

**COMPUTER VISION TECHNOLOGY TO ASSIST IN
REHABILITATION AND TRAINING**

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology
in
Computer Science and Engineering with
spec. in Blockchain Technology

by

ADWAITA RAJ MODAK
20BKT0095

Under the guidance of

Prof. NIHA K

*Assistant Professor Senior Grade 1,
SCOPE, VIT, Vellore*



MAY, 2024

DECLARATION

I hereby declare that the thesis entitled "*COMPUTER VISION TECHNOLOGY TO ASSIST IN REHABILITATION AND TRAINING*" submitted by me, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering with specialization in Blockchain Technology* to VIT is a record of bonafide work carried out by me under the supervision of Prof. Niha K.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date : 08/05/2024

Adwaita
Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled “Computer Vision Technology To Assist In Rehabilitation and Training” submitted by Adwaita Raj Modak (20BKT0095) , SCOPE, VIT, for the award of the degree of *Bachelor of Technology in Computer Science Engineering with specialization in Blockchain Technology*, is a record of bonafide work carried out by him / her under my supervision during the period, 05.01.2024 to 08.05.2024, as per the VIT code of academic and research ethics.


The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 08 / 05 / 2024


Signature of the Guide


Internal Examiner


External Examiner

Dr. Gopinath M P

Department of Information Security, SCOPE, HoD

ACKNOWLEDGEMENT

This project is really challenging for me, and the entire implementation would not have been possible without the support of my guide, Prof. Niha K.. I would also like to express my appreciation to my Dean (Prof. Ramesh Babu K) and Hod (Prof. Gopinath M. P.) for providing me with all the help and support and guidance required for the research. I came across new technologies and gained a lot of knowledge in the field of Computer vision usage in medical fields, during the course of this project. I would like to thank each and every one for inspiring me and encouraging me to overcome all challenges. The research and implementation introduced me to cutting edge technology and helped me dive into the fields of Artificial Intelligence that would surely help me in my future work.

ADWAITA RAJ MODAK

Student Name

Executive Summary

The project aims to develop a real-time posture recognition and correction system that aids patients in completing workouts related to their ailment. This system uses computer vision and machine learning techniques to provide patients with automatic, precise, and real-time feedback while performing yoga. The system gathers accurate and inaccurate postures related to the chosen illness or condition, trains a posture detection model using this dataset, and guides the patient through recommended workouts. The system then acquires landmarks on the user's body to form a rough skeletal structure consisting of 33 points covering the entire body. The system calculates angles between certain parts of the body to determine if the yoga is being performed accurately.

The device continuously evaluates the patient's motions throughout yoga sessions and compares them to predetermined proper positions. An alert system is set off when deviations are found, providing instant input on how to fix their form. This feedback system ensures timely instruction for safe and efficient activities.

The user interface provides real-time feedback on yoga technique and progress, enabling patients to take charge of their own rehabilitation. The system aims to prove its effectiveness in enhancing yoga adherence, form correctness, and patient outcomes through extensive testing and validation, including user trials with patients and healthcare professionals.

SRL NO.	CONTENTS	Page No.
	Acknowledgement	iv.
	Executive Summary	v.
	Table of Contents	vi.
	List of Figures	vii.
	List of Tables	ix.
	Abbreviations	ix.
1	INTRODUCTION	1
	1.1 Objectives	1
	1.2 Motivation	1
	1.3 Background	2
2	PROJECT DESCRIPTION AND GOALS	2
	2.1 Survey on Existing System	2
	2.2 Research Gap	6
	2.3 Problem Statement	7
3	TECHNICAL SPECIFICATION	8
	3.1 Requirements	9
	3.1.1 Functional	9
	3.1.2 Non-Functional	9
	3.2 Feasibility Study	10
	3.2.1 Technical Feasibility	10
	3.2.2 Economic Feasibility	10
	3.2.3 Social Feasibility	10
	3.3 System Specification	11
	3.3.1 Hardware Specification	11
	3.3.2 Software Specification	11
	3.3.3 Standards and Policies	11
4	DESIGN APPROACH AND DETAILS	12
	4.1 System Architecture	12
	4.2 Design	13
	4.2.1 Data Flow Diagram	13

	4.2.2 Use Case Diagram	14
	4.2.3 Class Diagram	15
	4.2.4 Sequence Diagram	16
	4.3 Constraints, Alternatives and Tradeoffs	17
5	SCHEDULE, TASKS AND MILESTONES	19
	5.1 Gantt Chart	19
	5.2 Module Description	22
	5.2.1 Module – 1	22
	5.2.2 Module – 2	23
	5.2.3 Module - 3	25
	5.3 Testing	26
	5.3.1 Unit Testing	26
	5.3.2 Integration Testing	28
6	PROJECT DEMONSTRATION	30
7	RESULT & DISCUSSION	34
8	SUMMARY	39
9	REFERENCES	40
	APPENDIX A – SAMPLE CODE	41

List of Figures

Figure No.	Title	Page No.
1.1	Latency of BlazePose GHUM with 3 Methods on Different Platforms	9
1.2	System Architecture	13
1.3	Data Flow Diagram	14
1.4	Use Case Diagram	15
1.5	Class Diagram	16
1.6	Sequence Diagram	17
1.7	Timeline Gantt Chart	20
1.8	Landmark map drawn on the body along with calculated angles displayed	24
1.9	Arctan2 of Two Vectors to Get the Angle	25
2.0	Video Feed Captured using OpenCV	27
2.1	Landmark Map drawn on the body	27
2.2	Body points on a 3-D plane	28
2.3	Angle calculated at the elbow joint	29
2.4	Starting screen	30
2.5	Landmarks Map of Google Media Pipe Pose	32
2.6	Landmark map formed on the body	33
2.7	Confusion Matrix for Adho Mukha Savasana	36
2.8	Confusion Matrix for Balasana	36
2.9	Correct Pose for Adho Mukha Savasana	37
3.0	Incorrect Pose for Adho Mukha Savasana	38
3.1	Correct Pose for Balasana	38
3.2	Incorrect Pose for Balasana	39

List of Tables

Table No.	Title	Page No.
1.1	Analysis test results	35

List of Abbreviations

AI	Artificial Intelligence
FPS	Frames Per Second
ML	Machine Learning
RGB	Red-Green-Blue
YOLO	You Only Look Once
CPU	Central Processing Unit
GPU	Graphics Processing Unit
SDK	Software Development Kit

1. INTRODUCTION

1.1. OBJECTIVE

The aim of this project is to use computer vision technologies to construct a real-time posture identification and correction system. By helping patients complete their training and rehabilitation activities precisely and securely, this system hopes to improve the therapeutic results. The initiative aims to enable patients to take control of their own rehabilitation while guaranteeing appropriate exercise practices by offering automated supervision and feedback. The need for a technology that can let patients manage their own exercise regimens and rehabilitation program independently while providing the guidance and feedback they need to complete their activities correctly without any human supervision. Thus, the goal is to make rehabilitation an automated process and help the person recover quickly from his ailment. In several applications, including full-body gesture control, sign language recognition, and workout quantification, human position estimate from video is essential. It can serve as the foundation for applications in yoga, dance, and fitness, for instance. It can also make it possible to use augmented reality, which is the superimposition of digital data and content on top of the actual environment. MediaPipe posture is a machine learning solution for high-fidelity body posture tracking.

1.2. MOTIVATION

The goal of computer vision, a branch of computer science, is to make it possible for machines to recognize and comprehend objects and people in pictures and movies. Similar to other forms of artificial intelligence, computer vision aims to execute and mechanize activities that mimic human skills. In this instance, computer vision aims to mimic human eyesight as well as human perception and interpretation of visual information.

Computer vision applications mimic the functioning of the human visual system by using data from sensing devices, artificial intelligence, machine learning, and deep learning. Algorithms used in computer vision applications are trained on vast volumes of visual data or cloud pictures. They identify patterns in this visual input and make inferences about the meaning of other images based on those patterns.

Google created the open-source MediaPipe framework, which lets programmers construct multimedia apps in real time for a range of platforms and gadgets. Advanced computer vision techniques and machine learning models are included, and they are utilized for tasks like position estimation, object identification, and facial recognition. These models are perfect for multimedia applications since they are real-time and very accurate. Additionally, MediaPipe has real-time performance-optimized computer vision algorithms for applications like segmentation, depth estimation, and image and video stabilization. Because the platform's pipeline is modular and easily extendable with custom components, developers may create highly customized applications using it.

1.3. BACKGROUND

In general, MediaPipe is a great option for programmers creating real-time multimedia applications. It leverages the BlazePose research, which drives the ML Kit Pose Detection API, to infer 33 3D landmarks on the whole body from RGB video frames. Real-time posture identification and correction through the use of machine learning models and sophisticated computer vision methods. Creating and executing an intuitive user interface to lead patients through prescribed exercise regimens.

Using patient and healthcare professional user trials to test and validate the system's efficacy. Concentrating on certain medical issues, such as spinal disabilities, in order to customize the system's operation to meet the requirements of various patient populations.

2. PROJECT DESCRIPTION AND GOALS

2.1 Survey on Existing Systems

[1] Body Posture Detection and Motion Tracking using AI for Medical Exercises and Recommendation System.

The study suggests using artificial intelligence (AI) and image processing to enhance and augment self-supervised exercise by creating a motion tracker that is software-

based and can monitor activities and offer postural feedback. Plotting points at different body joints and monitoring movement are done using the Media Pipe framework to provide a thorough study of body tracking. This body tracking analysis may be used to create an application that monitors registered persons' medical exercise, increasing the effectiveness of the workouts. By utilising databases, the programme may be further enhanced to match registered users with approved physicians who have access to diagnostic results and workout histories.

[2] Scoliosis: Review of diagnosis and treatment.

The paper discusses the diagnosis and treatment of idiopathic scoliosis, a common type of spine deformity. The significance of a patient's age, appearance, and natural history in determining the best course of therapy is emphasized. Numerous variables, including congenital, neuromuscular, syndrome-related, idiopathic, and secondary causes, such as spinal curvature, can result in scoliosis. Treatment for scoliosis is determined by the kind, severity of the curvature, amount of growth left, and patient's viewpoint. Determining the likelihood of curve development and the prognosis of therapy depends heavily on growth tracking using maturity markers and height measurements.

[3] Adolescent Idiopathic Scoliosis: Diagnosis and Management.

Two percent to four percent of teenagers have adolescent idiopathic scoliosis. Regular screening is not advised because of the risks. Radiologic testing might be guided by physical tests. Braces, surgery, and observation are available as forms of treatment. The onset and course of scoliosis are influenced by genetic factors. There is little data to support therapies like physical therapy and chiropractic adjustments. Patients who go untreated frequently have minimal physical disability.

[4] Media Pipe: A Framework for Building Perception Pipelines.

The MediaPipe framework is presented in this paper, addressing the difficulties in creating applications that take in their environment. Building prototypes, measuring system performance and resource consumption on target platforms, and choosing and developing machine learning models and algorithms are all made possible by the framework. Using MediaPipe, developers can build sophisticated cross-platform applications from scratch, integrate pre-existing perceptual components, and iterate enhance their apps to provide repeatable outcomes on several platforms and devices.

The study emphasizes how MediaPipe is now open-sourced, allowing programmers to utilize it as a development environment for algorithms and models.

[5] A Survey on Computer Vision for Assistive Medical Diagnosis From Faces.

The paper examines current developments in computer vision techniques for face image analysis used in medical diagnosis. It highlights the significance of multidisciplinary cooperation and the gathering of databases that are open to the public for future study. Additionally covered in the report is the applicability of various imaging modalities for diagnosing medical complaints. The lack of training data, however, continues to restrict the application of deep learning techniques and neural networks in clinical settings.

[6] Deep learning-enabled medical computer vision.

The study examines the developments in deep learning-driven computer vision methods for medical usage, with a particular emphasis on convolutional neural networks (CNNs) in imaging specialties such as ophthalmology, pathology, cardiology, and dermatology. It draws attention to the potential of AI models for things like reconstruction, retrieval, and picture registration. The study also demonstrates how AI algorithms might revolutionize oncology and pathology by employing digital pathology picture archives and annotations for diagnosis, prognosis, and prognosis. The usage of sub-micron tissue scanners, as well as their development and commercialization, are examined in this paper's discussion of AI-driven digital histopathology.

[7] Automated Body Postures Assessment From Still Images Using MediaPipe.

Body postures, stance, landmarks, and the use of MediaPipe and machine learning are the main topics of this work. Regardless of the user's and reference's age, gender, or picture and video backgrounds, the system functions effectively. Vertex angles, arm angles, wrist angles, and foot angles are among the aspects for the right side of the body that are provided in the article. The system is unaffected by variations in age, gender, backdrop, or image size. The suggested structure for automated evaluation of body posture using MediaPipe shown encouraging outcomes in side-view scenarios like standing and sitting. By identifying variations in the user's posture from the reference still image, the device functioned as a smart mirror, giving real-time corrective information. The technique was used to help people

avoid bad standing and sitting postures that might have unfavorable effects on their bodies, as well as to rectify their own posture. When the outcomes from the proposed framework were compared to judgements made manually, they were similar. However, there were some observations about the identification of neck and torso bending, the necessity of camera coordinate alignment, and the significance of choosing the reference image.

[8] Automatic detection of abnormal hand gestures in patients with radial, ulnar, or median nerve injury using hand pose estimation.

The study employs hand posture estimation to automatically identify aberrant hand gestures in patients with radial, ulnar, or median nerve damage. Using a smartphone, pictures of both healthy participants and patients with nerve damage were collected. Google MediaPipe Hands was used to extract landmark coordinates so that the characteristics could be calculated. The study examined how well rule-based models performed in comparison to machine learning methods such as random forest, logistic regression, and support vector machines.

According to the findings, the rule-based models were very successful in identifying injuries to the radial, ulnar, and median nerves with respect to accuracy, sensitivity, and specificity. Furthermore, all machine learning models exhibited sensitivity ranging from 37.5% to 100% and accuracy above 95%. The results imply that hand position estimation may be an easy-to-use screening technique for telemedicine and basic healthcare.

[9] Fitness Movement Types and Completeness Detection Using a Transfer-Learning-Based Deep Neural Network.

This research presents a deep transfer learning-based approach to build a fitness database and train a deep neural network to identify the kind and degree of fitness motions. Yolov4 and MediaPipe were utilized by the authors to quickly identify fitness motions, and they saved the 1D fitness signal of movement in order to create a database. With a high mAP of 99.71%, accuracy of 98.56%, and F1-score of 98.23%, the fitness movement types categorization performance was accomplished. The F1-score was 92.83%, accuracy was 92.84%, and precision was 92.85 when it came to the assessment of fitness movement completeness. Classifying the completeness of a fitness action is another area where the technique excels. According to experimental data, the suggested strategy performs more accurately

than alternative approaches.

[10] A Video Image Processing Method for Continuous Object Detection.

The research focuses on optimising video image post-processing for continuous object recognition. It tackles issues including interference between small units of many objects, variables that degrade picture quality, and skew/distortion of images that arise while recognising continuous objects in complicated settings. The suggested technique in the research makes use of past information to increase detection efficiency and accuracy. With regard to scene-based optimisation and object identification in various business settings, the research attempts to address the growing demands for continuous identifying objects.

2.2 Gaps Identified

[1] The effectiveness of the proposed system relies on the accuracy and reliability of the body posture detection and motion tracking algorithms. Any inaccuracies or errors in these algorithms may lead to incorrect feedback or recommendations.

[2] Does not discuss the controversies surrounding the treatment of scoliosis in specific patient groups, leaving some gaps in understanding the optimal approach for these cases.

[3] There is little data to support therapies like physical therapy and chiropractic adjustments. Patients who go untreated frequently have minimal physical disability.

[4] As the paper is an introduction to the MediaPipe framework, it does not provide in-depth analysis or evaluation of the framework's performance or comparison with other similar frameworks. Additionally, the paper does not discuss any specific challenges or issues that developers may face while using the MediaPipe framework.

[5] The lack of training data, however, continues to restrict the application of deep learning techniques and neural networks in clinical settings.

[6] It does not discuss the limitations or potential drawbacks of using deep learning in medical computer vision.

