ECE302-Embedded System and Design

Project Report

Monsoon Semester 2023

Guided by **Professor Sanket Patel**

Group Details:

Name	Enrollment Number
Het Patel	AU2140149
Dhruv Hingu	AU2140032
Parv Patel	AU2140180
Sankalp Patel	AU2140025
Saumil Patel	AU2140026
Ayush Patel	AU2140052
Rutul Patel	AU2140206
Nish Parikh	AU2140039
Vanaja Agrawal	AU2140213

Problem Statement:

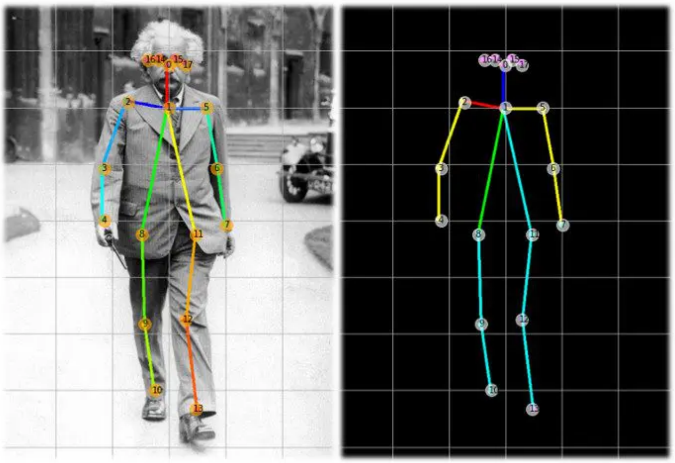
In the world of computer vision and embedded systems, the need for real-time and accurate human pose estimation has become paramount for various applications, ranging from health monitoring to immersive user experiences. The objective of this project is to design and implement an embedded pose estimation system that compares a user-provided reference pose image with real-time pose data captured by a sensor-equipped camera. The project leverages the capabilities of the OpenCV (cv2) library and the MediaPipe pose estimation model.

Literature Review:

Human pose estimation has garnered significant attention in computer vision and machine learning communities due to its wide range of applications, including human-computer interaction, activity recognition, and robotics. Over the years, researchers have explored various methodologies to accurately estimate the spatial configuration of human joints in images or videos. This literature review provides an overview of key developments in human pose estimation, highlighting influential approaches and recent advancements.

- **Classical Approaches:**

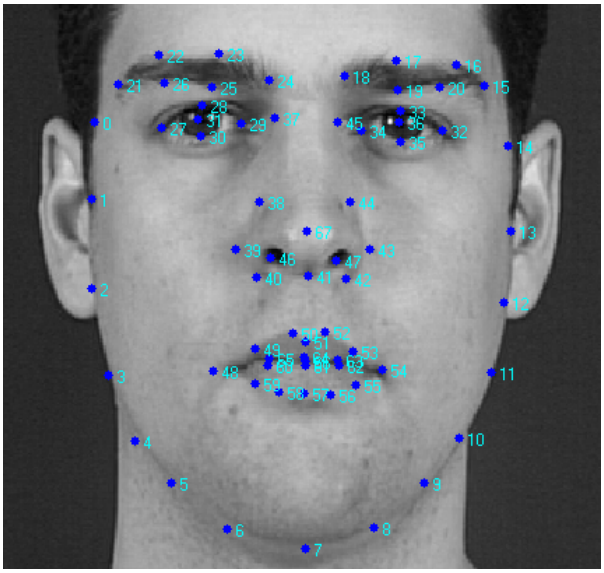
Early efforts in human pose estimation primarily relied on classical computer vision techniques. These methods often involved handcrafted features and rule-based algorithms to detect and localise body parts. While effective in controlled environments, these approaches struggled with challenges such as occlusions and variations in pose and appearance.



- **Model-based Approaches:**

Model-based methods emerged as a significant paradigm in human pose estimation. These techniques leverage articulated models of the human body, incorporating kinematic constraints to estimate pose parameters. While effective in capturing the inherent structure of the human body, model-based approaches faced limitations in handling complex poses and diverse datasets.

Example: The Active Shape Models (ASM) technique represents the statistical variation of shape and appearance in a training set, allowing it to adapt to different poses.



- **Deep Learning Revolution:**

The advent of deep learning, particularly convolutional neural networks (CNNs), revolutionised human pose estimation. Deep learning models demonstrated remarkable success in capturing intricate patterns and hierarchical representations from raw image data. Early architectures, such as Convolutional Pose Machines (CPM) and Hourglass Networks, paved the way for subsequent developments.

