | **38** |
| **7** | **CONCLUSION & FUTURE SCOPE** | **43** |
| **8** | **BIBILOGRAPHY** | **45** |

**TABLE OF CONTENTS**

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE.NO** | **NAME** | **PG.NO** |
| **4.1.1** | **ARCHITECTURE DIAGRAM** | **10** |
| **4.2.1** | **CLASS DIAGRAM** | **11** |
| **4.2.2** | **USE CASE DIAGRAM** | **14** |
| **4.2.3** | **ACTIVITY DIAGRAM** | **12** |
| **4.2.4** | **SEQUENCE DIAGRAM** | **13** |
| **5.1.1** | **ARCHITECTURE OF PYTHON** | **10** |

**CHAPTER 1**

**INTRODUCTION**

"Vision and Voice" is an innovative system designed to assist individuals with visual impairments by combining two advanced technologies: object detection and text-to-speech (TTS). The system uses a pre-trained deep learning model like YOLOv5 to detect and identify objects in images or video feeds, providing high accuracy and fast processing. Once an object is recognized, the system labels it and converts the label into spoken words using TTS technology, offering immediate vocal feedback to the user. This allows visually impaired individuals to understand and navigate their surroundings more independently. The system works in real-time, ensuring that users receive instant feedback as objects are detected.

**1.1PURPOSE AND OBJECTIVES**

|  |
| --- |
| The purpose of the "Vision and Voice" system is to enhance the lives of individuals with visual impairments by providing a real-time, accessible solution for identifying and understanding their surroundings. By integrating object detection and text-to-speech technologies, the system aims to serve as an assistive tool that functions as a virtual "eye" for visually impaired users. The system's goal is to help users navigate their environments, identify objects, and receive immediate auditory feedback, fostering greater independence and safety.  The objectives of this project are:   1. Real-time Object Detection: To develop a system that can identify and classify objects from images or video feeds in real-time, utilizing pre-trained deep learning models (such as YOLOv5) for accurate and fast object detection. 2. Voice Feedback for Object Identification: To provide auditory feedback to the user by converting object labels into speech using a text-to-speech (TTS) engine, allowing users to hear the name or description of detected objects instantly. 3. Enhance Accessibility for Visually Impaired Individuals: To create a system that helps visually impaired users identify objects around them, improving their ability to navigate spaces, identify hazards, and engage more effectively with their surroundings. 4. Seamless User Experience: To ensure that the system operates with minimal delay, providing quick and accurate object detection with vocal feedback in real-time, ensuring an uninterrupted user experience. 5. Adaptability and Customization: To allow for future customizations and adaptability to different environments or specialized use cases, such as recognizing specific objects in the home, public spaces, or workplace, thereby expanding the system's functionality. 6. Expand the Scope for Global Use: To extend the system's language and object detection capabilities, ensuring that it can cater to a diverse, global user base with support for multiple languages and region-specific objects. 7. Continuous Improvement: To continually improve the system’s accuracy, speed, and ability to detect a broader range of objects, ensuring it remains relevant and useful as technology advances and user needs evolve.   **1.2 EXISTING AND PROPOSED SYSTEM** **Existing System Limitations**  1. **Manual assistance required:** Blind individuals rely on human assistance for navigation and object identification. 2. **Limited accessibility:** Existing assistive technologies have limitations in terms of cost, training, and functionality. 3. **No real-time feedback:** Blind individuals lack real-time auditory feedback about their surroundings. 4. **Dependence on others:** Blind individuals may feel dependent on others for daily tasks and navigation. 5. **Limited mobility:** Existing systems may limit the mobility and independence of blind individuals.  **Proposed System Enhancements**  1. **Real-time object detection:** Detects objects in real-time, providing immediate auditory feedback. 2. **Increased independence:** Enables blind individuals to navigate and interact with their environment independently. 3. **Improved accessibility:** Enhances the quality of life for blind individuals, promoting greater mobility and independence. 4. **Enhanced safety:** Reduces the risk of accidents or injuries by providing real-time feedback about the environment. 5. **Increased confidence:** Empowers blind individuals with the confidence to navigate and interact with their environment. |

**1.3 SCOPE OF THE PROJECT**

The scope of the "Vision and Voice" project focuses on developing a real-time, user-friendly system designed to assist visually impaired individuals. The system leverages pre-trained deep learning models, such as YOLOv5, for object detection and integrates a text-to-speech (TTS) engine to provide instant auditory feedback about detected objects. The primary objective is to deliver an accessible solution that enables users to navigate their environment, identify objects, and receive real-time information to improve their independence and safety.

The project will be designed to work across various devices, including smartphones, smart glasses, or wearables, that are equipped with cameras and speakers. The system will provide real-time object detection with minimal delay, ensuring that users receive immediate vocal feedback. Additionally, it will support multiple languages and be adaptable to different environments, such as homes, public spaces, or workplaces, with future plans to integrate specialized object recognition for specific contexts.

As the project evolves, it will include the potential for integration with other assistive technologies like GPS, smart canes, or wearables, further enhancing the user's ability to interact with their surroundings. The system will be scalable, with future updates focused on improving object detection accuracy, expanding the object database, and incorporating advanced features. This approach ensures that "Vision and Voice" remains relevant, providing an increasingly sophisticated and inclusive solution for visually impaired individuals.

User interaction is a critical aspect of the project, ensuring that the virtual try-on experience is customizable and intuitive. Users should be able to input their body dimensions, or potentially scan their bodies using a smartphone or other devices to generate a more accurate virtual representation. The system would also benefit from incorporating user feedback loops, allowing users to rate their experience and provide input on how well the virtual clothes fit compared to their actual size. This feedback can help refine the recommendation algorithms and improve the overall try-on experience.

