सीडैक CDAC | प्रगत संगणन विकास केंद्र, नोएडा
Centre for Development of Advanced Computing, Noida

A PROJECT REPORT

ON

AI PERSONAL TRAINER

Submitted In Partial Fulfillment of the Requirement for The Award of

POST GRADUATE DIPLOMA IN ARTIFICIAL INTELLIGENCE

Under the guidance of

MR. RAVI PAYAL	**MRS. SARUTI GUPTA**
(Joint Director)	**(Project Guide)**

Submitted By

CHAITALI DHANDE (210920528012)

PRAJAKTA SURYAVANSHI (210920528031)

SHEETAL MATHUR (210920528033)

CDAC, B-30, Institutional Area, Sector-62
Noida (Uttar Pradesh)-201307

CDAC, NOIDA

CERTIFICATE

This is to certify that Report entitled **“AI PERSONAL TRAINER”** which is submitted by Kajal Panda in partial fulfillment of the requirement for the award of Post Graduate Diploma in Artificial Intelligence (PG-DAI) to CDAC, Noida is a record of the candidates own work carried out by them under my supervision.

The documentation embodies results of original work, and studies are carried out by the student themselves and the contents of the report do not from the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

MRS. SARUTI GUPTA

(Project Guide)

CONTENTS

ACKNOWLEDGEMENT

We would like to thank **CDAC Noida** for providing me the platform to contribute my knowledge into project work.

We would also like to express my best sense of gratitude & endeavor with respect to **Mrs. Saruti Gupta (Project Guide)** for suggesting the problems scholarly guidance and expert supervision during the course of this project. Special thanks to **Mr. Ravi Payal (Program Coordinator).**

Furthermore, we would like to thank my friends and my family members for encouraging and helping me to complete my project.

Chaitali	Prajaktas.	Sheetal Mathur
CHAITALI DHANDE	**PRAJAKTA SURYAVANSHI**	**SHEETAL MATHUR**

ABSTRACT

Everyone benefits from exercise and physical activity. Staying active can benefit you in a variety of ways, regardless of your health or physical abilities. In reality, research shows that “taking it easy” is dangerous. When older adults lose their ability to perform activities on their own, inactivity is often to fault rather than ageing. Lack of physical activity can also contribute to additional doctor visits, hospitalizations, and medication use for a range of conditions. This idea of a personal AI fitness trainer allows more people to get involved in the fitness field using this AI trainer makes sure everyone can perform exercises in correct way, minimizing the risk of injuries.

INTRODUCTION TO THE PROBLEM STATEMENT AND THE POSSIBLE SOLUTION

Staying at home for long periods of time can become boring, especially when most fun activities are done outdoors. Still, this is not an excuse to be unproductive and the extra available time is an excellent opportunity to work on your own health. Typical gyms come with a variety of equipment and trainers who can tell you what to do. The lack of these in one's home can often be the culprit that stops them from working out. Wouldn't it be great if there existed a personal trainer that could generate workouts for you at home? What if it could also count the repetitions of each exercise so that you can put all your concentration and energy to do one more push up? In order to overcome such problem Artificial Intelligence has worked one step ahead and introduced AI Personalise Trainer.

INTRODUCTION

Lifting weights is a great way to develop muscles, protect bones, burn calories, and stay fit. Maybe you don't know where to start or how to perform the exercises. You may be tempted to just copy the exercises your others are doing, but they may be doing things the wrong way. This is where your personal AI trainer comes in handy. With your personal AI trainer, you have immediate access to a world of knowledge to help you develop a weight-training routine that's safe and effective. Weight training and healthy diet is one of the best ways to get into shape and lead a healthy lifestyle. With each passing day people are getting conscious about their health. Many people today have a busy and hectic life and they cannot manage to go to the gym and take the guidance of any professional and during this pandemic many people started doing in home workouts and this they a have a big chance of getting injured. In this project, we will develop a personal AI fitness trainer which will help the user to do exercises in a correct form and posture.

1. AI Personal Trainer

Human activity recognition has emerged as an active research area in recent years. With the advancement in mobile and wearable devices, various sensors are ubiquitous and widely available gathering data a broad spectrum of peoples' daily life activities. Artificial Intelligence has evolved in many ways, the AI Personal Trainer as a subset consist of python and computer vision. In this project we will be building an AI Trainer using OpenCV and Python. We will use the pose estimation running on the CPU to find the correct points and using these points we will get the desired angles. Then based on these angles we find many gestures including the number of biceps curls. We will write the code in a way that you will be able to find angles between any 3 points with just a single line of code.

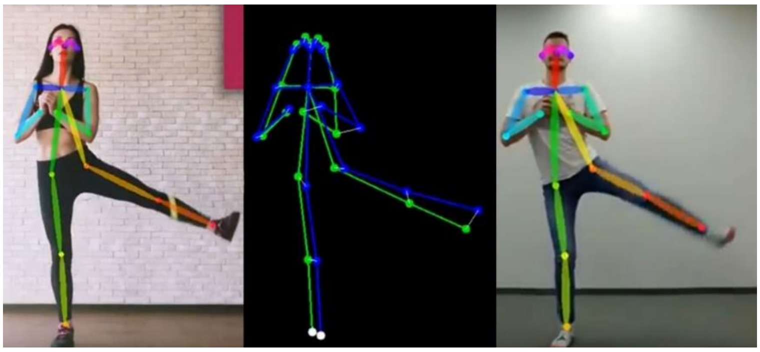


Figure 1: AI Personal Trainer with pose detection

2. **OpenCV**

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Image processing is a form of signal processing in which the input is an image such as a photograph or video frame, the output is an image or set of characteristics related to image. OpenCV is a library of programming functions mainly used for image processing. It is freely available on the open source Berkely Software Distribution license. It was started as a research project by Intel. OpenCV contains various tools to solve computer vision problems. It contains low level image processing functions and high level algorithms for face detection, feature matching and tracking. Some of the main image processing techniques are given below:

- **Image Filtering**:

  It is a technique for modifying or enhancing an image. Image filtering is of two types. The one is linear image filtering, in which, the value of an output pixel is a linear combination of the values of the pixels of the input pixel's neighborhood. The second one is the non-linear image filtering, in which, the value of output is not a linear function of its input.

- **Image Transformation**:

  Image transformation generates "new" image from two or more sources which highlight particular features or properties of interest, better than the original input images. Basic image transformations apply simple arithmetic operations to the image data. Image subtraction is often used to identify changes that have occurred between images collected on different dates.

  Main image transformation methods are :

  - ✓ Hough Transform: used to find lines in an image
  - ✓ Radon Transform: used to reconstruct images from fan-beam and parallel-beam projection data
  - ✓ Discrete Cosine Transform: used in image and video compression
  - ✓ Discrete Fourier Transform: used in filtering and frequency analysis

- ✓ Wavelet Transform: used to perform discrete wavelet analysis, denoise, and fuse images

- **Object Tracking:**

  Object tracking is the process of locating a object (or multiple objects) over a sequence of images. It is one of the most important components in a wide range of applications in computer vision, such as surveillance, human computer interaction, and medical imaging.

- **Feature Detection:**

  A feature is defined as an "interesting" part of an image and features are used as a starting point for many computer vision algorithms. Since features are used as the starting point and main primitives for subsequent algorithms, the overall algorithm will often only be as good as its feature detector. Feature detection is a process of finding specific features of a visual stimulus, such as lines, edges or angle. It will be helpful for making local decisions about the local information contents (image structure) in the image. The modules of openCV for image processing applications are given below CORE module contains basic data structures and basic functions used by other modules. IMGPROC module contains image processing related functions such as linear, non-linear image filtering and geometrical image transformations etc. VIDEO module contains motion estimation and object tracking algorithms. ML module contains machine-learning interfaces. HighGUI module contains the basic I/O interfaces and multi-platform windowing capabilities

3. **Computer Vision**

Computer Vision can be defined as a discipline that explains how to reconstruct, interrupt, and understand a 3D scene from its 2D images, in terms of the properties of the structure present in the scene. It deals with modelling and replicating human vision using computer software and hardware.

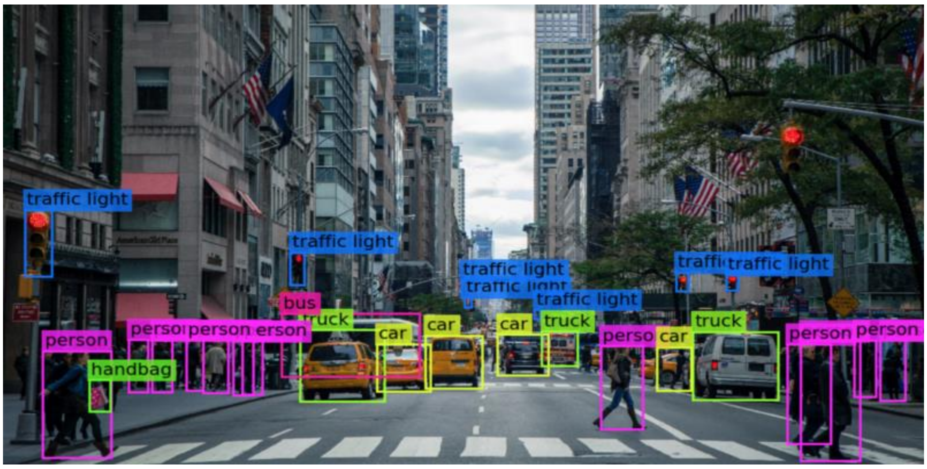


Figure2: Picture showing CV as an example

One of the most powerful and compelling types of AI is computer vision which you've almost surely experienced in any number of ways without even knowing. Here's a look at what it is, how it works, and why it's so awesome (and is only going to get better).Computer vision is the field of computer science that focuses on replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do. Until recently, computer vision only worked in limited capacity.Thanks to advances in artificial intelligence and innovations in deep learning and neural networks, the field has been able to take great leaps in recent years and has been able to surpass humans in some tasks related to detecting and labeling objects. One of the driving factors behind the growth of computer vision is the amount of data we generate today that is then used to train and make computer vision better.