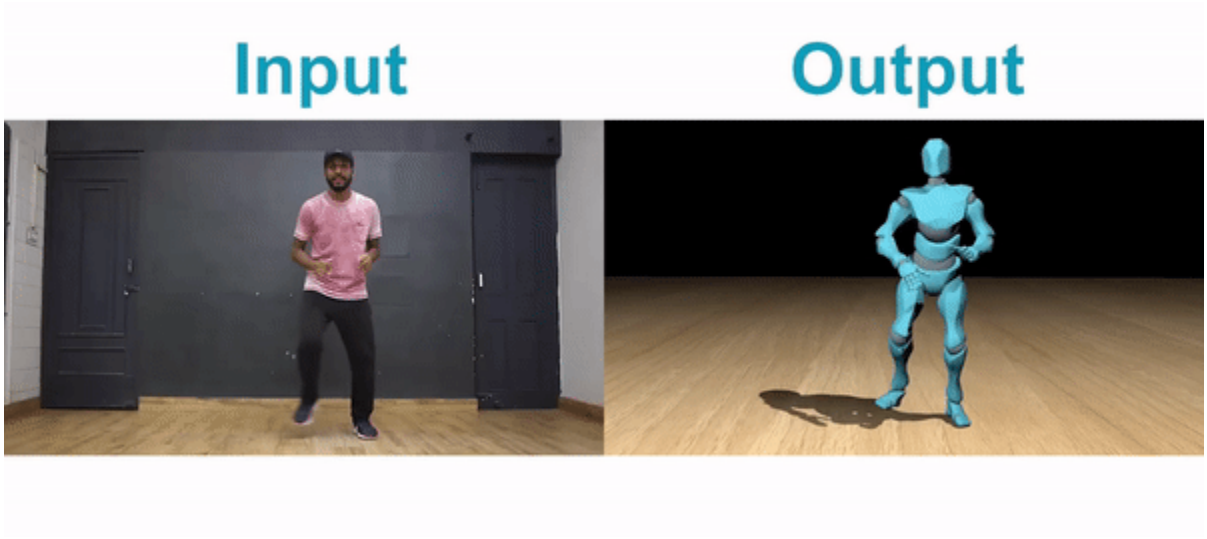
- **Single-Stage and Multi-Stage Architectures:**

Researchers explored both single-stage and multi-stage architectures for human pose estimation. Single-stage architectures, exemplified by the popular OpenPose framework, perform end-to-end prediction of joint locations. In contrast, multi-stage approaches, including stacked hourglass networks, iteratively refine predictions through cascaded stages, achieving enhanced accuracy.

- **3D Pose Estimation:**

Recent advancements have extended human pose estimation to the 3D domain. Integrating depth information, either from depth sensors or monocular cues, allows for the estimation of the

three-dimensional pose of human subjects. This capability is crucial for applications such as augmented reality and human-robot interaction.



- **Datasets and Benchmarks:**

The availability of diverse and challenging datasets, such as MPII Human Pose and COCO, played a pivotal role in advancing the field. These datasets facilitate benchmarking and comparison of different algorithms, fostering healthy competition and driving continuous improvements in performance**.**

<u>Types of Pose Estimation:</u>

1. Kinematic Model

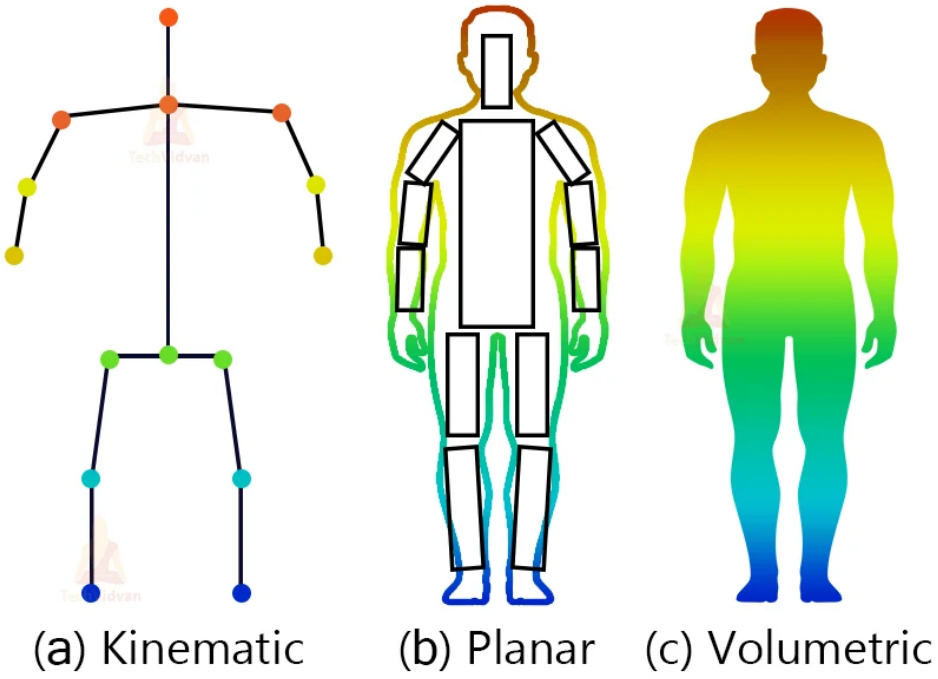
For 2D and 3D pose estimation, the kinematic model, also known as the skeleton-based model, is employed. This versatile and intuitive human body model provides a collection of joint configurations and limb orientations to depict the human body structure. As a result, kinematic pose estimation models are utilised to record the relationships between different body sections.

2. Planner Model

Planar Model also known as contour-based model, is used for 2D pose estimation. The shape and look of a human body are represented by the planar models. Body components are typically represented by several rectangles that approximate the human body shapes.

