

# **DEVELOPING AN SMART AI GYM TRACKER**

**1922708-A PROJECT REPORT-PHASE 1**

**BATCH NUMBER: 23PP108**

**Submitted by**

MERLINE.P 142220243023

NAVEEN.J 142220243026

RISHITHA.M 142220243032

**In partial fulfillment for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**



**SRM VALLIAMMAI ENGINEERING COLLEGE**

**(AN AUTONOMOUS INSTITUTION)**

**SRM NAGAR, KATTANKULATHUR,  
CHENGALPATTU**

**ANNA UNIVERSITY :: CHENNAI 600 025**

**NOVEMBER 2023**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**DEVELOPING AN SMART AI GYM TRACKER**” is the bonafide work of “MERLINE.P, NAVEEN.J, RISHITHA.M” who carried out the project under my Supervision.

### **SIGNATURE**

**Dr.B. MUTHU SENTHIL,B.E.,M.E.,Ph.D.**

### **PROFESSOR & HOD**

Department Of Artificial Intelligence

And Data Science

SRM Valliammai Engineering College

(An Autonomous Institution)

SRM Nagar, Kattankulathur

Chengalpattu – 603 203

### **SIGNATURE**

**Mr.P. SIRENJEEVI B.E.,M.Tech.,(Ph.D)**

### **SUPERVISOR**

Department Of Artificial Intelligence

And Data Science

SRM Valliammai Engineering college

(An Autonomous Institution)

SRM Nagar, Kattankulathur

Chengalpattu – 603 203

**Submitted for the viva voce held on ..... at SRM Valliammai Engineering College,Kattankulathur.**

**INTERNAL EXAMINER  
EXAMINER**

**EXTERNAL**

## ACKNOWLEDGEMENT

We express our sincere gratitude to our respected Director of our college **Dr. Chidhambararajan.B, M.E., Ph.D.**, for his constant encouragement, which has been our motivation to strive towards excellence.

We thank our Principal **Dr. Murugan.M, M.E., Ph.D.**, for his constant encouragement, which has been our motivation to strive towards excellence.

We thank our vice principal **Dr. Visalakshi.S, M.E., Ph.D.**, for her constant encouragement, which has been our motivation to strive towards excellence.

We also extend our heartfelt respect to our Head of the Department, **Dr. Muthu Senthil.B, M.E., Ph.D., Post Doc (South Korea), Professor** for offering his sincere support throughout the project work.

We thank our Project Coordinator **Mr.Sankaranarayanan.R, M.Tech., (Ph.D.)**, **Assistant Professor(O.G)** for his consistent guidance and encouragement throughout the progress of the project.

We are grateful to our Project guide **Mr.Sirenjeevi.P, B.E., M.Tech.,(Ph.D.)**, **Assistant Professor(Sr.G)** for his valuable guidance, support and active interest for the successful implementation of the project.

We would also thank all the **Teaching and Non-Teaching staff members** of our department, for their constant support and encouragement during the course of this project work.

We finally thank our friends and family for their support during the course of the project.

## **ABSTRACT**

The rapid advancement of technology has brought about transformative changes in various aspects of our daily lives, and fitness is no exception. This project work introduces the concept of a Smart AI Gym Tracker, an innovative system designed to enhance the fitness tracking experience for gym-goers. This AI-powered fitness tracker utilizes cutting-edge technologies to provide real-time, personalized guidance and feedback to users during their workouts.

The Smart AI Gym Tracker leverages machine learning and sensor technology to monitor and analyze various fitness metrics, such as heart rate, motion data, and workout duration. This data is processed and interpreted by the AI algorithm to provide personalized exercise recommendations, ensuring that users optimize their workout routines based on their fitness goals and progress. The AI system also offers guidance on proper exercise form and technique, reducing the risk of injuries and promoting long-term fitness success.

**KEYWORDS:** Smart AI Gym, Fitness, Machine Learning, Artificial Intelligence

# TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGENO.
1	<b>ABSTRACT</b>	iv
	LIST OF TABLES	vii
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	ix
	LIST OF SYMBOLS	x
1 .	<b>INTRODUCTION</b>	
	1.1 Introduction	1
	1.2 Problem definition	2
	1.3 Objectives	3
	1.4 Benefits	4
2.	<b>LITERATURE REVIEW</b>	
	2.1 Literature Review	5
	2.2 Summary of Literature Survey	6
3.	<b>SYSTEM ANALYSIS</b>	
	3.1 Existing System	8
	3.2 Disadvantages of existing system	8
	3.3 Proposed System	9
	3.4 Advantages Of Proposed System	9
	3.5 Architecture Diagram With Explanation	10
	3.6 Module Development	11
	3.7 UML Diagrams	17
	3.8 Performance Evaluation	21