4. **Python**

Python was chosen as the programming language for this project. For a variety of reasons, this was an obvious choice. Python as a programming language has a sizable user base. A trip to the stack overflow can simply solve any potential issues. Python is one of the most popular languages on the site, making it simple to get an immediate response to any query. Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

5. MediaPipe

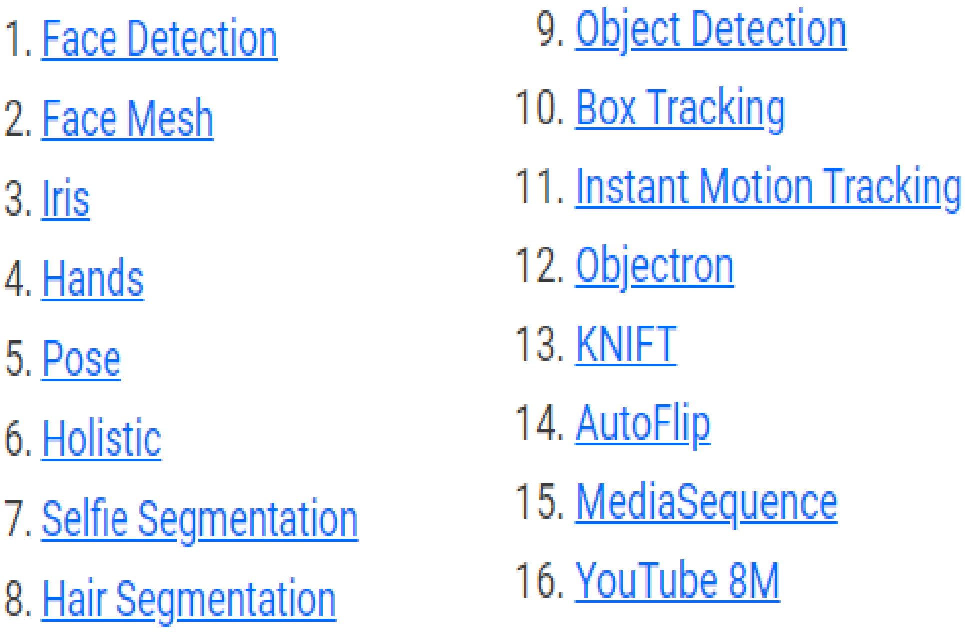
Google Cloud
Vision API
MediaPipe
LearnOpenCV.com

Figure 3: MediaPipe (Ref- https://learnopencv.com/introduction-to-mediapipe/)

MediaPipe is a framework mainly used for building multimodal audio, video, or any time series data. With the help of the MediaPipe framework, an impressive ML pipeline can be built for instance of inference models like TensorFlow, TFLite, and also for media processing functions. It is for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform Framework works in Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson Nano.

- **MediaPipe Solutions**

Solutions are open-source pre-built examples based on a specific pre-trained TensorFlow or TFLite model. You can check Solution specific models here. MediaPipe Solutions are built on top of the Framework. Currently, it provides sixteen solutions as listed below.



1. Face Detection
2. Face Mesh
3. Iris
4. Hands
5. Pose
6. Holistic
7. Selfie Segmentation
8. Hair Segmentation
9. Object Detection
10. Box Tracking
11. Instant Motion Tracking
12. Objectron
13. KNIFT
14. AutoFlip
15. MediaSequence
16. YouTube 8M

- **Synchronization and Performance Optimization**

MediaPipe supports multimodal graphs. To speed up the processing, different calculators run in separate threads. For performance optimization, many built-in calculators come with options for GPU acceleration. Working with time series data must be in proper synchronization; otherwise, the system will break. The graph ensures this so that flow is handled correctly according to the timestamps of packets. The Framework handles synchronization, context sharing, and inter-operations with CPU calculators.

6. NumPy

NumPy is a Python programming language library, adding support for multiple, multi-layered and mathematical editing, as well as a large collection of high-level mathematical functions running on these arrays. NumPy's ancestor, Numeric, was originally created byJim Hugunin with donations from several other developers. In 2005, Travis Oliphant made NumPy by adding the features of competing Numarray to Numeric, with multiple modifications. NumPy is open source software and has many sponsors

METHODOLOGY

We are going to recognize the pose of the model using pose estimation running on CPU to find the correct points and using these points we will get the desired angles, and then based on these angles we can find many gestures including number of workouts such as bicep curls. We want to find the angle using minimum three points to create two lines to tell us how much angle we are at and based on that we can do some calculations for the gestures. Below are the detailed steps discussed for how the model works.

1. **Install and setup**

   First up, we'll install and import our dependencies that we need which are MediaPipe and OpenCV. These two will help us gather data about the various joints in our body such as our wrists, shoulders, etc. for making our calculation with angles possible to count our repetitions with heavier weights.

   Secondly we would want NumPy which would help us with our trigonometry to calculate the angles.

2. **Make Detections**

   To make our detections possible, firstly we need to recolor our image because OpenCV renders the RGB image to BGR color format but for MediaPipe to work, we need to convert our BGR image back to RGB. Print the detections of our model.

   Lastly change the color format back to BGR format as OpenCV runs on BGR format, and then we can start rendering our detections.

3. **Determining Joints**

   There are 33 landmarks in total, starting from index 0. These represent the different joints within the pose. For instance, if we want to calculate the angle for our Right hand's bicep curl, we would require the joints of shoulder, elbow and wrist which are 12, 14 and 16 respectively as referred in **Figure 4**.

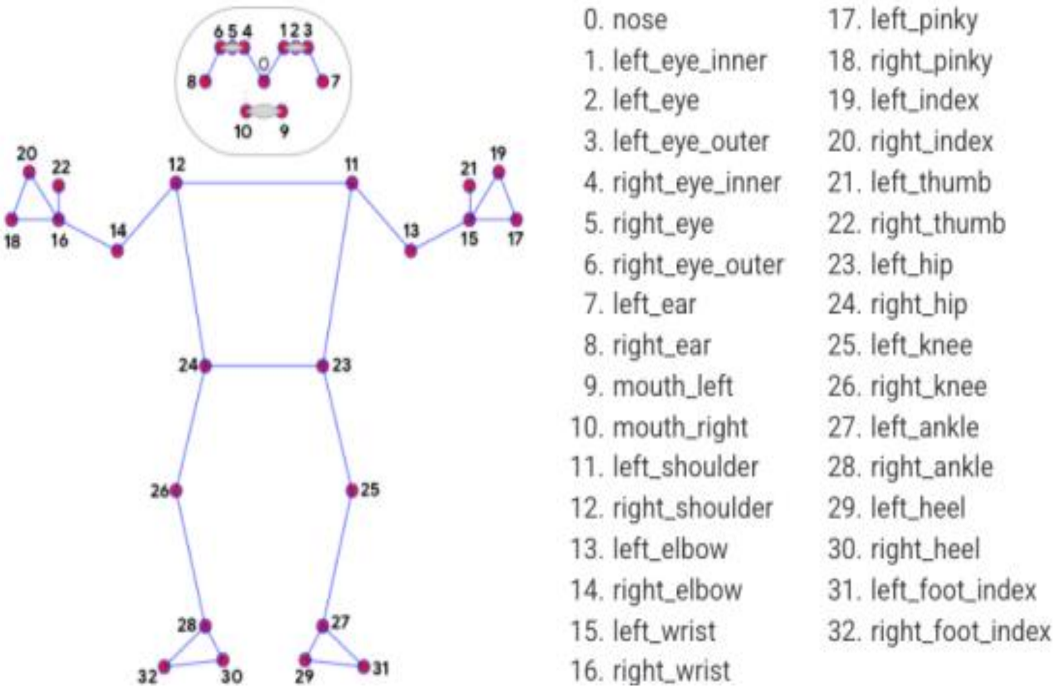
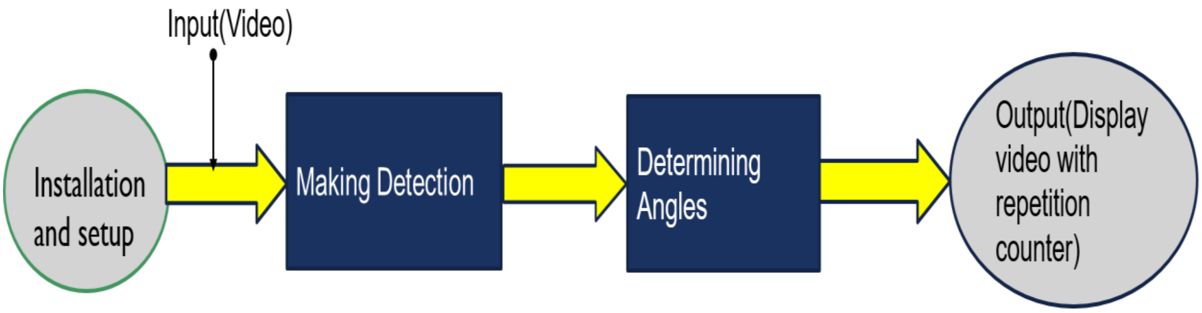


Figure 4: 33 points joints that would determine angle

4. Calculating Angles

First we get the coordinates of the three joints which we require to get the angle calculated. Then we can calculate the slopes of the joints using NumPy. Angles are calculated in radians which then can be converted into degrees.