3. Volumetric Model

Volumetric model is typically used in 3D pose estimation. It is made up of a variety of popular 3D human body models and poses represented by human geometric meshes and forms, which are generally captured for deep learning-based 3D human pose estimation.



Our Implementation:

In order to find the coordinates of the body we have used the mediapipe model. MediaPipe is a popular open-source library developed by Google. The Pose Estimation model in MediaPipe is designed to identify and localise key body landmarks or keypoints. These key points correspond to specific anatomical points on the human body, such as shoulders, elbows, wrists, hips, knees, and ankles. Atlast, storing those anatomical landmarks in one csv file which will be used later for comparison.

Further, getting input from a web camera using CV2 and storing 30 frames from it. For each frame we will find anatomical points in real time and compare it with each point stored in a csv file to find euclidean distance. Now if error or distance is greater than the threshold set then it will show a box which indicates that in this particular point there is an error.

We then normalise the distance between consecutive coordinates by dividing the x coordinates by the distance between the length of shoulder and y-coordinates by the length of torso. After finding the distance between the coordinates we compare the frames of the image with the reference coordinates and calculate the euclidean distance between them. We then provide the suggestion according to the distance.

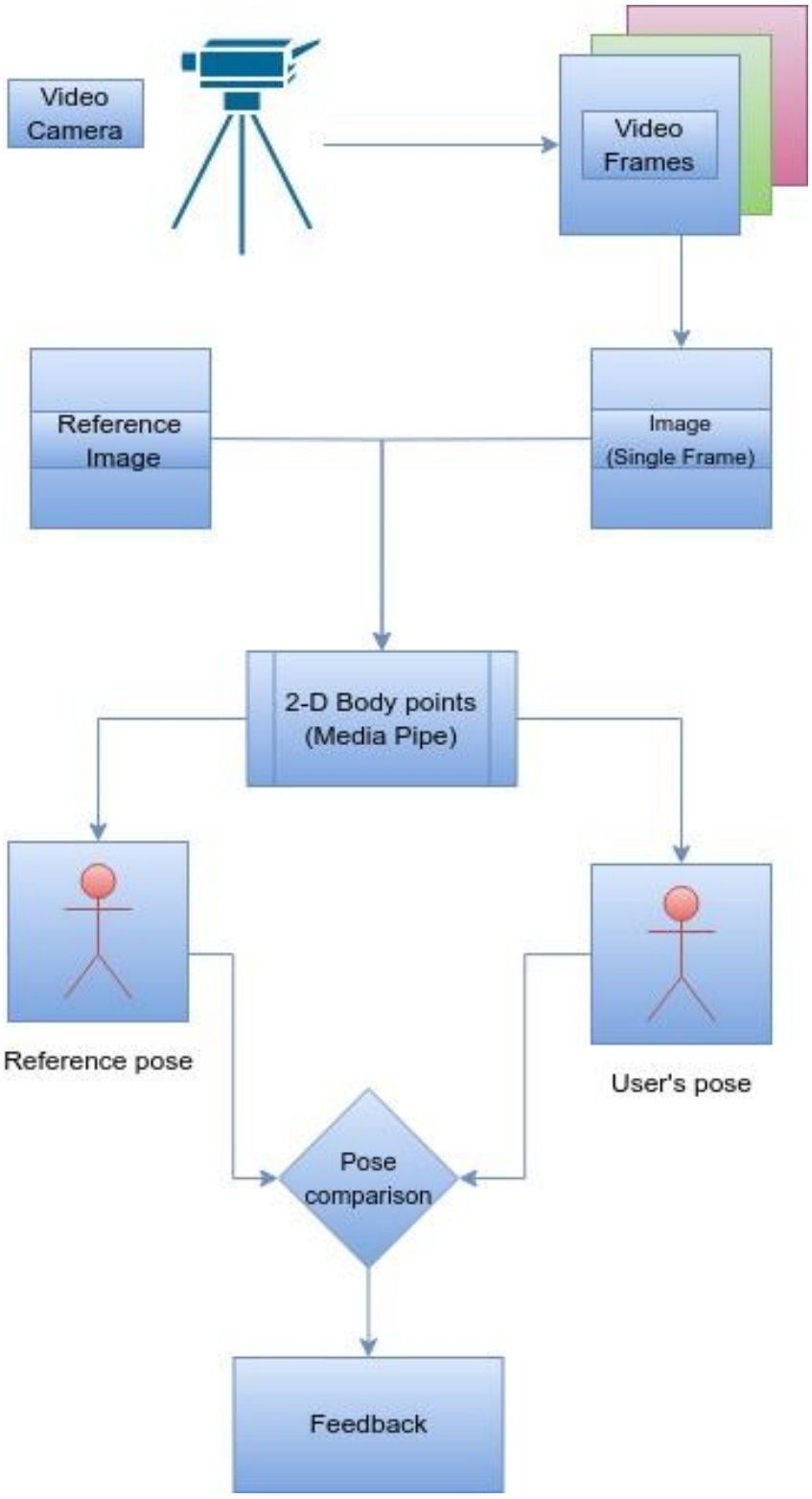
The problem in our implementation:

In our implementation there were two basic problems in our implementation. One of them was, if in the reference image the object is in the middle of the frame and the person who is standing in front of the camera is skewed toward any one side. In this particular scenario, although the person is perfectly imitating correctly, it will still display an error. Another problem is what if a different person has different physical parameters (like height, width, etc.) and is standing in front of the camera. It will display different distances/errors for each person.