[8] The paper does not provide information on the sample size used for each classification task, which could affect the generalizability of the results. The paper does not mention the specific features used for the rule-based models, making it difficult to replicate the study or understand the rationale behind feature selection. The paper does not discuss the limitations of using smartphone images for hand pose estimation, such as potential variations in image quality or the need for standardized image capture protocols. The paper does not provide information on the demographic characteristics of the study participants, such as age, gender, or occupation, which could influence the generalizability of the findings.

[9] The paper does not provide a detailed discussion on the limitations of using wearable devices for fitness movement detection, which could have helped in understanding the drawbacks of this approach. The drawbacks of body-node-based and image-deep-learning-based methods for fitness movement detection are mentioned, but the paper does not provide a comprehensive analysis of these limitations. There is no mention of the potential limitations or constraints of using YOLOv4 and Mediapipe for instant detection of fitness movements

[10] A Video Image Processing Method for Continuous Object Detection. The research focuses on optimising video image post-processing for continuous object recognition. It tackles issues including interference between small units of many objects, variables that degrade picture quality, and skew/distortion of images that arise while recognising continuous objects in complicated settings. The suggested technique in the research makes use of past information to increase detection efficiency and accuracy. With regard to scene-based optimisation and object identification in various business settings, the research attempts to address the growing demands for continuous identifying objects.

2.3 Problem Statement

- Take input from the user regarding the disease and suggest exercises for rehabilitation.
- Capture video feed to get the posture of the user and convert it into frames to form the landmark on the human body.

- Calculate the angle formed between certain pin points to ensure that the activity is performed properly, and if there is an error in the posture then voice alert message is generated otherwise accuracy and count of repetitions is displayed.

Goals :

- Gather sufficient dataset and in case limited number of data is available then data augmentation needs to be performed in order to standardize the number of data.
- Develop a model that reads images as input from folders and runs pose estimation on them to determine the angles formed by certain joints and store them in a csv file.
- Training and testing the data for developing the model.
- Develop a small framework to accept the choice of yoga from the user, and triggering the required function at the backend.
- Through video input source, live feed will be captured and the methods and classes from mediapipe will be run on them for estimating the pose.
- Trigonometric functions will be used to calculate the angles between any three body points at a time.
- The calculated angles will be compared to the pre-determined angles and based on this the desired output will be displayed.

3. TECHNICAL SPECIFICATION

A computer vision system using Google's MediaPipe technology for rehabilitation and training requires detailed technical specifications covering technologies, hardware requirements, software specifications, user interaction, and expected functionalities. In this section the requirements, technical, economic and social feasibility have been discussed. This comprehensive breakdown is essential for the project's success. In order to guarantee high accuracy, an excellent user experience, real-time performance, and dependability, the paper explores both functional and non-functional needs. The feasibility assessment confirms the project's viability and revolutionary potential by addressing its technical, economic, and social components. Along with standards and procedures to guarantee adherence to data protection and healthcare legislation, comprehensive hardware and software specifications are offered to steer development.

When handling video streams, including camera inputs, Google MediaPipe Pose with BlazePose GHUM [Fig – 1.1] has better accuracy and faster processing speed than YOLO-Pose. Nonetheless, for a single perspective photograph, the YOLO-Pose may estimate many people's poses. Given that the Google MediaPipe Pose would perform better on video streams and process information quicker across a variety of platforms, including PCs and smartphones, we should pay closer attention to it in this scenario.

Method	Latency	Latency
	Pixel 3 TFLite GPU	MacBook Pro (15-inch 2017)
BlazePose GHUM Heavy	53 ms	38 ms
BlazePose GHUM Full	25 ms	27 ms
BlazePose GHUM Lite	20 ms	25 ms

Fig – 1.1 : Latency of BlazePose GHUM with 3 Methods on Different Platforms

3.1 Requirements

3.1.1 Functional

- Disease and its corresponding exercise plan selection: The system takes the disease as input from user and suggests a list of exercises for rehabilitation. The user is required to make the selection and accordingly the corresponding function will be triggered in the backend.
- Real-time posture recognition and correction: a system that is able to detect and correct the posture of the user in real-time during exercise session.
- Feedback Mechanism: A system to provide proper feedback to the user to keep them updated if they are performing the exercise correctly or if their posture is incorrect.

3.1.2 Non-Functional

- Accuracy: The posture detection and recommendation system should have a high accuracy to be able to provide reliable guidance and feedback to the users.
- User Experience: The UI should be easy to use and intuitive, so that any kind

of user is able to easily interact with the application.

- Real-time performance: The system has to perform in real-time to produce instantaneous feedback to the users during the exercise sessions.
- Reliability: While in operation, the system should be reliable with minimum error and downtime.

3.2 Feasibility Study

3.2.1 Technical Feasibility

- The concept has a high technological viability as it makes use of machine learning and current computer vision technologies (the MediaPipe framework) for posture detection and correction.
- The concept is technically feasible since hardware components like cameras and computers with enough processing power are readily available.

3.2.2 Economic Feasibility

- A number of variables, including the price of software development, hardware components, and possible licencing costs for using third-party frameworks, affect the project's economic viability.
- The idea may, however, be worth it in the long run since computerised rehabilitation procedures might lead to better patient outcomes and lower healthcare expenses.

3.2.3 Social Feasibility

- The initiative tackles the increasing significance of physical therapy in the management of diverse medical disorders from a social standpoint.
- The idea encourages patient empowerment and autonomy in their rehabilitation process by giving them automated instruction and feedback during exercise sessions.

3.3 System Specification

3.3.1 Hardware Specification

- Hardware for the system, such as cameras that can record clear video feeds for posture recognition and correction, is needed.
- Moreover, in order to execute the posture detection model and furnish the user with immediate feedback, computing devices possessing adequate processing capabilities are essential.
- *System Requirement: Operating System:* Windows 11, *CPU:* Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz, *RAM:* 24 GB, and *GPU:* NVIDIA GeForce GTX 1650 Ti.
- High resolution webcam or integrated camera capable of capturing at least 720p video at 30 FPS.

3.3.2 Software Specification

- The creation of the posture detection and correction algorithm utilising machine learning methods and the MediaPipe architecture is part of the software specs.
- To enable communication between the user and the system, an intuitive user interface must be created and put into use.
- *Software Requirements :* python \geq 3.8.0, mediapipe 0.8.9.1, opencv-python 4.5.5.62, pandas 1.4.1, and scikit-learn 1.0.2.
- *Development Environment* - PyCharm, Visual Studio Code, or any python supportive IDE.

3.3.3 Standards and Policies

- The project abides by all applicable laws, regulations, and guidelines regarding data security and privacy, particularly while gathering and using user data for exercise sessions.
- It might also be needed to comply with the healthcare guidelines and regulations, specifically if the system is being used for public service in the clinics.

4. DESIGN APPROACH AND DETAILS

The main idea of this project design is to extract and convert the MediaPipe pose estimation into certain parameters that can be processed in a readable way to fetch valuable output.

The angles of the primary body landmarks will be the primary parameters extracted from the MediaPipe Pose Estimation, taking into account the fact that each of us has a unique body form and length. Various diagrams have been drawn in order to make the flow of the entire project errorfree, and also to enable everyone to have a seamless user experience and understanding. The diagrams will be providing information about how the system has been designed, the workflow, how the data is flowing in the program. Through the class diagram we can come to know how the different functions and methods have been used in the execution of this program.

4.1. System Architecture

The execution starts with the user selecting his ailment and the yoga that he/she intends to perform. The video feed is processed in the pre-processing stage where it is broken into multiple frames and converted into RGB mode, so that Media Pipe can carry out all pose estimations easily. The processed video feed is fed to MediaPipe, that uses certain member methods and classes to acquire all the landmark points on the human body. Based on the pin points determined coordinated are assigned to them. A trigonometric function is used to calculate the angle formed at a joint using a set of three points by using the `arctan2()` function. This data is then verified with the previously obtained data to check if the yoga is being done correctly or not [Fig – 1.2].

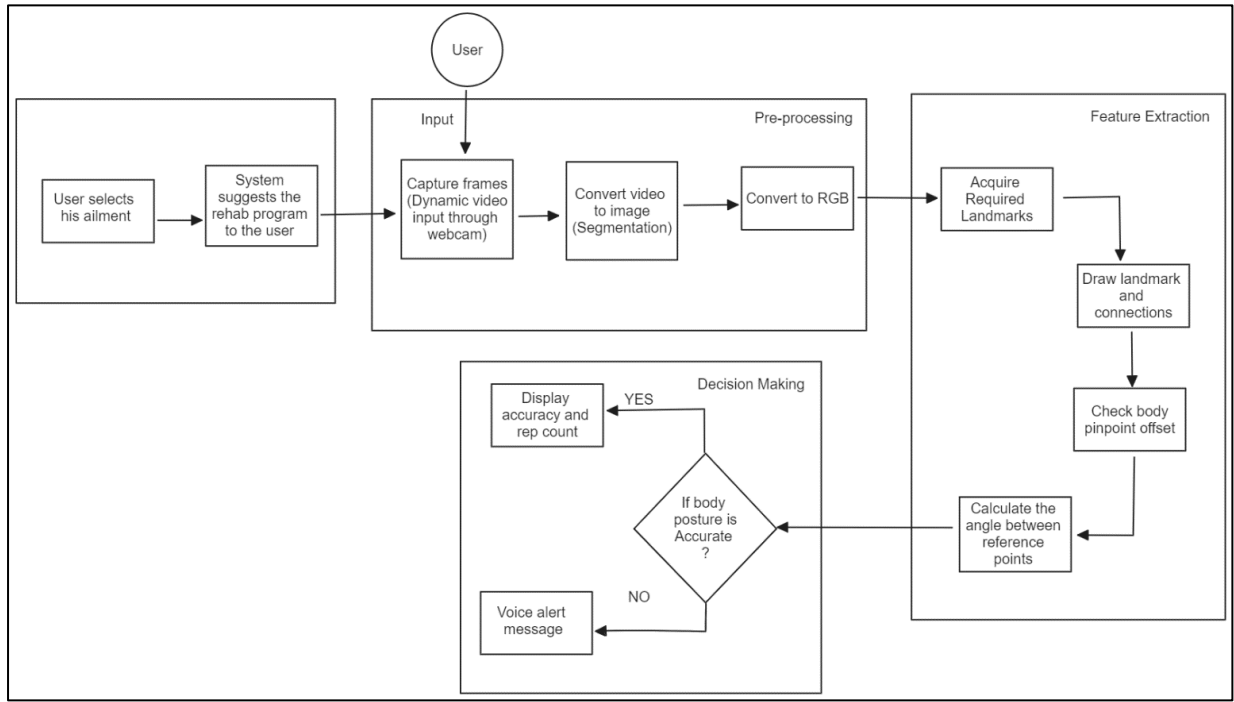


Fig - 1.2 : System Architecture

4.2. Design

4.2.1. Data Flow Diagram

A data flow diagram (DFD) shows the movement of data through a system or process in graphic form. DFDs display the inputs, outputs, storage locations, and pathways between each destination using symbols and text labels. In this project, the flow of data happens in the following format. The user makes his selection, the corresponding function is triggered in the backend, and the video feed dialog box pops up that captures the video through an input source and also performs all processing on it to provide suitable results to the user. The angles are calculated and the correctness of each value is checked and the result is displayed on the screen to the user [Fig – 1.3].

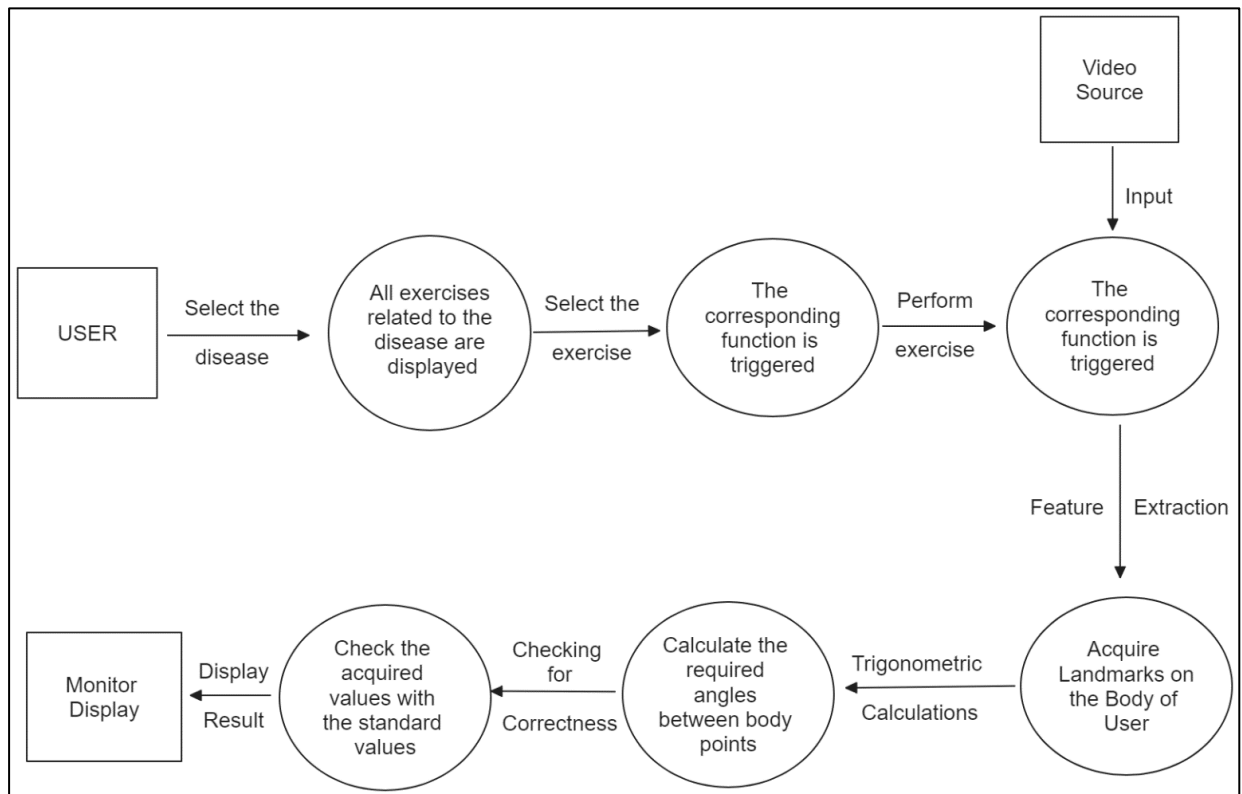


Fig – 1.3 : Data Flow Diagram

4.2.2. Use Case Diagram

An illustration of a user's possible interactions with a system is called a use case diagram. It illustrates the roles played by the players in using the system's functionalities using a particular notation. Stick figures are used to depict actors in use case diagrams, while circles or ellipses are used to illustrate use cases. Actors are outside entities that work with the system to accomplish particular objectives. In this project, the actors include the user, webcam, and an external display. The user avails a certain set of functionalities which include choice selection, pre-processing functionalities. The webcam is responsible for capturing live video feed and the external monitor displays the output after all checks for correctness are carried out [Fog – 1.4].