ARCHITECTURE OF PROJECT

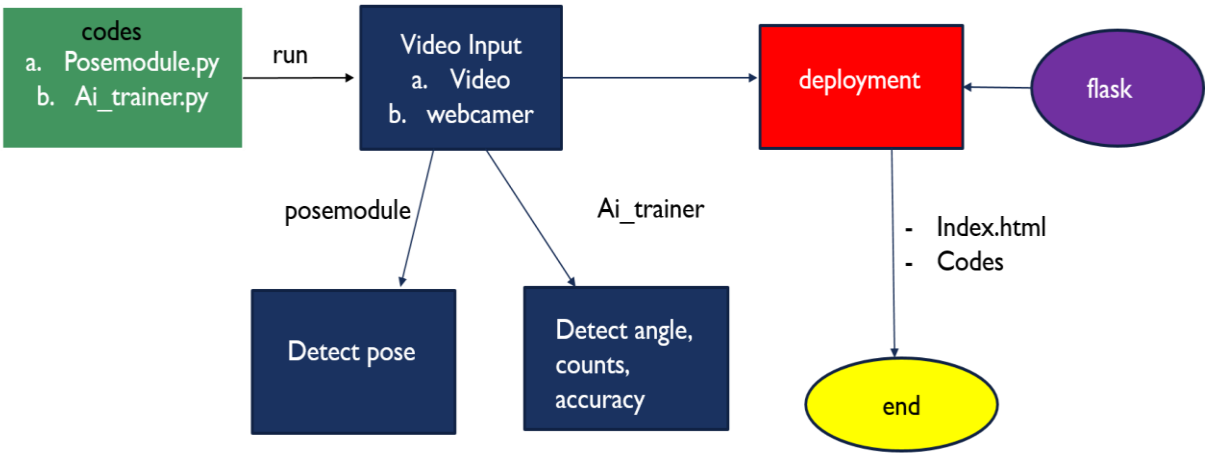


Figure: architecture of project

The project contain 2 python(.py) files when it gets executed the pose.py module detects the pose of the body where as the AI_Trainer.py Module detects angle, percentage and set counts. In the last stage we tried deployment where a index.html file was created with CSS and html files script. And deployment was concluded using flask.

DISCUSSION

1. Python

We have made 2 project modules named as (a)posemodule (b)AI_trainer

a) Posemodule.py

```
#Import Libraries
import numpy as np
import time
import cv2
import mediapipe as mp
import math
class poseDetector():
    def __init__(self, mode=False, upBody=False, smooth=True,
                 detectionCon=True, trackCon=True):
        self.mode = mode
        self.upBody = upBody
        self.smooth = smooth
        self.detectionCon = detectionCon
        self.trackCon = trackCon

        self.mpDraw = mp.solutions.drawing_utils
        self.mpPose = mp.solutions.pose
        self.pose = self.mpPose.Pose(self.mode, self.upBody, self.smooth,
self.detectionCon, self.trackCon)

    def findPose(self, img, draw=True):
        imgRGB=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
        self.results=self.pose.process(imgRGB)
        if self.results.pose_landmarks:
            if draw:
                self.mpDraw.draw_landmarks(img, self.results.pose_landmarks,
                                           self.mpPose.POSE_CONNECTIONS)
        return img
```

```
def findPosition(self, img, draw=True):
    self.lmList = []
    if self.results.pose_landmarks:
        for id, lm in enumerate(self.results.pose_landmarks.landmark):
            h, w, c = img.shape
            # print(id, lm)
            cx, cy = int(lm.x * w), int(lm.y * h)
            self.lmList.append([id, cx, cy])
            if draw:
                cv2.circle(img, (cx, cy), 3, (255, 0, 255), cv2.FILLED)
    return self.lmList

def findAngle(self, img, p1, p2, p3, draw=True):
    # Get the landmarks
    x1, y1 = self.lmList[p1][1:]
    x2, y2 = self.lmList[p2][1:]
    x3, y3 = self.lmList[p3][1:]
    # Calculate the Angle
    angle = math.degrees(math.atan2(y3 - y2, x3 - x2) -
                         math.atan2(y1 - y2, x1 - x2))
    print("Angle",angle)
    if angle < 0:
        angle += 360   # if angle i negative we use this line to make it positivw
    if draw:
        cv2.line(img, (x1, y1), (x2, y2), (255, 255, 255), 3)  #for lines b/w x and y
        cv2.line(img, (x3, y3), (x2, y2), (255, 255, 255), 3)
        cv2.circle(img, (x1, y1), 5, (0, 0, 255), cv2.FILLED)
        cv2.circle(img, (x1, y1), 10, (0, 0, 255), 2)
        cv2.circle(img, (x2, y2), 5, (0, 0, 255), cv2.FILLED) # for inner circle
        cv2.circle(img, (x2, y2), 10, (0, 0, 255), 2) #for outer circle
        cv2.circle(img, (x3, y3), 5, (0, 0,255), cv2.FILLED)
        cv2.circle(img, (x3, y3), 10, (0, 0, 255), 2)
        cv2.putText(img, str(int(angle)), (x2 - 10, y2 + 20),
                    cv2.FONT_HERSHEY_PLAIN, 3, (255, 120, 0), 3)
        return angle

def main():
    cap=cv2.VideoCapture("C://Users//Admin//Desktop//p2.mp4")
    pTime = 0
    detector = poseDetector()
    while True:
        success, img = cap.read()
        img = cv2.resize(img, (720, 720))
        img = detector.findPose(img,draw=True)
        lmList = detector.findPosition(img, draw=False)
        #print(lmList)
        if len(lmList) != 0:
            print(lmList[22])
            cv2.circle(img, (lmList[22][1], lmList[22][2]), 10, (130, 255, 255),
cv2.FILLED)
        cTime = time.time()
        fps = 1 / (cTime - pTime)
        pTime = cTime
        cv2.putText(img, str(int(fps)), (70, 50), cv2.FONT_HERSHEY_PLAIN, 1, (255,
0, 0), 1)
        cv2.imshow("Image", img)
        key=cv2.waitKey(1)
        if key == ord("q") or key == ord("Q"):
            break
    cap.release()
    cv2.destroyAllWindows()

if __name__=="__main__":
    main()
```

b) AI_Trainer

```
"IMPORTING THE LIBRARIES"

import posemodule as pm

from flask import Flask, render_template, Response, request, redirect, url_for

import numpy as np

import cv2

app = Flask(__name__)

@app.route('/')

def index():


    return render_template('index.html')


@app.route('/show')

def plot():

    cap=cv2.VideoCapture("C://Users//Admin//Desktop//p2.mp4")

    detector=pm.poseDetector()

    count = 0

    dir = 0

    while True:

        success, img=cap.read()

        img=cv2.resize(img,(600,400))

        img=detector.findPose(img,False) #for draw line

        lmList=detector.findPosition(img,False) #for creating dots

        #print(lmList)
```

```
if len(lmList)!=0:
    #These points is for left arm
    angle=detector.findAngle(img,11,13,15)
    per = np.interp(angle, (210,310), (0, 100)) #2nd shows the  to 100%
    print("PERCENT LEVEL",per)
    # Check for the Angle
    bar = np.interp(angle, (220, 310), (400, 100))  #2nd shows bar height n width
    # Check for the dumbbell curls
    color = (255, 0, 255)

    if per == 100:
        color = (0, 255, 0)
        if dir == 0:
            count += 0.5
            dir = 1
    if per == 0:
        color = (0, 0, 255)
        if dir == 1:
            count += 0.5
            dir = 0
    print("SETS COUNTS",count)
    # Draw Bar
    cv2.rectangle(img, (250, 100), (200, 400), color, 2)
    cv2.rectangle(img, (250, int(bar)), (200, 400), color, cv2.FILLED)
```

```
            cv2.putText(img, f'{int(per)} %', (210, 65), cv2.FONT_HERSHEY_PLAIN, 4,
                        color, 4)
            #DRAW CURLS
            cv2.rectangle(img, (0, 550), (100, 700), (0, 255, 0), cv2.FILLED)
            cv2.putText(img, str(int(count)), (20, 620), cv2.FONT_HERSHEY_PLAIN, 5,
                        (255.0,0), 5)


        cv2.imshow("WEBCAM",img)
        key=cv2.waitKey(1)
        if key == ord("q") or key == ord("Q"):
            break


    cap.release()
    cv2.destroyAllWindows() #realesing the webcam


    return render_template('index.html')


if __name__ == '__main__':
    app.run(host='0.0.0.0',debug=False,port=5000)
```