**CHAPTER 2**

**LITERATURE SURVEY**

[Ayan Ravindra Jambhulkar](https://ieeexplore.ieee.org/author/37090038334); [Akshay Rameshbhai Gajera](https://ieeexplore.ieee.org/author/37090035562)[Chirag Manoj Bhavsar](https://ieeexplore.ieee.org/author/37090034593); [Shilpa Vatkar](https://ieeexplore.ieee.org/author/37090037197) et al. [1] use the YOLO\_v3 algorithm with the MS COCO dataset to detect and classify objects in real-time and provide corresponding audio feedback. We used gTTS (Google Text to Speech) API for generating the audio feedback. The audio feedback is generated using an audio processing techniques and deep learning algorithms. We evaluated on a dataset, and achieved an average detection accuracy of 90%. The proposed system provides a practical and effective solution for enhancing accessibility and independence for visually impaired individuals, and demonstrates the potential of using advanced deep learning algorithms and datasets for real-time object detection and audio feedback systems.

[Lakshmi Haritha Medida](https://ieeexplore.ieee.org/author/37089997909); [Selvi S](https://ieeexplore.ieee.org/author/37089993339); [A. Thilagavathy](https://ieeexplore.ieee.org/author/37089724185); [D. Naveen Raju](https://ieeexplore.ieee.org/author/37089015449); [Abbireddy Sai Jahnavi](https://ieeexplore.ieee.org/author/37089998165); [Abitha. P](https://ieeexplore.ieee.org/author/37089996700) et al. [2] a research study that intends to convert the image to text and then text to voice. Here, an intelligent system is proposed to recognize the common objects and generates verbal feedback to inform the user about the object's position. The best YOLO (You Only Look Once) version is selected after being compared. Following the process of object detection, the proposed model determines its spatial location before generating the speech output.

[P. Likhitha](https://ieeexplore.ieee.org/author/806387775773176); [Aftab Rauf Naik](https://ieeexplore.ieee.org/author/462268247071288); [Kishan Nagesh Chari](https://ieeexplore.ieee.org/author/333350123308905); [Sukanya Dessai](https://ieeexplore.ieee.org/author/37089452517); [Pratiksha Shetgaonkar](https://ieeexplore.ieee.org/author/37087087881) et al. [3] presented SightSpeak Object detection and speech generation for visually challenged people. This novel approach is to enhance accessibility for visually impaired individuals by integrating object detection and speech generation using YOLOv5 model on the COCO dataset, along with gtts for speech generation. The proposed system, named SightSpeak, aims to assist Konkani speakers in Goa by providing real-time auditory feedback about their surroundings. Through extensive data collection and analysis, the system identifies various common objects crucial for navigation. Employing YOLOv5 for object detection enables efficient processing and accurate recognition of objects in images. The system further utilizes Google Text-to-Speech to generate spoken descriptions of detected objects in Konkani, ensuring usability for the target user group. Evaluation metrics including precision, recall, and mean average precision (mAP) and MOS which validate the effectiveness of the proposed approach in delivering timely and relevant auditory cues to aid navigation and enhance the independence of visually impaired individuals in Goa.

[Naga Praneeth Kumar Chinni](https://ieeexplore.ieee.org/author/570436992364386); [Sai Pranav Reddy Kaamaala](https://ieeexplore.ieee.org/author/163439074216683); [Bhasham Vishva Vardhan](https://ieeexplore.ieee.org/author/950361500368693); [Adari Uday Kishan](https://ieeexplore.ieee.org/author/321915452638350); [Vasimalla Sunny Richards](https://ieeexplore.ieee.org/author/997962429121635); [Ramadasu Nooka Harsh Vardhan](https://ieeexplore.ieee.org/author/357234039076952) et al. [4] presented Vision Sense: Real-Time Object Detection And Audio Feedback System For Visually Impaired Individuals.This project introduces a transformative object detection system designed to enhance the navigational capabilities of visually impaired individuals through the application of advanced computer vision technologies. Utilizing the You Only Look Once (YOLO) model, paired with the Comprehensive Object Collection (COCO) dataset, this system provides real-time, accurate object detection and classification. The core functionality of the application allows for the processing of both static images and live video feeds, enabling blind users to receive auditory announcements of nearby objects, thereby assisting with spatial awareness and environmental interaction. The system leverages a pre-trained YOLO model to ensure robust detection performance, achieving a peak detection accuracy of 99%. By delivering object labels and bounding box coordinates audibly, the application serves as a critical tool in improving the daily independence and quality of life for people with visual impairments. This project not only highlights the potential of deep learning in assistive technologies but also underscores the importance of adaptive solutions in inclusive technology development.

[Kinnary Uday Panchal](https://ieeexplore.ieee.org/author/37088925812); [Dhruvisha Chetan Khara](https://ieeexplore.ieee.org/author/37088927994); [Taher Juzer Gari](https://ieeexplore.ieee.org/author/37088927776); [Vaishali Chavan](https://ieeexplore.ieee.org/author/37088927570)  et al. [5] presented Companion: Easy Navigation App for Visually Impaired Persons. They have proposed a system to aid visually impaired persons in their day-to-day navigation by providing them information about their surroundings using audio output (voice feedback) in real-time. The execution of this system is divided into three steps. The first is object detection using the You Only Look Once version 3 (YOLOv3) algorithm. Although multiple objects as detected in each frame, it is not practical to relay each object to the (visually impaired) user. There is a need to find the closest and most relevant object for the user. This is done in the second stage by estimating the depth and finding the closest object with the help of a depth map and comparing the size of bounding boxes in real-time. The output of the first two steps is generated in text format. The last step is to convert the text into audio output using the Text-To-Speech (TTS) API. As this system is intended to be inexpensive and readily available, this system is implemented on Android smartphones and has no additional hardware requirements.

