1

FITNESS TRACKER USING MEDIAPIPE POSE CLASSIFICATION

A PROJECT REPORT

Submitted by

Kesav.S.J

Madhav Hari

Barath.J

in partial fulfillment for the award of the

degree of

Bachelor Of Technology

IN

Information Technology

SRI VENKATESWARA COLLEGE OF ENGINEERING
(An Autonomous Institution; Affiliated to Anna University, Chennai -600 025)
ANNA UNIVERSITY :: CHENNAI 600 025

May 2023

SRI VENKATESWARA COLLEGE OF ENGINEERING
(An Autonomous Institution; Affiliated to Anna University, Chennai -600 025)

ANNA UNIVERSITY, CHENNAI – 600 025

BONAFIDE CERTIFICATE

Certified that this project report **“Sentiment Analysis On Twitter Tweets”** is the bonafide work of “**Kesav.S.J,Madhav Hari,Barath.J ”** who carried out the project work under my/our supervision.

SIGNATURE	**SIGNATURE**
Dr.V.Vidhya, M.E.Ph.D.,	**Dr.V.M.Sivagami,M.E,Ph.D.,**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	PROFESSOR
INFORMATION TECHNOLOGY	**INFORMATION TECHNOLOGY**

Submitted for the project viva-voice examination held on

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Fitness tracker makes exercise easier for people to maintain a healthy lifestyle despite their busy schedules. By using the mediapipe pose classification system, users can get accurate activity scores to track their progress and stay motivated. The app does not rely on large datasets and can be easily deployed on any cloud environment and device. It is designed to work with any type of fitness workout, whether rep-based or duration-based, and can quickly scale in any cloud environment.

Change of approach from rep-based to joint angle-based resulted in increased accuracy of action detection and comparison. However, unnecessary joint angles caused less accuracy for a particular exercise. Hence implementation of an angle-omitting logic using exercise category and point visibility was required to eliminate pointless joint angles.

Real-time workout videos and user testing showed the algorithm can handle most edge cases. Once integrated with the web, the app will be ready for real-world users. Our app has the potential to help people prioritize their fitness in today's fast-paced lifestyle.

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to Sri Venkateswara College Of Engineering for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

It is a matter of immense pleasure to express our sincere thanks to Dr. V.Vidhya, Head of the Departments, Information Technology, Sri Venkateswara College Of Engineering ,for providing the right academic guidance that made our task possible.

We would like to thank our guide Dr.V.M Sivagami Professor, Dept. of Information Technology , Sri Venkateswara College Of Engineering, for sparing her valuable time to extend help in every step of our project work, which paved the way for smooth progress and the fruitful culmination of the project.

We also would like to thank all the staff members of Information Technology for their support.

We are also grateful to our family and friends who provided us with every requirement throughout the course. We would like to thank one and all who directly or indirectly helped us in the Project work.

Kesav.S.J

Madhav Hari

Barath.J

TABLE OF CONTENTS

LIST OF FIGURES

8.11	**Scoring logic(Multi-pose).**	
9.1.1	**Single-pose exercise reference**	
9.1.2	**Single-pose exercise**	
9.1.3	**Single-pose scoring**	
9.1.4	**Multi-pose exercise reference**	
9.1.5	**Multi-pose exercise reference**	
9.1.6	**Multi-pose exercise**	
9.1.7	**Multi-pose exercise**	
9.1.8	**Multi-pose scoring**	
9.2.1	**Without doing the exercise**	
9.2.2	**Displaying score**	
9.2.3	**No one in the frame**	
9.2.4	**Displaying score**	

CHAPTER - 1

INTRODUCTION

CHAPTER 1 INTRODUCTION

In today's fast-paced world, maintaining a healthy and active lifestyle has become increasingly important. Fitness applications have emerged as powerful tools to support individuals in achieving their fitness goals by providing workout tracking, guidance, and motivation. However, ensuring the efficiency and accuracy of workouts remains a significant challenge. To address this, our project focuses on leveraging MediaPipe Pose Classification, a cutting-edge technology, to enhance the efficiency and accuracy of workouts within a fitness application.

MediaPipe Pose Classification is a component of the MediaPipe framework developed by Google, which provides a comprehensive solution for building cross-platform, real-time computer vision applications. Pose detection refers to the task of estimating the pose or body landmarks of a person in an image or video. It involves identifying key points on the human body, such as the positions of the joints (e.g., elbows, knees) and other body parts (e.g., head, shoulders). These body landmarks are crucial for fitness tracking.

MediaPipe Pose Classification excels at recognizing and categorizing specific exercise poses based on the detected body landmarks. By integrating this technology into a fitness web application, the system can accurately identify exercises such as push-ups, squats, lunges, or yoga poses the user performs by comparing their detected poses with the expected ideal poses uploaded by the administrator. The application can provide immediate feedback to users, this real-time feedback helps users maintain proper form, prevent injuries, and optimize their workout efficiency.

The primary objective of this project is to develop a fitness application that utilizes MediaPipe Pose Classification to improve the efficiency and accuracy of users' workouts. MediaPipe Pose Classification typically involves training a machine learning model on labeled pose data to classify different poses or actions. This model can then be integrated into the pose detection pipeline to enhance the system's capabilities.

1.1 PURPOSE

The purpose of this project is to develop a fitness application that utilizes MediaPipe Pose Classification to enhance the efficiency and accuracy of workouts. By leveraging the advanced pose recognition capabilities of MediaPipe.

The goal of this project is to make it easier for people to live a healthy lifestyle and to inspire them to exercise despite their hectic schedules. To obtain an exact efficiency of the workouts performed by users using our application, which might assist them in improving their efficiency.

1.2 SCOPE

This model will assist those who have a hectic schedule and are unable to incorporate exercise into their routine. Because we live in a fast-paced world, many of us will require an effective and accurate online application that can give the accuracy of the exercises we complete and is user friendly. This application will benefit every busy citizen in the country, and we discovered that existing solutions are inadequate when we compared them.

CHAPTER 2
LITERATURE REVIEW

[1] B. Natarajan, E. Rajalakshmi. This paper contributes to the development of a deep learning framework for end-to-end sign language recognition, translation, and generation. We addressed the challenges that persist with earlier SL recognition and video generation approaches using the proposed H-DNA framework. We evaluated the model performance using the RWTH-PHOENIX- Weather 2014T dataset, the How2Sign dataset, and the ISL-CSLTR datasets quantitatively and qualitatively. The proposed H-DNA framework is also evaluated qualitatively using various quality metrics.

[2] Sanjal Gupta , Sadhana Singh. In this study, a yoga posture classifier that is excellent for photos, static video, and live video of any user was successfully constructed. The construction of the study's setting is the first step, and then open data sources are used to collect data.

[3] Ardra Anilkumar , Athulya K.T. This system made it easier to do exercises without the need for a special trainer. Reduce injuries due to improper technique.

[4] Khushi Sidana.This study successfully implemented systems for the detection and evaluation of yoga poses based on deep learning have been developed in the past using a variety of different techniques.

[5]. Shuo Zhang,Wanmi Chen.This paper presents a deep squat motion detection model based on a combination of mediapipe and the network structure of yolo v5, the extremely fast detection speed of the mediapipe algorithm combined with the accuracy of the yolo v5 detection target improves.

[6] Yejin Kwon, Dongho Kim. The program proposed in this study converts images of users obtained through webcams by OpenCV and uses the Landmark model of MediaPipe Pose to estimate body landmarks

.

[7] Barathi Subramanian , Jeonghong Kim.Recognition of patient emotions has become increasingly important as it helps in many different areas, including the medical sector for personalized healthcare. The ability to identify an individual's emotions can help healthcare centers build smart diagnostic tools such as an ER system that can detect depression and stress among the patients in the early stages so that the medication can be given in prior.

CHAPTER 3
EXISTING AND PROPOSED SYSTEM

CHAPTER 3 EXISTING AND PROPOSED SYSTEM

This chapter presents an overview of the existing and proposed System.

3.1 EXISTING SYSTEM

There are few existing solutions, for example, Kaia Health App provides a similar kind of application where the user's movement will be tracked and real-time suggestions will be given to the users. But a major drawback is, It only works for Rep-based workouts and they have Predefined Workouts (Very Limited) there is no freedom to the administrator to add any exercise. Kia Health App does not provide action-scoring; instead, it only delivers feedback. Truerep is a mobile application that uses computer vision and position estimation to deliver real-time feedback that provides a comparable solution; however, it is only available for completing squats; no other exercises can be performed.