Possible Solution of it:

One of the possible solutions for this type of problem is to use cosine similarity. Cosine similarity works by comparing the vector of the pose orientation instead of just using the magnitude for comparison. By normalising the distances between consecutive coordinates and utilising cosine similarity, the algorithm becomes more robust to changes in body positioning and physical attributes.

Flow Chart:


Video Camera
Video Frames
Reference Image
Image (Single Frame)
2-D Body points (Media Pipe)
Reference pose
User's pose
Pose comparison
Feedback

Web Implementation:

Tech Stack Used :

Frontend : HTML, CSS, Javascript

Backend : Flask (Python Framework)

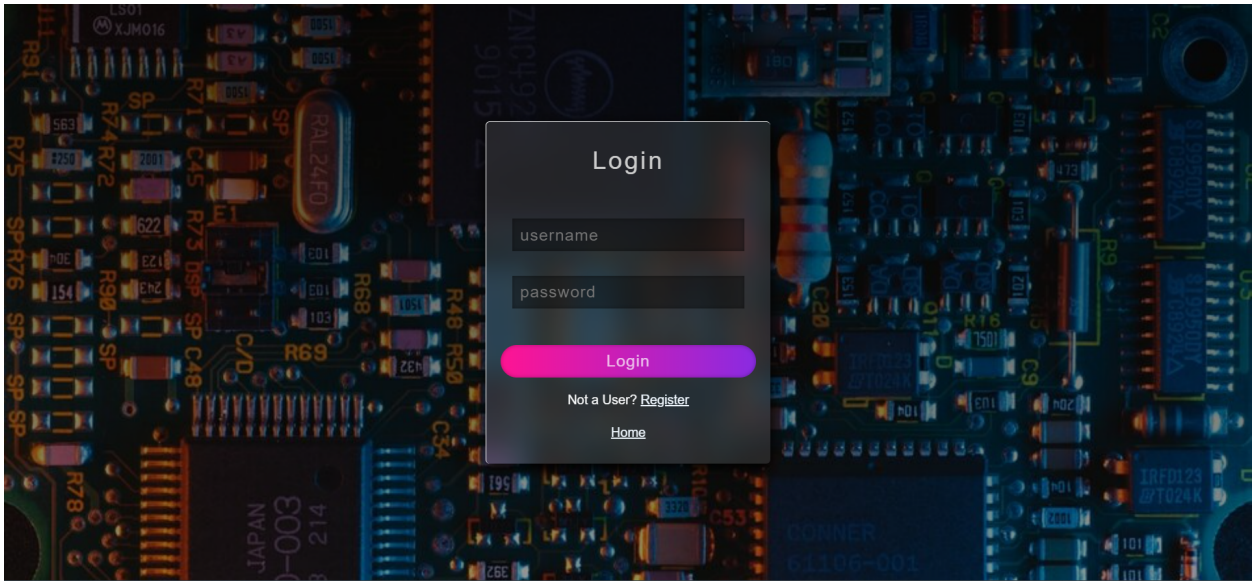
Functionality :

Login and Singup for user :

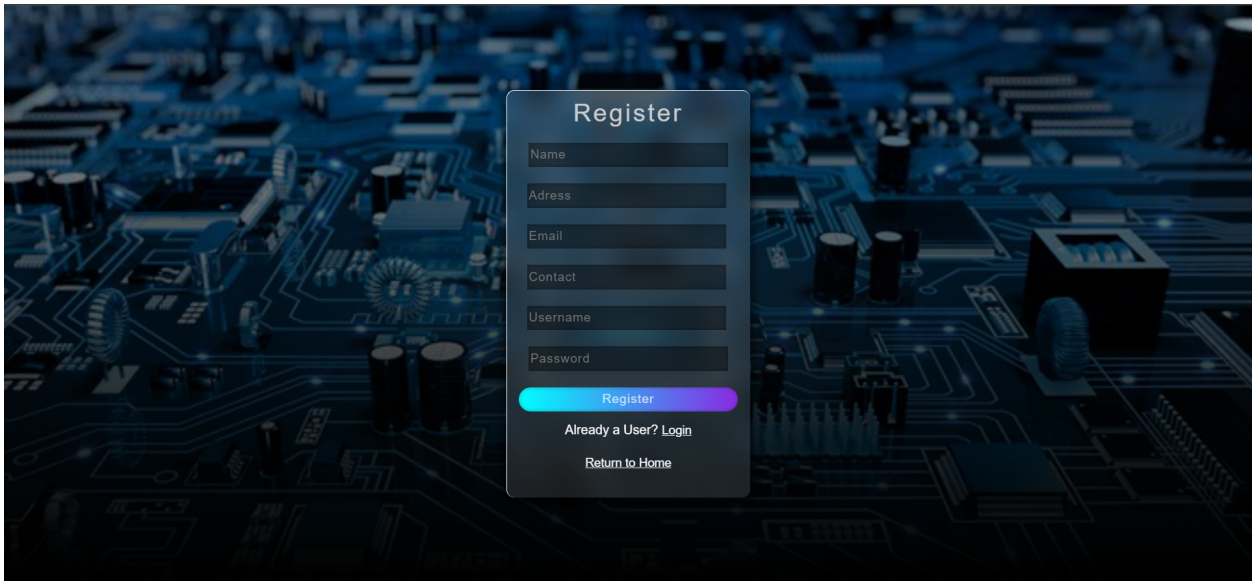
User can signup and login for accessing the website