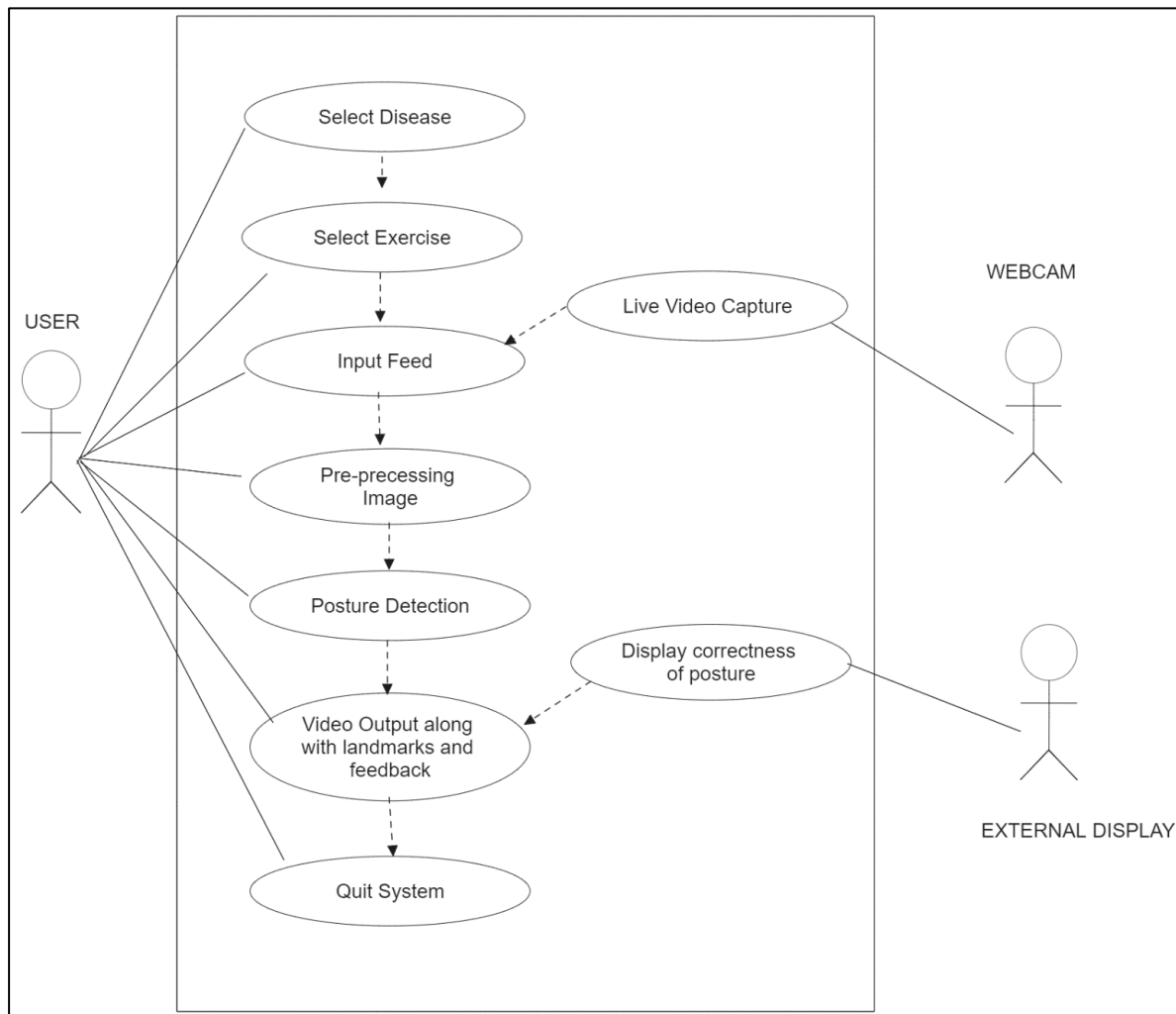


Fig – 1.4 : Use Case Diagram

4.2.3. Class Diagram

An application's static view is shown in the class diagram. It depicts the many kinds of things found in the system as well as their connections to one another. A class can inherit from other classes in addition to consisting of its own objects. An executable software code may be created and many parts of the system can be visualised, described, and documented using a class diagram. In Fig [1.5] the class diagram shows the different member methods and member variables that come under the different objects, and how the flow of process happens from one module to another. the various member variables are used as parameters to accept values or for other computations. The member functions denote the classes or functions that use the member variables to perform respective functions.

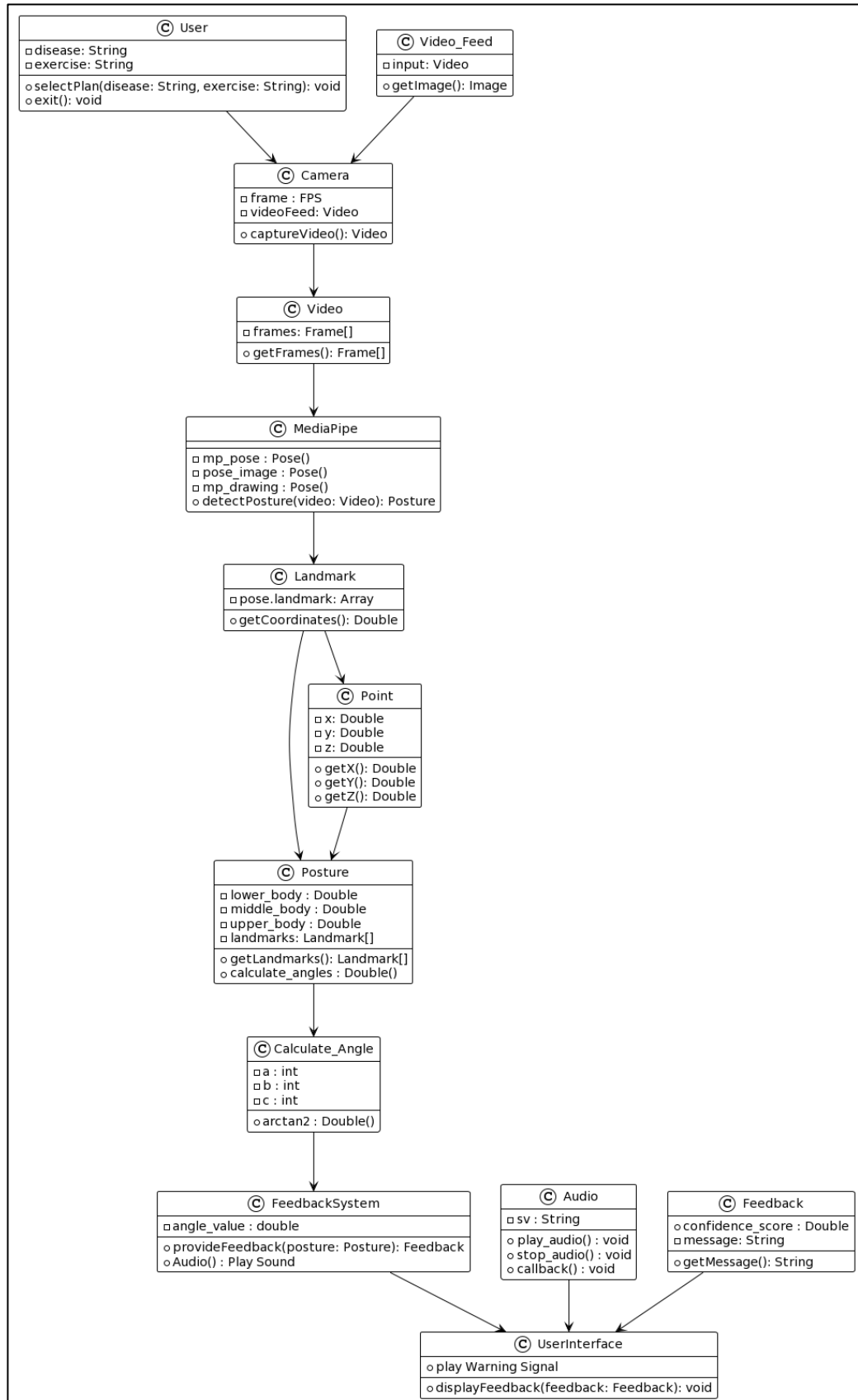


Fig – 1.5 : Class Diagram

4.2.4. Sequence Diagram

A sequence diagram is a diagram created using the Unified Modelling Language (UML) that shows the messages that are sent back and forth between objects during an interaction. A sequence diagram shows a collection of items, symbolised by lifelines, together with the messages they exchange during the course of their relationship. A sequence diagram displays the messages that are sent back and forth between objects. Diagrams of sequences can also display the control networks among items. Fig [1.6] shows the sequence in which the different operations are carried out. The values generated in one function, that might be required by another function are also shown here.

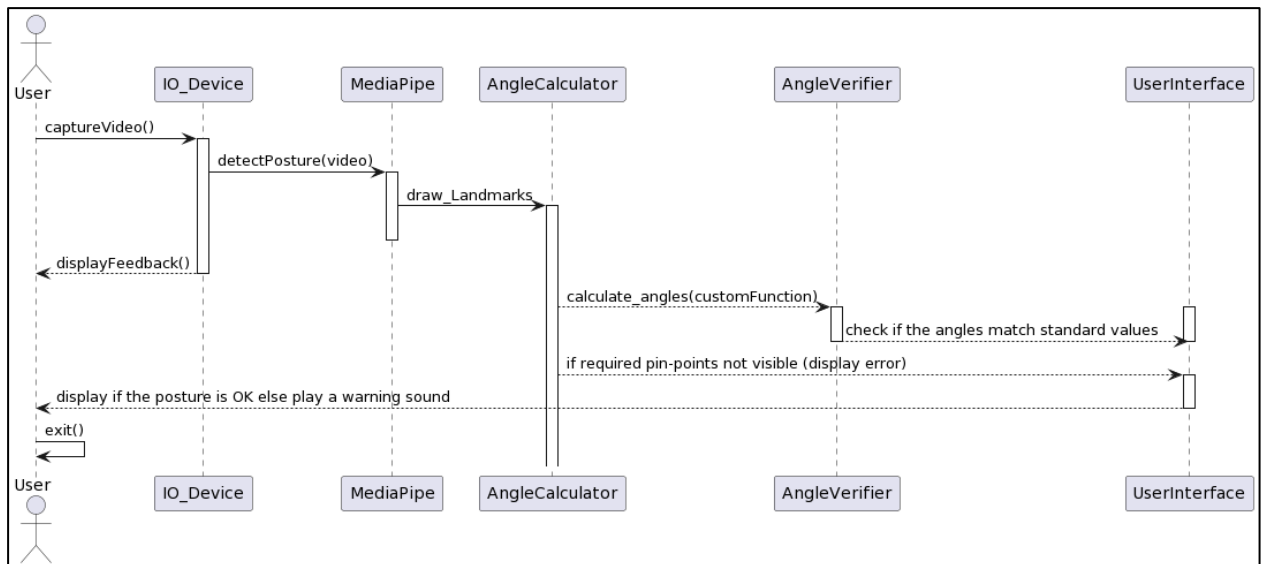


Fig – 1.6 : Sequence Diagram

4.3. Constraints, Alternatives

Constraints

- Environmental Dependence - Pose estimate accuracy might vary greatly depending on the backdrop, illumination, and camera characteristics. Inadequate illumination or intricate backdrops may hinder the system's precision in identifying crucial locations.
- Hardware Requirements - Considerable processing power is needed for real-time posture estimation, which may be limited on mobile devices or low-end hardware,

perhaps resulting in latency or decreased frame rates.

- Optimization needs - For real-time applications in particular, it might be needed to further optimise the model to achieve a balance between speed and accuracy, depending on how complicated the application is.
- Data Sensitivity - Human pose estimation applications may give rise to privacy problems, particularly in cases where recordings, storage, or transmission of photos or videos are involved.

Alternative

- YoloV7 is another pose detection model that can be used but has a certain number of drawbacks as compared to Media Pipe.
- Media Pipe is mainly used for pose detection of a single person, whereas Yolo can detect multiple people in a frame, but that is not the purpose of this project. MediaPipe has 33 landmark points, whereas Yolo has only 17 landmark points. Yolo has 960p default input size as compared to Media Pipe's 256p x 256p, which is well trained for smaller images.
- Yolo requires GPU for smooth operation, whereas Media Pipe works well on CPU.
- Yolo flickers a lot while in use, as compared to MediaPipe that works without much ambiguity.

5. SCHEDULE, TASKS AND MILESTONES

5.1. Gantt Chart

ID.	TASK NAME	START DATE	END DATE	Duration(in days)
1	Deciding the Topic	01-10	01-14	5
2	Research	01-15	01-30	16
3	Requirement (Functionality, Overview and Documentation)	02-01	02-07	7
4	Analysis	02-08	02-14	7
5	Design (System Design and Diagrams)	02-18	02-29	12
6	Pre-processing	03-01	03-09	9
7	Feature Extraction	03-11	03-21	11
8	Decision Making	03-23	04-02	11
9	Coding	03-05	03-31	27
10	Training and Testing	04-07	04-21	15
11	Maintenance	04-22	04-30	9

TASKS

- Literature Survey, Research and Analysis - Extensive research was carried out to determine the project topic, and all major related research articles were thoroughly surveyed and relevant data was gathered.
- Design – during the design phase, the system architecture, class , use-case, sequence diagrams were built for better understanding of the project and the workflow. Multiple sources were referred to while building the diagrams and UML softwares were used to build these diagrams.
- Pre-Processing – take input from the user regarding the disease and suitable yoga was suggested to him.
- Feature Extraction – Capture video feed to get the posture of the user and convert it into frames to form the landmark on the human body.
- Decision Making - a Calculate the angle formed between certain pin points to ensure that the activity is performed properly, and if there is an error in the posture then voice alert message is generated otherwise accuracy and count of repetitions is displayed. Detailed Gantt chart has been displayed below [Fig – 1.7].
- Coding and Implementation – writing codes for training and testing the data and then preparing the model for dynamic video feed implementation.

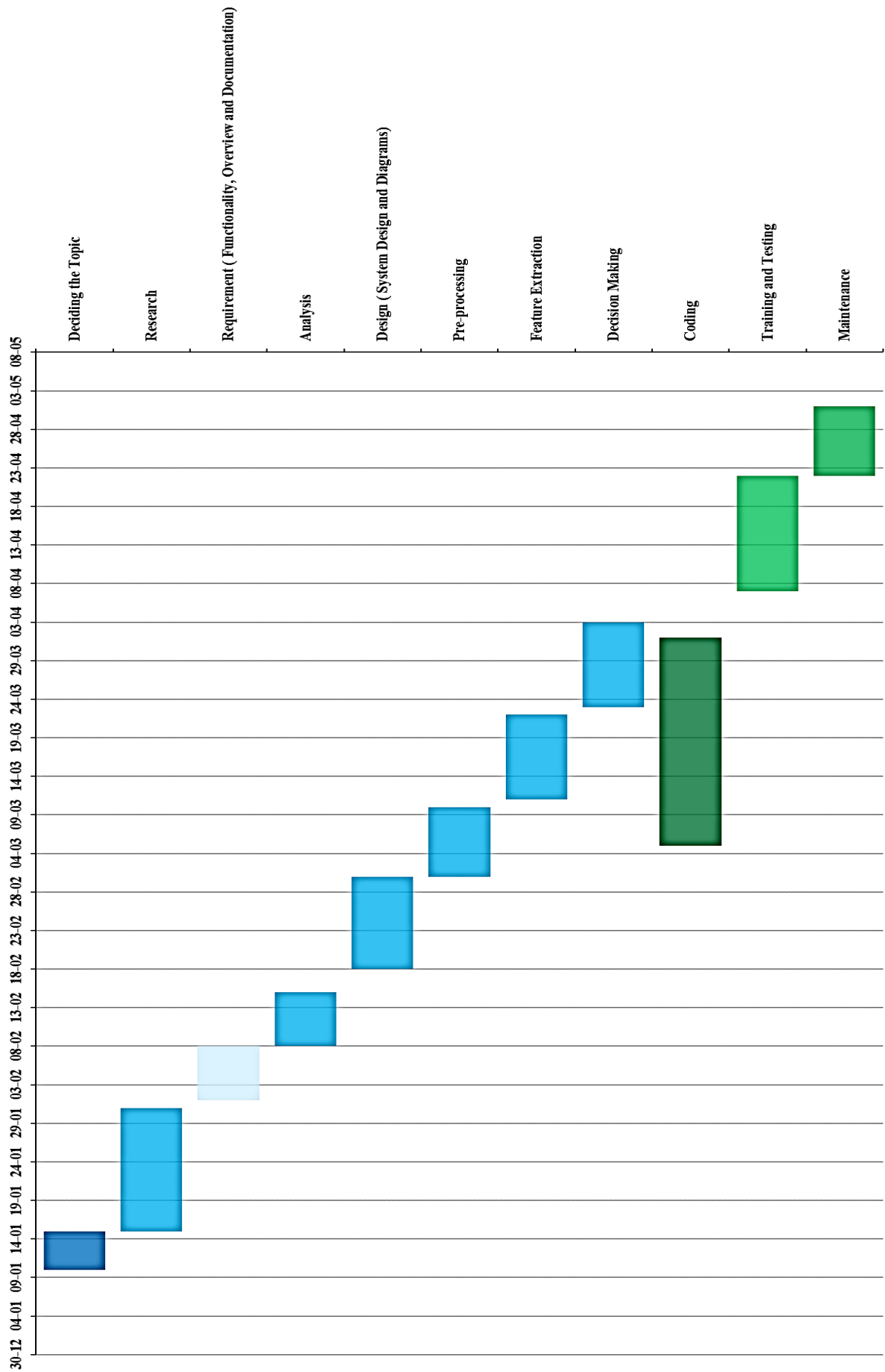


Fig – 1.7 : Timeline Gantt Chart

MILESTONES

Review – 1 (15/02/2024)

- The final project topic was proposed in this review.
- The objectives, introduction and problem statement were provided.
- A detailed literature survey was presented from research papers related to the topic of research.
- A detailed system architecture diagram was prepared to show the proposed workflow.