3.2 PROPOSED SYSTEM

An approach that is solely algorithmic, more precise and effective than image classification methods, more accurate and effective than a distance computation method, provides values that can be utilized to calibrate users' cameras, Real-time, and post-workout scoring, Needs no dataset at all; a reference workout video will do. Profile information is gathered, which can be used to add biases to the scoring logic. suitable for desktop and mobile contexts, As normalized coordinates are used to calculate joint angles, body types are unaffected. The software is easily deployable in any type of cloud environment and is readily accessible for all types of devices because it does not require any kind of substantial datasets for computer vision and action scoring. The detection and scoring components of the software are also designed as algorithmic components, allowing for quick scaling in any cloud environment.

CHAPTER 4
PROJECT DESCRIPTION

CHAPTER 4 PROJECT DESCRIPTION

4.1 PROPOSED DESIGN:

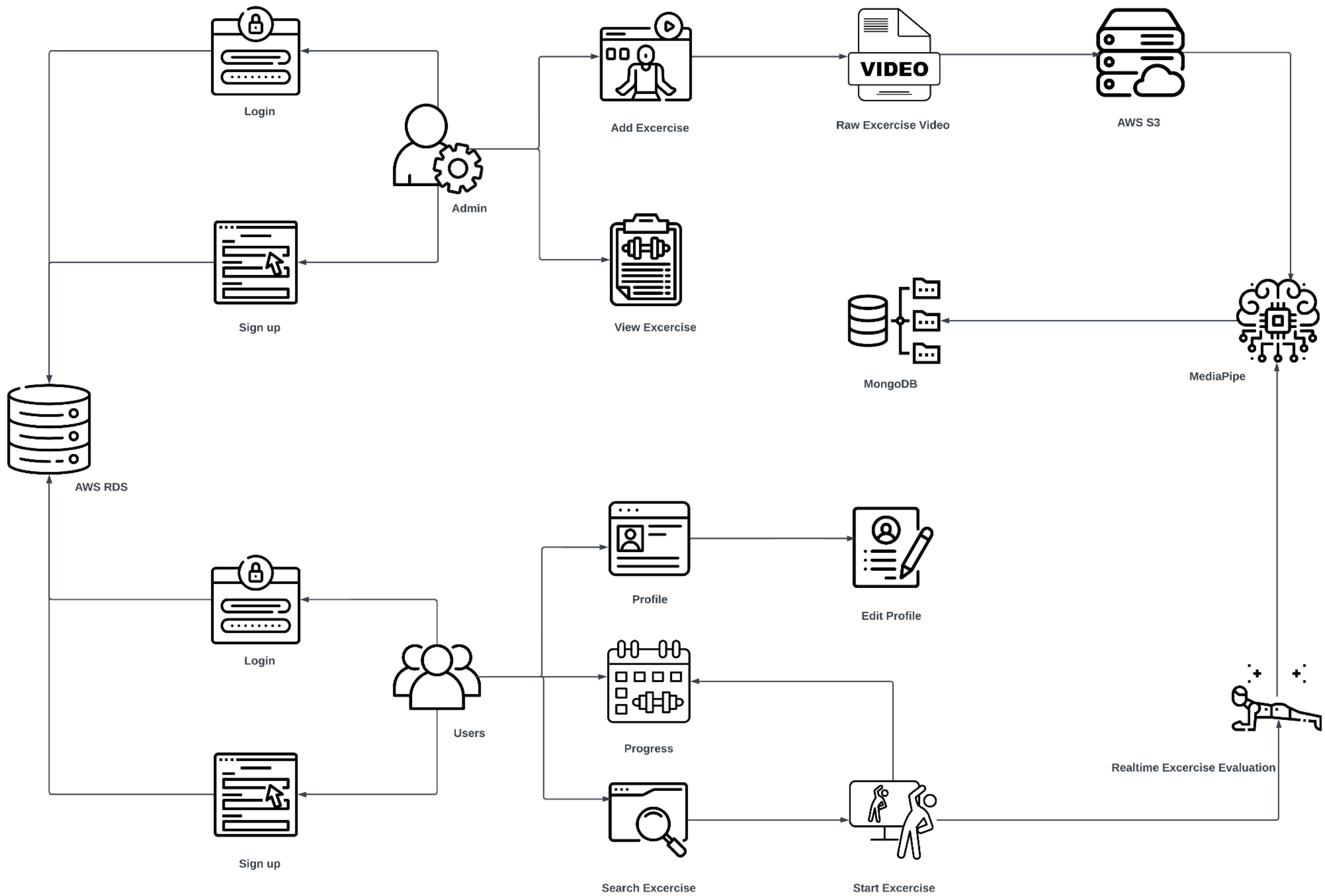

Fig 4.1.Fitness Tracker Architecture Diagram.

As seen in the architecture diagram, administrators can sign up for/login to our application. After signing in, the administrator can post an exercise video. In order for the angle omitting logic to operate, admins need to define the sort of workout they are adding, such as lower-body workout, upper-body workout, and full-body workout. If it is an upper-body workout, the algorithm will omit the lower-body angles, and vice versa for lower-body workouts. No angles will be omitted if it is a full-body workout. Administrators could also define whether the workout is single- or multi-purpose, They should also indicate the duration of the exercise if it is a single-pose workout and the number of reps if it is a multi-pose workout. Administrators can view the videos they have posted after uploading them.

Users of our application can sign up and login to the website; after signing in, they must provide their personal information such as name, age, height, and weight. BMI will be calculated once they have updated their profile with this information. They can occasionally edit their profile information.

Users can then look for the workout they need to undertake. Once they've found the program, they should begin the workout by clicking the button and performing it correctly in front of the web camera; our application will provide live scoring during the workout. Users can determine whether or not to repeat the practice after seeing its efficiency and accuracy. When they are finished, they can check their progress on our website's dedicated progress section.

CHAPTER 5
MODULE DESCRIPTION

CHAPTER 5 MODULE DESCRIPTION

5.1 ADMIN MODULE

● **Login/Signup** - New admins can register to the website with an email address, account password and other basic details.
● **Add exercise** - administrators can add any workout video into the app by uploading it from this module.
● **View exercise** - Admins can view all the videos that have been uploaded.

5.2 USER MODULE

● **Login/Signup** - New users can register to the website with an email address, account password, and other basic details.
● **Add/Edit/View** - They have their own dedicated page for viewing and editing their profile.
● **Search exercise** - Users can search for the workout they want in this module.
● **Start workout** - Once they find an exercise they can start working out.
● **Progress -** In this page users can see the progress of the exercise which they did previously.

CHAPTER 6
REQUIREMENTS

CHAPTER 6 REQUIREMENTS

6.1. FUNCTIONAL REQUIREMENTS:

In software engineering, a functional requirement defines a function of a software system or its component. Here, the system has to perform the following tasks:

●Has to calculate the 12 major angles of the human body.

●Has to compare these angles frame by frame with the angles calculated from the user side and provide accuracy.

6.2. NON-FUNCTIONAL REQUIREMENTS:

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviour. This should be contrasted with functional requirements that define specific behaviour or functions. The plan for implementing functional requirements is detailed in the system design. Other terms for non-functional requirements are ―constraints, "quality goals", "quality of service requirements" and "non-behavioral requirements". Some of the quality attributes are as follows:

Accessibility:

Accessibility is a general term used to describe the degree to which a product, device, service, or environment is accessible by as many people as possible. In our project any number of people can register and can access the service. User interface is simple and efficient and easy to use.

Maintainability:

In software engineering, maintainability is the ease with which a software product can be modified in order to:

● Correct defects

● Meet new requirements

New functionalities can be added in the project based on the user requirements just by adding the appropriate files to existing projects using python scripting languages.

Scalability:

System is capable of handling increased total throughput under an increased load when resources are added. System can work normally under situations such as low bandwidth and a large number of users.

Portability:

Portability is one of the key concepts of high-level programming. Portability is the software code base feature to be able to reuse the existing code instead of creating new code when moving software from an environment to another. Project can be executed under different operation conditions provided it meets its minimum configurations.

6.3. SOFTWARE REQUIREMENTS:

- Visual Studio Code
- Python 3.0 or above versions
- Windows 8 or higher OS

5.4. HARDWARE REQUIREMENTS:

- Processor: Any processor above 500 MHz
- RAM: 512Mb
- Hard Disk: 10GB
- Input device: Standard Keyboard and Mouse
- Output device: High Resolution Monitor
- Nvidia CUDA version 10.0.0 and above (additional requirement for faster processing)

CHAPTER 7
METHODOLOGY

CHAPTER 7 METHODOLOGY

7.1. MEDIAPIPE DESCRIPTION:

Google's MediaPipe Pose Classification is a component of the MediaPipe framework, which offers a comprehensive solution for developing cross-platform, real-time computer vision applications.

Using machine learning models trained on large datasets, MediaPipe Pose Classification analyzes input data, typically in the form of images or video frames, to detect and classify specific human body poses. It identifies and tracks the positions of key body landmarks or joints, such as the wrists, elbows, shoulders, hips, knees, and ankles.