User Interface :

Login :



Sign Up :



User DashBoard :

After login we are able to see the Dashboard which has two functionalities

1) Default Dashboard :

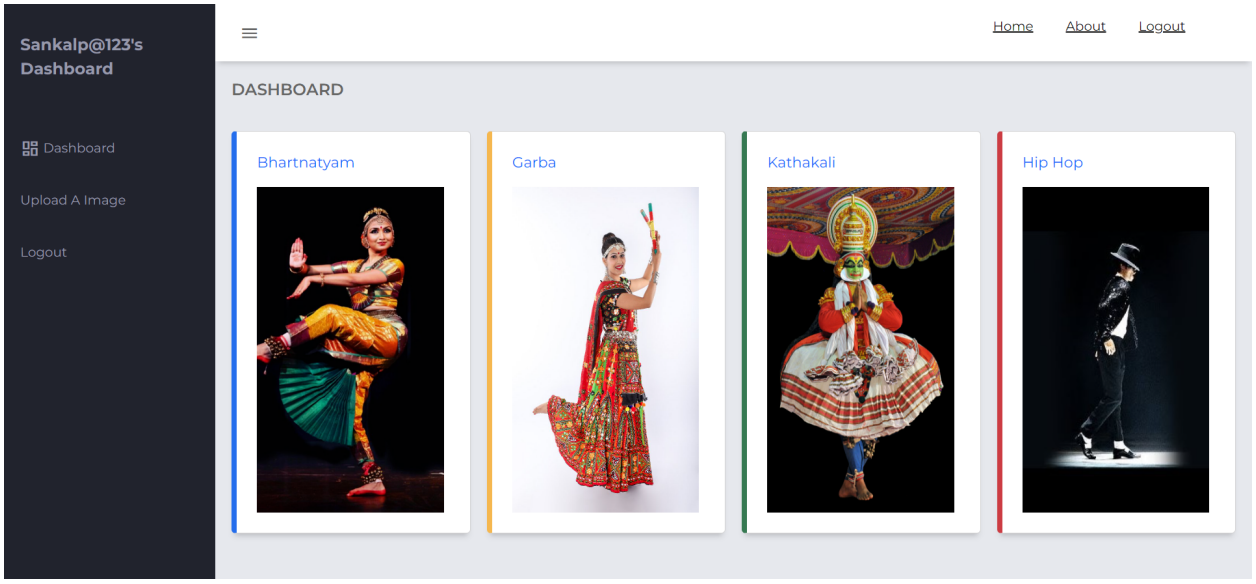
The Default Dashboard has 4 well known dance forms from which the user can select one (by clicking on the picture) and can perform pose estimation.

2) Upload Reference Image and then do the pose :

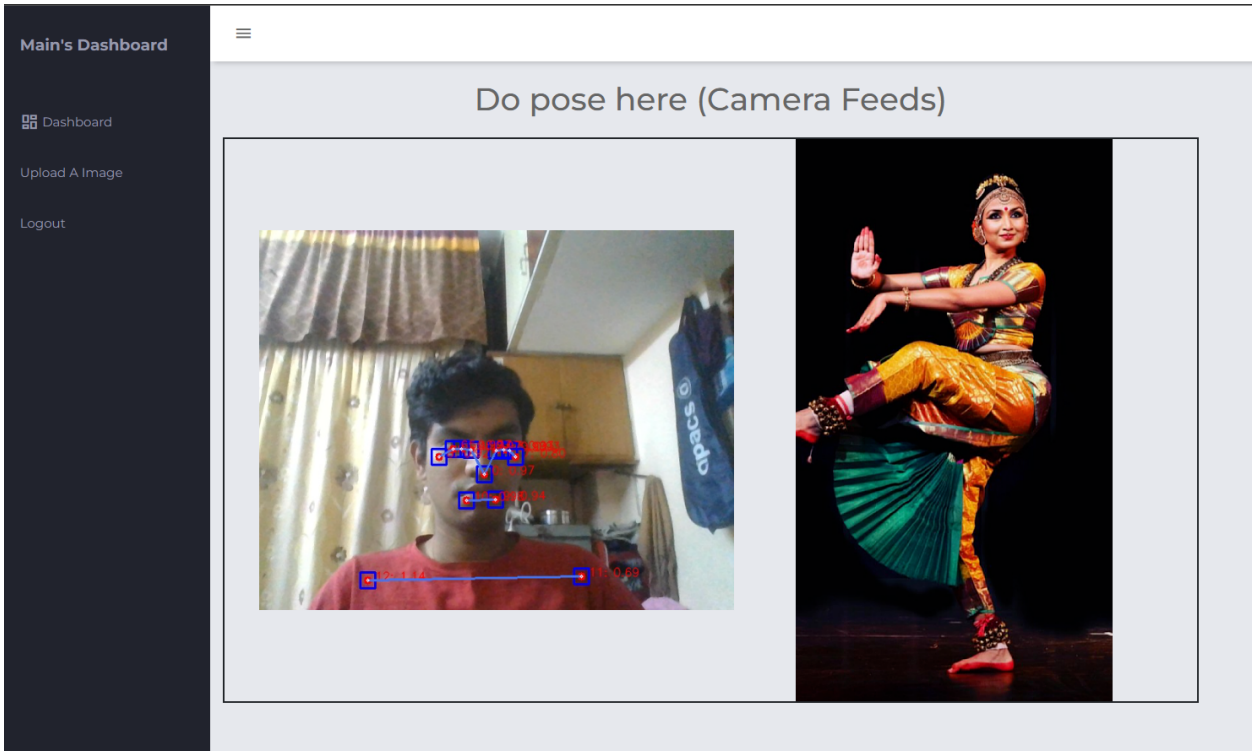
Now ! What if the user wants to do pose estimation for a dance form other than the 4 default ones. So, we have a functionality in which the user can upload the reference Image and then perform the pose estimation for the same.