2) Input video file

- Some home exercise video have been recorded provided as input this is a simple hand exercise to and fro movements

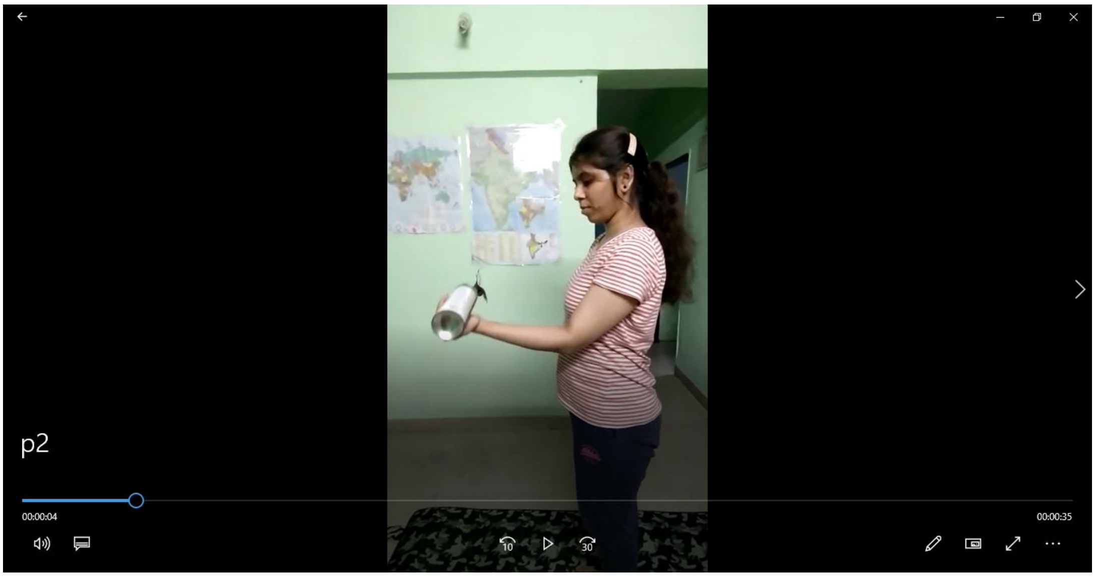


Figure 5: Video name "p2.mp4"

- Using web camera live exercise and inputs can be captured too

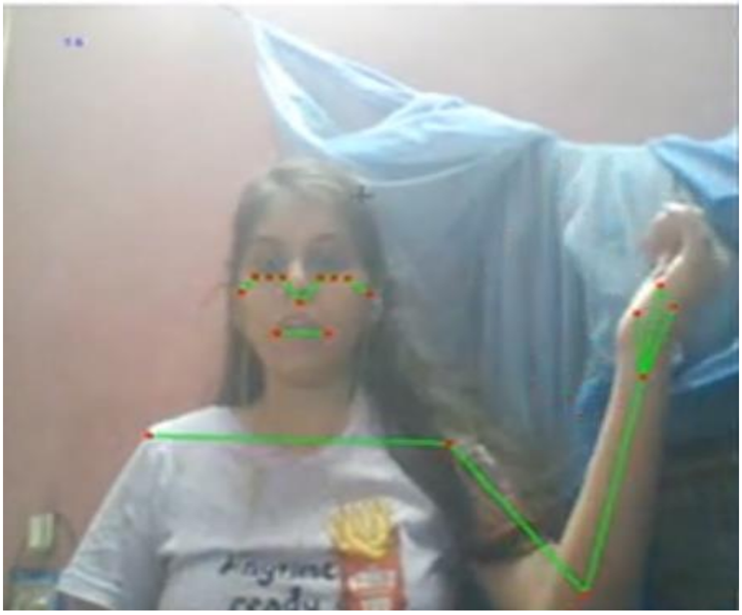


Figure 6: Live Video Input shown using web camera

In the above video the posemodule will detect the body pose and adjoint with the circular points.

- In the AI_TRAINER module there is a prediction of angles, percentage and then the counts.

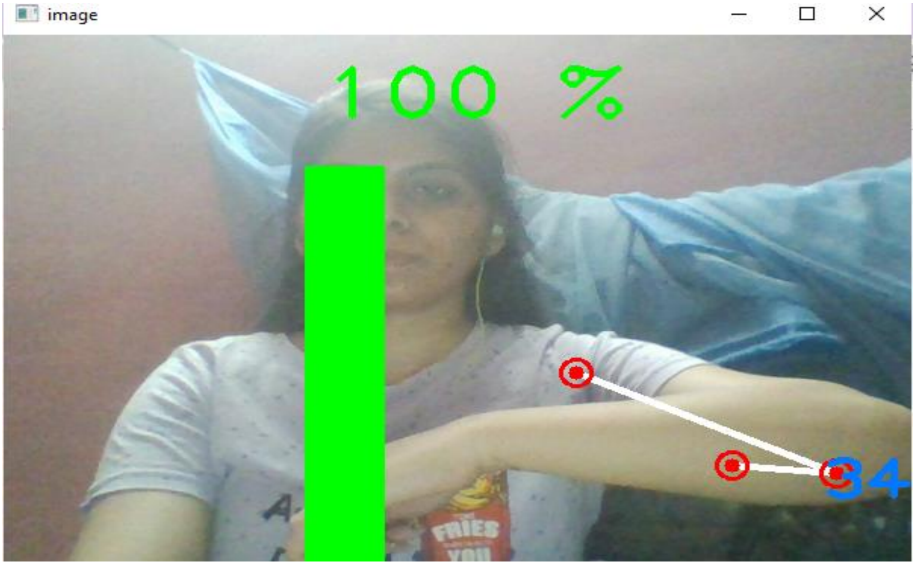
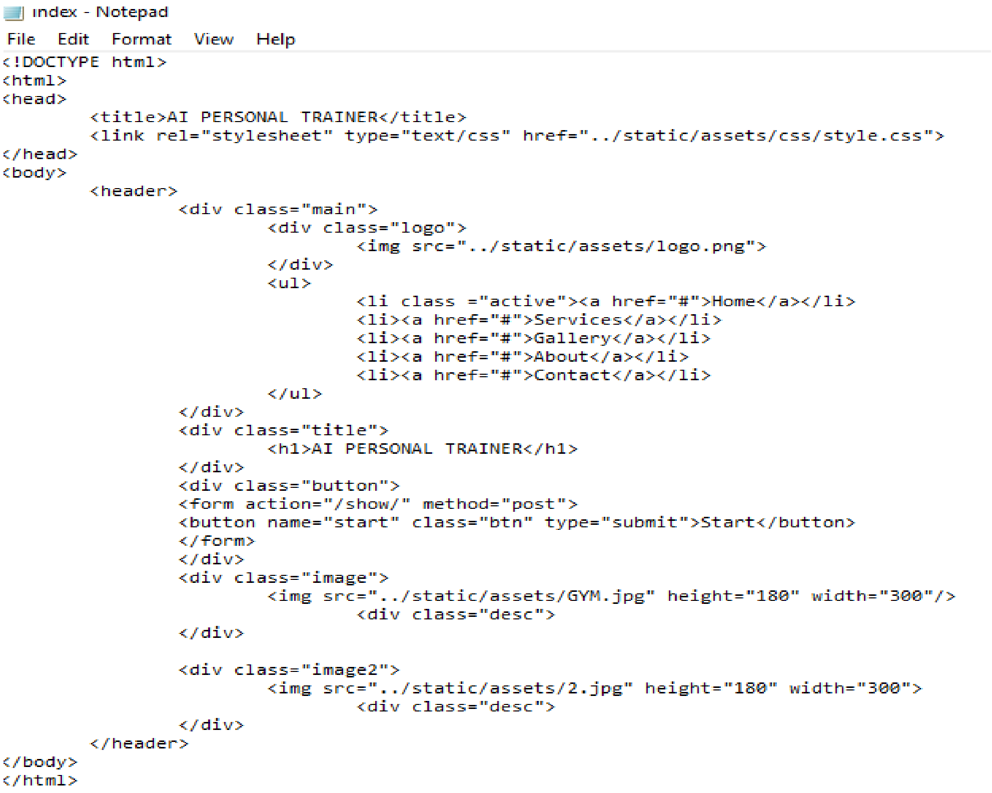