**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
* Mobile OS:

Ios (13.0 or later)

Android (8.0 or later)

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE**

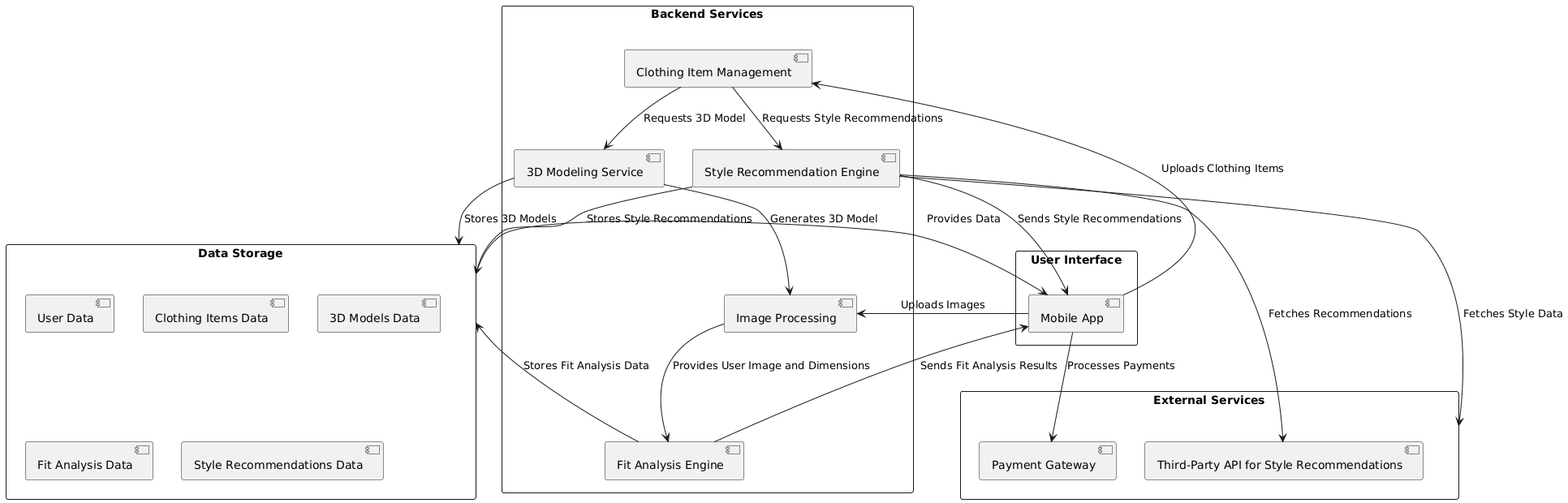
System requirements are the functionality that is needed by a system in order to satisfy the customer's requirements. System requirements are abroad and a narrow subject that could be implemented to many items. The requirements document allows the project team to have a clear picture of what the software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