The system uses deep learning algorithms to detect and localize the positions of key body landmarks or joints in the input data. This involves identifying the coordinates of each joint, enabling the system to understand the body's overall pose and structure. The model tracks the movements of the detected body landmarks across consecutive frames or images, allowing for real-time pose analysis and motion estimation.

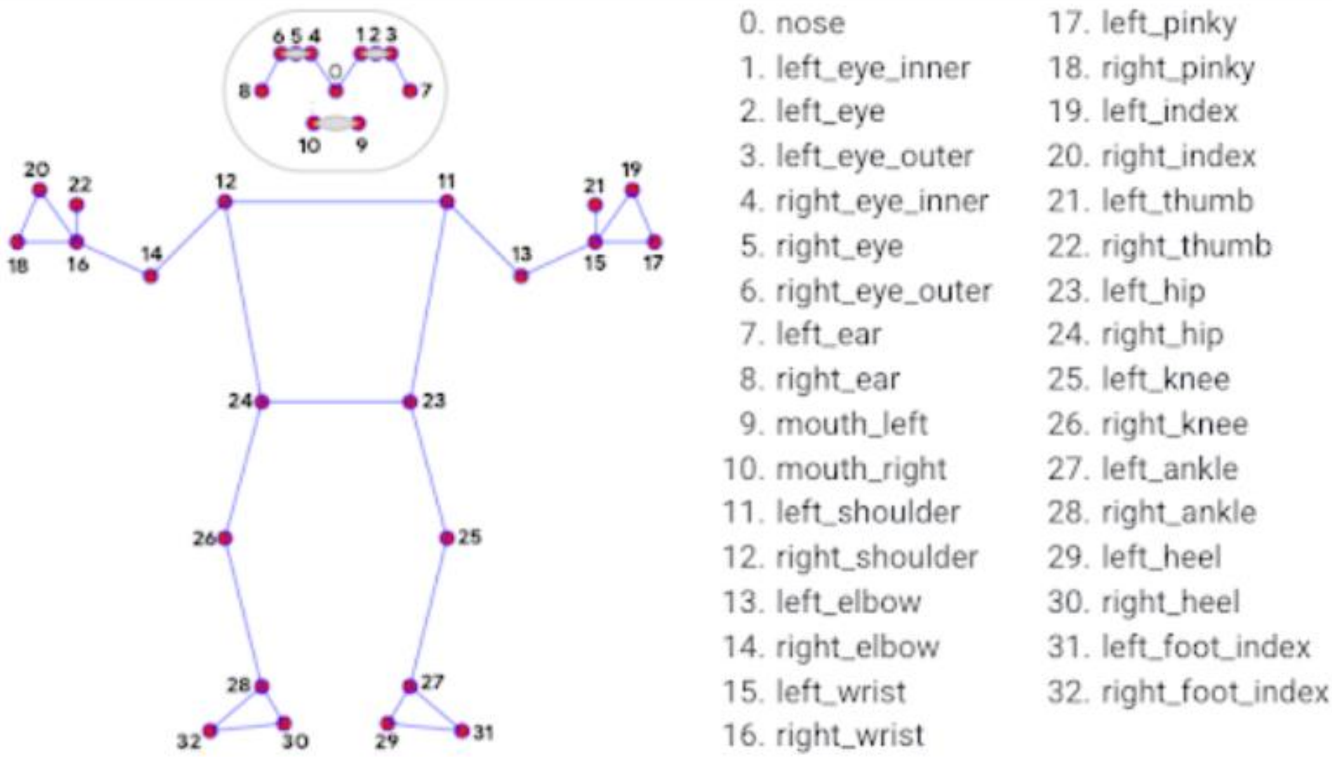

Fig 7.1.Major points of the human body.

7.2. SINGLE-POSE METHODOLOGY:

If the workout is a single pose duration workout then for each frame in the video a list of all major 12 angles will be generated. All these list of 12 angles will be given into a average function to get a final list of 12 angles. Following that, in order to have only the important angles in the pose and to increase the accuracy of scoring the users, an omitted angle list is constructed using a new criterion called visibility criteria together with the type of exercise (full,upper,lower). In addition, we build a list of 12 z co-ords from each frame, which will be used to calibrate the user to the camera before beginning the activity. Finally, the angles, visibility, z coordinates, and omitted angles will be saved in a NOSQL database.

If the requested workout is a single pose, a database request is made to retrieve the reference angles, omitted angles, and duration of the activity. Workout begins, and the timer begins. For each received frame, the major 12 angles are calculated, as well as the angle difference between the current list of angles and the reference angle list. The preceding technique is repeated for all frames of the duration. After the workout, the average angle difference and visibility of all 12 primary angles for all frames are calculated.

The angle difference is specified to a maximum value of 100 points using the visibility criteria. The unnecessary points are removed from the list of angles that differ with the help of the omitted angles list, and the average angle difference is then scored using the final scoring logic. The scoring logic that we used distributes the average angle difference over a range of 0 to 100. It is a distributed range-oriented scoring logic. Following scoring, both the user's progress and the result are kept in the database.

7.3 MULTI-POSE METHODOLOGY

If the workout is a multipose reps workout then for each frame in the video a list of all major 12 angles will be generated. All of these 12 angles will be entered into an average function and compared to a list of maximum angles and a list of minimum angles, both of which are initially empty. However, when more frames are added, both the maximum angle list and the minimum angle list will eventually be created.

Following this, a list of omitted angles is also generated using a new criterion called visibility criteria along with the type of exercise (full, upper, lower) in order to omit some angles from the list in order to have only the important angles in the pose and to increase the accuracy of scoring the users. We additionally generate lists of 12 z-coordinates for the user's maximum and minimum positions from each frame, which will be utilized to calibrate the user to the camera prior to the exercise. Finally the max_angles, min_angles, max_visibility, min_visibility, max_z_coords, min_z_coords, omitted angles will be stored in a mongodb database.

When a multi-pose workout is requested, the appropriate database request is made to retrieve the exercise's reference maximum and minimum angles, omitted angles, and reps. A threshold value is determined using the reference video's max and min angles list, above which the user is in the maximum state and below which the user is in the minimum state. To keep track of the current rep and adjust appropriately for the current user position, user-specific initial state, count, and rep variables are updated.Some angles are left out of each frame that is received, and the average is then taken to compare it with the threshold average and accordingly the rep is updated.

Once the value of a rep increases, the max angle list and the min angle list for that rep are calculated, along with the max angle and min angle difference from the reference max angle and reference min angle, respectively. The scoring logic is then applied, and the score for each rep is printed. The user progress is kept in the database along with the average score once the user has completed all the reps.

CHAPTER 8

IMPLEMENTATION

```html
{% load static %}
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <link rel="apple-touch-icon" sizes="76x76" href="assets/img/apple-icon.png">
  <link rel="icon" type="image/png" href="assets/img/favicon.png">
  <title>
    SSG
  </title>
  <!--     Fonts and icons     -->
  <link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700,900|Roboto+Slab:400,700" />
  <!-- Nucleo Icons -->
  <link href="{% static 'assets/css/nucleo-icons.css' %}" rel="stylesheet" />
  <link href="{% static 'assets/css/nucleo-svg.css' %}" rel="stylesheet" />
  <!-- Font Awesome Icons -->
  <script src="https://kit.fontawesome.com/42d5adcbca.js" crossorigin="anonymous"></script>
  <!-- Material Icons -->
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons+Round" rel="stylesheet">
  <!-- CSS Files -->
  <link id="pagestyle" href="{% static 'assets/css/material-kit.css' %}" rel="stylesheet" />
</head>

<body class="about-us bg-gray-200">
  <!-- Navbar Transparent -->
  <nav class="navbar navbar-expand-lg position-absolute top-0 z-index-3 w-100 shadow-none my-3  navbar-transparent ">
    <div class="container">
      <a class="navbar-brand  text-white " href="https://demos.creative-tim.com/material-kit/presentation" rel="tooltip" title="Designed
        Fitness tracker
      </a>
      <button class="navbar-toggler shadow-none ms-2" type="button" data-bs-toggle="collapse" data-bs-target="#navigation" aria-controls=
```

Fig 8.1 HTML code for index.

```html
        </div>
      </header>
      <!-- -------- END HEADER 7 w/ text and video ------- -->
      <div class="card card-body shadow-xl mx-3 mx-md-4 mt-n6">
      <!-- Section with four info areas left & one card right with image and waves -->
      <section class="py-7">
        <div class="container">
          <div class="row align-items-center">
            <div class="col-lg-6">
              <div class="row justify-content-start">
                <div class="col-md-6">
                  <div class="info">
                    <i class="material-icons text-3xl text-gradient text-info mb-3">public</i>
                    <h5>Flexible</h5>
                    <p>Adaptable for any workouts, role, or device so you can choose when, where and how</p>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="info">
                    <i class="material-icons text-3xl text-gradient text-info mb-3">payments</i>
                    <h5>Rewards</h5>
                    <p>Get rewarded with SSG coins which makes more desirable</p>
                  </div>
                </div>
              </div>
              <div class="row justify-content-start mt-4">
                <div class="col-md-6">
                  <div class="info">
                    <i class="material-icons text-3xl text-gradient text-info mb-3">apps</i>
                    <h5>Inclusive</h5>
                    <p>Equal experiences for everyone regardless of geography, fitness</p>
                  </div>
                </div>
```

Fig 8.2 HTML code for index.