Review – 2 (03/04/2024)

- Aim and scope of the project were included in the introduction.
- The requirement analysis was framed, that consisted of the functional and non-functional requirements, then the technical, economical and social feasibility.
- The hardware and software specifications required for the model to run were provided.
- Design diagrams that were prepared for smooth operating of the project were included in the document.
- The code implementation was carried out for the project and the demo was shown to the panel members during the review.

Review – 3 (08/05/2024)

- The remaining 30% of the implementation was carried out and the thesis document was prepared.
- A detailed presentation was prepared for presenting to the panel members.
- Couple of additions and modifications were done to the code, certain graphical representations and characteristics were also included.

5.2. Module Description

5.2.1. Module - 1 (*Pre-Processing*)

Capturing Frames through Dynamic/Manual Input:

This module covers all the processes from starting the system to the formation of required image ready to be fed to the MediaPipe library for processing.

As the user opens the application, he is required to select his ailment and hit the submit button. A list of recommended and medically approved exercises will be displayed on the screen and the proper visuals will be provided if the user does not know how to perform the exercise correctly. Once the user selects the required exercise, the concerned function will be triggered. The webcam or any other video feed source used by the user, goes live and starts capturing live feed. The user will be required to stay within the frame of the camera. Real-time video frames are continuously captured by the camera from the webcam. At a frame rate of at least 30 frames per second (FPS), these frames are consecutive photographs of the scene taken by the camera at regular intervals.

In the case of training the model with dataset obtained from Kaggle (<https://www.kaggle.com/datasets/shrutisaxena/yoga-pose-image-classification-dataset>), I sorted out the list of exercises, performed data loading, on the images that were recommended for scoliosis. Since the size of the dataset was rather small, I had to perform data augmentation, and I increased the size of each image dataset to nearly 500 images per exercise. This enabled me to train my model on various angles and various image quality photos and get the best possible output.

Segmentation:

After the camera captures the video frames, then we have to segment the video into individual images frames. Each frame in the captured video represents an image snapshot captured at a specific point of time. Through this process, each frame is isolated so that it can be processes individually. Segmentation is the process of taking individual frames out of the video stream and saving them as different picture files. Each frame may be processed independently as this procedure, makes it possible to analyze, manipulate, and extract characteristics or information from the visual data.

Converting frames to RGB:

After segmentation, each individual frame is converted from its original format(usually YUV or BGR) to the RGB(Red-Green-Blue) color space. It is a standard color model that is used in digital imaging. Wherein each and every pixel is represented by the (red, green and blue) color channels. The RGB format conversion guarantees color representation consistency and compatibility with image processing algorithms and libraries that frequently employ RGB pictures as input data. Additionally, it makes the visualization and comprehension of the picture data for subsequent analysis simpler.

5.2.2. Module - 2 (Feature Extraction)

Acquiring landmarks using Media Pipe:

Setting up a new instance of the MediaPipe feed, it allows to access the pose estimation model. Then the detection and tracking confidence is passed to maintain the state, and after this the detections have to be rendered.

The MediaPipe module is used in this stage to identify and obtain landmarks on the pictures or live stream that the camera has recorded. MediaPipe uses deep learning models for position estimation, which allows it to recognize important landmarks that correspond to different body parts, including the torso, limbs, and joints. In order to analyze the user's posture during workouts, several markers are essential.

Drawing Landmark and connections on the video Feed :

After the landmarks are obtained, a graphical depiction of the detected stance is produced by visualizing them on the human stream (picture or live video). Usually, landmarks are shown as points on the picture that symbolize certain body sections. MediaPipe gives us certain drawing utilities to draw all the landmarks and connection on our body. All the landmarks are extracted and stored in a variable called landmarks. Then all the (x,y) coordinates need to be extracted from the landmarks array for each part of the body by passing certain parameters, and stored in respective variables so that they can be used to calculate the angles smoothly. In addition, the skeletal structure and position configuration are illustrated by drawing links between landmarks [Fig - 1.8].

Creating landmarks and links aids in the visualization of the detected pose and gives the user feedback on how the system is recognizing their body.

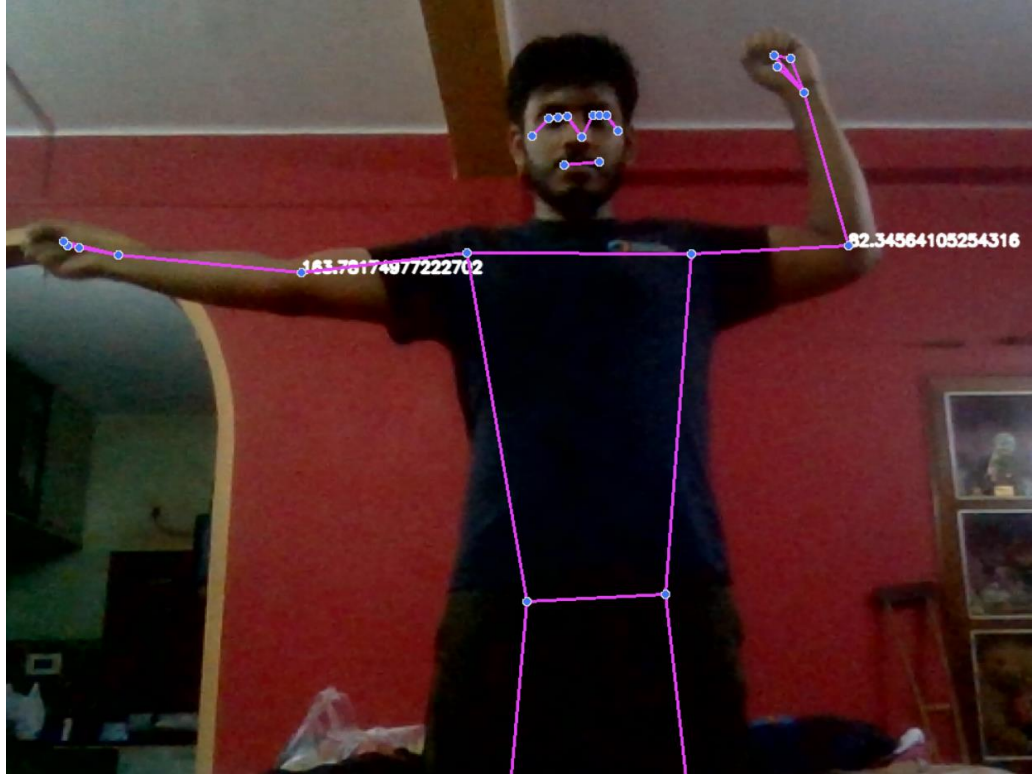


Fig - 1.8 : Landmark map drawn on the body along with calculated angles displayed

Checking body pinpoint offset:

The algorithm looks for any body pinpoint offset—that is, differences or inconsistencies between the detected landmarks and their intended positions—after generating landmarks and connections. To find any misalignments or errors in the posture estimate, this stage compares the discovered landmarks with pre-established reference points or templates. The utilization of body pinpoint offset detection contributes to the precision and dependability of the pose estimation procedure, allowing the system to provide accurate posture feedback to the user.

$$V_1 = \begin{bmatrix} w1 \\ w2 \end{bmatrix}$$

$$V_2 = \begin{bmatrix} v1 \\ v2 \end{bmatrix}$$

$$\Theta = \arctan2 (w_2v_1 - w_1v_2, w_1v_1 + w_2v_2)$$

Calculating angles between reference points:

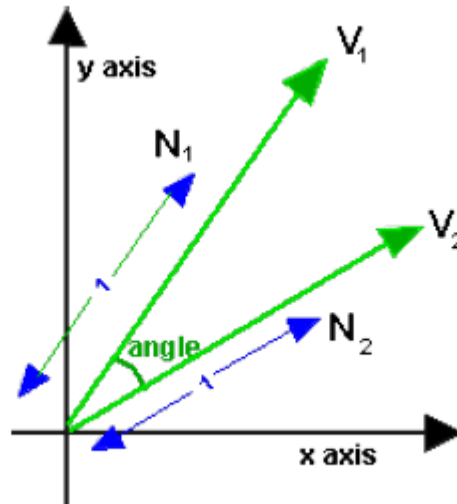


Fig - 1.9 : Arctan2 of Two Vectors to Get the Angle

In the end, the system determines the angles between various body reference points when an exercise is being done. This entails assessing the spatial correlations among particular landmarks in order to ascertain the alignment and orientation of body parts throughout the workout. Any number of three set coordinates that are being stressed during the exercise have to be passed to the `calculate_angles()` function that takes three coordinated at a time, as input and calculates the angle formed by them in radians and for any angle greater than 180 degree it subtracts that particular angle from 360 [Fig – 1.9]. This gives the angle that is being formed by those three parts of the body if they could be connected using 2 straight lines.

The system evaluates the user's posture, detects any deviations from the ideal form or alignment, and sends out an alarm to rectify the posture by computing the angles between reference points.

Accuracy obtained during testing – 91 %

5.2.3. Module – 3 (Decision Making)

In the last part, after feature extraction is done, a couple of if-else conditions are run and the obtained angles are checked with the pre-determined training values obtained beforehand. If the posture is not- correct a beep alert is audible or else if it is correct, the OK is displayed, and all this keeps on happening in real time.

The angles computed between various body reference locations are contrasted with training values or thresholds that have been previously established. These thresholds reflect the optimal angles for carrying out activities safely and

successfully. They are determined based on proper posture alignments for different exercises. Since they act as a standard for assessing the user's present posture, these training values are essential. They are often ascertained by in-depth exercise physiology data gathering, biomechanical research, and expert discussions.

Conditional statements, are used to assess if the computed angles are within the permitted ranges of the pre-established training values. In this stage, each crucial angle that was obtained during the feature extraction process is compared to its corresponding threshold.

The system sounds a beep alarm or other comparable aural indication if it determines that the posture is inaccurate or out of alignment with the optimal training values. The user receives rapid feedback when their posture or workout form needs to be adjusted. The decision to utilize an audio signal is important because it may rapidly grab the user's attention without forcing them to glance at a screen, which can interfere with their workout.

The system shows a "OK" message if the posture is proper and all angles are within the specified permissible ranges. The user is encouraged to retain their present posture by this feedback, which reassures them that they are executing the exercise correctly. New video frames are processed as they are received from the camera by this real-time decision-making and feedback loop. The user's posture may be dynamically and continuously monitored during the workout session thanks to this. The user may instantly correct their posture while exercising because the procedure is real-time, which guarantees prompt and pertinent feedback.

5.3. Testing

5.3.1. Unit Testing

- Video Feed Capture

OpenCV is used to capture live feed through the webcam. Using the VideoCapture function from cv2, the video is captured. The webcam continues to capture the video feed, till the user hits 'q' on the key-board [Fig – 2.0].

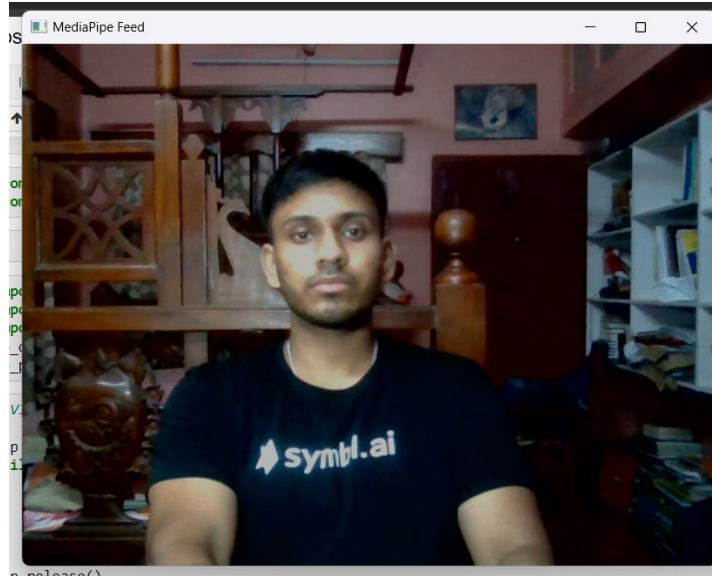


Fig - 2.0 : Video Feed Captured using OpenCV

- Drawing landmark map on the body

For this testing, certain yoga pose images were taken from the dataset. The algorithm was run on these images, and the images with the landmark map on them were obtained. To make it clearer, the array of landmark points was also collected, that gave the coordinated on the location of the 33 landmark points within the frame, and the visibility of each point [Fig – 2.1].

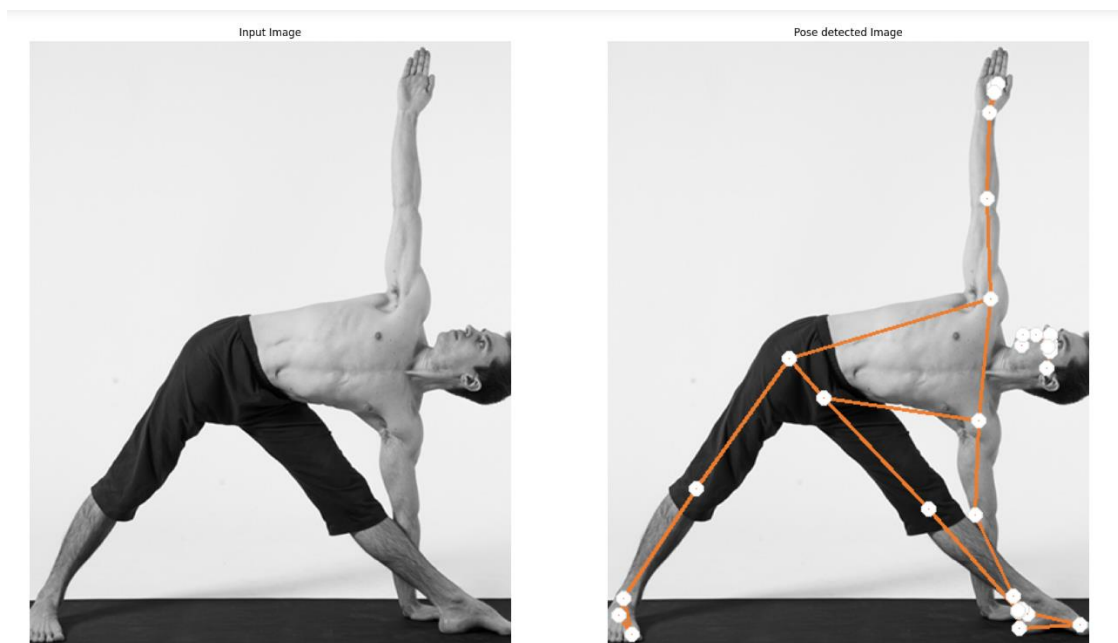


Fig - 2.1 : Landmark Map drawn on the body

- Draw the landmark map on a 3-D plane

Only the landmark points were plotted in a 3-D plane for better understanding of the working of the model. It gives us a view of all the detected landmark points connected through straight lines, forming a raw skeletal structure of the human figure, resembling the posture. The `plot_landmark` method from the `mp_drawing` class was used to fetch the landmarks of the image and `POSE_CONNECTIONS` feature of the pose class was used to connect all the points accordingly on the 3-D plane [Fig – 2.2].