## **4**

### **REQUIREMENTS**

4.1 Hardware Requirements	24
4.2 Software Requirements	24
4.3 Tools	24
4.4 Libraries Used	25
4.5 Methodology	28
4.6 Results	36

## **5**

### **CONCLUSION AND FUTURE WORKS**

5.1 Conclusion	37
5.2 Future & Scope	37
Appendix	38
References	47

## **LIST OF TABLES**

<b>TABLE NO.</b>	<b>TABLE NAME</b>	<b>PAGE NO.</b>
1	Summary Of Literature Survey	6

## **LIST OF FIGURES**

<b>FIGURE NO</b>	<b>FIGURE NAME</b>	<b>PAGE NO</b>
1	System Architecture	10
2	Uml Diagrams	17
3	Use Case Diagram	18
4	Class Diagram	19
5	State Chart Diagram	20
6	Overall Output	38






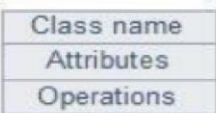



## **LIST OF ABBREVIATIONS**

### **ABBREVIATIONS**

### **EXPANSIONS**

ML	Machine Learning
NLP	Natural Language Processing
RL	Reinforcement Learning
NN	Neural Networks
RAM	Random Access Memory
ROM	Read Only Memory
OS	Operating System

## LIST OF SYMBOLS

SYMBOLS	DESCRIPTION
	An entity is a source of data or a destination for data.
	A data flow shows the flow of Information from its source to its destination
	A process shows a transformation or manipulation of data flow within the system.
	Classes are used to represent objects. Objects can be anything having properties and responsibility.
	The start symbol represents the beginning of a process or workflow in an activity diagram.
	The end symbol shown represents the completion of a process or workflow.
	A use case is the specification of a set of actions performed by system, which yields an observable result that is typically of value for one or more actors or other stakeholders of the system.

# **CHAPTER 1**

## **INTRODUCTION**

## **1.1 INTRODUCTION:**

In an era where technological innovations have become deeply integrated into our daily lives, the realm of fitness and health is no exception. With a growing emphasis on personal well-being and the pursuit of healthier lifestyles, the need for smarter, more sophisticated tools to aid individuals in their fitness journeys has never been more pronounced. Enter the Smart AI Gym Tracker, a groundbreaking solution designed to revolutionize the way we approach and achieve our fitness goals.

The concept of the Smart AI Gym Tracker is born through the integration of cutting-edge artificial intelligence and the world of physical fitness. It represents a significant leap forward in the way we monitor, track, and enhance our exercise routines, leveraging the power of data analysis, real-time feedback, and personalized guidance. Whether you're an experienced athlete striving for peak performance or someone just beginning their fitness journey, this innovative technology promises to be a game-changer.

## **1.2 PROBLEM STATEMENT:**

The problem is to design and create a Smart AI Gym Tracker that can assist gym-goers in tracking, optimizing, and improving their fitness routines. The system should address the following challenges:

Gym-goers often struggle to accurately track their workout performance, including the number of sets and repetitions, exercise form, and progress over time. The AI Gym Tracker should offer real-time monitoring and feedback to ensure users maintain proper form and execute their workouts correctly. Progress Tracking: Keeping track of progress and results can be cumbersome. The AI Gym Tracker should automatically log workout data, analyze it, and generate visual representations of progress over time, making it easier for users to evaluate their achievements and make data-driven decisions.

Adaptive Guidance and Motivation: Many users lack motivation or require additional guidance during their workouts. The AI Gym Tracker should offer real-time feedback, motivation, and adaptive recommendations to help users push their boundaries and stay on track to achieve their fitness goals. Preventing Injuries: Injuries can occur due to improper form, overtraining, or pushing beyond one's limits.

### **1.3 OBJECTIVES**

The objective of the development of a Smart AI Gym Tracker is to provide users with a comprehensive and real-time view of their exercise sessions. This involves the collection and analysis of various fitness metrics, such as heart rate, exercise duration, calorie expenditure, and motion data. By integrating advanced sensor technology, the system ensures accurate data collection, allowing users to track their performance and progress with precision. This objective aims to empower users with the information needed to make informed decisions and optimize their fitness routines for better results.

The aim is to design a system that is user-friendly and easy to navigate, ensuring that individuals of all fitness levels and age groups can use it effectively. Accessibility considerations also encompass cross-device compatibility to provide a seamless user experience on various platforms, such as mobile devices and tablets. By achieving this objective, the Smart AI Gym Tracker aims to make fitness tracking and guidance readily available and usable by a broad and diverse user base.