User Interface :

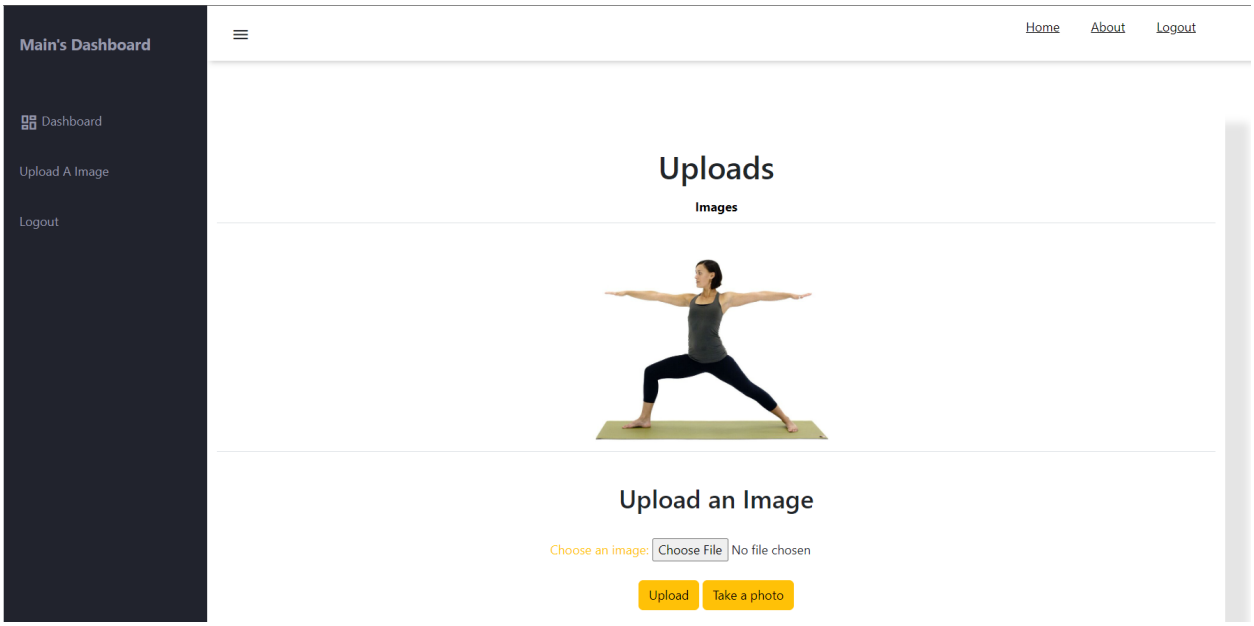
Default Poses Page :