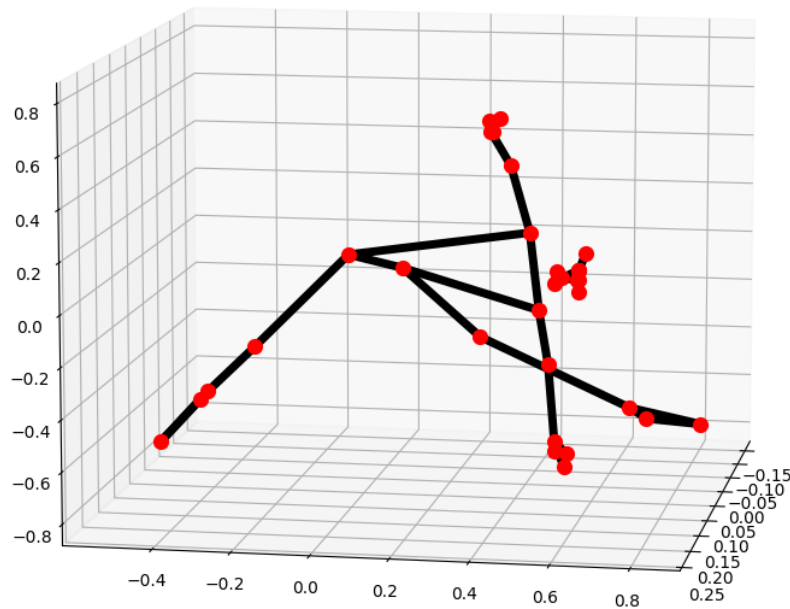


Fig - 2.2 : Body points on a 3-D plane

5.3.2. Integration Testing

- Calculating angles at the joints

A custom function was built to calculate the joint angles. This trigonometric function accepted three points as parameters, and using vector theory calculated the angles formed by the three points when connected by a line. The calculated angle was displayed next to the landmark point in the output window. As the points keep on changing within the frame, the angle also keeps on changing, since the calculation is being done frame by frame. For this, the `putText` feature from the `cv2` class was used to put the data on the

image frame itself [Fig – 2.3].

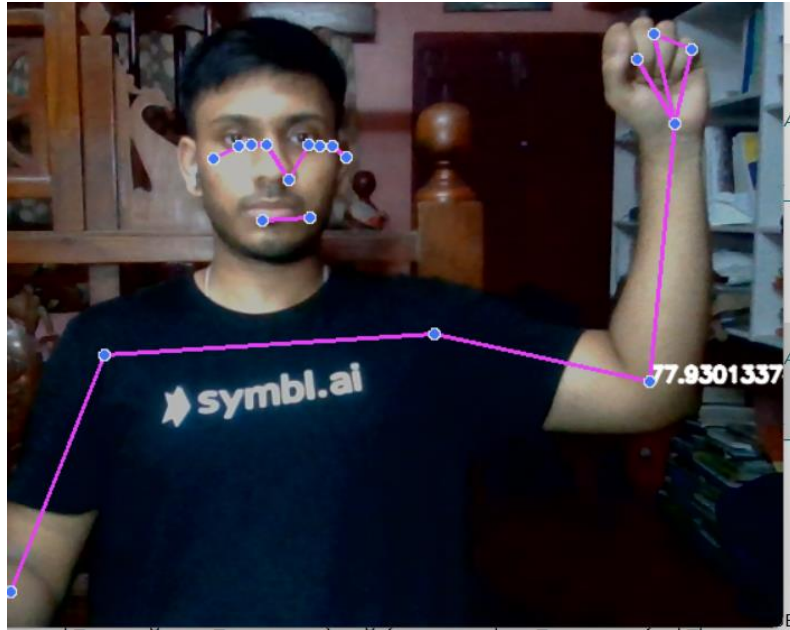


Fig - 2.3 : Angle calculated at the elbow joint

- Testing the model

All the units were executed and tested. They were combined together to build a system that takes video feed as input and converts it into RGB mode for processing. Since the input is generally in BGR mode, it has to be converted to RGB for any action by mediapipe. Then the landmark map was drawn on the pose detected. The sets of body-points were then selected and passed to the mathematical function. The angle values were returned in degrees. They were cross checked with the standard values as determined during testing. Accordingly, the decision was taken and either an “OK” message or an “INCORRECT” message was displayed. In-case of an error in posture, an alert warning was triggered. This is implemented using the Pygame library. The confidence score calculated based on the visibility of the points as detected was displayed on the top left corner of the output window.

6. PROJECT DEMONSTRATION

To run the code, one has to have any IDE that supports python. One should have python installed in his system form beforehand to avoid any unwanted errors. All the modules and packages required to run the code are present in the code file itself. The user should have a stable video source to enable capturing of live feed. One should also have a display to view the results/ output. The different modules of this project have been discussed in this section.

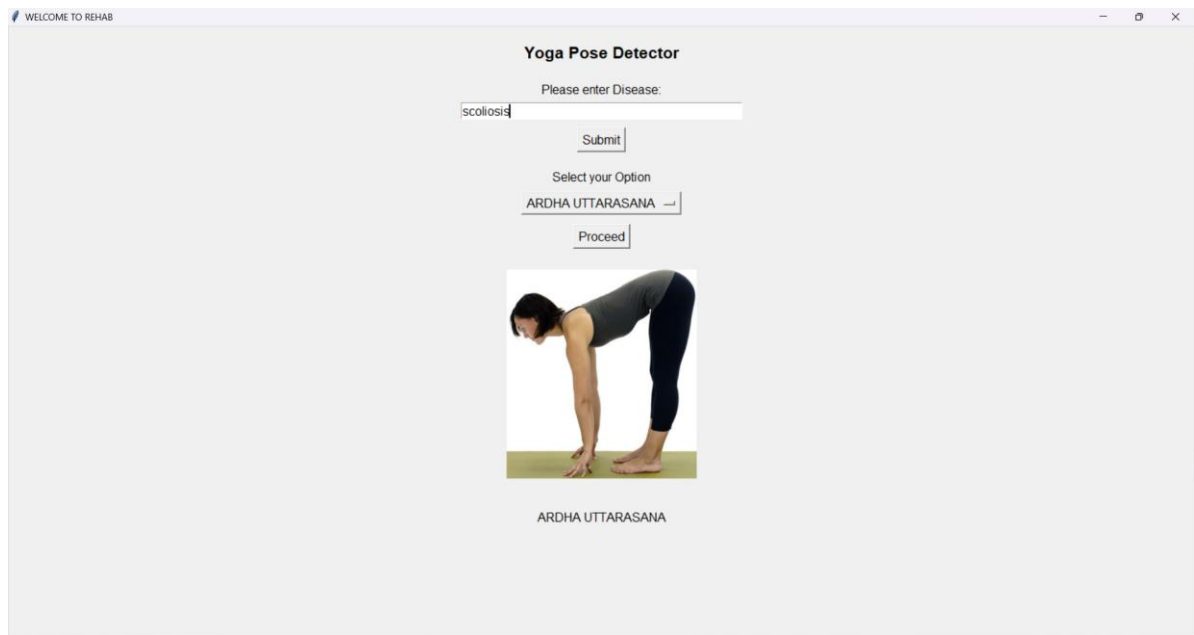


Fig - 2.4 : Starting screen

This project is based on computer vision technology, where the medical condition that is being taken into account is scoliosis.

When the user starts the system, he will be prompted to such a screen Fig – [2.4], where one has to enter the disease for which rehabilitation is needed. If the particular disease is present in the database, then on clicking the submit button, the list of yoga will be enabled. As soon as the user selects a particular yoga pose, an image demonstrating the pose will be made visible on the screen. Once the user knows how the pose is to be performed, he can click the proceed button, and the corresponding function will be triggered in the backend.

When an individual stands in front of the camera while the model is running, his video feed will be captured and converted into RGB format for processing.

The video input will be recorded using cv2 in order to utilise the MediaPipe Pose Estimation. If there is only one connected camera device, the address for video capture (address) is typically 0. Next, depending on how precise the findings should be, parameters for the pose estimation will be assigned. These parameters include tracking confidence, model complexity, on and off of the static picture mode, and minimum detection and on.

Then, as long as the camera is on or the video is playing, the programme will continuously read the frames from the video input. MediaPipe Pose will then process each frame one after the other to obtain the landmarks.

But before users can view and compare, the camera and video inputs must be visualised. The Red-Green-Blue in the pictures will need to be rearranged because it is actually in the Blue-Green-Red sequence. Additionally, the video will be mirrored for a better visualisation experience because the camera or video feed is not mirrored.

After processing this feed will be fed to Google Media Pipe, that will make use of the minimum detection and tracking confidence to perform pose estimation by calling the required methods from the `pose()` class.

The individual can typically be located using the Nose, which is the "landmark 0," which is returned by MediaPipe stance along with a stance prediction based on 33 landmarks. Using the posture landmark submodule, Google MediaPipe displayed the pose landmark graph. Cross-platform settings, such as static picture mode, model complexity, minimum detection confidence, and minimum tracking confidence, are provided by Google MediaPipe Pose and may be applied in many scenarios. For picture input, the static image mode may be turned on; for video stream inputs, it can be turned off.

The model's level of confidence in detecting and tracking the input is determined by the minimum detection and tracking confidence. We may set the minimum detection and tracking confidence with a lower value and set the model complexity configuration to "2" if the posture was very complicated, however this would slow down the procedure pace. Upon detecting the posture from the picture or frame, the model will return coordinates including its width, height, depth, and visibility.

The landmarks will be determined on the body of the user and displayed as live feed on the monitor. Then the user can perform the required yoga pose. A trigonometric function has been prepared that takes any three points as input and calculated the angles between them using the `arctan2` function. During the training and the testing phase the angles formed by a certain set of points will be calculated to be used during the dynamic implementation.

While the user performs the yoga pose, the angles formed by the body points that are specifically required for this yoga will be displayed next to the point in the output. If the user is able to correctly perform the yoga, then "OK" will be displayed, otherwise, a sound alert will be triggered to make the user aware that he is out of form. Along with this, the confidence score will also be displayed so that the user can understand what percent of the

body points are being captured by the frame.

The project is divided into three modules :

Module 1 – Pre-processing stage

This module outlines the process of creating an image for processing in the MediaPipe library. Users select their ailment and submit exercises, which are displayed on a screen. The camera captures real-time video frames at a frame rate of at least 30 frames per second (FPS). These frames are then segmented into individual images frames, which can be processed individually. The frames are then converted from their original format to the RGB color space, ensuring consistency and compatibility with image processing algorithms and libraries. This process simplifies visualization and analysis of the image data.

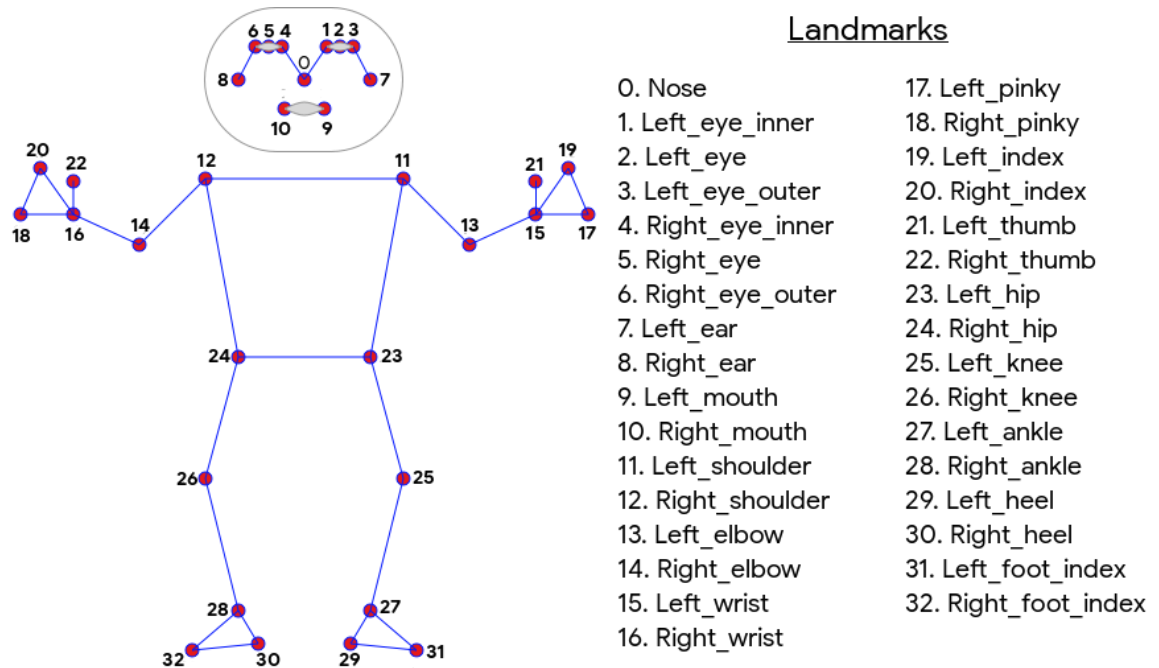


Fig – 2.5 : Landmarks Map of Google Media Pipe Pose

Module 2 – Feature Extraction

The MediaPipe feed is a tool used to analyze user posture during workouts. It uses deep learning models to identify landmarks on images or live streams, allowing it to recognize important body parts like the torso, limbs, and joints. Landmarks are then drawn on the video feed, allowing for a graphical depiction of the detected stance. These landmarks are stored in a variable called landmarks, and the skeletal structure and position configuration are illustrated by drawing links between landmarks [Fig – 2.5].

The algorithm then looks for body pinpoint offset, which is differences or inconsistencies between detected landmarks and their intended positions. This stage compares the discovered landmarks with pre-established reference points or templates to identify any misalignments or errors in the posture estimate. Body pinpoint offset detection contributes to the precision and dependability of the pose estimation procedure, allowing the system to provide accurate posture feedback to the user.

The angle of various body sections can be determined using the x, y, and z coordinates from the posture estimate result and the coordinates that were returned. The radians of the line formed by the first and middle landmarks, the line of origin, and point (1,0) as well as the radians of the line formed by the middle and end landmarks, as well as the line of origin, are obtained using the arctan2 function (1,0). Subsequently, the angle formed by the three landmarks will represent the substitution of the two radians. The angles will be translated from radians to degrees for easier comprehension [Fig – 2.6]. We shall maintain the original angle values for a more accurate comparison since, when considering body positions, it is possible that the angles generated by the body components are more than 180 degrees.



Fig – 2.6 : Landmark map formed on the body

The system then determines the angles between various body reference points during an exercise, assessing spatial correlations among landmarks to determine the alignment and orientation of body parts. The calculate_angles() function takes three coordinates as input and calculates the angle formed by them in radians. The system evaluates the user's posture, detects deviations from the ideal form or alignment, and sends an alarm to rectify the

posture by computing the angles between reference points.

Module 3 – Decision Making

The system uses a real-time feedback loop to monitor the user's posture during a workout session. After feature extraction, if-else conditions are run, the angles obtained are checked against pre-established training values. If the posture is incorrect, a beep alert is audible, and if correct, an OK message is displayed. These training values are based on proper posture alignments for different exercises and are essential for assessing the user's current posture. If the angles are within the permitted ranges of the pre-established training values, the system sounds a beep alarm or other aural indication. The system then displays an "OK" message if the posture is correct and all angles are within the specified ranges. This real-time decision-making and feedback loop allows the user to instantly correct their posture while exercising, ensuring they are performing the exercise correctly.

7. RESULTS AND DISCUSSION

After successful implementation of all the modules, successful results have been obtained. The model was thoroughly trained using a large number of datasets. This ensured that the model can make predictions from various angles and for pictures with low-pixel quality, and give error-free results. Certain exercises have been performed and the output of the model has been recorded.

For calculating the angles formed at the different joints, the major regions that are being stressed are spotted out. They are the elbow-shoulder-hip, shoulder-hip-knee, hip-knee-ankle. The three sets of joints were passed into the custom function, one at a time, and the angles were calculated in radians using arctan2 function on 2 vectors. The value in radians was converted back to degrees, and any value above 180 was brought down to the range of 0-180. The model was trained by using a variety of yoga pose images and then it was tested using a specific set of images. For each type of pose, the false positive(fp), false negative(fn), true positive(tp), true negative(tn) were generated and using certain formulas these results were obtained. Confusion matrix for two yoga postures [Fig – 2.7, 2.8] were determined using the above values, using certain formulas. The confusion matrix helps to determine the accuracy, precision, recall and F1-score, for better analysis and understanding of the model.

$$Accuracy = \frac{(tn+tp)}{(tn+fp+tp+fn)}$$

$$Precision = \frac{(tp)}{(fp+tp)}$$

$$Recall = \frac{(tp)}{(tp+fn)}$$

$$F1\ Score = 2 * \frac{(precision * recall)}{(precision + recall)}$$

Table – 1.1 : From the testing phase, these results were obtained.