This involves the implementation of features designed to inspire and encourage users, such as gamification elements, challenges, and rewards. By fostering a sense of achievement and progress, the system aims to maintain user commitment and motivation. Additionally, providing social connectivity features allows users to share their accomplishments and engage with fitness communities, creating a supportive and motivating environment. This objective strives to enhance the overall fitness experience and drive users toward their goals.

### **1.3 BENEFITS**

- ✓ **Personalized Workouts:** AI can analyze your fitness goals, physical capabilities, and progress to create customized workout plans tailored to your needs and preferences.
- ✓ **Real-time Feedback:** AI trackers can provide real-time feedback on your form, technique, and performance during exercises, helping you make immediate adjustments for better results and injury prevention.
- ✓ **Progress Tracking:** They can keep a detailed record of your workouts, helping you monitor your progress, set achievable goals, and adapt your training routine as needed.
- ✓ **Motivation:** AI gym trackers can motivate users through gamification, challenges, and goal-setting features, making the fitness journey more engaging.
- ✓ **Corrective Suggestions:** AI can identify and provide suggestions for correcting common exercise mistakes, reducing the risk of injury and improving efficiency.
- ✓ **Time Efficiency:** AI can help optimize your workout routines to make the most of your time, focusing on exercises that offer the greatest benefits for your goals.
- ✓ **Nutrition and Recovery Guidance:** Some smart AI gym trackers may also offer nutritional advice and recovery tips to support your overall health and fitness.

## **CHAPTER 2**

### **LITERATURE REVIEW**



## **2.1 LITERATURE REVIEW**

**1. “trAIner - An AI Fitness Coach Solution” , Vaibhav Singh, Gaurang Pawar[1], in 18 July 2022.**

It is a representation of the type of AI-driven fitness coaching platforms that have become increasingly popular in recent years. Such solutions aim to make fitness more accessible, convenient, and effective by providing personalized guidance and support based on individual goals and capabilities.

**2. Employing Machine Learning Techniques to Categorize users in a Fitness Application, Shyamali Das; Pamela Chaudhury[2], in 19-20 November 2022.**

It involves using artificial intelligence and data-driven methods to group app users into different categories or segments based on various characteristics, behaviors, or preferences. This can help fitness app developers provide more tailored and personalized experiences to their users.

**3. “AI & multi-resolution Temporal Processing for Accurate Counting of Exercises Repetitions”, Youssef Fayad; Ahmed Khairy Mahmoud[3], 13-15 July 2021 published in 2020.**

The phrase "AI & multi-resolution temporal preprocessing for accurate counting of exercise repetitions" refers to a sophisticated technique that utilizes artificial intelligence (AI) and a multi-resolution temporal preprocessing approach to improve the accuracy of counting the number of exercise repetitions performed by an individual.

**4. “Recognition technology of human body movement behavior in fitness exercise based on transfer learning”, Sahu et al [4] published in 2021.**

The "Recognition technology of human body movement behavior in fitness exercise based on transfer learning" is a technological approach that leverages pre-trained models and transfer learning to accurately identify and analyze how individuals perform fitness exercises.

## 2.2 SUMMARY OF LITRATURE SURVEY

S.NO	TITLE	YEAR OF PUBLICATION	AUTHOR NAME	CONCEPT
1.	“trAIner - An AI Fitness Coach Solution”	18 July 2022	Vaibhav Singh, Gaurang Pawar[1]	The app obtains users’ motion data by the use of a webcam, and then applies human pose estimation assisted with repetition counting and form evaluation via voice based real time feedback.
2.	“Employing Machine Learning Techniques to Categorize users in a Fitness Application”	19-20 November 2022	Shyamali Das; Pamela Chaudhury[2]	The K-means machine learning technique is used to cluster App users based on a variety of factors, and whether they are eligible for bonuses or reward points. This paper's research focused on user categorization using unsupervised learning based on cluster.

3.	“AI & multi-resolution Temporal Processing for Accurate Counting of Exercises Repetitions”	13-15 July 2021	Youssef Fayad; Ahmed Khairy Mahmoud[3]	In this paper a new algorithm relay on temporal processing for the combination of Computer Vision and deep machine learning techniques are introduced to analyze frames data accurately and report a feedback on the repetitions of exercises.
4.	“Recognition technology of human body movement behavior in fitness exercise based on transfer learning”	09-11 April 2021	Sahu et al[4]	The detection of human body movements in fitness exercise, the normative correction of human body movements in fitness exercise is carried out.