Figure 7: Output with the angles and accuracy shown

2. HTML CSS webpage

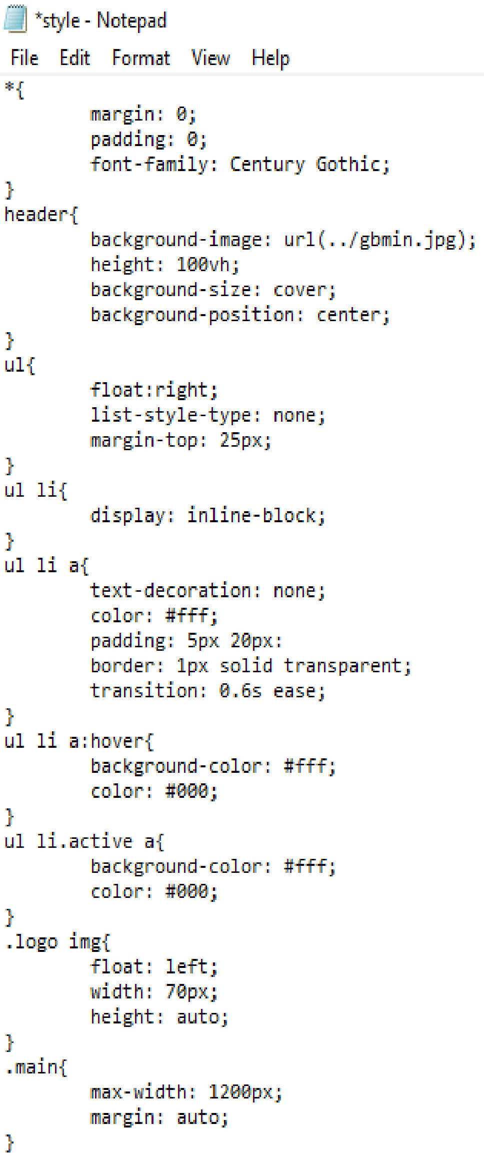
Html page



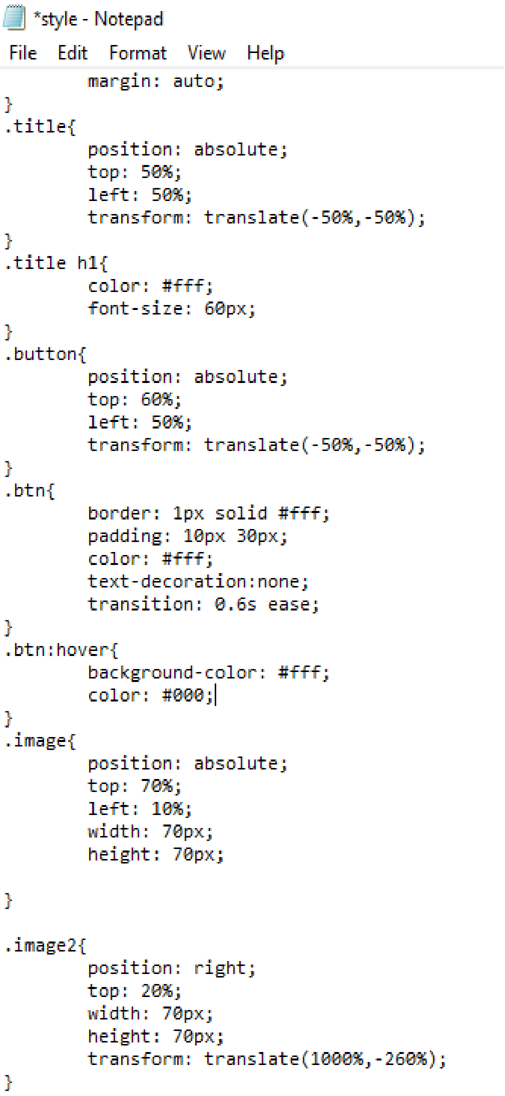
```
<!DOCTYPE html>
<html>
<head>
        <title>AI PERSONAL TRAINER</title>
        <link rel="stylesheet" type="text/css" href="../static/assets/css/style.css">
</head>
<body>
        <header>
                <div class="main">
                        <div class="logo">
                                <img src="../static/assets/logo.png">
                        </div>
                        <ul>
                                <li class ="active"><a href="#">Home</a></li>
                                <li><a href="#">Services</a></li>
                                <li><a href="#">Gallery</a></li>
                                <li><a href="#">About</a></li>
                                <li><a href="#">Contact</a></li>
                        </ul>
                </div>
                <div class="title">
                        <h1>AI PERSONAL TRAINER</h1>
                </div>
                <div class="button">
                <form action="/show/" method="post">
                <button name="start" class="btn" type="submit">Start</button>
                </form>
                </div>
                <div class="image">
                        <img src="../static/assets/GYM.jpg" height="180" width="300"/>
                                <div class="desc">
                </div>

                <div class="image2">
                        <img src="../static/assets/2.jpg" height="180" width="300">
                                <div class="desc">
                </div>
        </header>
</body>
</html>
```

Figure 8: HTML PAGE

Css page



```
*{
	margin: 0;
	padding: 0;
	font-family: Century Gothic;
}
header{
	background-image: url(../gbmin.jpg);
	height: 100vh;
	background-size: cover;
	background-position: center;
}
ul{
	float:right;
	list-style-type: none;
	margin-top: 25px;
}
ul li{
	display: inline-block;
}
ul li a{
	text-decoration: none;
	color: #fff;
	padding: 5px 20px:
	border: 1px solid transparent;
	transition: 0.6s ease;
}
ul li a:hover{
	background-color: #fff;
	color: #000;
}
ul li.active a{
	background-color: #fff;
	color: #000;
}
.logo img{
	float: left;
	width: 70px;
	height: auto;
}
.main{
	max-width: 1200px;
	margin: auto;
}
```



```
	margin: auto;
}
.title{
	position: absolute;
	top: 50%;
	left: 50%;
	transform: translate(-50%,-50%);
}
.title h1{
	color: #fff;
	font-size: 60px;
}
.button{
	position: absolute;
	top: 60%;
	left: 50%;
	transform: translate(-50%,-50%);
}
.btn{
	border: 1px solid #fff;
	padding: 10px 30px;
	color: #fff;
	text-decoration:none;
	transition: 0.6s ease;
}
.btn:hover{
	background-color: #fff;
	color: #000;
}
.image{
	position: absolute;
	top: 70%;
	left: 10%;
	width: 70px;
	height: 70px;

}

.image2{
	position: right;
	top: 20%;
	width: 70px;
	height: 70px;
	transform: translate(1000%,-260%);
}
```

Figure 8: css page

Output:

The Html and CSS webpages are made as shown and the output is as follows:

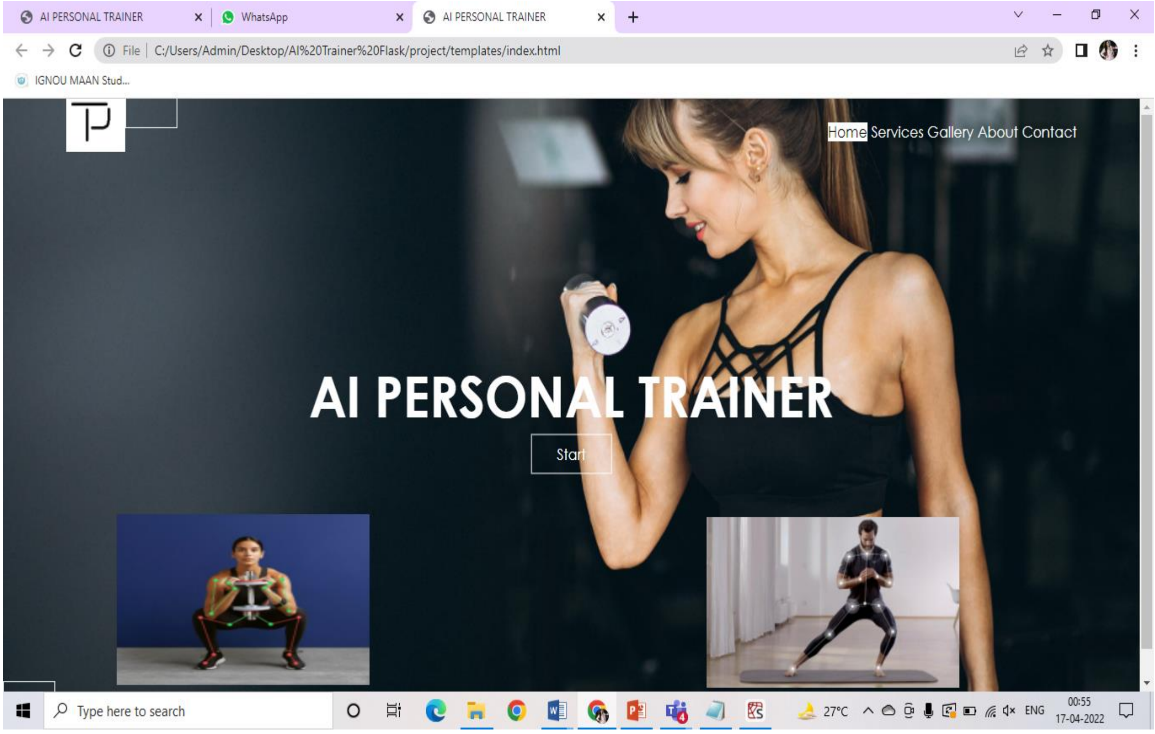
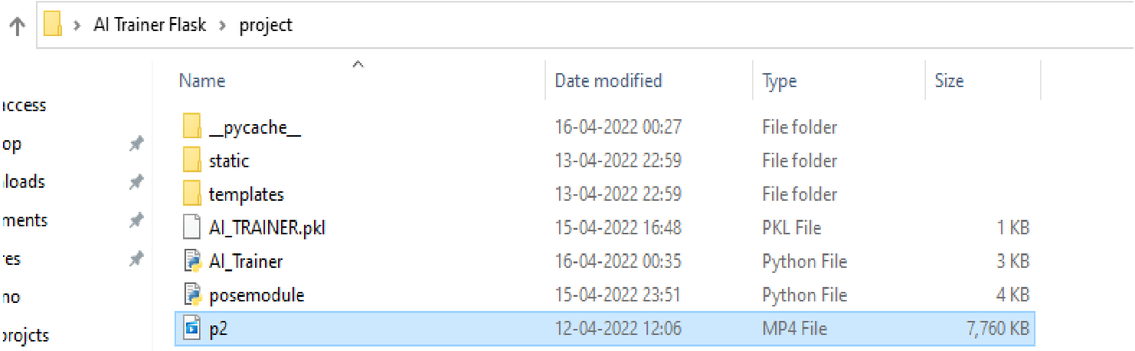