In the above code snippets, we can see the HTML codes for the index page of our application

```python
angles = [(23,11,13),(14,12,24),(11,13,15),(12,14,16),(13,15,19),(14,16,20),(11,23,25),(12,24,26),(23,25,27),(24,26,28),(25,27,31),(26,

def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    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

def omit(exercise_type,omitted_angles):
    if exercise_type == "upper_body":
        for i in [8,9,10,11]:
            if i not in omitted_angles:
                omitted_angles.append(i)
    elif exercise_type == "lower_body":
        for i in [0,1,2,3,4,5]:
            if i not in omitted_angles:
                omitted_angles.append(i)
    omitted_angles.sort()

@login_required(login_url='admin_login')
def add_exercise(request):
    if request.method=="POST":
        exercise_name = request.POST['exercise_name']
```

Fig 8.3 Angle calculation from admin side.

In the above code snippet, we calculate the angles for the major points of our body for the reference video uploaded by the admin.

```
import cv2
import mediapipe as mp
import numpy as np
import time
import requests
import pymongo
from bson.objectid import ObjectId

exercise_id = '632478ae-28bb-4c88-874d-8853f0a3db30'
user_id = 2
pose = requests.get('http://127.0.0.1:8000/admin_app/return_pose/',{'exercise_id':exercise_id,'user_id':user_id}).json()
pose_id = pose['pose_id']
duration = pose['duration']

client = pymongo.MongoClient("mongodb+srv://2019it0530:finalyear@cluster0.w91uonf.mongodb.net/?retryWrites=true&w=majority")
db = client["ssg"]
col = db["pose"]


data = col.find_one({'_id': ObjectId(str(pose_id))})




#returned from the reference:
ref_angles = data['ref_angles']
omitted_angles = data['omitted_angles']

def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End
```

Fig 8.4 Implementation of Single-pose methodology.

In the above code snippet, we start the implementation of single-pose methodology.

```
    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

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
angles = [(23,11,13),(14,12,24),(11,13,15),(12,14,16),(13,15,19),(14,16,20),(11,23,25),(12,24,26),(23,25,27),(24,26,28),(25,27,31),(26,28
cap = cv2.VideoCapture(0)
angle_diff=[]
avg_visib=[]
c=0
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as pose:
    start = time.time()
    while cap.isOpened():
        ret, frame = cap.read()
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        image.flags.writeable = False
        results = pose.process(image)
        image.flags.writeable = True
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
        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)
```

Fig 8.5 Angle calculation from user end(single-pose).

In the above code snippet, we calculate angles from the user end for a single pose workout.

```python
    cap.release()
    cv2.destroyAllWindows()
print(c)
angle_diff = [i/c for i in angle_diff].copy()
avg_visib = [i/c for i in avg_visib].copy()

for i in range(0,len(angle_diff)):
    if avg_visib[i]<0.5:
        angle_diff[i]=100

t=0
for i in omitted_angles:
    del angle_diff[i-t]
    t+=1

avg_angle_diff = sum(angle_diff)/len(angle_diff)

score = None
if avg_angle_diff>40:
    score=0          #Workout is not done
elif avg_angle_diff<40 and avg_angle_diff>15:
    score = (40-avg_angle_diff) * 3.6
else:
    score = (15-avg_angle_diff) * 0.67
    score = score + 90


print("score:", score)

requests.get('http://127.0.0.1:8000/user_app/score_user/',{'exercise_id':exercise_id,'user_id':user_id,'score':score})
```

Fig 8.6 Scoring logic(Single-pose).

In the above code snippet, we implement scoring logic for single-pose workouts.

```python
import cv2
import mediapipe as mp
import numpy as np
import requests
import pymongo
from bson.objectid import ObjectId

exercise_id = '99c8a6b1-fd17-4b20-883f-438f27b67800'
user_id = 4
pose = requests.get('http://127.0.0.1:8000/admin_app/return_pose/',{'exercise_id':exercise_id,'user_id':user_id}).json()
pose_id = pose['pose_id']
reps = pose['reps']

client = pymongo.MongoClient("mongodb+srv://2019it0530:finalyear@cluster0.w91uonf.mongodb.net/?retryWrites=true&w=majority")
db = client["ssg"]
col = db["pose"]

data = col.find_one({'_id': ObjectId(str(pose_id))})

def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    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
```

Fig 8.7 Implementation of Multi-pose methodology.

In the above code snippet, we start the implementation of multi-pose methodology.

```python
ref_max_angles = data['ref_max_angles']
ref_min_angles = data['ref_min_angles']
omitted_angles = data['omitted_angles']

t=0
for i in omitted_angles:
    del ref_max_angles[i-t]
    del ref_min_angles[i-t]
    t+=1

threshold = []
for i in range(0,len(ref_max_angles)):
    threshold.append((ref_max_angles[i]+ref_min_angles[i])/2)

avg_threshold = sum(threshold)/len(threshold)

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
angles = [(23,11,13),(14,12,24),(11,13,15),(12,14,16),(13,15,19),(14,16,20),(11,23,25),(12,24,26),(23,25,27),(24,26,28),(25,27,31),(26,28
cap = cv2.VideoCapture(0)
state = "max"
rep = 0
max_avg = 0
min_avg = 0
max_angle = []
min_angle = []
count = 0
initial_state = 0
total_score = 0
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as pose:
```

Fig 8.8 Min angle and Max angle calculation.

In the above code snippet, we calculate minimum and maximum angles for the multi-pose workout.

```python

                    min_angle_diff = []
                    for i in range(0,len(ref_min_angles)):
                        min_angle_diff.append(abs(ref_min_angles[i]-min_angle[i]))
                    avg_min_angle_diff = sum(min_angle_diff)/len(min_angle_diff)

                    if avg_min_angle_diff>40:
                        min_score=0          #Workout is not done
                    elif avg_min_angle_diff<=40 and avg_min_angle_diff>15:
                        min_score = (40-avg_min_angle_diff) * 3.66
                    else:
                        min_score = (15-avg_min_angle_diff) * 0.67
                        min_score = min_score + 90

                    print(rep," : ",(max_score+min_score)/2)
                    total_score += (max_score+min_score)/2
                    if rep==reps:
                        cap.release()
                        cv2.destroyAllWindows()
                    max_angle = curr_angles.copy()
                    max_avg = avg_curr_angles
                    initial_state = 1
                    count=0
                else:
                    pass

            elif state == "max" and avg_curr_angles >avg_threshold:
                if initial_state==0 or initial_state==1:
                    count +=1
                    initial_state =2
```

Fig 8.9 Rep counting using threshold.

In the above code snippet, we implement rep counting using a threshold value. For each time the user reaches from minimum angle to maximum angle and returns back to minimum angle we increase the count of rep.

```
        initial_state =2
    if max_avg==0 and len(max_angle)==0:
        max_avg=avg_curr_angles
        max_angle = curr_angles.copy()
    elif avg_curr_angles > max_avg:
        max_avg = avg_curr_angles
        max_angle = curr_angles.copy()
elif state == "min" and avg_curr_angles < avg_threshold:
    if initial_state == 4:
        count +=1
        initial_state =3
    if avg_curr_angles < min_avg:
        min_avg = avg_curr_angles
        min_angle = curr_angles.copy()
elif state=="max" and avg_curr_angles < avg_threshold:
    if initial_state==2:
        count +=1
    if count==2:
        initial_state=4
        state = "min"

        max_angle_diff = []
        for i in range(0,len(ref_max_angles)):
            max_angle_diff.append(abs(ref_max_angles[i]-max_angle[i]))
        avg_max_angle_diff = sum(max_angle_diff)/len(max_angle_diff)
        if avg_max_angle_diff>40:
            max_score=0         #Workout is not done
        elif avg_max_angle_diff<=40 and avg_max_angle_diff>15:
            max_score = (40-avg_max_angle_diff) * 3.66
        else:
            max_score = (15-avg_max_angle_diff) * 0.67
```

Fig 8.10 Scoring logic(Multi-pose).

```
                    avg_max_angle_diff = sum(max_angle_diff)/len(max_angle_diff)
                    if avg_max_angle_diff>40:
                        max_score=0         #Workout is not done
                    elif avg_max_angle_diff<=40 and avg_max_angle_diff>15:
                        max_score = (40-avg_max_angle_diff) * 3.66
                    else:
                        max_score = (15-avg_max_angle_diff) * 0.67
                        max_score = max_score + 90

                    min_angle = curr_angles.copy()
                    min_avg = avg_curr_angles

        except Exception as e:
            print(e)
    cap.release()
    cv2.destroyAllWindows()

total_score /= rep
requests.get('http://127.0.0.1:8000/user_app/score_user/',{'exercise_id':exercise_id,'user_id':user_id,'score':total_score})
```

Fig 8.11 Scoring logic(Multi-pose).