**CHAPTER 3**  
**SYSTEM ANALYSIS**

### **3.1 EXISTING SYSTEM:**

In the existing work the development of a Smart AI Gym Tracker is a fundamental component that leverages artificial intelligence (AI) and machine learning technologies to enhance the user's fitness experience. This module plays a central role in monitoring, analyzing, and optimizing workout routines.

Most existing systems provide static workout plans that don't adapt to the user's changing fitness level or preferences. Many existing systems are limited in their adaptability to different types of fitness equipment and often lack scalability for future feature enhancements.

### **3.2 DISADVANTAGES OF EXISTING SYSYTEM**

- **Limited Exercise Recognition:** Many existing smart AI gym tracker systems may have limitations in accurately recognizing and categorizing various exercises, especially for unconventional or customized workouts. This can lead to incorrect tracking and analysis.
- **Data Privacy Concerns:** Privacy issues can arise if user data, such as workout history and biometrics, are not adequately protected. Unauthorized access to this data could lead to privacy breaches and misuse.
- **User Dependency:** Users may become overly reliant on the AI system, potentially neglecting their ability to self-monitor and make informed decisions about their workouts.
- **Inaccurate Form Analysis:** While some systems offer form analysis, the accuracy of these analyses may vary.

### **3.3 PROPOSED SYSTEM**

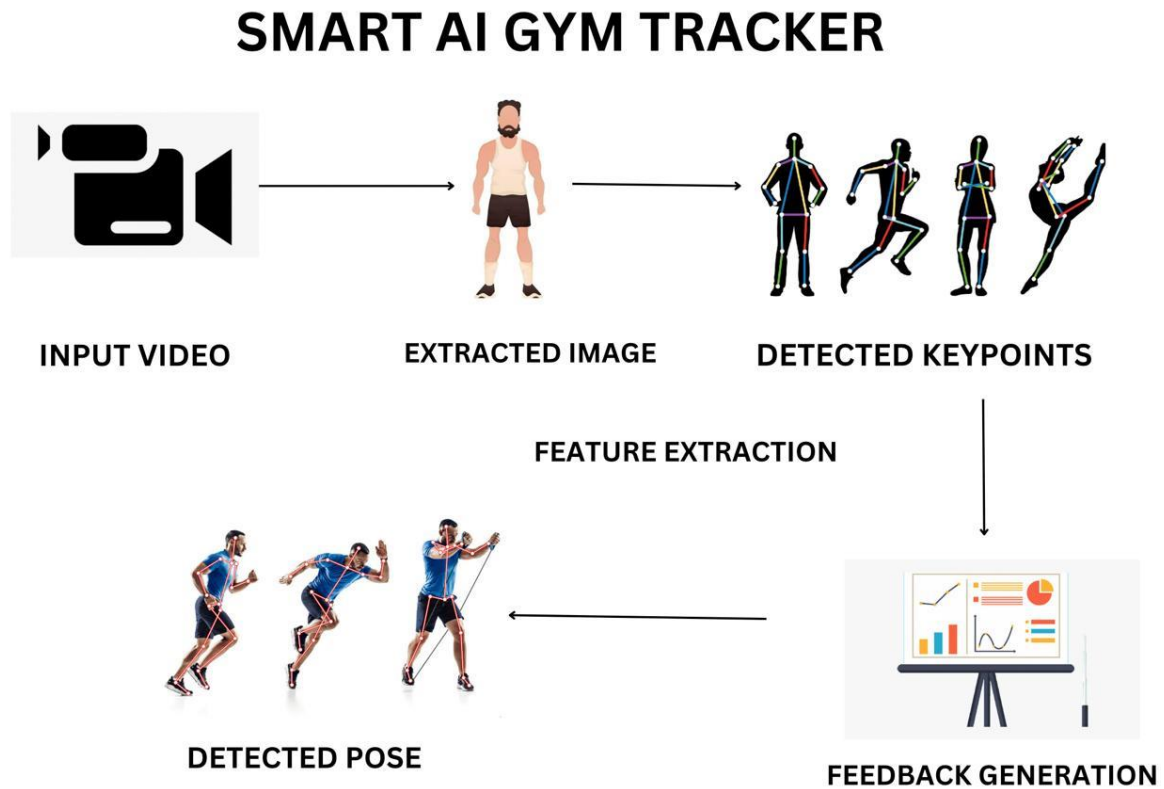
The proposed system for a Smart AI Gym Tracker represents an innovative and advanced solution that builds upon the limitations of the existing fitness tracking systems. It leverages AI and modern technology to offer an intelligent, personalized, and engaging fitness tracking experience.

The proposed system incorporates AI algorithms to enable real-time monitoring of the user's workouts. It continuously collects and analyzes data from various sensors, providing immediate feedback on exercise performance. The proposed system is designed to be scalable, accommodating new features and compatibility with various fitness equipment.