Srl No.	Yoga	Accuracy	Precision	Recall	F1-Score
1.	Adho Mukha Savasana	0.976744	1.0	0.814815	0.897959
2.	Ardha Uttanasana	0.986047	1.0	0.888889	0.941176
3.	Balasana	0.962791	1.0	0.703704	0.826087
4.	Bhujangasana	0.95814	0.875	0.777778	0.823529
5.	Setu Bandha Sarvangasana	0.972093	0.954545	0.807692	0.875
6.	Utthita Trikonasana	0.990698	1.0	0.925926	0.961538
7.	Cat	0.976744	0.84375	1.0	0.915254
8.	Camel	0.990698	1.0	0.925926	0.961538

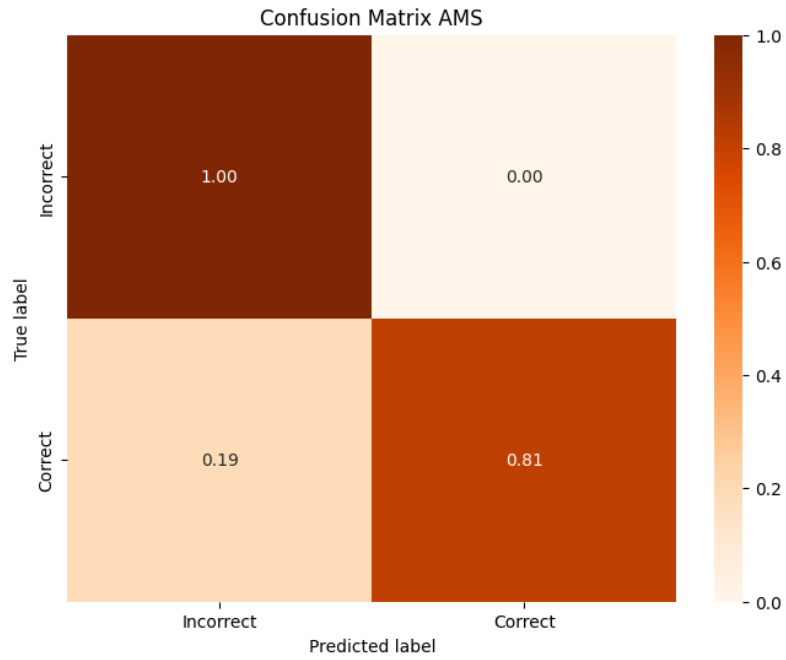


Fig – 2.7 : Confusion Matrix for Adho Mukha Savasana

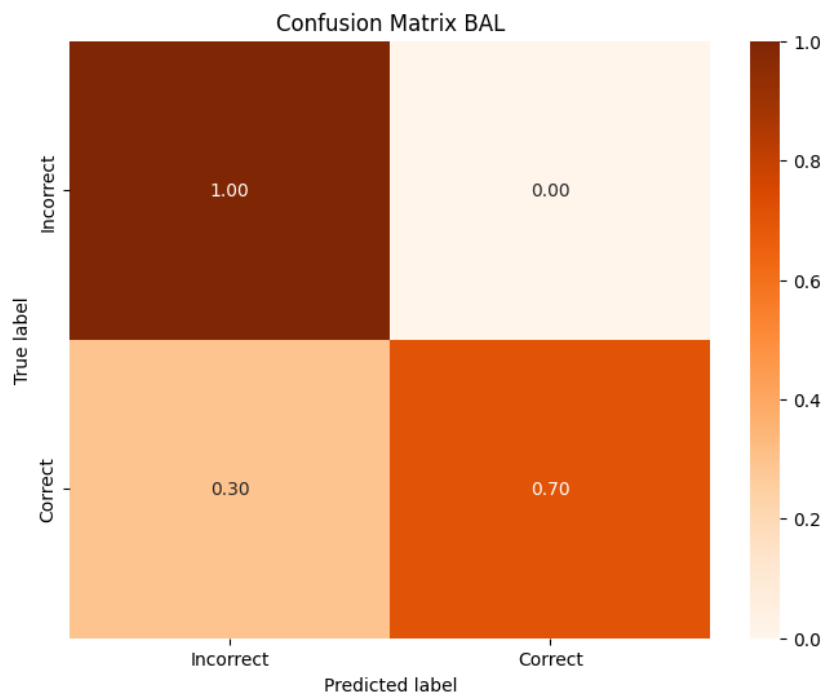


Fig – 2.8 : Confusion Matrix for Balasana

In Fig[2.9] the yoga, adho mukha savasana is being performed. This is the actual correct pose and it has been correctly detected by the system. The angles being formed at certain joints on the body are displayed right next to the joint itself. Therefore the “OK” message is shown in the stage section. The “CONF.” section shows the confidence score. It is the percentage of points that MediaPipe has been able to detect and make use of. For every landmark point on the body, MediaPipe provides a (landmark.visibility) feature that enables

us to calculate the visibility of a point in the frame.

For this pose, the three sets of landmark points that are being used to calculate the angles are as follows, the Lower Body (ankle-knee-hip), the Middle Body (knee-hip-shoulder), the Upper Body (hip-shoulder-elbow).

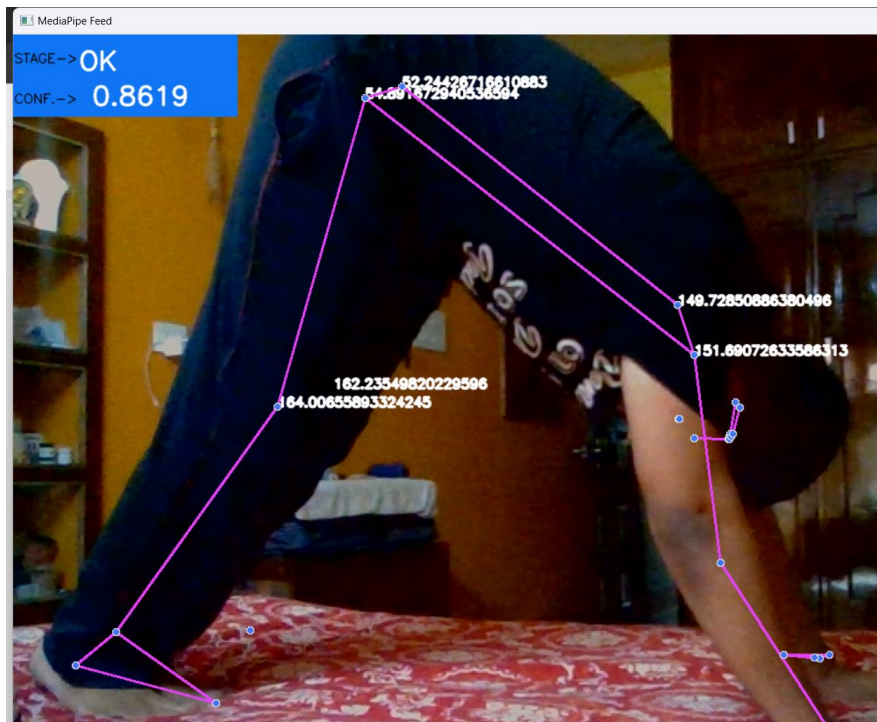


Fig – 2.9 : Correct Pose for Adho Mukha Savasana

In Fig[3.0], the previous yoga is being performed, but in an incorrect manner. It is clearly visible that the legs have been folded from the knee and there is a change in the angle. The new angle formed in the knee region does not correspond to the optimal angle range and therefore an “INCORRECT” message is displayed. In this result, 82% of the landmark points are visible .

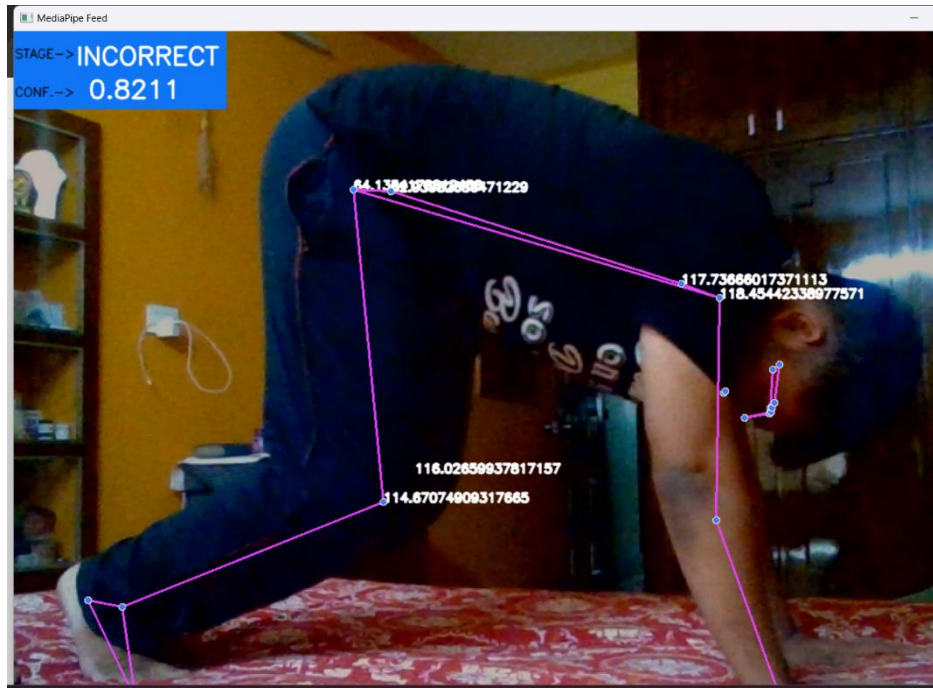


Fig – 3.0 :Incorrect Pose for Adho Mukha Savasana

In Fig[3.1], balasana is being performed. The pose is correct as per the standards and all the angles measurements are in correspondence to the pre-determined values. In this output, 75.69% of the landmark points are visible in the screen.

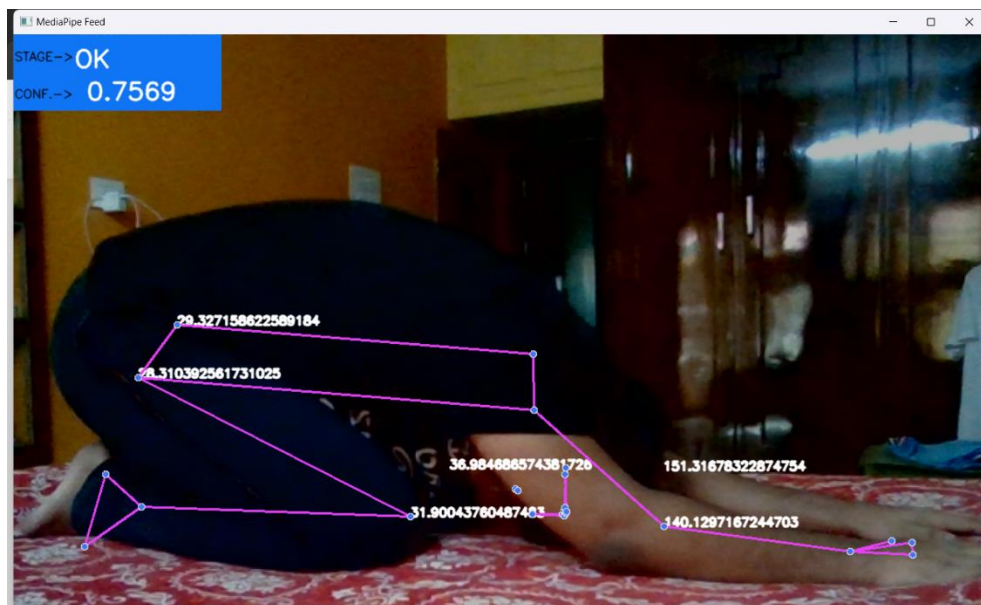


Fig – 3.1 : Correct Pose for Balasana

In Fig[3.2] it is visible that the angle formed at the elbow and knee have changes drastically, and therefore the overall value changes and this is identified as a wrong pose. 80.97 % of the landmark points are visible in the frame.

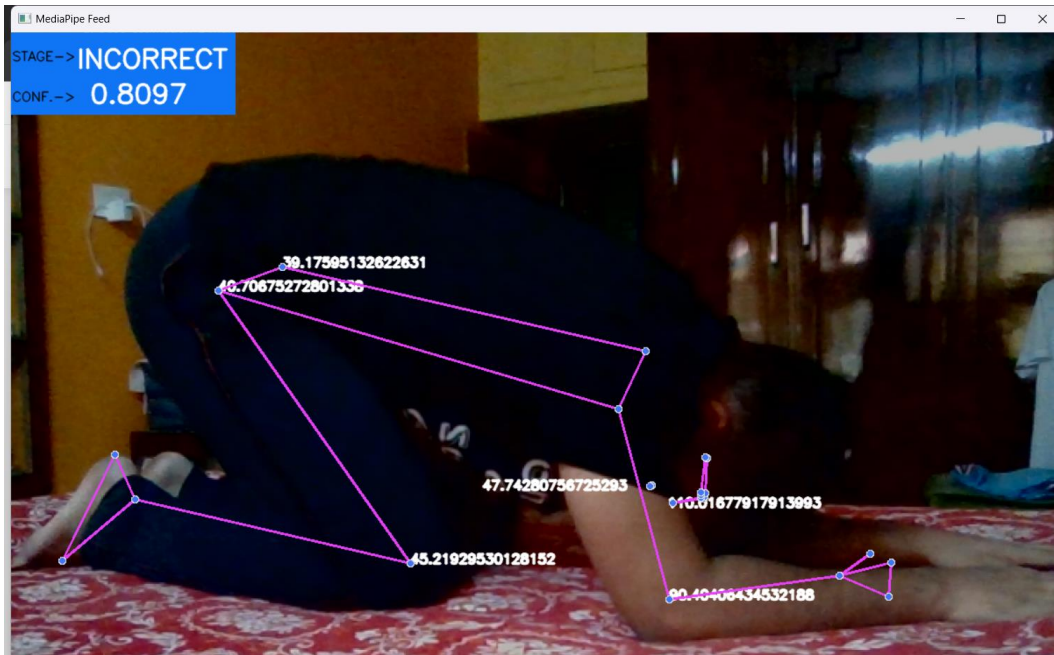


Fig – 3.2 : Incorrect Pose for Balasana

8. SUMMARY

Therefore, by developing such a system we can make the entire system automated, and yet provide accurate results over time. The patients will be supervised and guided by the system, tracking and posture correction will go on concurrently at the same time. The biggest challenge is that we have to get a large number of datasets that has to be used to train the model so that it can have greater efficiency and provide the best of results. The user should also ensure that he is facing the camera so that the media pipe technology can detect the points on his body accurately and calculate all the desired angles with greater efficiency. Its effectiveness in enhancing exercise adherence, form correctness, and ultimately, patient outcomes is sought to be proven by this system through extensive testing and validation, including user trials with patients and healthcare professionals. The potential to transform the monitoring and management of disease-specific workouts via the use of technology, improving patient quality of life and rehabilitation outcomes, is possessed by this project. As part of the future scope, a proper front-end website can be created that will also act as a portal to guide the users, and through the website the users can start their rehab program. It will also provide a forum to the users to communicate with other fellow users and also post question and answers. A feature can be added which will ask the user permission every time camera access is required, so that the personal rights of any individual are not disturbed. A

wider range of yoga postures can be included to serve for different types of injuries and rehabilitation stages. Utilize augmented reality to overlay corrective suggestions directly onto the user's view, making it easier to understand and follow guidance. Run trials on patients suffering from scoliosis. Use cloud service to host the model and scale the service efficiently to handle a large number of users.