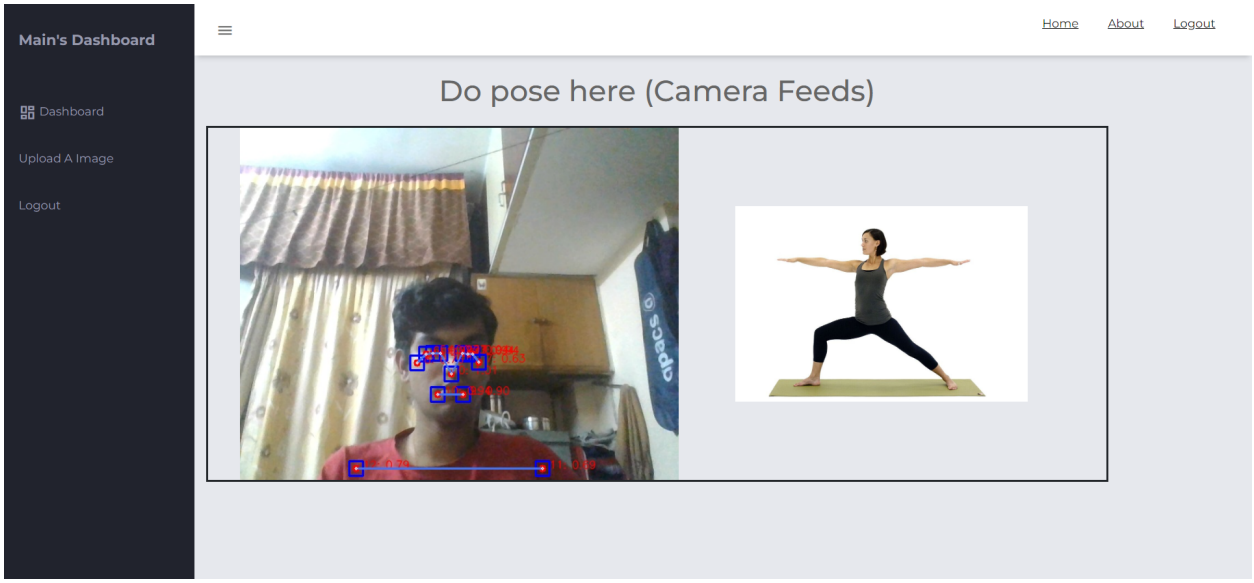
Pose Estimation for default poses page :



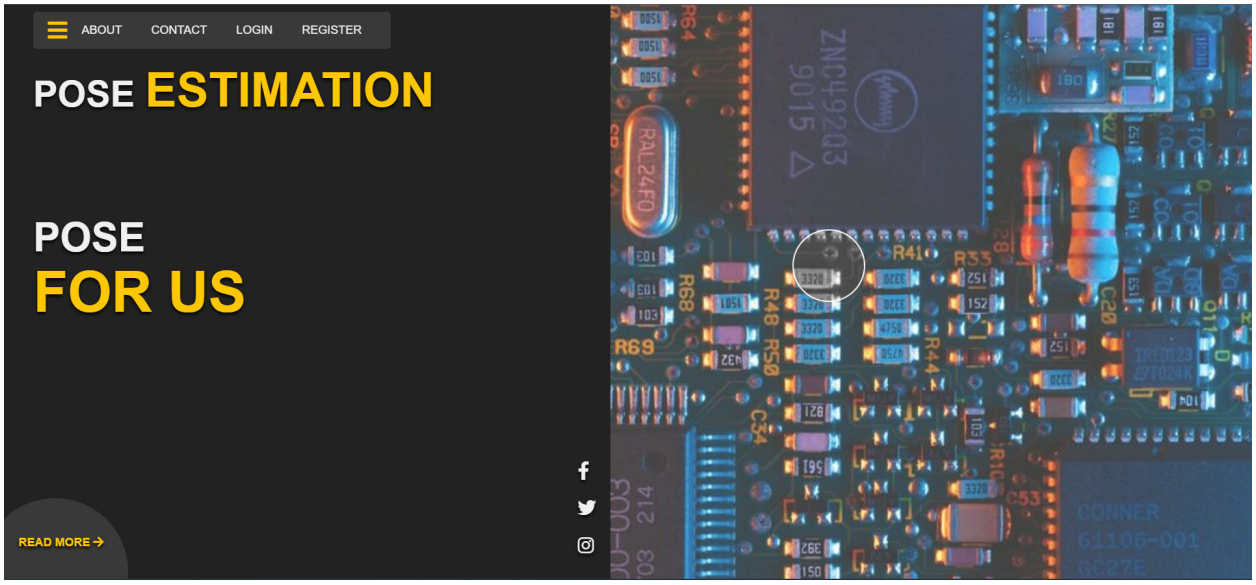
Upload reference pose page :