In the above code snippets, we implement scoring logic for multi-pose workouts.

CHAPTER 9

TESTING AND RESULTS

CHAPTER 9 TESTING AND RESULTS

9.1. RESULTS

Fig 9.1.1 Single-pose exercise reference

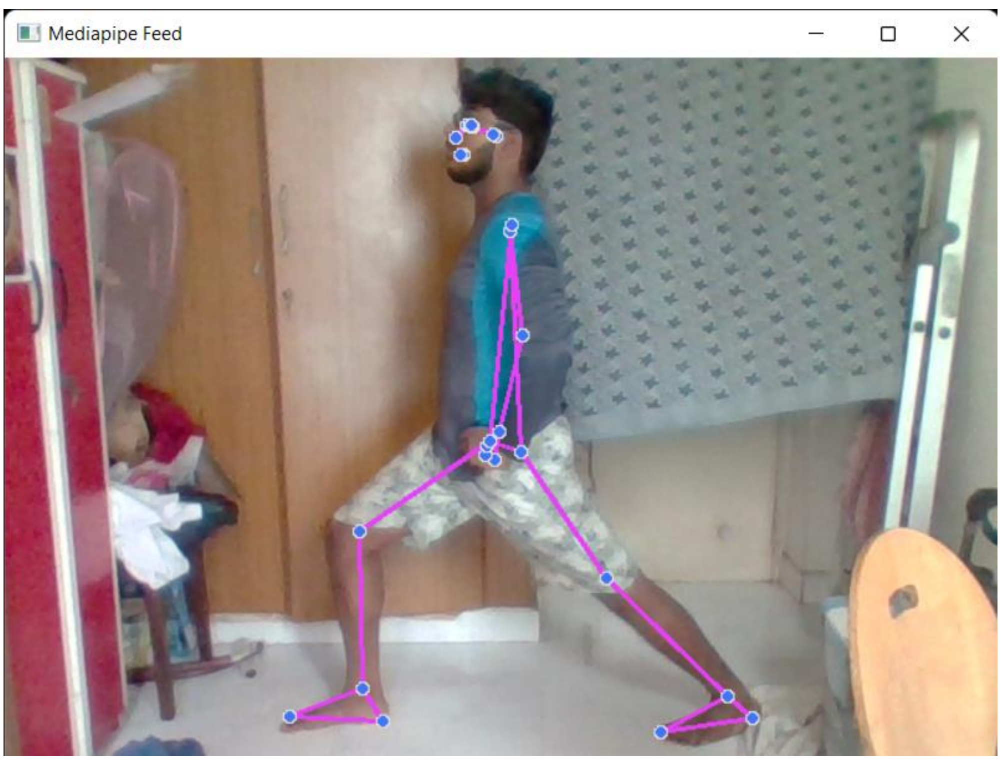

Fig 9.1.2 Single-pose exercise

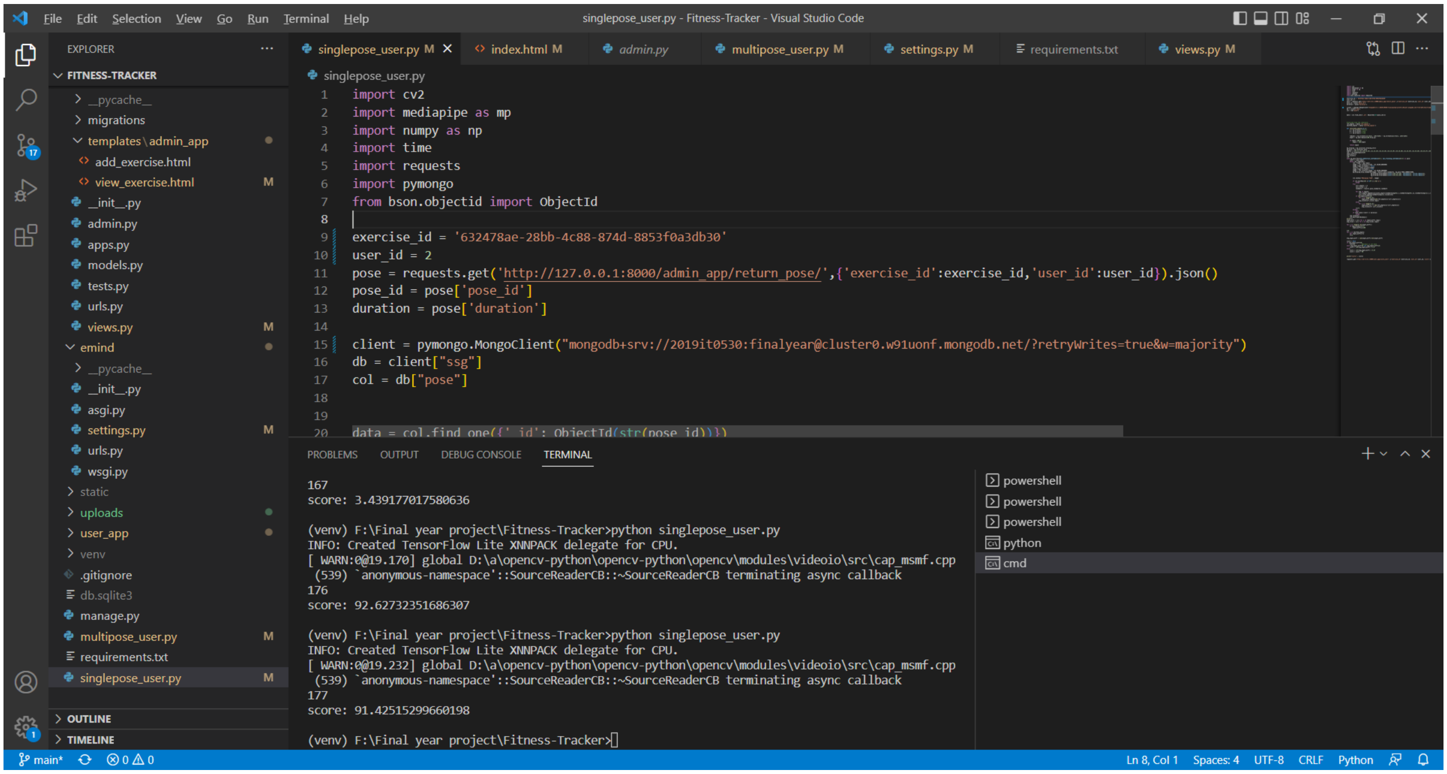

Fig 9.1.3 Single-pose scoring

In the above pictures, we can observe the results of a single-pose exercise using mediapipe pose classification by comparing the user end webcam with the reference video. The Accuracy for this particular exercise is 92.67%.