### **3.4 ADVANTAGES OF PROPOSED SYSTEM**

- **Precise Form Analysis:** The system is expected to provide more accurate form analysis, helping users avoid injuries and improve the effectiveness of their exercises.
- **Customized Workouts:** Personalized workout recommendations based on user goals, progress, and preferences will help users tailor their fitness routines to meet their specific needs.
- **Accessibility and Affordability:** Efforts will be made to ensure that the system is accessible and affordable for a wide range of users, making it inclusive for individuals with different budgets.
- **Seamless User Experience:** The system will prioritize an intuitive and user-friendly interface, reducing the learning curve and ensuring a smooth user experience.

### 3.5 ARCHITECTURE DIAGRAM WITH EXPLANATION



#### EXPLANATION:

- The data will be collected as human information and send it will be preprocessed.
- After preprocessing the data collected will be stored in the database.
- The data base will segregate the data and sends it to the network layer.
- Futher the data will be classified by logistic regression algorithm.
- The operation performed in the Basic training layer will be carried out by the training agent, it will detect key points and extracts.
- The preprocessed data which is stored in the data base will be performed using reinforcement learning algorithm.
- Finally it provides the output and display the result.

## 3.6 MODULE DESCRIPTION AND EXPLANATION

- ❖ **Dependencies Module**
- ❖ **Detecting Joints Module**
- ❖ **Determining Angles Module**
- ❖ **Curl Counter Module**
- ❖ **AI Powered Gym Tracker Module**

### 3.6.1 DEPENDENCIES MODULE

A "dependencies module" typically refers to a component or library used in software development to manage dependencies, which are external code or libraries that a software project relies on. These dependencies can be in the form of packages, libraries, modules, or plugins that provide specific functionality or features to the software application. A dependencies module helps keep track of these external dependencies and ensures they are properly managed.

#### **Code:**

```
import cv2
import mediapipe as mp
import numpy as np
mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
```



### 3.6.2 DETECTING JOINTS MODULE

A "detecting joints module" in the context of computer vision, robotics, or motion tracking typically refers to a component or software module that is designed to identify and locate key joint positions in human or object movements. This module plays a crucial role in various applications, such as gesture recognition, motion capture, human pose estimation, robotics, and even in the development of fitness applications like the "smart AI gym tracker."

#### Code:

```
cap = cv2.VideoCapture(0)

## Setup mediapipe instance

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

    while cap.isOpened():

        ret, frame = cap.read()

        # 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)

        # Extract landmarks

        try:

            landmarks = results.pose_landmarks.landmark

            print(landmarks)
```

```

except:

    pass

    # 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'):

        break

cap.release()

cv2.destroyAllWindows()

```

### 3.6.3 DETERMINING ANGLES MODULE

It refers to a software component or module designed to calculate and analyze angles between different objects, points, or entities. Such a module can be applied in various fields, including engineering, computer graphics, robotics, motion analysis, and more. a system that is designed to measure and calculate angles between different objects or points in various contexts.

Code:

```
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

```

### 3.6.4 CURL COUNTER MODULE

It is typically a software component or feature integrated into a fitness application or device designed to count and track the number of bicep curls performed during a strength training or weightlifting workout. Bicep curls are a common exercise used to build and tone the muscles in the upper arms (biceps). This module is useful for individuals who want to keep precise records of their workout progress.

#### Code:

```

cap = cv2.VideoCapture(0)

# Curl counter variables
counter = 0
stage = None

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

```

```

ret, frame = cap.read()

# 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)

```

### 3.6.5 AI POWERED GYM TRACKER MODULE

An AI-powered gym tracker is a central component in the development of a Smart AI Gym Tracker. This sophisticated module integrates artificial intelligence and machine learning to offer a comprehensive and personalized fitness tracking experience. Based on the user's fitness goals, historical data, and real-time performance, the AI generates personalized exercise recommendations. These recommendations adapt to the user's progress and preferences. Strong data encryption, access controls, and compliance with data protection regulations ensure the security and privacy of user data.

The AI-powered gym tracker is the heart of the Smart AI Gym Tracker, bringing intelligence, personalization, and real-time guidance to fitness tracking. It enables users to achieve their fitness goals with precision and motivation while promoting a safe and engaging workout experience.