9. REFERENCES

1. Anuj Patil, D.V. Divakara Rao, Kaustubh Utturwar, T. P. Shelke, Ekta Sarda, "Body Posture Detection and Motion Tracking using AI for Medical Exercises and Recommendation System", ITM web of conferences-Vol. 44, pp 03043-03043, 01 Jan 2022
2. JA Janicki, B Alman. Scoliosis, "Scoliosis: Review of diagnosis and treatment", Paediatr Child Health;12(9):771-776, 2007
3. John P. Horne, Robert Flannery, Saif Usman, "Adolescent Idiopathic Scoliosis: Diagnosis and Management", American Family Physician (Am Fam Physician)-Vol. 89, Iss: 3, pp 193-198, 01 Feb 2014
4. Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georgl, Matthias Grundmann , "MediaPipe: A Framework for Building Perception Pipelines", Google1- arXiv: Distributed, Parallel, and Cluster Computing, 14 Jun 2019
5. Jérôme Thevenot, Miguel Bordallo López, Abdenour Hadid, "A Survey on Computer Vision for Assistive Medical Diagnosis From Faces", University of Oulu, IEEE Journal of Biomedical and Health Informatics (IEEE)-Vol. 22, Iss: 5, pp 1497-1511, 01 Sep 2018
6. Andre Esteva, Katherine Chou, Serena Yeung, Nikhil Naik, Ali Madani, Ali Mottaghi, Yun Liu, Eric J. Topol, Jeffrey Dean, Richard Socher, "Deep learning-

enabled medical computer vision”, Salesforce.com, Google, Stanford University, Scripps Health-Vol. 4, Iss: 1, pp 1-9, 08 Jan 2021

7. Mazin H. Aziz, Hamed A. Mahmood, “Automated Body Postures Assessment From Still Images Using Media Pipe”, Computer Engineering Department, University of Mosul, Mosul Journal of Optimization & Decision Making 2(2), 240-246, 202
8. Fanbin Gu, Jingyuan Fan, Chengfeng Cai, Zhaoyang Wang, Xiaoming Liu, Jiantao Yang, Qingtang Zhu, “Automatic detection of abnormal hand gestures in patients with radial, ulnar, or median nerve injury using hand pose estimation”, -Frontiers in Neurology-Vol. 13, 07 Dec 2022
9. Kuan-Yu Chen, Jungpil Shin, Md. Al Mehedi Hasan, Jiun-Jian Liaw, Yuichi Okuyama, Yoichi Tomioka, “Fitness Movement Types and Completeness Detection Using a Transfer-Learning-Based Deep Neural Network”, -Sensors-Vol. 22, Iss: 15, pp 5700-5700, 29 Jul 2022
10. Wenting Ma, Xiaohang Yuan, Jian-Ye Zhu, Hao Zhang, Zhipeng Zhang, “A Video Image Processing Method for Continuous Object Detection”, -pp 1015-1020, 21 Apr 2023

APPENDIX

This Code is for Adha Mukha Savasana. For implementing the other functions similar method has to be followed.

```
!pip install mediapipe opencv-python  
  
from pydub import AudioSegment  
  
from pydub.playback import play  
  
import cv2  
  
import mediapipe as mp
```

```

import numpy as np

mp_drawing = mp.solutions.drawing_utils

mp_pose = mp.solutions.pose

import matplotlib.pyplot as plt

# Initialize mediapipe pose class.

mp_pose = mp.solutions.pose

# Setup the Pose function for images - independently for the images standalone
processing.

pose_image = mp_pose.Pose(static_image_mode=True,
min_detection_confidence=0.5)

# Initialize mediapipe drawing class - to draw the landmarks points.

mp_drawing = mp.solutions.drawing_utils

import pandas as pd

import cv2

import mediapipe as mp

import matplotlib.pyplot as plt

import os

from PIL import Image

from pydub import AudioSegment

from pydub.playback import play

import numpy as np

from tkinter import *

from winsound import PlaySound, SND_FILENAME, SND_LOOP,
SND_ASYNC

import winsound

import pygame

# CALCULATE ANGLES

```

```

def calculate_angles(a,b,c):

    a = np.array(a) # First
    b = np.array(b) # Second
    c = np.array(c) # Third


    radians = np.arctan2(c[1]-b[1], c[0]-b[0]) - np.arctan2(a[1]-b[1], a[0]-b[0])
    angle = np.abs(radians*180.0/np.pi)


    if angle > 180.0:
        angle = 360-angle

    return angle

pygame.mixer.init()

def play_audio():

    pygame.mixer.init()

    sound =
pygame.mixer.Sound("C:/Users/User/Downloads/f1_team_radio_loud.wav")

    if not pygame.mixer.get_busy():

        sound.play(-1) # -1 indicates looping indefinitely


def stop_audio():

    pygame.mixer.stop()


def callback(sv):

    if sv == "PLAY":

        play_audio()

    else:

        stop_audio()

```

```

# MAKE DETECTIONS

cap = cv2.VideoCapture(0)

counter = 0

stage = None

audio_duration = 2

sum = 0

with mp_pose.Pose(min_detection_confidence=0.5,
min_tracking_confidence=0.5) as pose : while cap.isOpened():

    ret, frame = cap.read()

    width = 1200

    height = 880

    dim = (width, height)

    frame = cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)

    # DETECT STUFF AND RENDER

    #RECOLOR IMAGE TO RGB

    image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    image.flags.writeable = False

    # MAKE DETECTION

    results = pose.process(image)

    #RECOLOR BACK TO BGR

    image.flags.writeable = True

    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

    # PLAY AUDIO ALERT

#     sum=

    # EXTRACT LANDMARKS

```

```

try:

    landmarks = results.pose_landmarks.landmark

    # GET COORDINATES

    right_shoulder =
[landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].y]

    left_shoulder =
[landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x,landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].y]

    right_elbow =
[landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].y]

    left_elbow =
[landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x,landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]

    right_wrist =
[landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].y]

    left_wrist =
[landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x,landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].y]

    right_hip =
[landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value].y]

    left_hip =
[landmarks[mp_pose.PoseLandmark.LEFT_HIP.value].x,landmarks[mp_pose.PoseLandmark.LEFT_HIP.value].y]

    right_knee =
[landmarks[mp_pose.PoseLandmark.RIGHT_KNEE.value].x,landmarks[mp_pose.PoseLandmark.RIGHT_KNEE.value].y]

    left_knee =
[landmarks[mp_pose.PoseLandmark.LEFT_KNEE.value].x,landmarks[mp_pose.P

```

```

poseLandmark.LEFT_KNEE.value].y]

    right_ankle =
[landmarks[mp_pose.PoseLandmark.RIGHT_ANKLE.value].x,landmarks[mp_pose
e.PoseLandmark.RIGHT_ANKLE.value].y]

    left_ankle =
[landmarks[mp_pose.PoseLandmark.LEFT_ANKLE.value].x,landmarks[mp_pose
.PoseLandmark.LEFT_ANKLE.value].y]

    nose =
[landmarks[mp_pose.PoseLandmark.NOSE.value].x,landmarks[mp_pose.PoseLan
dmark.NOSE.value].y]

    # CALCULATE THE ANGLE

    lower_body_1 = calculate_angles(left_ankle,left_knee,left_hip)
    lower_body_2 = calculate_angles(right_ankle,right_knee,right_hip)

    middle_body_1 = calculate_angles(left_knee,left_hip,left_shoulder)
    middle_body_2 =
calculate_angles(right_knee,right_hip,right_shoulder)

    upper_body_1 = calculate_angles(left_hip,left_shoulder,left_wrist)
    upper_body_2 = calculate_angles(right_hip,right_shoulder,right_wrist)

    # VISUALIZE ANGLE

    cv2.putText(image,str(lower_body_1),
                tuple(np.multiply(left_knee, [1200,880]).astype(int)),
                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255), 2,
cv2.LINE_AA
                )

    cv2.putText(image,str(lower_body_2),
                tuple(np.multiply(right_knee, [1200,880]).astype(int)),
                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255), 2,
cv2.LINE_AA
                )

```

```

cv2.putText(image,str(middle_body_1),
            tuple(np.multiply(left_hip, [1200,880]).astype(int)),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255), 2,
cv2.LINE_AA
        )
cv2.putText(image,str(middle_body_2),
            tuple(np.multiply(right_hip, [1200,880]).astype(int)),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255), 2,
cv2.LINE_AA
        )
cv2.putText(image,str(upper_body_1),
            tuple(np.multiply(left_shoulder, [1200,880]).astype(int)),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255), 2,
cv2.LINE_AA
        )
cv2.putText(image,str(upper_body_2),
            tuple(np.multiply(right_shoulder, [1200,880]).astype(int)),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255,255,255), 2,
cv2.LINE_AA
        )

#      CHECKING CONDITIONS

lb = (lower_body_1 + lower_body_2)/2
mb = (middle_body_1 + middle_body_2)/2
ub = (upper_body_1 + upper_body_2)/2

sum=0

for it in landmarks:

    sum+=it.visibility

avg=sum/len(landmarks)

```



```

    avg="{0:.4f}".format(avg)

    avg=str(avg)

    if((lb>=160 and lb<=180) and (mb>=50 and mb<=90) and (ub>=150
and ub<=180)):

        stop_audio()

        stage="OK"

        counter+=1

        print(stage)

    else:

        stage="INCORRECT"

except:

    avg="0"

    pass

    # setup status box

    cv2.rectangle(image,(0,0), (255,93), (245,117,16), -1)

    # STAGE DATA

    cv2.putText(image, 'STAGE-> ', (1,32),

                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1,

cv2.LINE_AA)

    cv2.putText(image, stage,

                (75,40),

                cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2,

cv2.LINE_AA)

    cv2.putText(image, 'CONF.-> ', (1,78),

                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1,

cv2.LINE_AA)

    cv2.putText(image, avg,

                (90,80),

```

```

        cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2,
cv2.LINE_AA)

# RENDER DETECTIONS

    mp_drawing.draw_landmarks(image, results.pose_landmarks,
mp_pose.POSE_CONNECTIONS,

        mp_drawing.DrawingSpec(color=(245,117,66),
thickness=2, circle_radius=2),

        mp_drawing.DrawingSpec(color=(245,66,230),
thickness=2, circle_radius=2),

    )

    cv2.imshow('MediaPipe Feed', image)

    if cv2.waitKey(10) & 0xFF == ord('q'):

        stop_audio()

        break

    cap.release()

    cv2.destroyAllWindows()

```




COMPUTER VISION TECHNOLOGY IN REHABILITATION AND TRAINING

20BKT0095 | ADWAITA RAJ MODAK | Prof. NIHA K. | SCOPE

Introduction

One of the research papers, proposes using AI and image processing through **Media Pipe** for a motion tracker that monitors and provides feedback on exercises, enhancing workout effectiveness and physician coordination. The gaps identified in this research were, that any inaccuracies or errors in these algorithms may lead to incorrect feedback or recommendations as these rely on **body-posture detection** and **motion tracking algorithms**.

Motivation

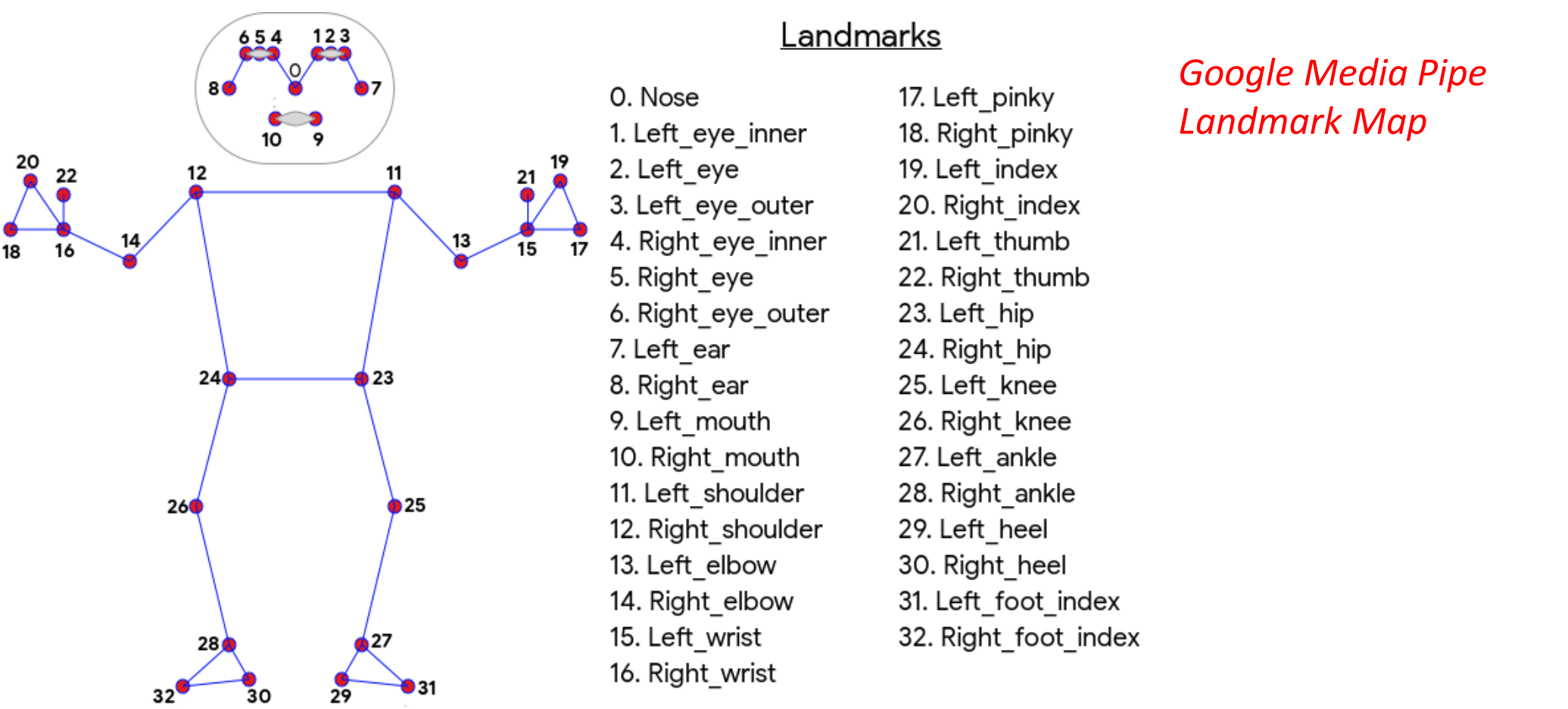
- Technological Integration** : The rapid advancement of computer vision technology allows for innovative applications in healthcare, particularly in patient rehabilitation.
- Need for Precision** : Accurate posture detection is crucial in rehabilitation exercises to prevent injuries and ensure the effectiveness of therapeutic interventions.
- Empowerment and accessibility** : By automating the supervision of exercises, patients can independently manage their rehabilitation process with reduced need for physical therapist intervention.

Scope of the Project

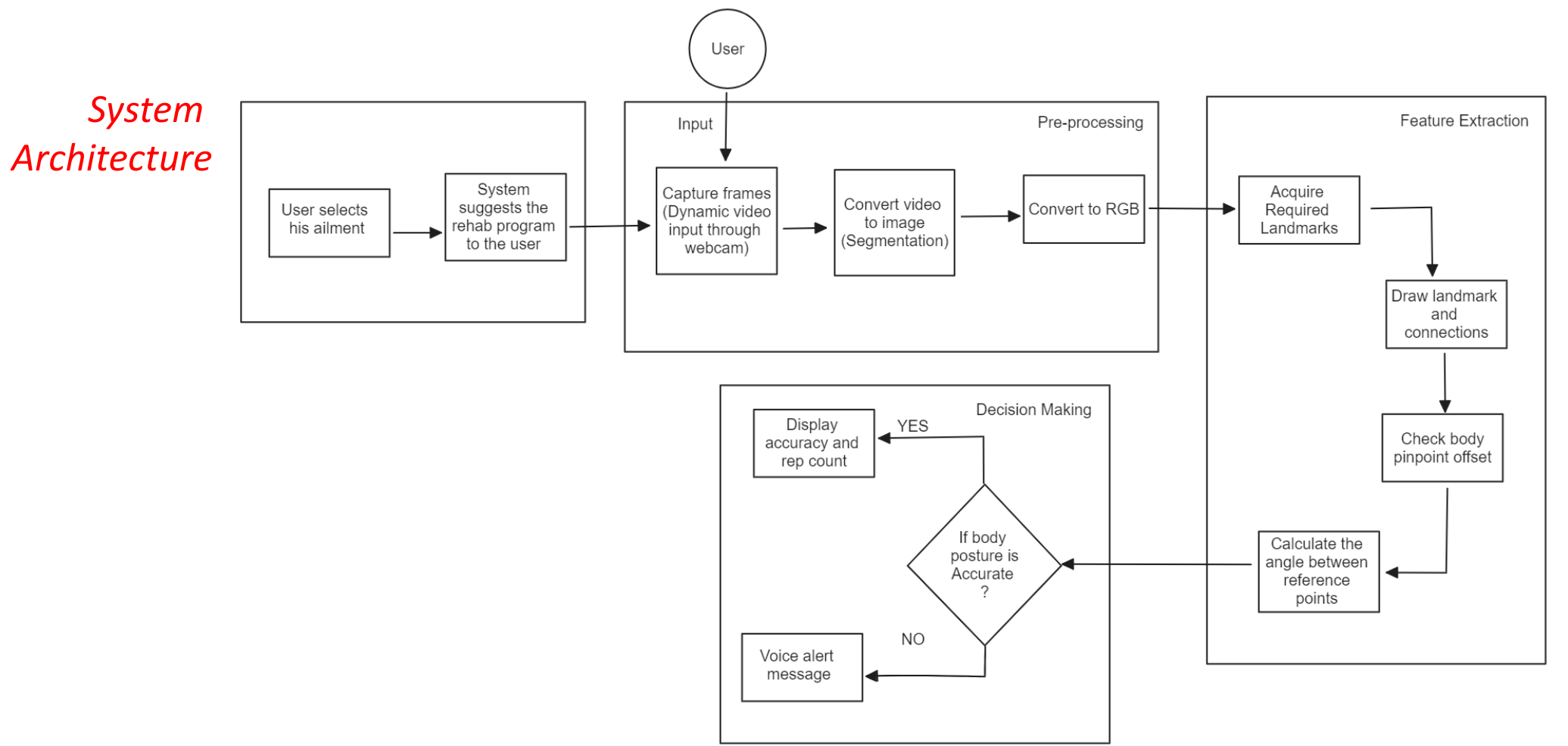
- Real-time posture identification and correction** through the use of machine learning models and sophisticated computer vision methods. Creating and executing an intuitive user interface to lead patients through prescribed exercise regimens.
- Using patient and healthcare professional user trials to test and validate the system's efficacy. Concentrating on certain medical issues, such as spinal disabilities, in order to customize the system's operation to meet the requirements of various patient populations.