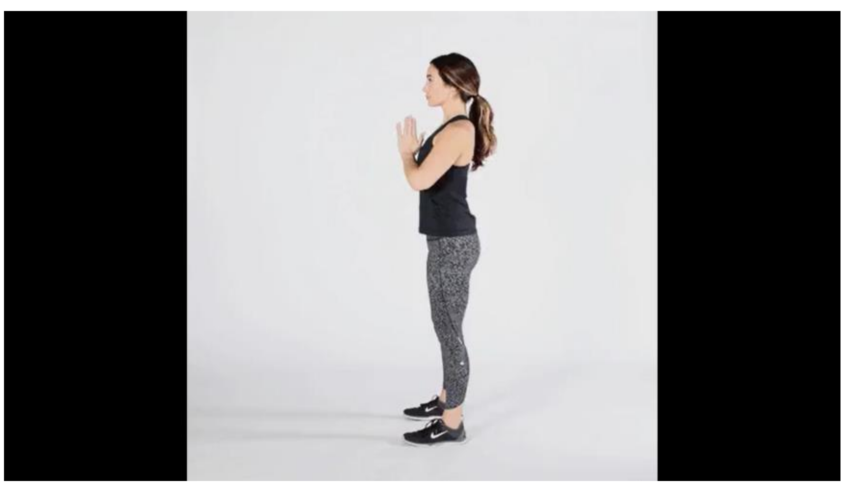

Fig 9.1.4 multi-pose exercise reference

Fig 9.1.5 multi-pose exercise reference

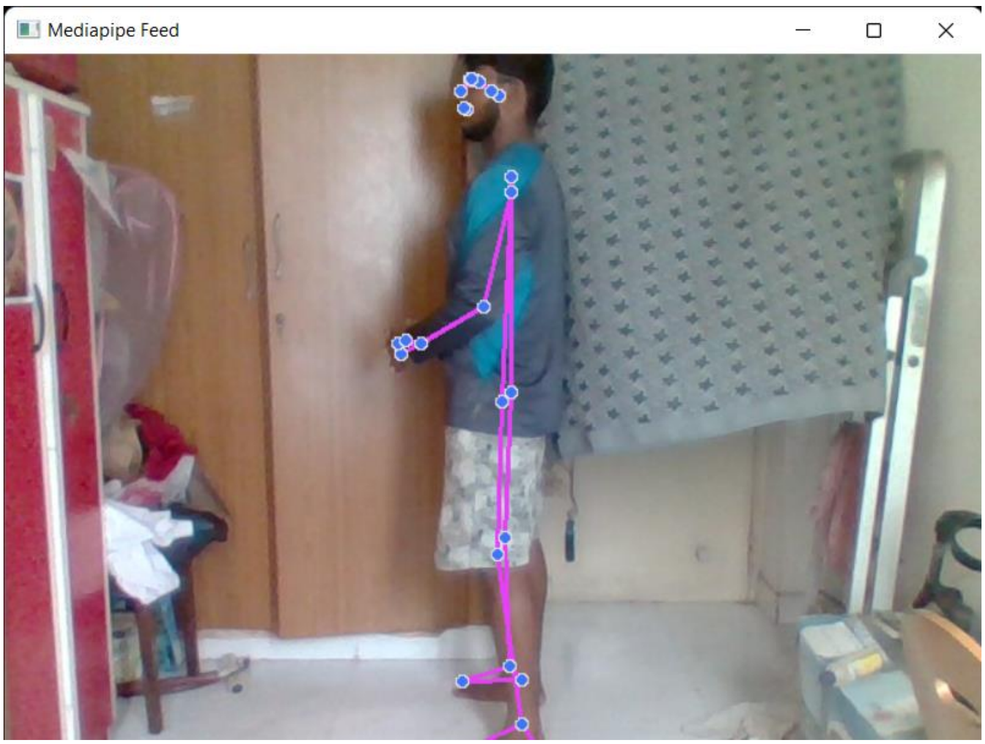

Fig 9.1.6 multi-pose exercise

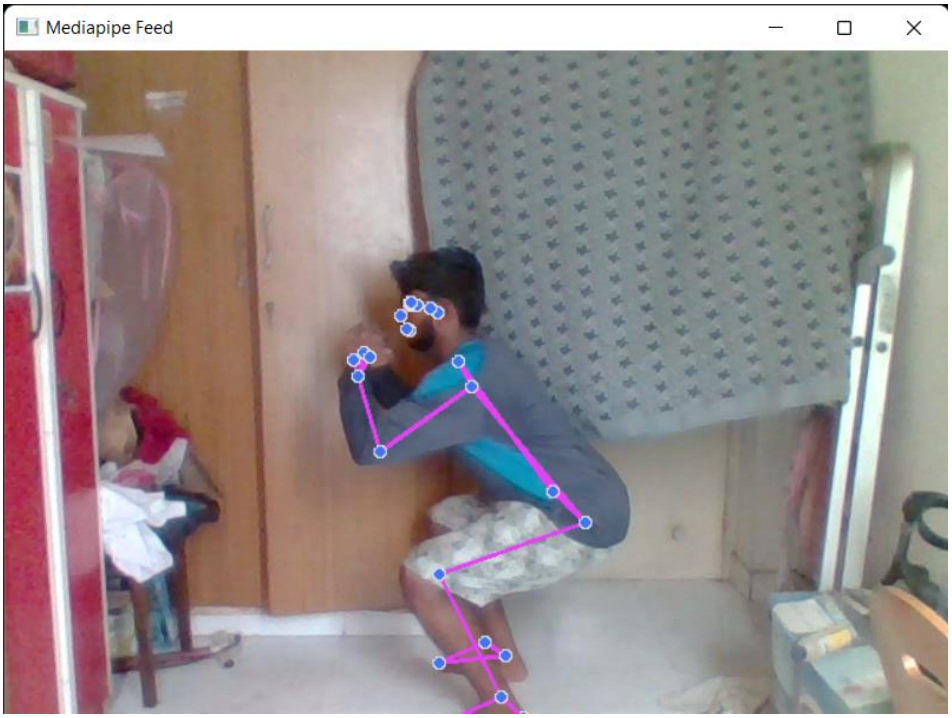

Fig 9.1.7 multi-pose exercise

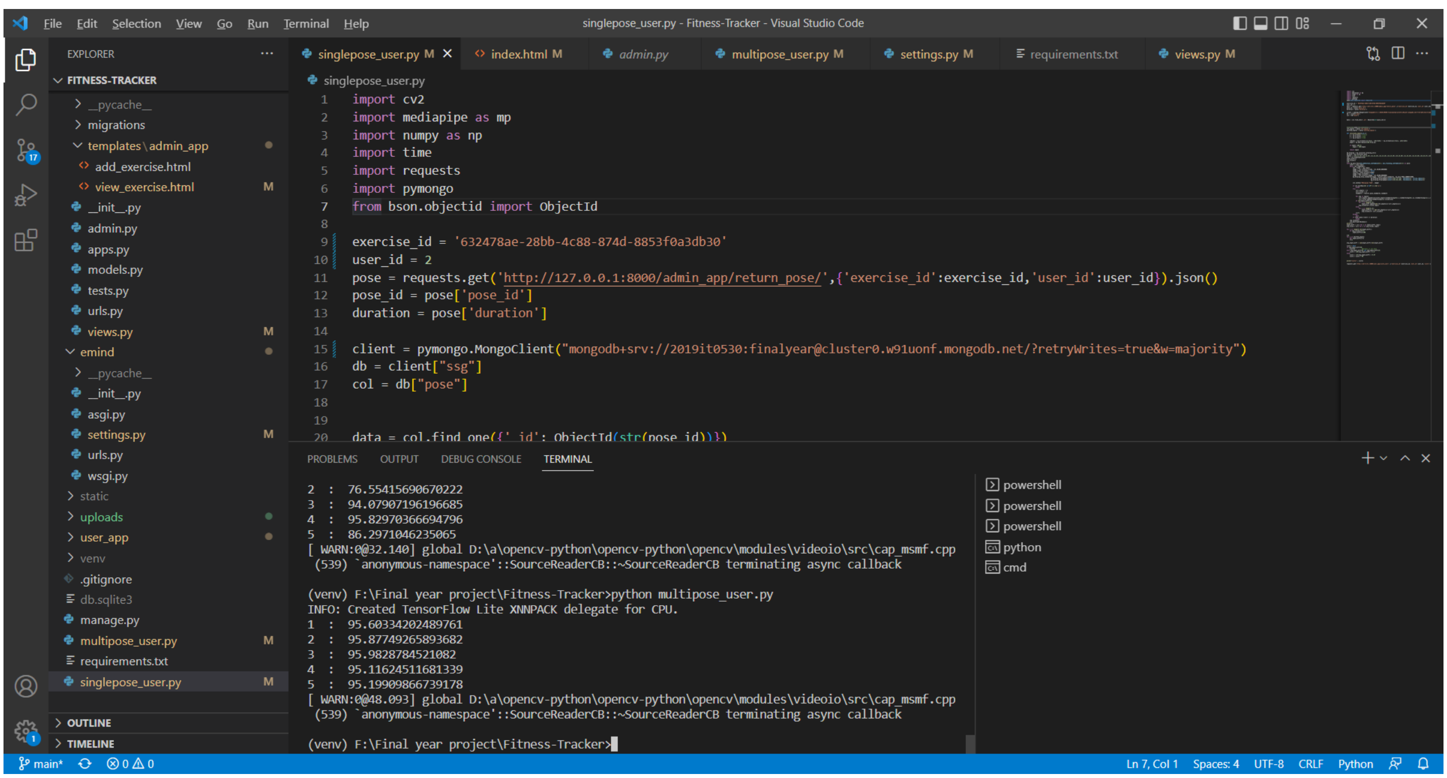

Fig 9.1.8 multi-pose scoring

In the above pictures we can observe the results of a multi-pose exercise using mediapipe pose classification by comparing the user end webcam with the reference video. Accuracy for each rep has been calculated.