Figure 9: HTML and CSS Webpage Output

Folder for DEPLOYMENT

- All this files to be placed in a single folder for deployment



Name	Date modified	Type	Size
__pycache__	16-04-2022 00:27	File folder	
static	13-04-2022 22:59	File folder	
templates	13-04-2022 22:59	File folder	
AI_TRAINER.pkl	15-04-2022 16:48	PKL File	1 KB
AI_Trainer	16-04-2022 00:35	Python File	3 KB
posemodule	15-04-2022 23:51	Python File	4 KB
p2	12-04-2022 12:06	MP4 File	7,760 KB

Figure 10: Local Drive folder which consists of all the files

ANALYSIS AND RESULT

I) Video as input

a) Posemodule.py

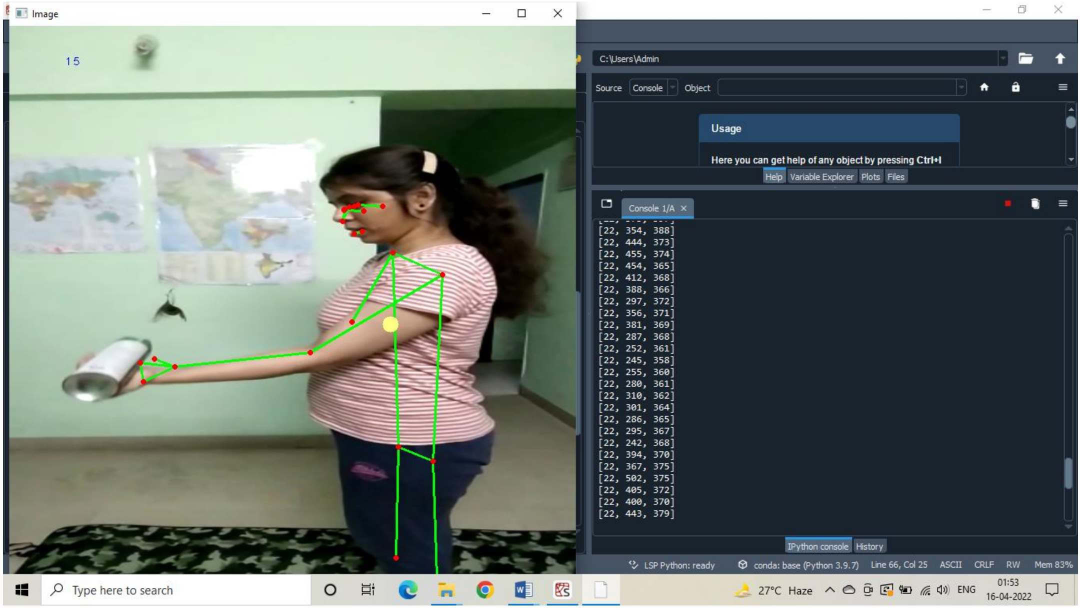
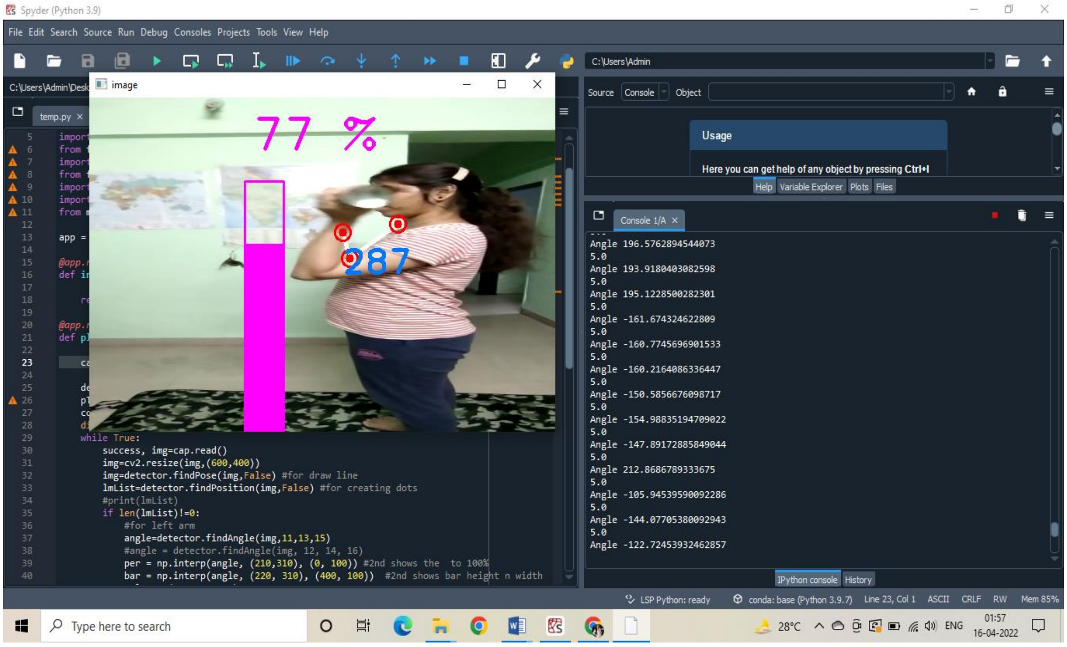


Image
15
C:\Users\Admin
Source Console Object
Usage
Here you can get help of any object by pressing Ctrl+I
Help Variable Explorer Plots Files
Console 1/A
IPython console History
LSP Python: ready conda: base (Python 3.9.7) Line 66, Col 25 ASCII CRLF RW Mem 83%
Type here to search
27°C Haze ENG 01:53 16-04-2022

```
[22, 354, 388]
[22, 444, 373]
[22, 455, 374]
[22, 454, 365]
[22, 412, 368]
[22, 388, 366]
[22, 297, 372]
[22, 356, 371]
[22, 381, 369]
[22, 287, 368]
[22, 252, 361]
[22, 245, 358]
[22, 255, 360]
[22, 280, 361]
[22, 310, 362]
[22, 301, 364]
[22, 286, 365]
[22, 295, 367]
[22, 242, 368]
[22, 394, 370]
[22, 367, 375]
[22, 502, 375]
[22, 405, 372]
[22, 400, 370]
[22, 443, 379]
```

Figure 11: posemodule with detecting joints on consoles

b) AI_Trainer.py


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Admin\Desk
temp.py
image
77 %
287
C:\Users\Admin
Source Console Object
Usage
Here you can get help of any object by pressing Ctrl+I
Help Variable Explorer Plots Files
Console 1/A
IPython console History
LSP Python: ready conda: base (Python 3.9.7) Line 23, Col 1 ASCII CRLF RW Mem 85%
Type here to search
28°C ENG 01:57 16-04-2022

```
import
from
import
from
import
import
from

app =

@app.
def i

    re

@app.
def pl

    ca

    de
    pl
    co
    di
    while True:
        success, img=cap.read()
        img=cv2.resize(img,(600,400))
        img=detector.findPose(img,False) #for draw line
        lmList=detector.findPosition(img,False) #for creating dots
        #print(lmList)
        if len(lmList)!=0:
            #for left arm
            angle=detector.findAngle(img,11,13,15)
            #angle = detector.findAngle(img, 12, 14, 16)
            per = np.interp(angle, (210,310), (0, 100)) #2nd shows the  to 100%
            bar = np.interp(angle, (220, 310), (400, 100))  #2nd shows bar height n width
```

