VIRTUAL DRESSING ROOM

using Python, Flask, OpenCV & MediaPipe

Sree Deepti AP23110011616
Keerthana AP23110011620
Prakarshi AP23110011621
Dikshya AP23110011673

cse-Y

Project Motivation

Online shopping often lacks visual try-on experiences
Users want to see how clothes fit before purchasing
Real-time virtual try-ons solve this issue using AI & Computer Vision
Inspired by fashion-tech apps (like Snapchat AR filters)

Objectives

- Use live webcam feed to track user body.
- Overlay wardrobe items (e.g. shirts) virtually.
- Detect shoulders and upper body using pose landmarks.

Tech Stack

Python – *main programming language*
OpenCV – *for video processing and image overlay*
MediaPipe – *for accurate pose detection*
Flask – *for the app*
static/wardrobe/ – *directory storing clothes to overlay*

How It Works

1. *Capture webcam feed using OpenCV.*
2. *Detect key pose landmarks using MediaPipe.*
3. *Track shoulder coordinates for shirt placement.*
4. *Overlay images of clothes using coordinate math.*

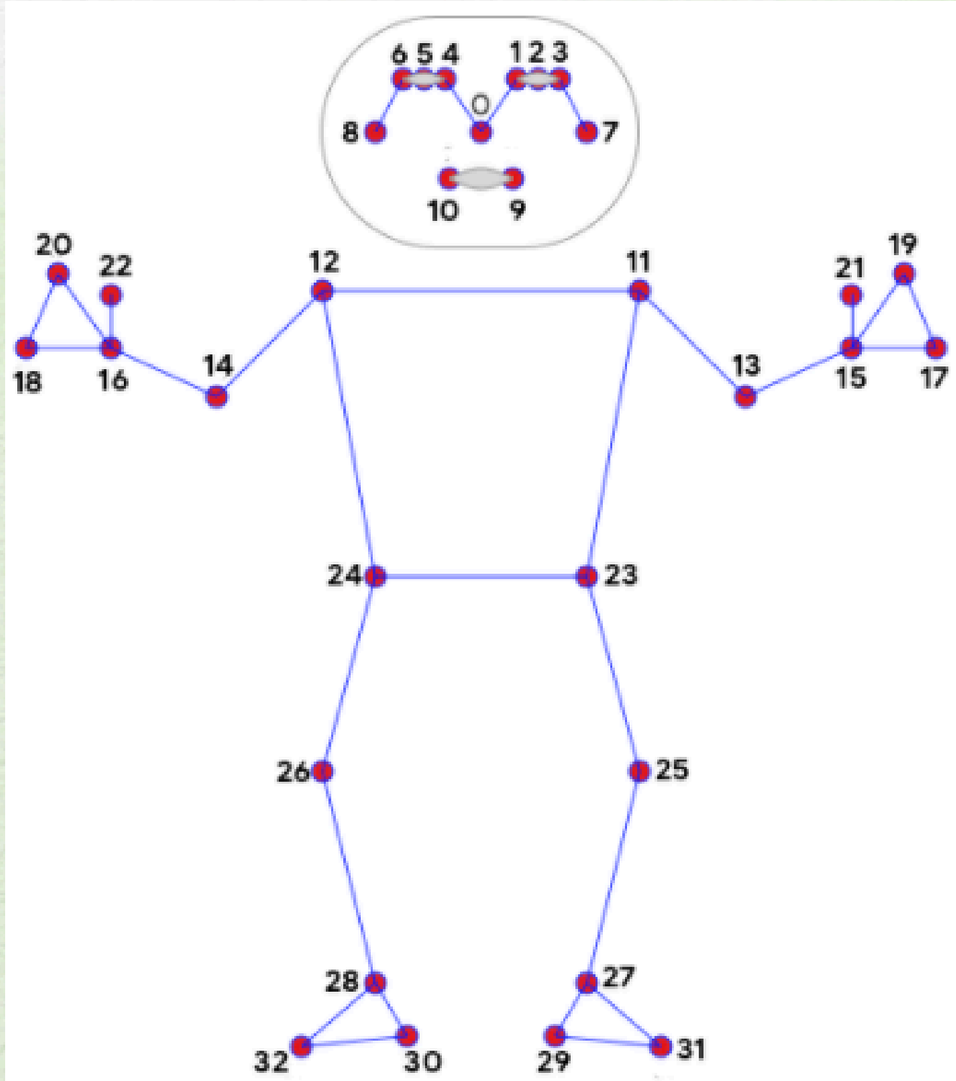
Code Structure

- ***generate_frames()*** *– Captures webcam feed, runs pose detection, overlays shirt in real time*
- ***overlay_shirt()*** *– Overlays transparent PNG shirt on the user's torso based on pose*

Uses:

- ***cv2.VideoCapture()*** *for webcam*
- ***mediapipe.Pose()*** *for landmark detection*
- ***cv2.resize()*** *and* ***NumPy*** *math for shirt placement*

mediapipe pose detection module :



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Landmark Detection Logic

Extracts LEFT_SHOULDER and RIGHT_SHOULDER from MediaPipe:

l_sh = lm[mp_pose.PoseLandmark.LEFT_SHOULDER]

r_sh = lm[mp_pose.PoseLandmark.RIGHT_SHOULDER]

Calculates:

- shoulder_dist – distance between shoulders
- shirt_w, shirt_h – for realistic sizing
- x_offset, y_offset – for overlay alignment
- Aligns shirt just below the shoulders to avoid overlapping the face

HTML Templates Overview

Template	Purpose / Role
base.html	The main layout file. Contains shared components like navbar, CSS links, etc.
index.html	The landing/home page. Welcomes the user and gives an intro to the project.
home.html	Placeholder for a future dashboard or profile page if needed.
camera.html	Displays live webcam feed with overlaid virtual shirt. Receives selected shirt.
wardrobe.html	Shows all shirt images from /static/wardrobe/ for the user to choose from.

Flask App Routes

Route	Method	Description
/	GET	Loads the home page (index.html). Introduction or welcome screen.
/camera	GET	Loads camera.html where the virtual try-on is shown. Accepts shirt param.
/video_feed	GET	Streams real-time webcam video with shirt overlay using OpenCV + MediaPipe.
/wardrobe	GET	Displays available shirt images from static/wardrobe/.

Challenges

1. *Aligning clothes perfectly across various body types*
2. *Lighting conditions affecting pose detection*
3. *Handling occlusions (arms crossing over torso, etc.)*

Future Improvements

- *Add full-body try-on (pants, skirts)*
- *AI-based clothes fitting or scaling*
- *Use a better backend (e.g., TensorFlow.js for web AR)*
- *Integrate with online shopping APIs*

thank you :)