Rep 1- 95.60%

Rep 2- 95.87%

Rep 3- 95.98%

Rep 4- 95.11%

Rep 5- 95.19%

8.2. TEST CASES:

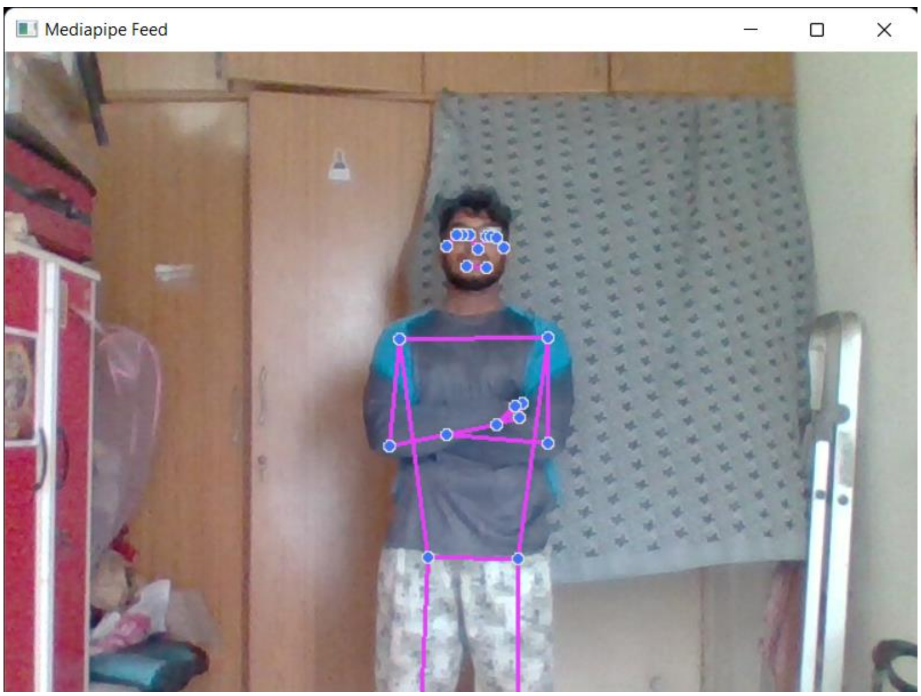

Fig 9.2.1. Without doing the exercise

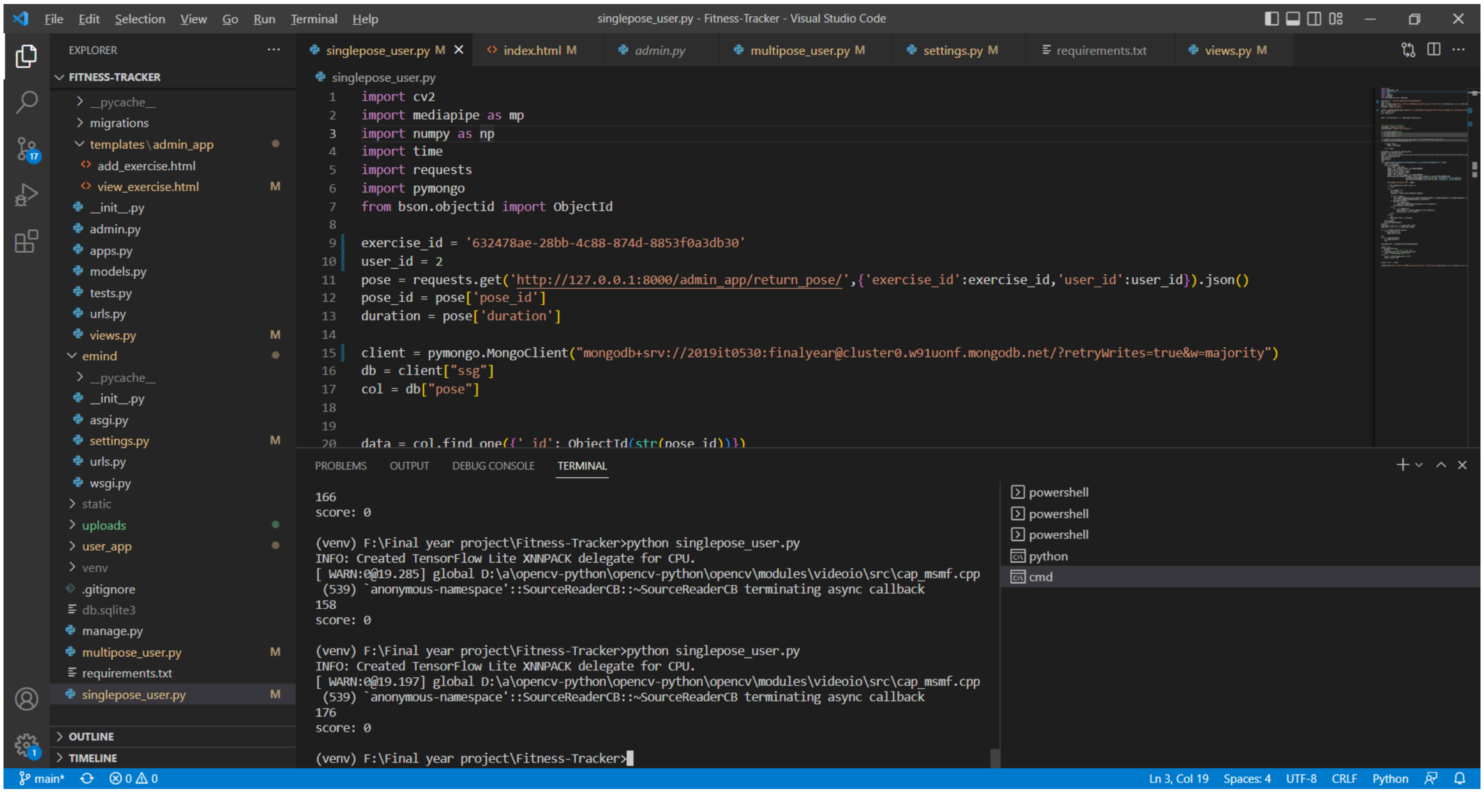

Fig 9.2.1. Displaying score

Our first case involves the user trying to do nothing in the workout (8.2.1) which in turn displays a score as 0 because the angles of the reference video and the angles of this test case don't match .