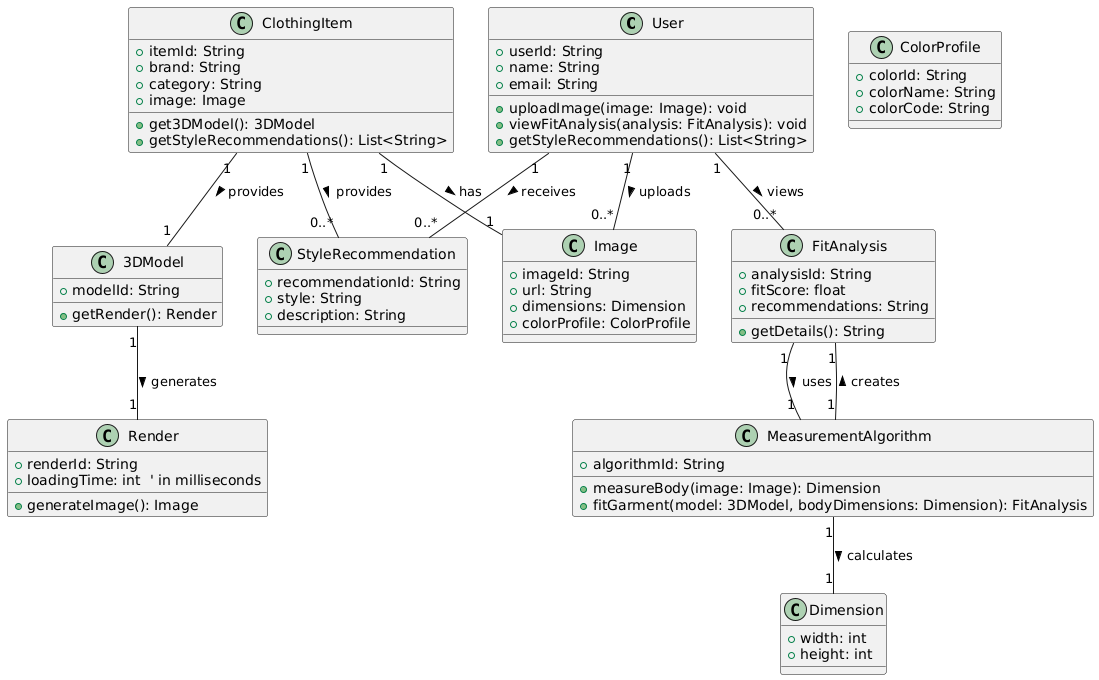


**Fig: 4.1.1 Architecture Diagram**

**4.2 UML DIAGRAMS**

**Class Diagram**

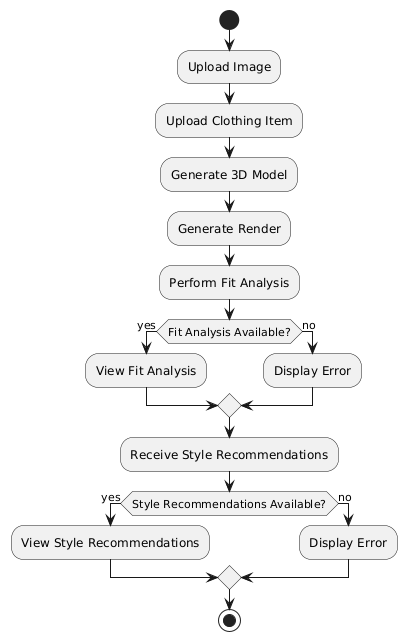
In software engineering, a class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and the relationships among objects. It is a representation of an object in many ways, 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**