#### Code:

```

cv2.rectangle(image, (0,0), (225,73), (245,117,16), -1)
# Rep data
cv2.putText(image, 'REPS', (15,12),

```

```

        cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE_AA)
    cv2.putText(image, str(counter),
                (10,60),
                cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2,
cv2.LINE_AA)
    # Stage data
    cv2.putText(image, 'STAGE', (65,12),
                cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE_AA)
    cv2.putText(image, stage,
                (60,60),
                cv2.FONT_HERSHEY_SIMPLEX, 2, (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'):
        break

cap.release()
cv2.destroyAllWindows()

```

### 3.7 UML DIAGRAM

Unified Modelling Language is a visual language for specifying, constructing and documenting the artifacts of systems. The UML defines various UML profiles that specialize subsets of the notation for common subject areas, such as diagramming Enterprise JavaBeans(with the UML EJB profile). The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

UML diagrams make abstract ideas and software systems easier to understand through visualization. This is beneficial for software engineers who need to collaborate when building software. UML diagrams also make it easier for software engineering teams to communicate with external stakeholders. Software engineers can use the UML model to explain how the system works to non-tech-savvy people.

.

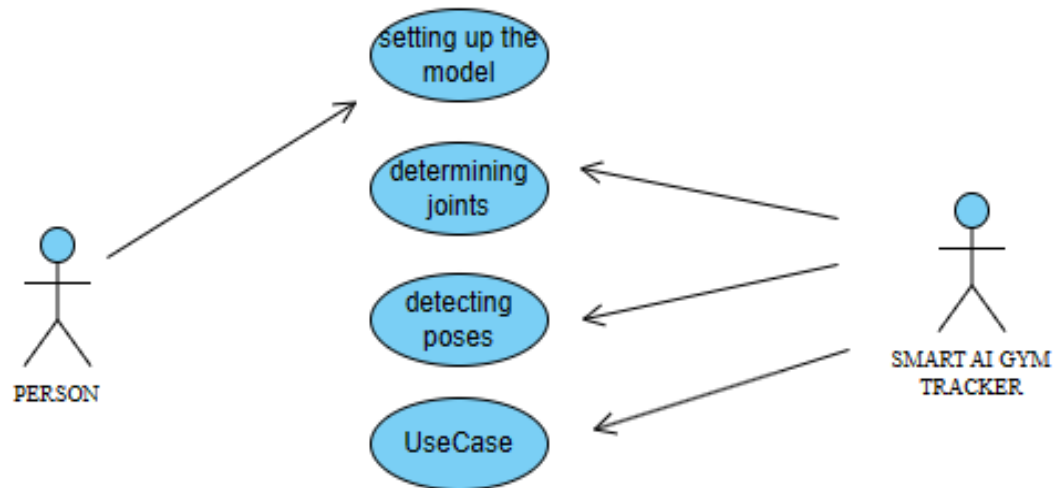
#### Types of UML diagrams

There are two subcategories of UML diagrams: structural diagrams and behavioral diagrams.

- Structural diagrams depict the components that make up a system and the relationship between those components. These diagrams show the static aspects of a system.
- Behavioral diagrams represent what happens within a system. They show how all the components interact with each other and with other systems or users.

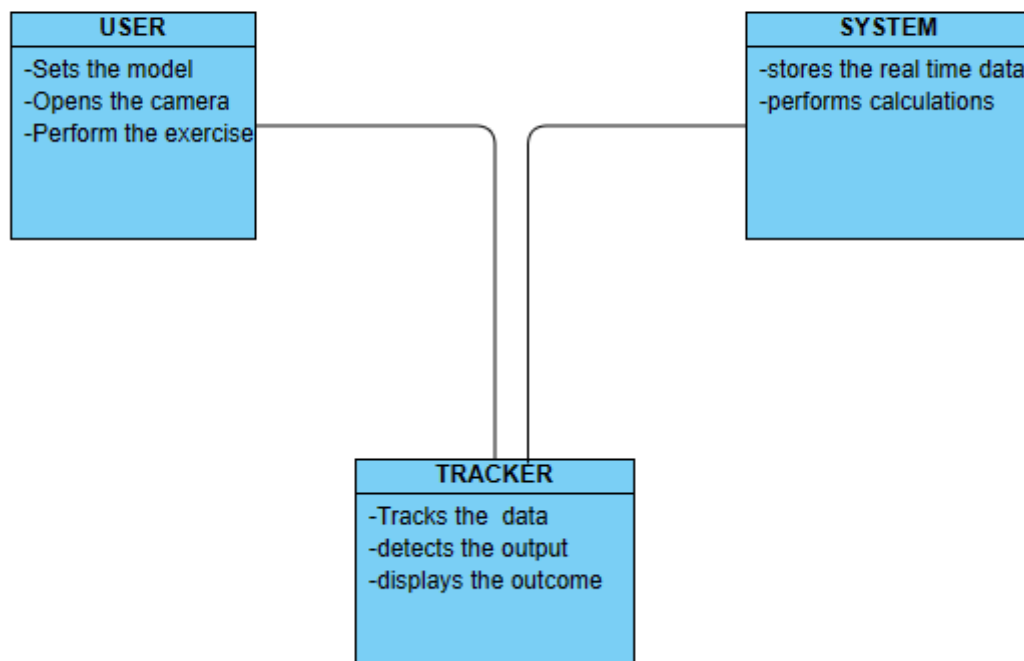
### 3.7.1 USE CASE DIAGRAM

Use case is a collection of related success and failure scenarios that describes an actor using a system to support a goals.