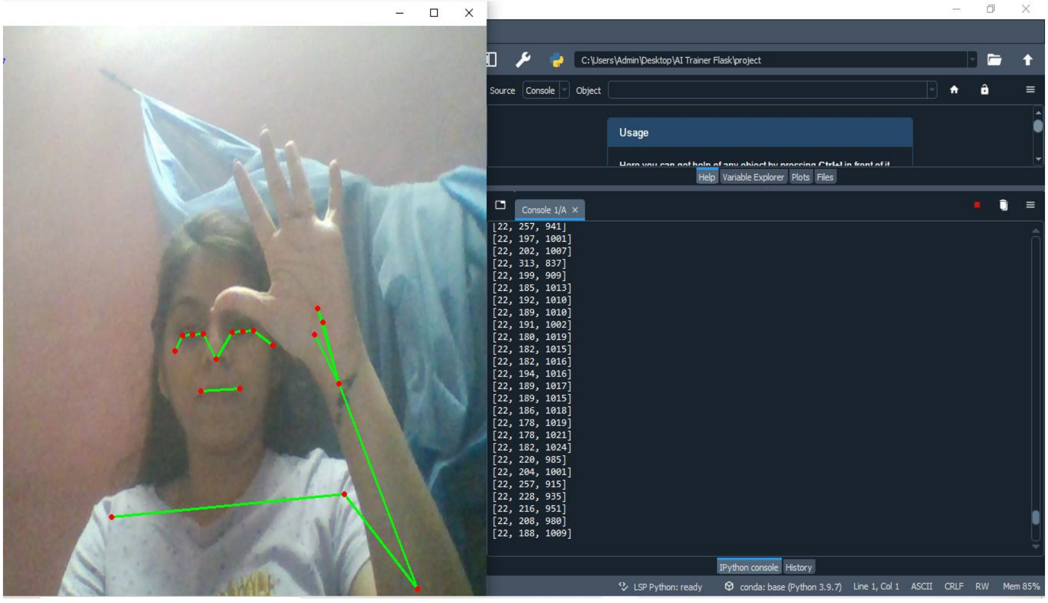
```
Angle 196.5762894544073
5.0
Angle 193.9180403082598
5.0
Angle 195.1228500282301
5.0
Angle -161.674324622809
5.0
Angle -160.7745696901533
5.0
Angle -160.2164086336447
5.0
Angle -150.5856676098717
5.0
Angle -154.98835194709022
5.0
Angle -147.89172885849044
5.0
Angle 212.8686789333675
5.0
Angle -105.94539590092286
5.0
Angle -144.07705380092943
5.0
Angle -122.72453932462857
```

Figure 12: AI_Trainer angles

II) Web camera as input

1. Python

a) posemodule.py

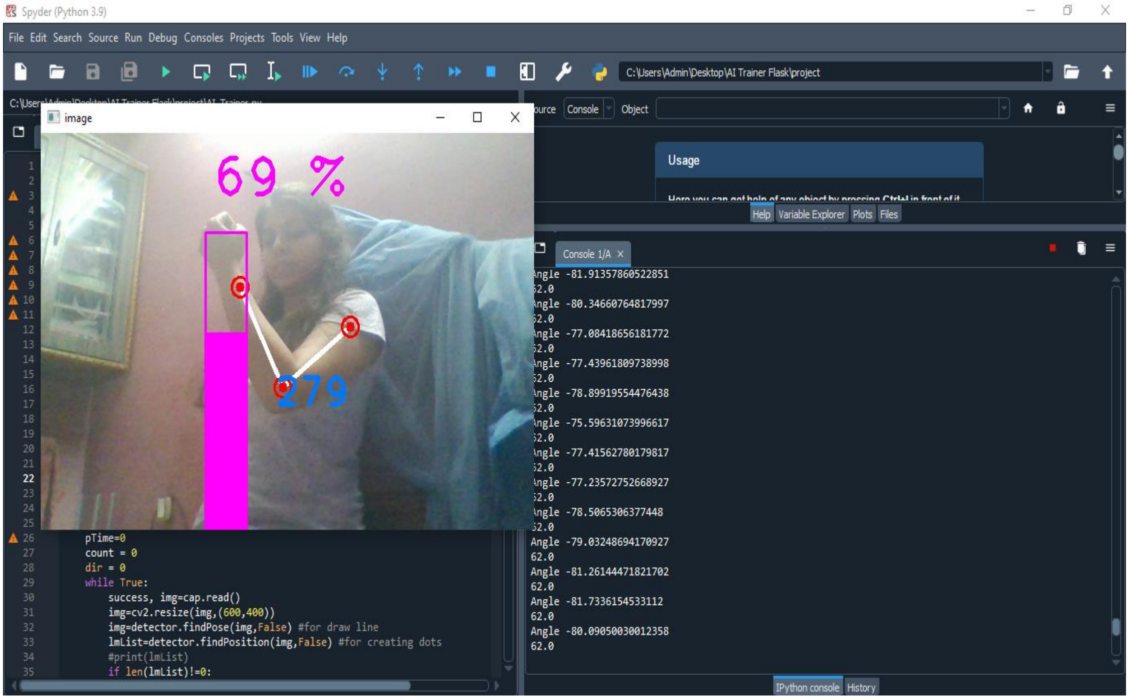


```
[22, 257, 941]
[22, 197, 1001]
[22, 202, 1007]
[22, 313, 837]
[22, 199, 909]
[22, 185, 1013]
[22, 192, 1010]
[22, 189, 1010]
[22, 191, 1002]
[22, 180, 1019]
[22, 182, 1015]
[22, 182, 1016]
[22, 194, 1016]
[22, 189, 1017]
[22, 189, 1015]
[22, 186, 1018]
[22, 178, 1019]
[22, 178, 1021]
[22, 182, 1024]
[22, 220, 985]
[22, 204, 1001]
[22, 257, 915]
[22, 228, 935]
[22, 216, 951]
[22, 208, 980]
[22, 188, 1009]
```

Figure 13: posemodule and console output

The posemodule.py files shows the estimation pose of the body and it calculates the x,y,z joints

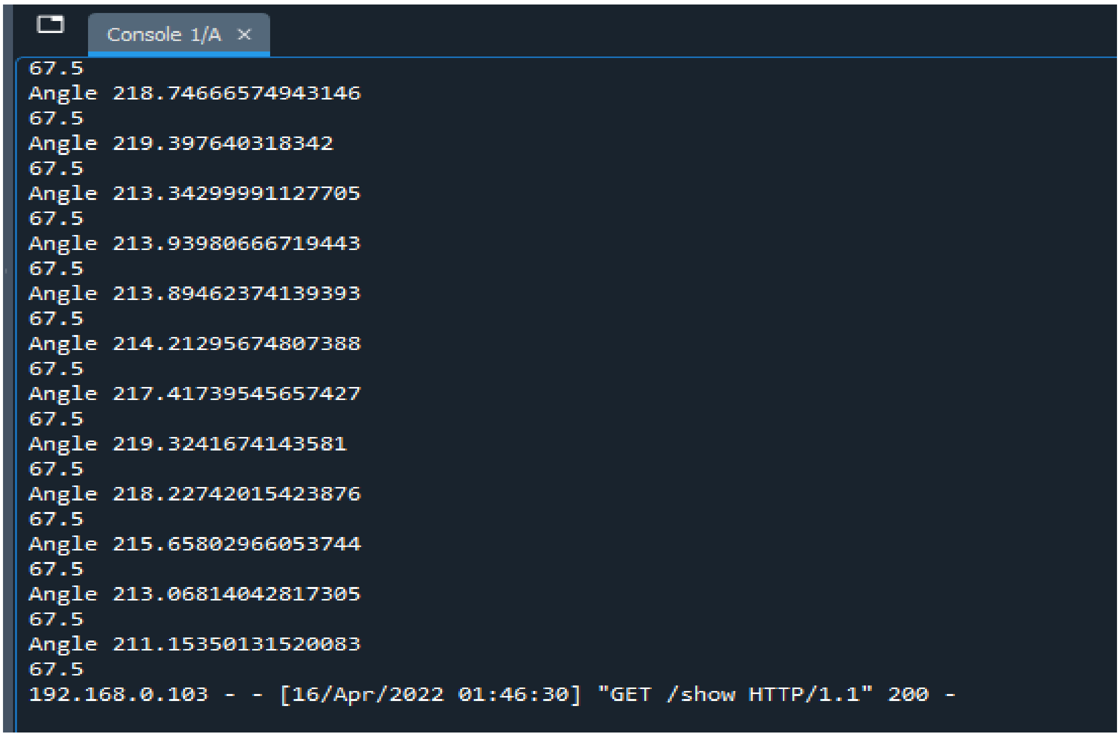
b) AI_trainer.py



```
pTime=0
count = 0
dir = 0
while True:
    success, img=cap.read()
    img=cv2.resize(img,(600,400))
    img=detector.findPose(img,False) #for draw line
    lmList=detector.findPosition(img,False) #for creating dots
    #print(lmList)
    if len(lmList)!=0:
```

```
Angle -81.91357860522851
62.0
Angle -80.34660764817997
62.0
Angle -77.08418656181772
62.0
Angle -77.43961809738998
62.0
Angle -78.89919554476438
62.0
Angle -75.59631073996617
62.0
Angle -77.41562780179817
62.0
Angle -77.23572752668927
62.0
Angle -78.5065306377448
62.0
Angle -79.03248694170927
62.0
Angle -81.26144471821702
62.0
Angle -81.7336154533112
62.0
Angle -80.09050030012358
62.0
```