Methodology

- The methodology adopted for this project, focused on real-time posture recognition and correction during exercises, integrates a combination of **computer vision, machine learning, and real-time feedback mechanisms**.

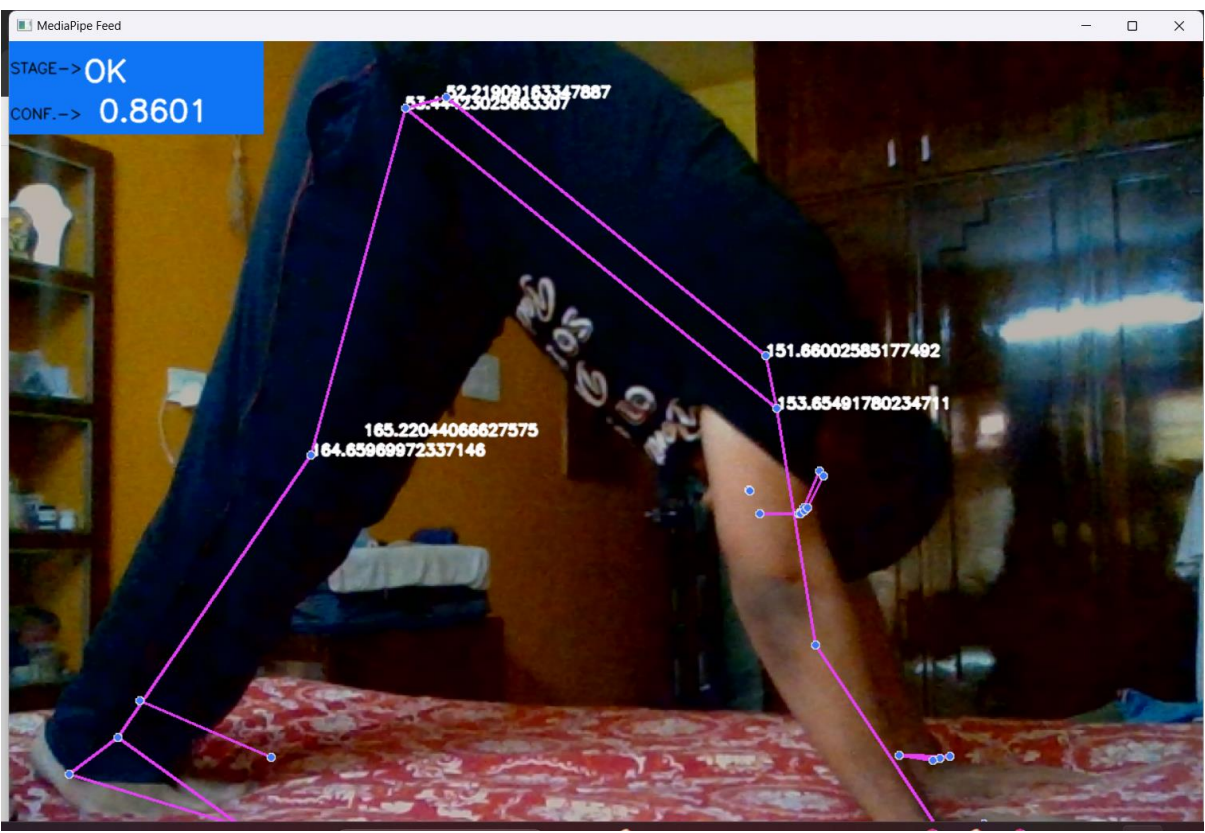


- The project utilized **OpenCV** for capturing webcam feeds and **MediaPipe** for identifying 33 body landmarks to create skeletal models, with data augmentation used to enhance the training dataset.



- Video Feed input and converting image frames to RGB make up the **Pre-Processing** stage, next up, the pose() class of MediaPipe, processing the frames to draw landmarks on the body of the person is the **Feature Extraction** phase.
- In the **Decision Making** phase, joint angles are calculated, compared with pre-determined values to provide feedback if the posture is correct or not.

Results



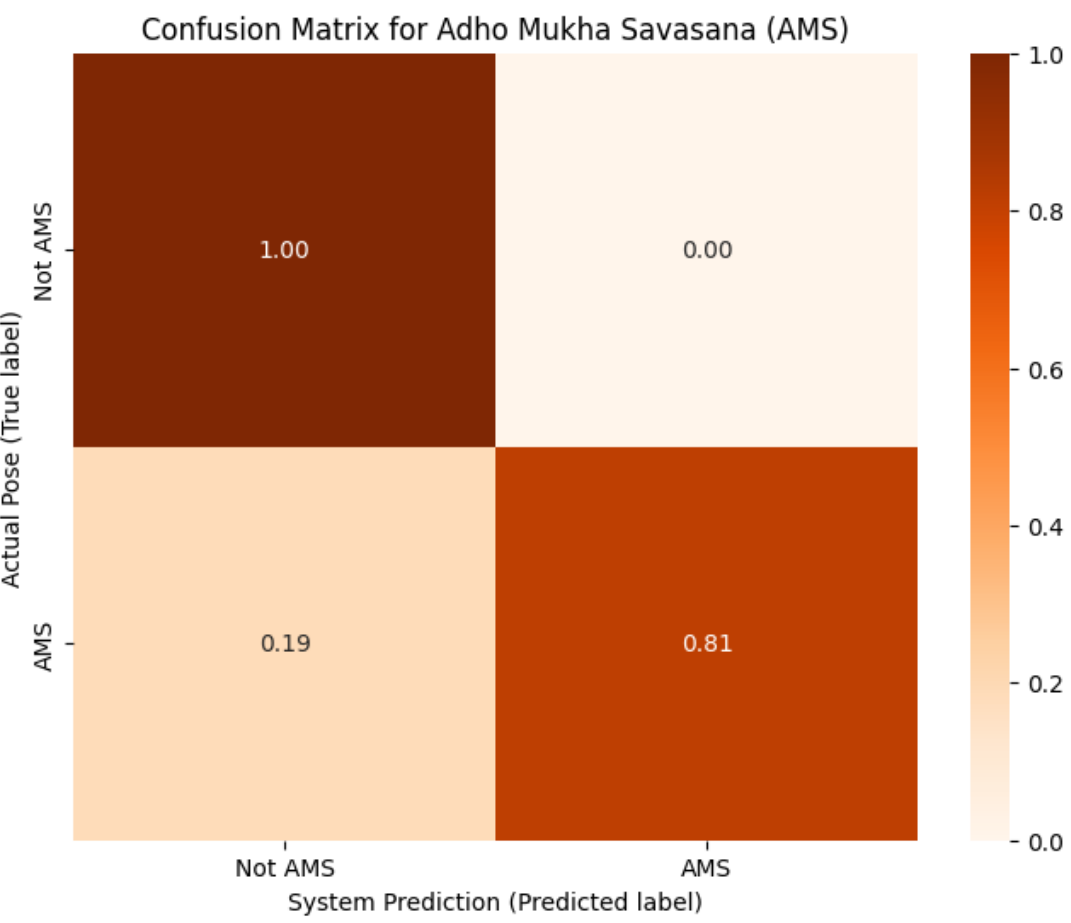
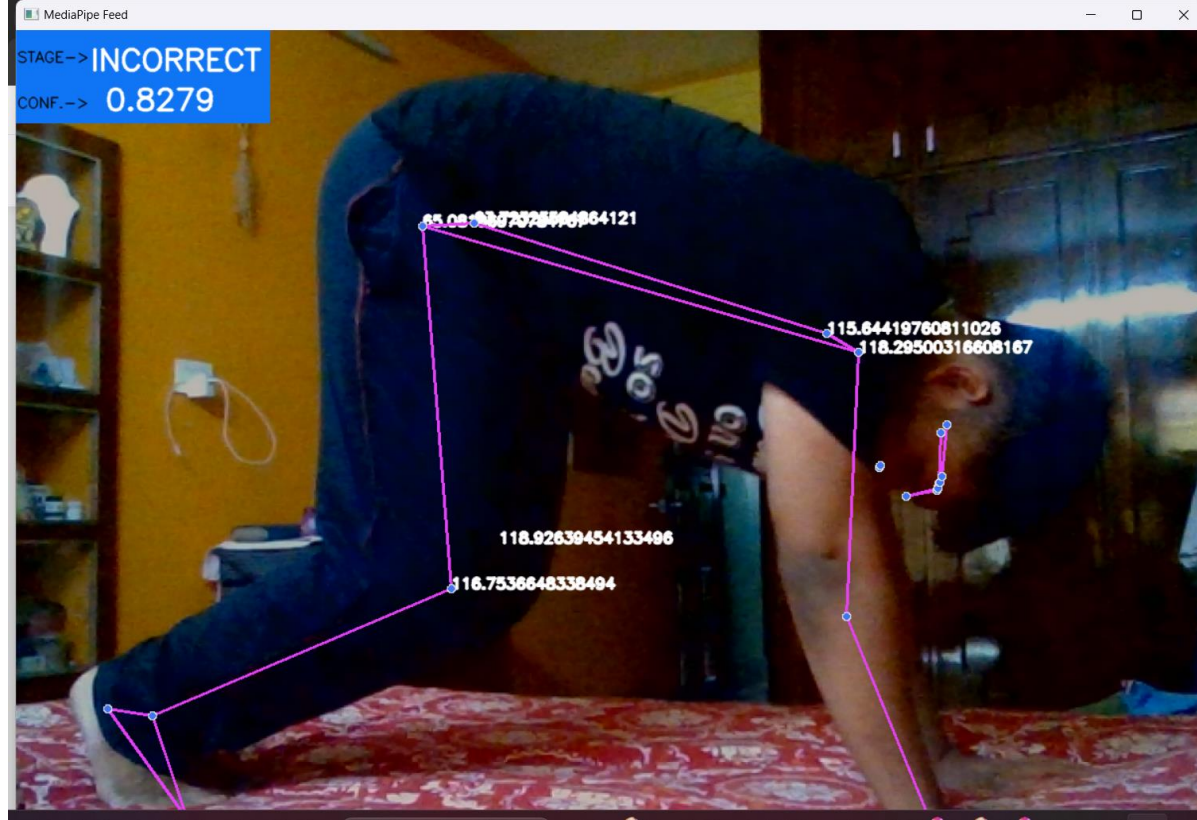
Correct Pose for Adho Mukha Savasana

Landmark points that are being used to calculate the angles are as follows, the

- Lower Body**(ankle-knee-hip)
- Middle Body**(knee-hip-shoulder)
- Upper Body**(hip-shoulder-elbow).

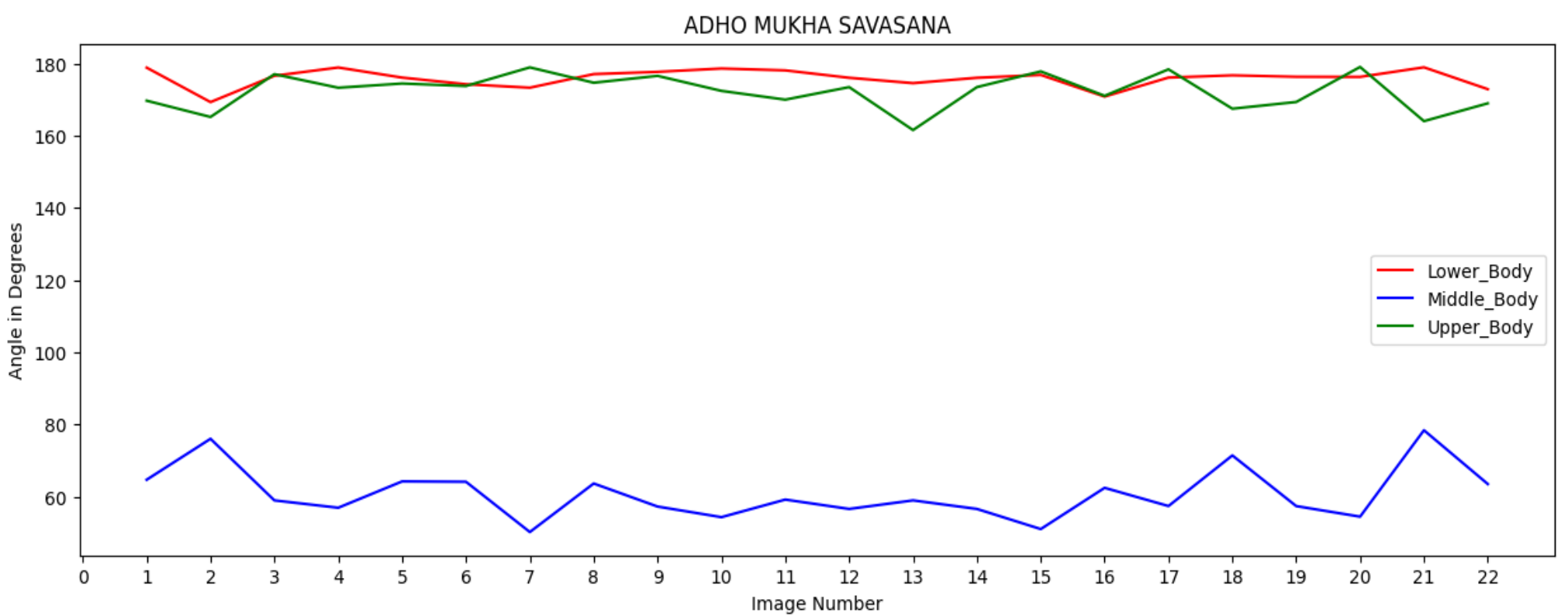
Incorrect Pose for Adho Mukha Savasana

It is clearly visible that the legs have been folded from the knee and there is a change in the angle. The new angle formed in the knee region does not correspond to the optimal angle range and therefore an "INCORRECT" message is displayed.



Confusion Matrix obtained for Adho Mukha Savasana after Testing

The confusion matrix helps to determine the **accuracy, precision, recall and F1-score**, for better analysis and understanding of the model.



Range of angles for body sections for the Yoga Pose

- For calculating the angles formed at the different joints, the major regions that are being stressed are spotted out.
- The sets of joints were passes into the custom function, one at a time, and the angles were calculated in radians using **arctan2** function on 2 vectors.
- The value in radians was converted back to degrees.
- The model was trained by using a variety of yoga pose images and then it was tested using a specific set of images.

Conclusion

Therefore, by developing such a system we can make the entire system automated, and yet provide accurate results over time. The patients will be supervised and guided by the system, tracking and posture correction will go on concurrently at the same time. The biggest challenge is that we have to get a large number of datasets that has to be used to train the model so that it can have greater efficiency and provide the best of results.

References

- Anuj Patil, "Body Posture Detection and Motion Tracking using AI for Medical Exercises and Recommendation System", ITM web of conferences-Vol. 44, pp 03043-03043, 01 Jan 2022
- <https://developers.google.com/mediapipe/solutions/vision/poselandmarker>