### 3.7.2 CLASS DIAGRAM

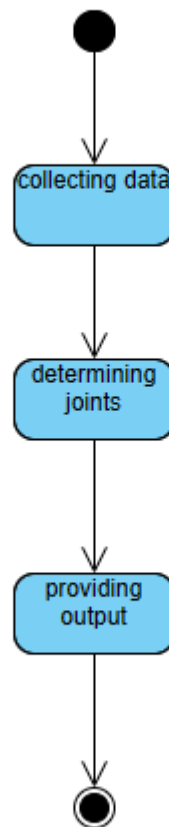
Class diagram are used to show the different objects in a system, their attributes, operations and a relationship between them. They are used for static object modelling. In figure.no.3.6.2.1 describes class diagram analysing diseases for climatic changes using ontology. It helps in identifying the relationships between objects and provides a clear picture of the overall structure of the system. The class diagram also helps in code generation and provides a blueprint for the software development team.





### 3.7.3 STATE CHART DIAGRAM

State chart diagram is used to model the dynamic nature of a system. It illustrates the interesting events and states of an object, the behaviour of an object in an event. Transitions are shown as arrows, labelled with their events. Statechart diagrams can be used to model complex systems .They are useful for visualizing and analyzing the behavior of a system, identifying potential problems, and improving system performance and reliability.