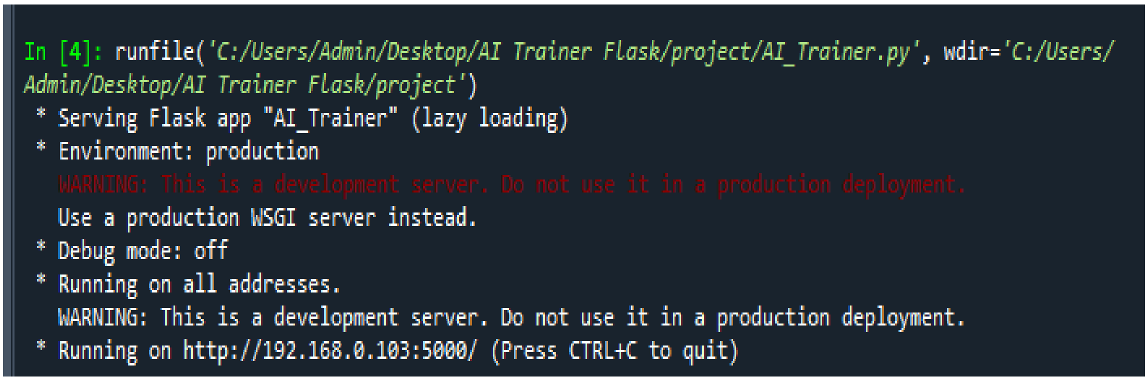
Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Admin\Desktop\AI Trainer Flask\project
image
69 %
279
Console
Object
Usage
Help
Variable Explorer
Plots
Files
Console 1/A
IPython console
History



```
67.5
Angle 218.74666574943146
67.5
Angle 219.397640318342
67.5
Angle 213.34299991127705
67.5
Angle 213.93980666719443
67.5
Angle 213.89462374139393
67.5
Angle 214.21295674807388
67.5
Angle 217.41739545657427
67.5
Angle 219.3241674143581
67.5
Angle 218.22742015423876
67.5
Angle 215.65802966053744
67.5
Angle 213.06814042817305
67.5
Angle 211.15350131520083
67.5
192.168.0.103 - - [16/Apr/2022 01:46:30] "GET /show HTTP/1.1" 200 -
```

Figure 14: Angles with counts and accuracy

AI_Trainer.py



```
In [4]: runfile('C:/Users/Admin/Desktop/AI Trainer Flask/project/AI_Trainer.py', wdir='C:/Users/
Admin/Desktop/AI Trainer Flask/project')
 * Serving Flask app "AI_Trainer" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://192.168.0.103:5000/ (Press CTRL+C to quit)
```

Figure 15: AI_Trainer module generating https link to be run on browser

This https link is generated after deployment else it wont be generated if fot deployed. Once the link is thrown in browser we displayed the below webpage as shown in **figure 16**

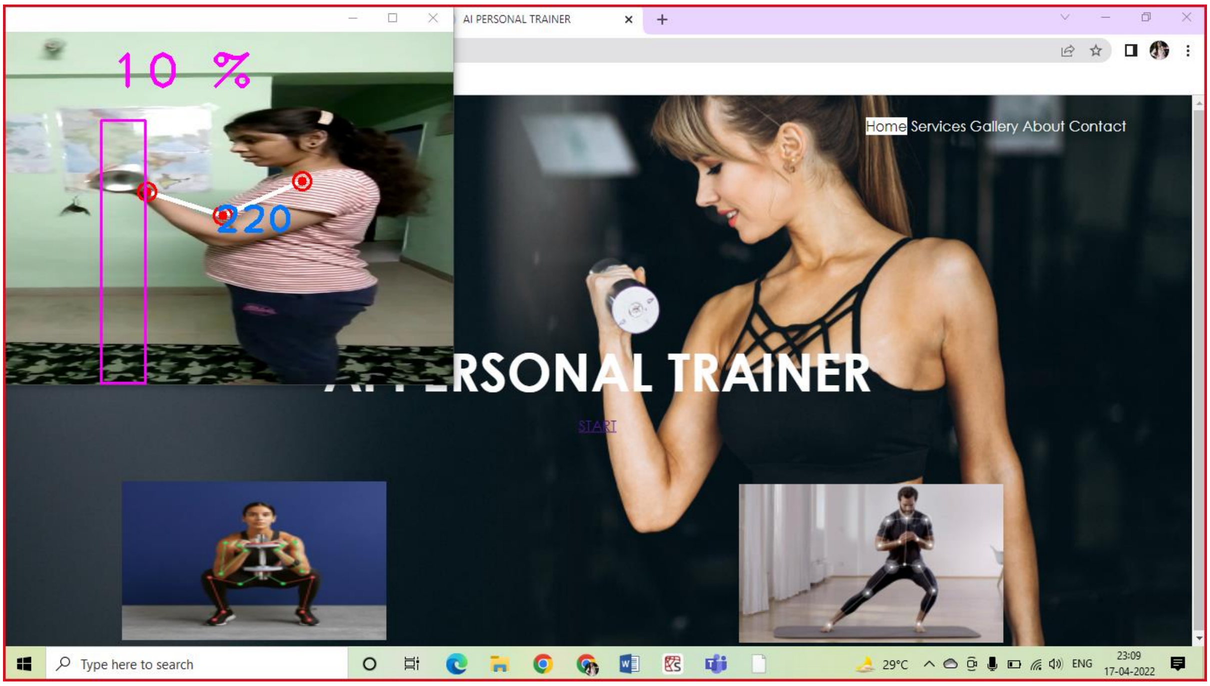


Figure 16: WEBPAGE after deployment

Console Output:

angle	Percentage level	Set count
194.43322848540345	0.0	3.0
-0.7953144929356427	100.0	3.5
336.6115507488483	100.0	3.5
212.7632043319199	2.7632043319198942	3.0

Figure 17: Console output showing angle, perct. Level, set counts

BENEFIT:

Saving time to go to a gym and get a personal trainer might cost you an arm and a leg in this rapidly and chaotic existence. We're not joking! We're not kidding! Personal trainers with significant expertise may not be cheap for everybody. But integrated fitness applications with artificial intelligence are a fantastic approach for those who want their fitness objectives to be achieved.

More excitingly, the AI trainers operate like human trainers; sure, they advise you on the proper way to exert. You may also have a one-on-one setting and receive feedback throughout an exercise on your posture.

AI-driven human pose estimation technology is a breakthrough that is interesting for AI personal trainer applications.

CONCLUSION AND FUTURE WORK

Nowadays our life is becoming busier and we hardly find time in our schedules to be healthy and fit and exercise daily. This has caused many diseases and health issues. Implementation of Artificial Intelligence in the field of fitness can solve many problems. The health-related applications and devices are making our lives easier and eases our fitness journey. Individuals can use this application in their own workouts, hence making them more efficient are less error-prone. In this process, we learnt how to use the OpenCV library and package and how the application of machine learning can be beneficial to humans.

There is a lot of scope of development in this project. The project can be upgraded to support more exercises. A User interface can be added for easy navigation through the exercises. The data collected by the AI trainer can be saved and processed for the next sessions. Daily steps tracker can also be added. The trainer will suggest you workout plan and its intensity according to your body type and weight. This application can be developed into a complete android/ios application for ease of use.

From the brief insight provided above, it shows that AI- based workout assistant and fitness guide uses some concepts of blaze pose, requires a camera to capture the body pose as input to the system generated and with the help of pose estimator, will provide the stats of calories burnt and exercise count as output in human-readable form.

Future work may include the movement of the camera vertically and horizontally to capture another wide variety of exercises or it may include the use of multiple cameras to capture the body pose from various angles in order to feed the template of other exercises.

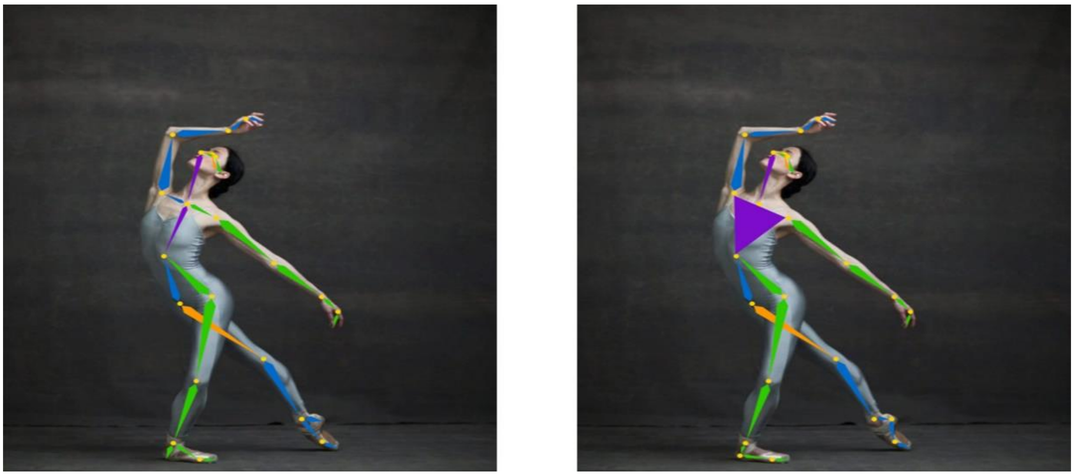


Figure 17: AI Personal Dance Trainer

REFERENCES

1. https://www.computervision.zone/courses/personal-ai-trainer/
2. https://www.rs-online.com/designspark/personal-ai-trainer-project-introduction-part-i
3. Baumbach, Sebastian & Bhatt, Arun & Ahmed, Sheraz & Dengel, Andreas. (2018). Towards a Digital Personal Trainer for Health Clubs - Sport Exercise Recognition Using Personalized Models and Deep Learning. 438-445. 10.5220/0006590504380445.
4. http://www.ijaresm.com/uploaded_files/document_file/Shwetank_KardamqCQK.pdf
5. https://google.github.io/mediapipe/solutions/pose