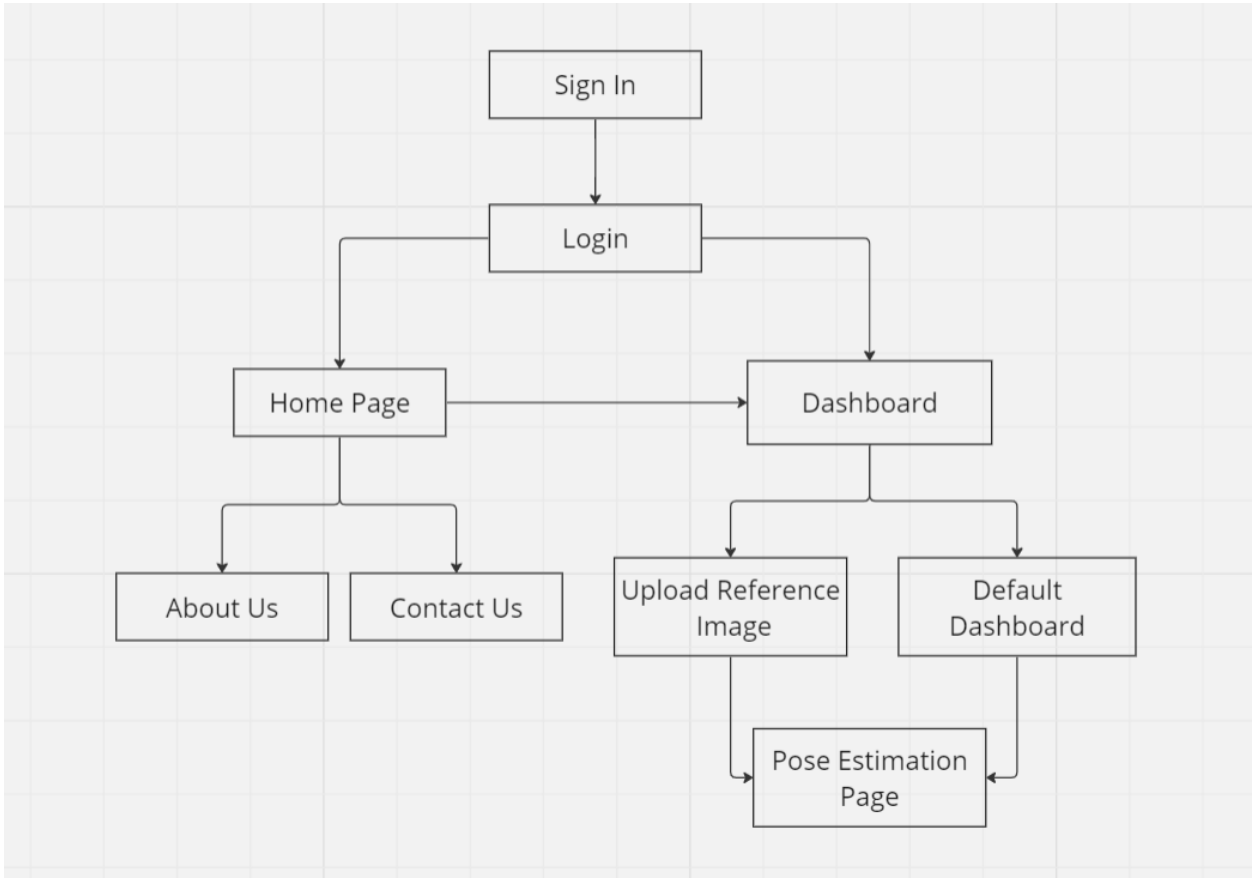
And By clicking on the picture of the pose we will be directed to the pose estimation page



Home Page User Interface:



Web Application Flow chart :



Contribution:

Name	Contribution
Het Patel	● Logic builder ● Implemented normalisation using static points and distance manipulation using data in .csv file. ● Contributed to the development of the project report
Dhruv Hingu	● Set up and manage the video processing pipeline. ● Handled visualisation, debugging, and overall program flow with exception handling and code organisation. ● Contributed to the development of the project report
Saumil Patel	● Developed the imagePoints function to read and analyse reference images, including landmark identification and Euclidean distance calculation. ● Implemented logic for generating correction suggestions based on Euclidean distances.
Ayush Patel	● Integrated OpenCV and MediaPipe libraries for real-time pose detection. ● Developed the logic for capturing video frames and extracting relevant pose information.
Rutul Patel	● Developed functionalities for drawing landmarks and highlighting deviations on the video stream. ● Optimised the video processing and display pipeline for smooth performance.
Parv Patel	● Designed and developed the user interface for all user-facing functionalities except login and registration. ● Debugged and resolved all front-end errors from styling side and HTML side, ensuring optimal functionality and responsiveness across diverse devices.
Sankalp Patel	● Designed and implemented backend APIs for uploading user data and reference images. ● Defined and implemented data structures and protocols for seamless communication between the backend and frontend components. ● Played a key role in troubleshooting and resolving backend issues, ensuring the smooth operation of the system.

Nish Parikh	• Developed interactive elements like the selection of 4 default dance forms and the "Upload Reference Image" functionality, allowing users to choose their desired pose estimation scenarios. • Debugged and tested the user interface thoroughly, resolving functionality issues and enhancing user experience. • Implemented logic for data transfer between both systems, ensuring accurate and timely data delivery.
Vanaja Agarwal	• Crafted a user-friendly and visually appealing user dashboard, facilitating a seamless user experience. • Identified and diagnosed camera-related bugs that hampered the pose estimation process. • Played a crucial role in bridging the gap between the frontend user interface and the backend system.

Drive Link :

Drive

References:

- https://www.analyticsvidhya.com/blog/2022/01/a-comprehensive-guide-on-human-pose-estimation/
- https://ijcrt.org/papers/IJCRT2110194.pdf
- https://www.researchgate.net/publication/343088066_The_Progress_of_Human_Pose_Estimation_A_Survey_and_Taxonomy_of_Models_Applied_in_2D_Human_Pose_Estimation
- https://mobidev.biz/blog/human-pose-estimation-technology-guide
- https://www.cs.sjsu.edu/~bruce/fall_2016_cs_160_lecture_active_shape_modelling_ASM_and_active_appearence_modelling_AAM.html
- https://www.v7labs.com/blog/human-pose-estimation-guide
- https://towardsdatascience.com/using-hourglass-networks-to-understand-human-poses-1e40e349fa15
- https://syncedreview.com/2020/07/27/adobe-and-stanford-unveil-sota-method-for-human-pose-estimation/