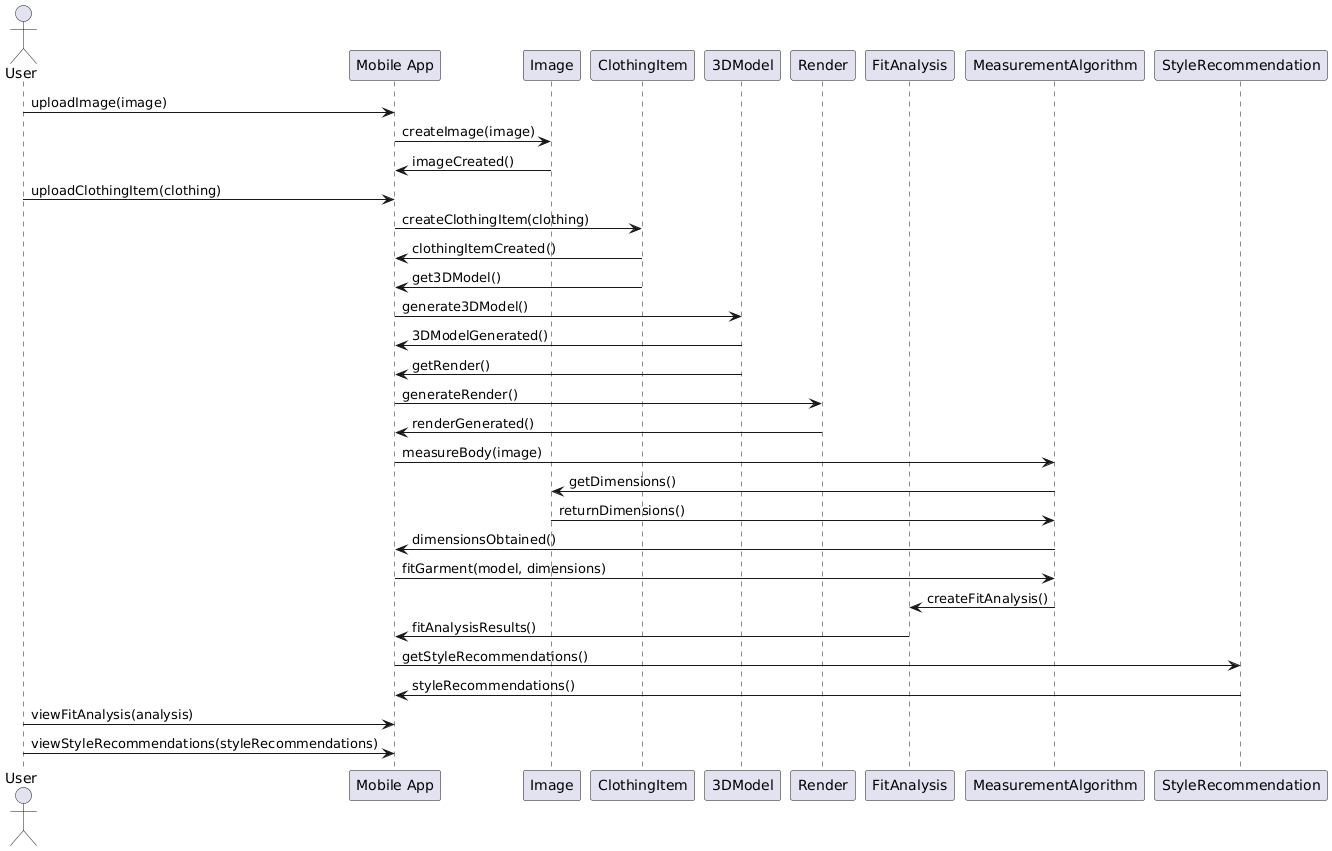
**Activity Diagram**

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**

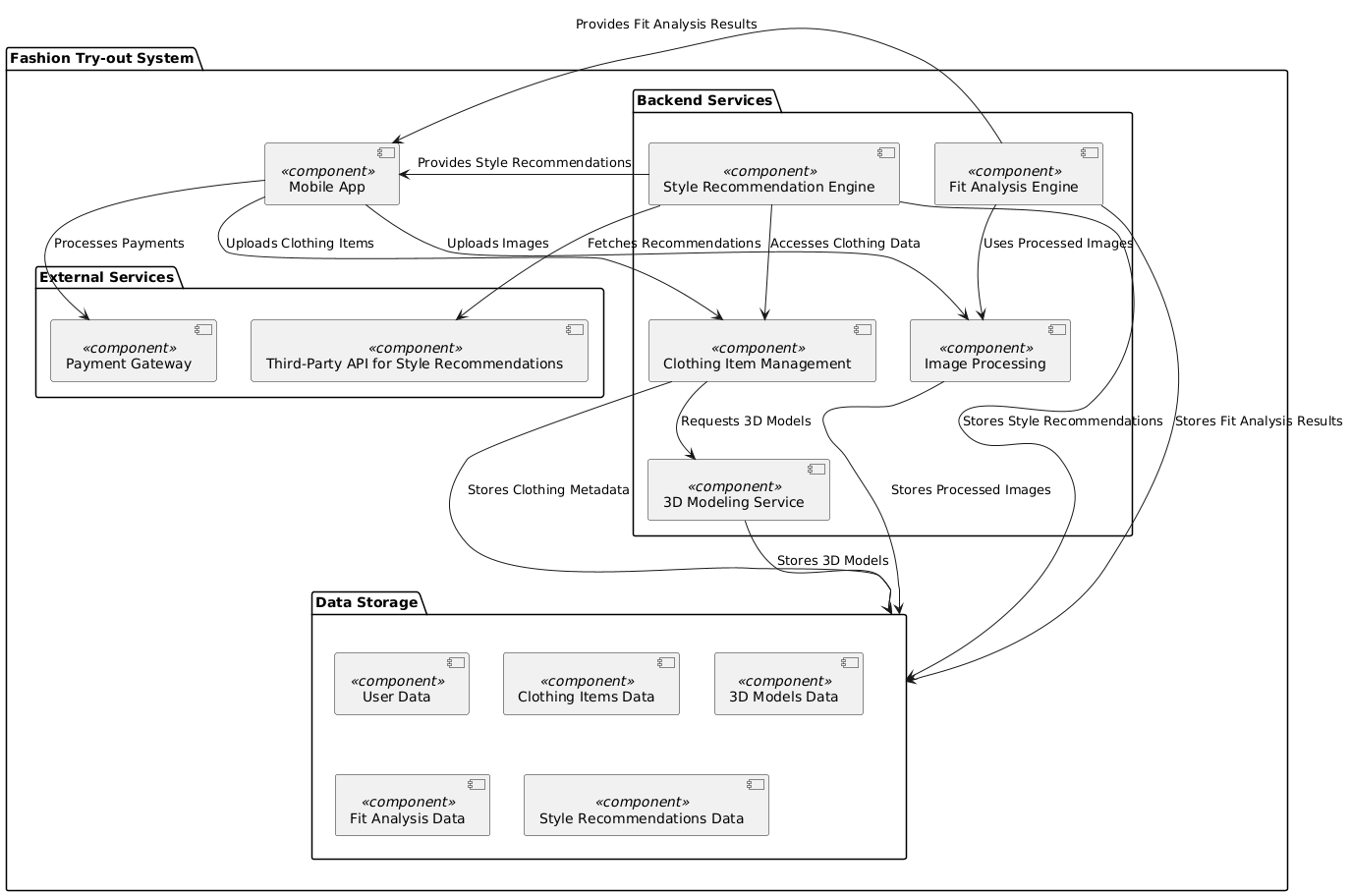
**Sequence Diagram**

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)

      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]

    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

    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)

    # 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]

     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

  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

     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]

  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:

    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')

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

  else:

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

  return output

`

**6.2 OUTPUT SCREENS**

**Login Screen:**

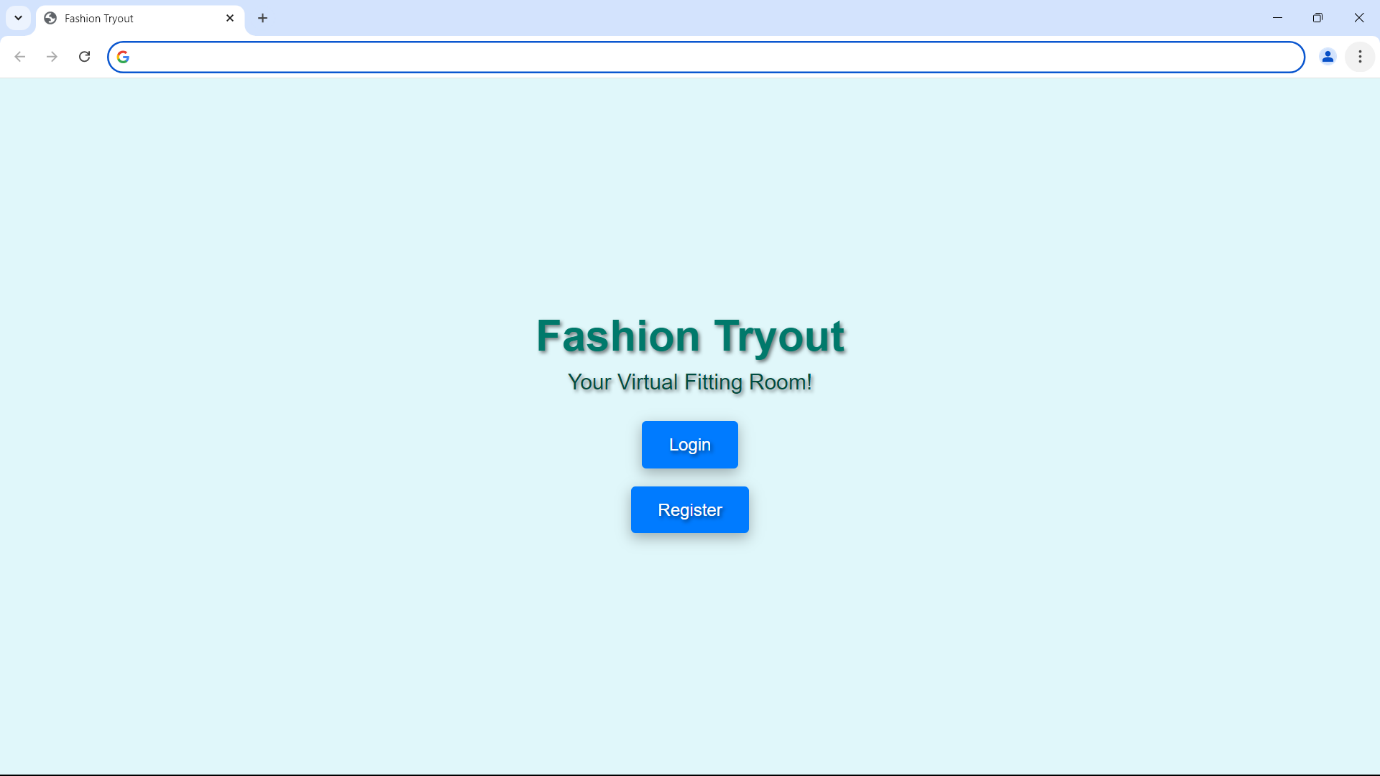


Fig : 6.2.1

**Home Screen:**

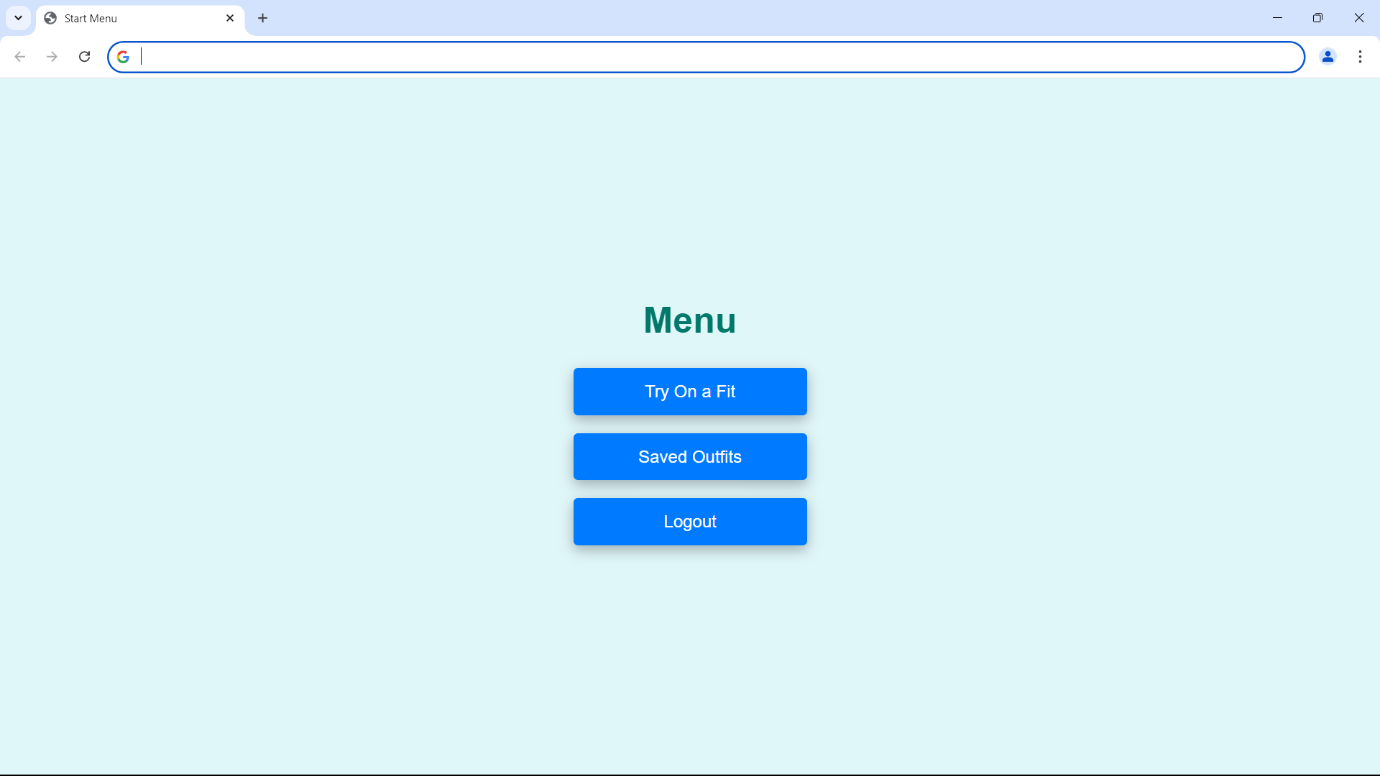


Fig : 6.2.2

**Image Processing:**

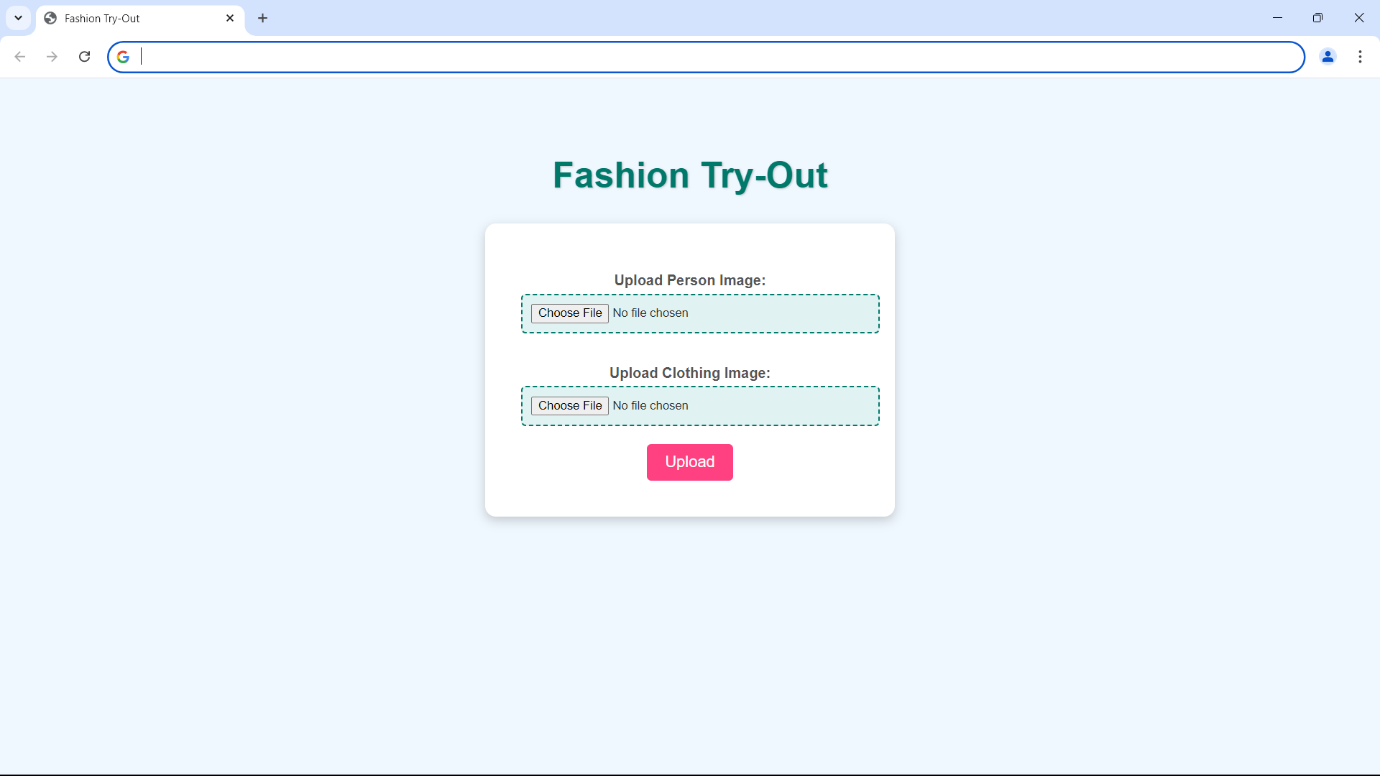
****

Fig : 6.2.3

**Result:**

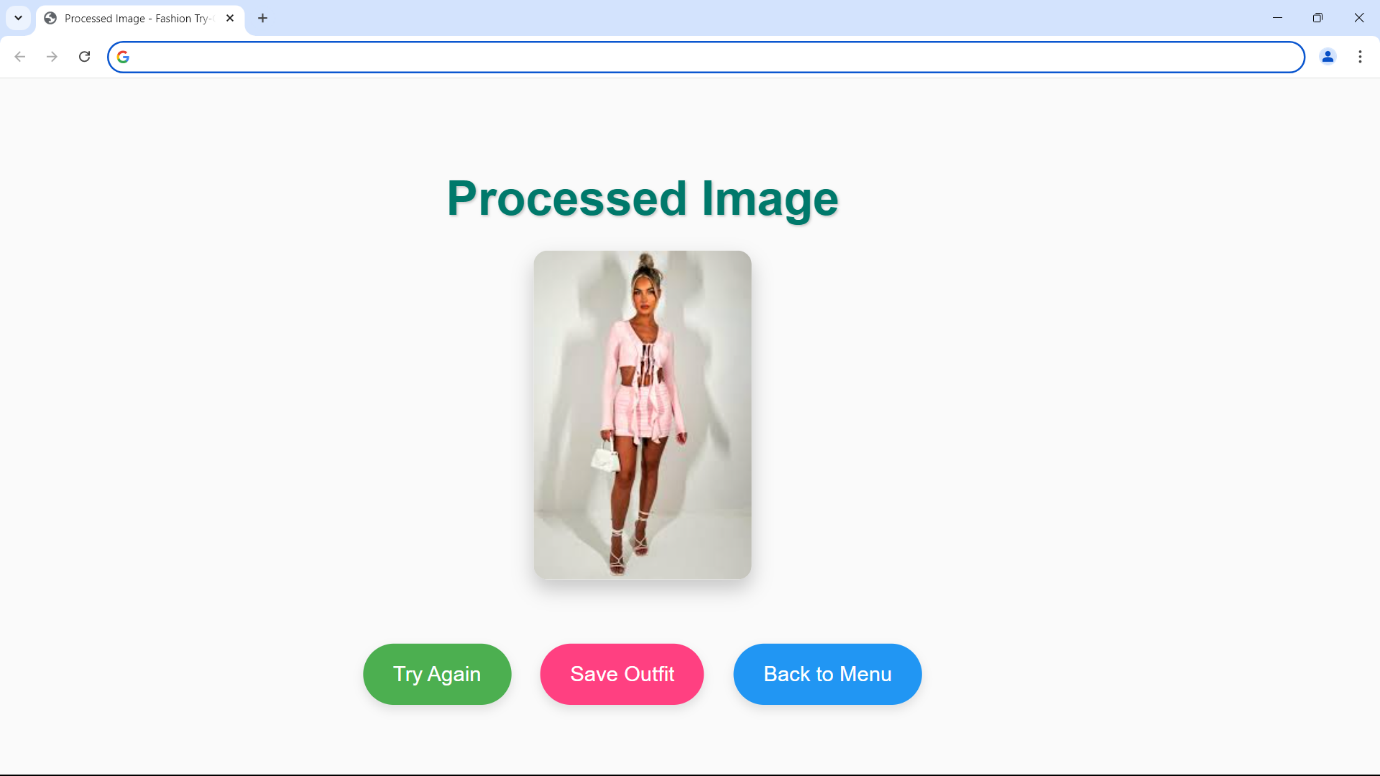
****

Fig : 6.2.4

**Saved Outfits:**

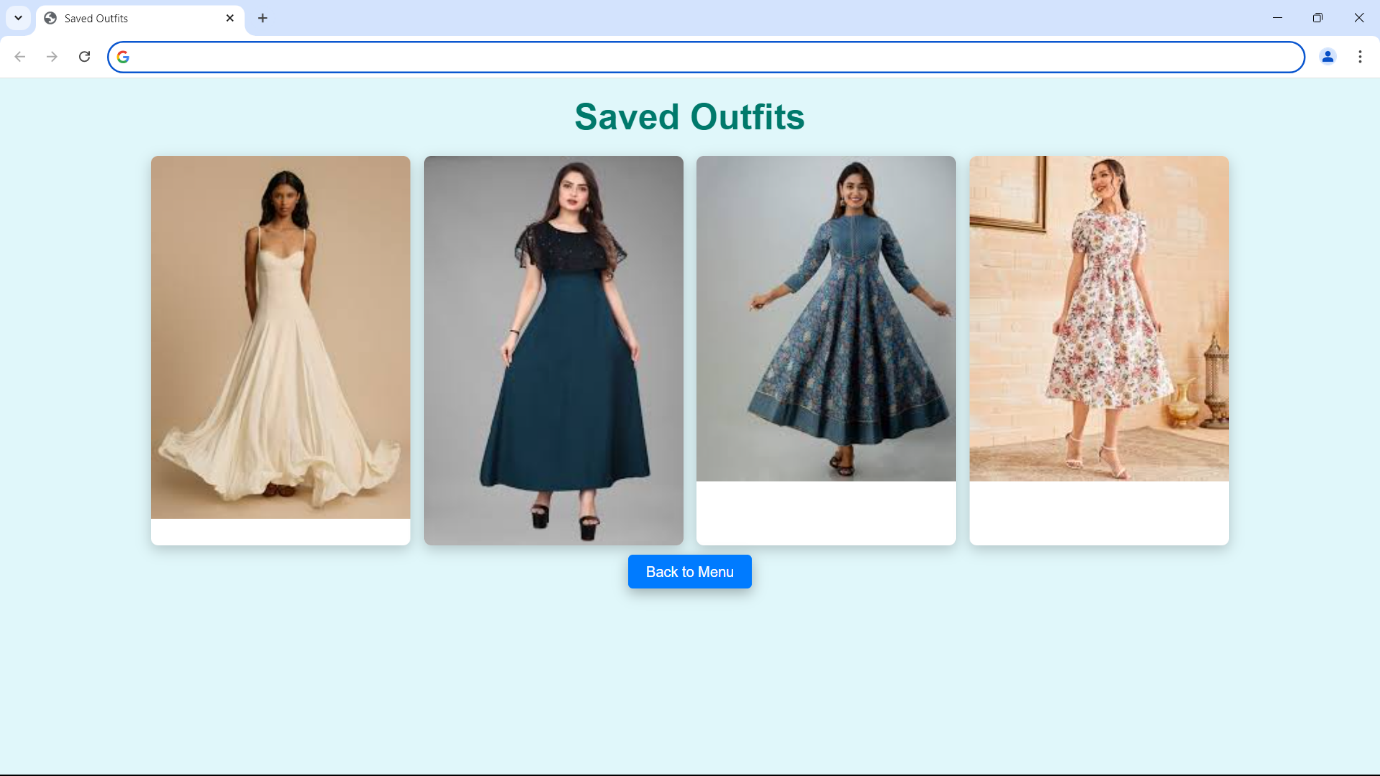
****

Fig : 6.2.5

**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.

[6] Z. Huang, M. J. Black, and J. S. D. Z. “Wearable Clothing Generation: A Generative Adversarial Network Approach for Virtual Try-On.” IEEE International Conference on Computer Vision (ICCV), 2019.

[7] J. T. Zhou, Z. Li, W. Chen, and H. Ling. “Fashioning an Image: A Comprehensive Survey on Fashion Image Generation and Manipulation.” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[8] R. P. Li, X. Zhang, H. Wu, and C. Li. “DeepFashion2: A Comprehensive Dataset for Fashion Understanding and Generation.” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[9] J. Liu, Y. Chen, and S. M. Y. Z. “VITON: An Image-based Virtual Try-on Network.” IEEE International Conference on Computer Vision (ICCV), 2019.

[10] T. J. Wu, Y. Xu, Y. C. Huang, and M. H. K. “FashionGAN: A Novel Network for Image-Based Virtual Try-On.” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.