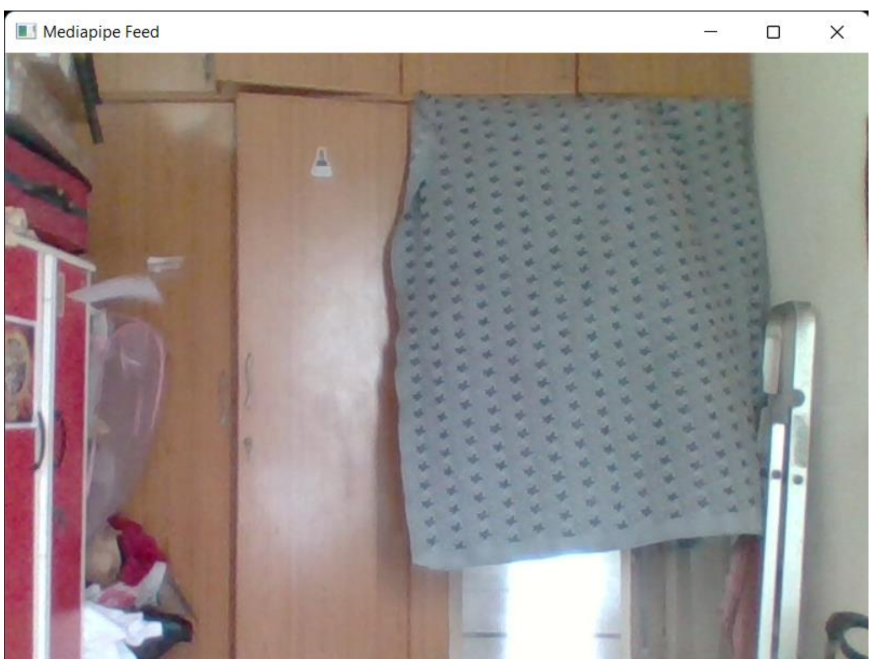

Fig 9.2.3 no one in the frame

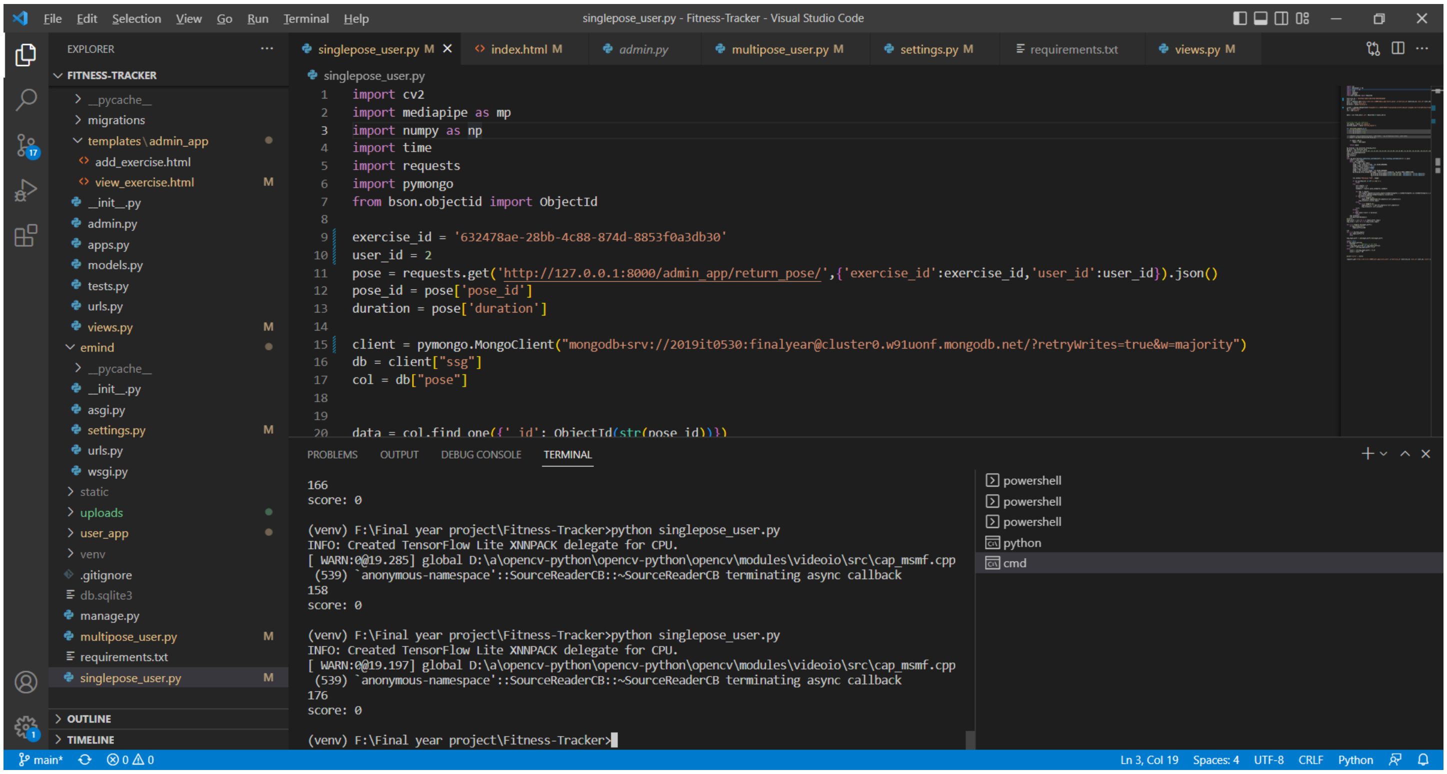

Fig 9.2.4. Displaying score

Our second case involves the user trying not to be in the frame (8.2.3) which in turn displays a score as 0 because the angles of the reference video and the angles of this test case don't match .

CHAPTER 10

CONCLUSION AND FUTURE WORKS

CHAPTER 10 CONCLUSION AND FUTURE SCOPE

10.1 CONCLUSION:

A website and a methodology that supports everyone in their efforts to exercise. This application allows any novice to assess their own performance in terms of exercise efficiency and proper form. Every busy person can find time for working out with the help of a fitness tracker because it can be accessed from anywhere and does not require special instruction from anyone about form or someone to evaluate their session. Our application provides accurate efficiency of the workout that the user has performed.

10.2 FUTURE SCOPE:

We intend to add numerous features in the future to improve the functionality of our application, such as bias. We want to use each person's unique medical history to add bias to our scores. For instance, if an obese person exercises, it is obvious that they cannot perform the exercise as well as a fit person. The same is true for seniors and people with physical disabilities. In order to determine which user will have greater bias, we will categorize the users by age. For example, if a user is 60 years or older, their score will be biased more. This bias may aid In self-motivation for that individual.

REFERENCES

REFERENCES:

[1] B. Natarajan et al.,“"Development of an End-to-End Deep Learning Framework for Sign Language Recognition, Translation, and Video Generation”" in IEEE Access, vol. 10, pp. 104358-104374, 2022, doi: 10.1109/ACCESS.2022.3210543.

[2] Shuo Zhang, Wanmi Chen, Chen Chen, Yang Liu”Human deep squat detection method based on MediaPipe combined with Yolov5 network” Proceedings of the 4^{1s}t Chinese Control Conference July 25-27, 2022, Hefei, China

[3] Khushi Sidana “REAL TIME YOGA POSE DETECTION USING DEEPLEARNING: A REVIEW” International Journal of Engineering Applied Sciences and Technology, 2022 Vol. 7, Issue 7, ISSN No. 2455-2143, Pages 61-65 November 2022.

[4] Yejin Kwon, Dongho Kim”Real-Time Workout Posture Correction using OpenCV and MediaPipe” Journal of KIIT. Vol. 20, No. 1, pp. 199-208 Jan. 31, 2022

[5] Sanjal Gupta , Sadhana Singh , Pratibha Sharma , Sadhana Maurya , Sunil Yadav “Yoga Pose Detection Using Deep Learning”Volume 3, Issue 6 (NovemberDecember 2022), PP: 92-94.

[6] Ardra Anilkumar , Athulya K.T., Sarath Sajan , Sreeja K.A. “Pose Estimated YogaMonitoring System” 2 nd International Conference on IoT Based Control Networks and Intelligent Systems (ICICNIS 2021)

[7] B. Subramanian, J. Kim, M. Maray and A. Paul,“"Digital Twin Model: A Real-Time Emotion Recognition System for Personalized Healthcare”" in IEEE Access, vol. 10, pp. 81155-81165, 2022, doi: 10.1109/ACCESS.2022.3193941.

APPENDIX:

```
import cv2
import mediapipe as mp
import numpy as np
import time
import requests
import pymongo
from bson.objectid import ObjectId

exercise_id = '632478ae-28bb-4c88-874d-8853f0a3db30'
user_id = 2
pose =
requests.get('http://127.0.0.1:8000/admin_app/return_pose/',{'exercise_id':exercise_id,'user_id':
user_id}).json()
pose_id = pose['pose_id']
duration = pose['duration']

client =
pymongo.MongoClient("mongodb+srv://2019it0530:finalyear@cluster0.w91uonf.mongodb.net/
?retryWrites=true&w=majority")
db = client["ssg"]
col = db["pose"]


data = col.find_one({'_id': ObjectId(str(pose_id))})




#returned from the reference:
ref_angles = data['ref_angles']
omitted_angles = data['omitted_angles']

def calculate_angle(a,b,c):
    a = np.array(a) # First
    b = np.array(b) # Mid
    c = np.array(c) # End

    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

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
angles                                                                                                              =
[(23,11,13),(14,12,24),(11,13,15),(12,14,16),(13,15,19),(14,16,20),(11,23,25),(12,24,26),(23,25,
27),(24,26,28),(25,27,31),(26,28,32)]
cap = cv2.VideoCapture(0)
angle_diff=[]
avg_visib=[]
c=0
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as pose:
    start = time.time()
    while cap.isOpened():
        ret, frame = cap.read()
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        image.flags.writeable = False
        results = pose.process(image)
        image.flags.writeable = True
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
        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'):
            break
        try:
            curr_angles = []
            curr_visib = []
            landmark = results.pose_landmarks.landmark

            for ang in angles:

curr_angles.append(calculate_angle([landmark[ang[0]].x,landmark[ang[0]].y],[landmark[ang[1]
].x,landmark[ang[1]].y],[landmark[ang[2]].x,landmark[ang[2]].y]))
                curr_visib.append(landmark[ang[1]].visibility)
            if len(angle_diff)==0:
                for i in range(0,12):
```

```
                angle_diff.append(abs(ref_angles[i]-curr_angles[i]))
            avg_visib=curr_visib.copy()
        else:
            for i in range(0,12):
                angle_diff[i] += abs(ref_angles[i]-curr_angles[i])
                avg_visib[i]+= curr_visib[i]
        c=c+1
    except:
        pass
    if time.time()-start >= duration:
        break
cap.release()
cv2.destroyAllWindows()
print(c)
angle_diff = [i/c for i in angle_diff].copy()
avg_visib = [i/c for i in avg_visib].copy()

for i in range(0,len(angle_diff)):
    if avg_visib[i]<0.5:
        angle_diff[i]=100

t=0
for i in omitted_angles:
    del angle_diff[i-t]
    t+=1

avg_angle_diff = sum(angle_diff)/len(angle_diff)

score = None
if avg_angle_diff>40:
    score=0        #Workout is not done
elif avg_angle_diff<40 and avg_angle_diff>15:
    score = (40-avg_angle_diff) * 3.6
else:
    score = (15-avg_angle_diff) * 0.67
    score = score + 90


print("score:", score)

requests.get('http://127.0.0.1:8000/user_app/score_user/',{'exercise_id':exercise_id,'user_id':user
_id,'score':score})
```