Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: Capstone Project	**Ideation and stakeholder need analysis - Intermediate Review**	
	Date: 24/09/2025	**Enrolment No: 92200133001**

Team Member:

1. Ritesh Sanchala (92200133001)

Problem Statement:

Tailors, shoppers and especially in the fashion and tailoring domains, face difficulties in visualizing how a garment will look on their own body before purchase or stitching. This results in high product return rates, customer dissatisfaction, and inefficiencies in tailoring processes due to miscommunication about design, fit, or fabric. Existing virtual try-on solutions are expensive, or fail to handle complexities such as varied body measurements, fabric textures, and occlusions like hair overlapping clothing.

Virtual try-on system that leverages **computer vision, pose estimation, and segmentation techniques** to realistically overlay garments on a user's image while addressing accuracy, usability, and scalability.

Modular Design:

Frontend Module (User Interface)

- Built with HTML, CSS, and JavaScript, served by Flask templates.
- Allows users to upload personal images and garment files.
- Displays the processed try-on results.

Backend Module (Flask Server)

- Handles routing, API endpoints, and request/response cycles.
- Integrates AI model inference pipelines.
- Manages caching and session data to reduce redundant computation.

AI/ML Module

- **Pose Estimation:** Detects user body landmarks (shoulders, hips).
- **Segmentation:** Identifies the garment area and overlays it onto the target body image.
- **Post-processing:** Handles occlusions like hair overlapping garments.

Storage Module

- Stores temporary uploads (user images, garment images).
- No permanent database in v1; can be extended to use MongoDB or PostgreSQL for production.

Deployment Module

- Deployed on Hugging Face with Docker containers for portability.
- Can be extended to Render/AWS/Azure for scaling.

Tech Stack:

Languages & Frameworks

- **Python (Flask):** Chosen for lightweight server handling and easy integration with AI libraries.
- **JavaScript, HTML, CSS:** For user-facing interface.

AI/ML Libraries

- **PyTorch:** For loading and running deep learning models.
- **Transformers (Hugging Face):** For SAM (Segment Anything Model) and pre-trained pose estimation.
- **OpenCV:** For image preprocessing and post-processing tasks.

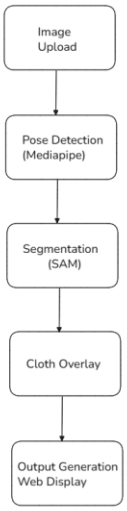
Deployment Tools

- **Docker:** Ensures portability across environments.
- **Hugging Face Spaces:** Provides public hosting with GPU/CPU support.

Justification of Choices:

- Flask is lightweight compared to Django, making it ideal for a prototype with AI integration.
- PyTorch dominates in research and offers flexibility with pre-trained models ([Paszke et al., NeurIPS 2019]).
- Docker ensures reproducibility and reduces dependency conflicts.
- Hugging Face is widely used in the ML community for hosting inference applications.

Flow Chart:



1) User Uploads Inputs

- Upload photo of person
- Upload garment image

2) Preprocessing

- Resize and normalize images
- Detect key landmarks (shoulders, hips, etc.)

3) Pose Estimation & Segmentation

- Apply pose estimation to identify body regions
- Use SAM / segmentation models to isolate upper-body clothing area

4) Garment Processing

- Align garment image with detected body landmarks
- Scale, warp, and adjust orientation of garment

5) Overlay & Blending

- Apply garment mask over person's image
- Blend edges for realistic appearance

6) Output Generation

- Final composite image shown to user
- Option to re-try with different garments