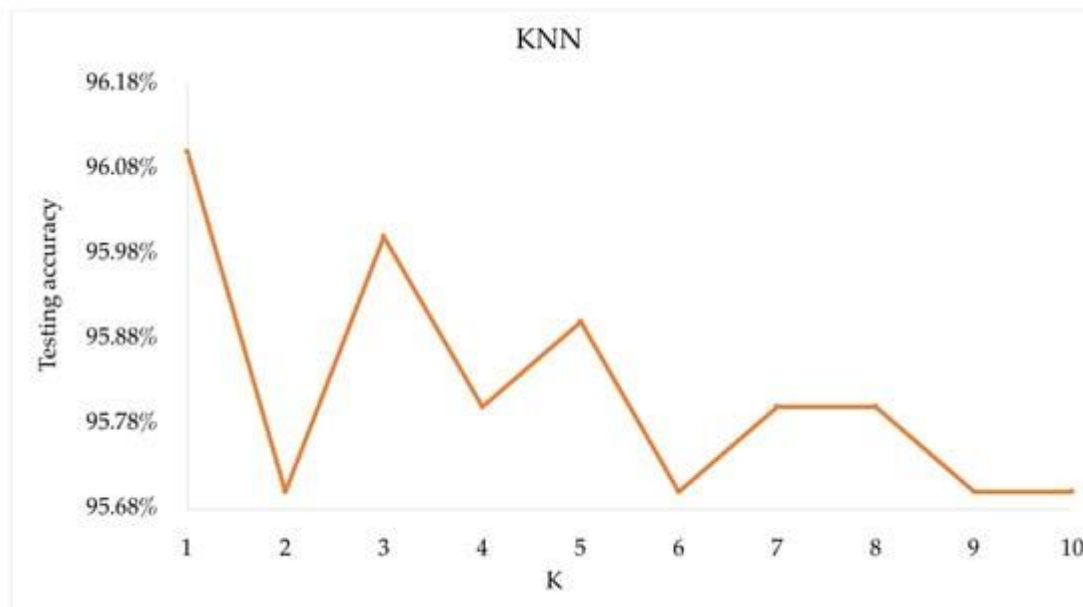


### 3.8 PERFORMANCE EVALUATION:

#### Hyper-Parameter Tunning

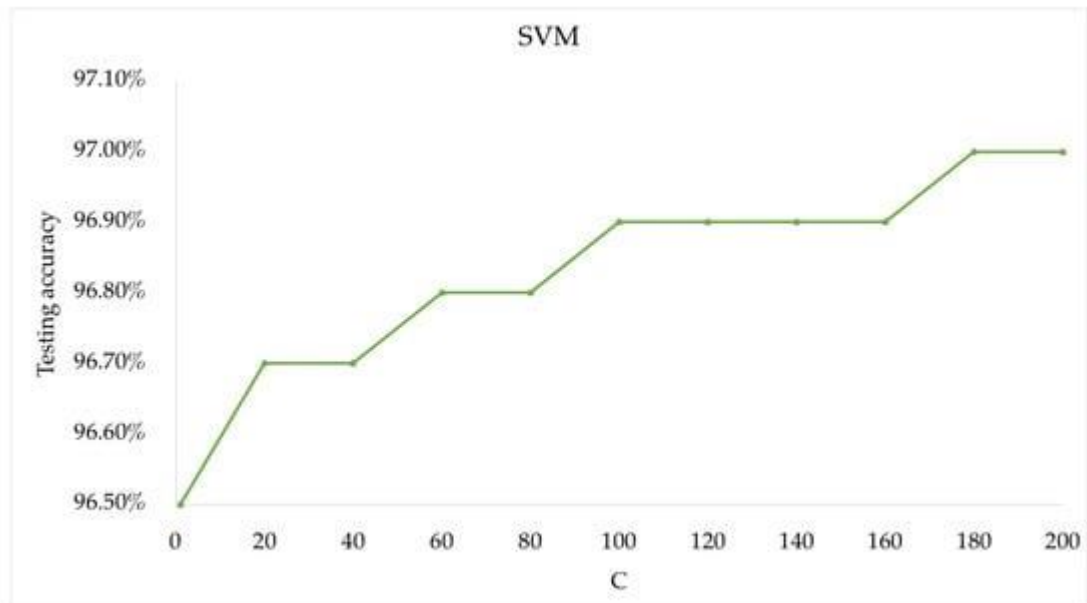
In this section, we present the process of hyper-parameter tuning to all four algorithms. The number of neighbors is tuned for KNN; the value of C is tuned for SVM; the number of neurons and the activation function are tuned for SNN and DNN; an additional hyper-parameter, the number of hidden layers, is also adjusted for DNN.

**Figure 10** offers how the testing accuracy of KNN responds to the change of the number of neighbors. The testing accuracy starts at 0.961 and suffers an overall drop to 0.957 at the number of neighbors of 10. The trend of the graph shows the optimal number of neighbors is one.



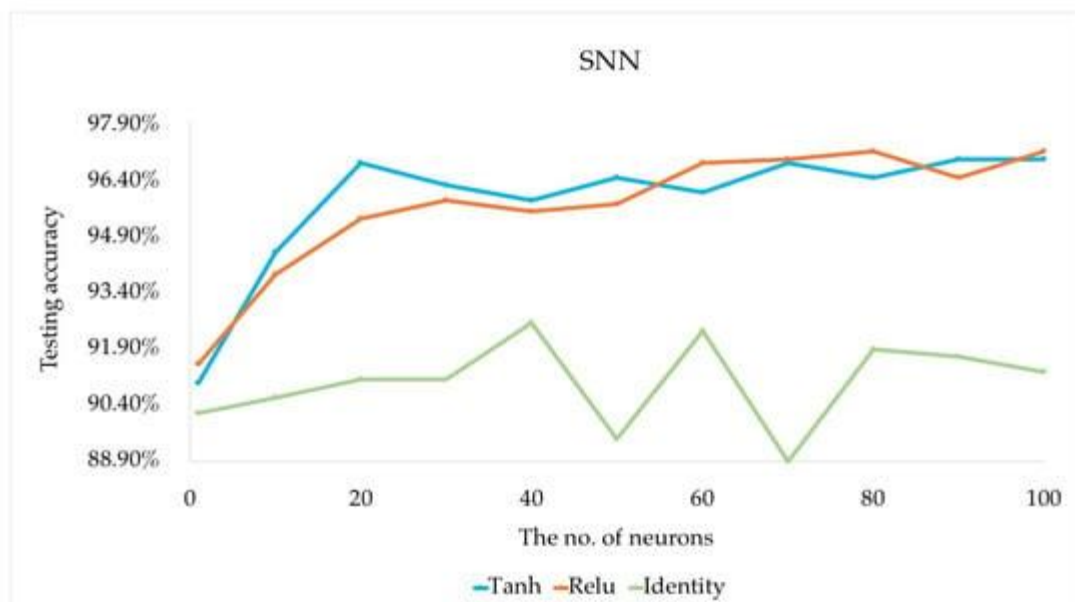
**Figure 10.** The testing accuracy of KNN for  $k = 1$  to  $k = 10$ .

The result obtained from **Figure 11** is that when C is equal to 180, the testing accuracy reaches its peak of testing accuracy 0.970 and remains at that level, although C is turned to 200.



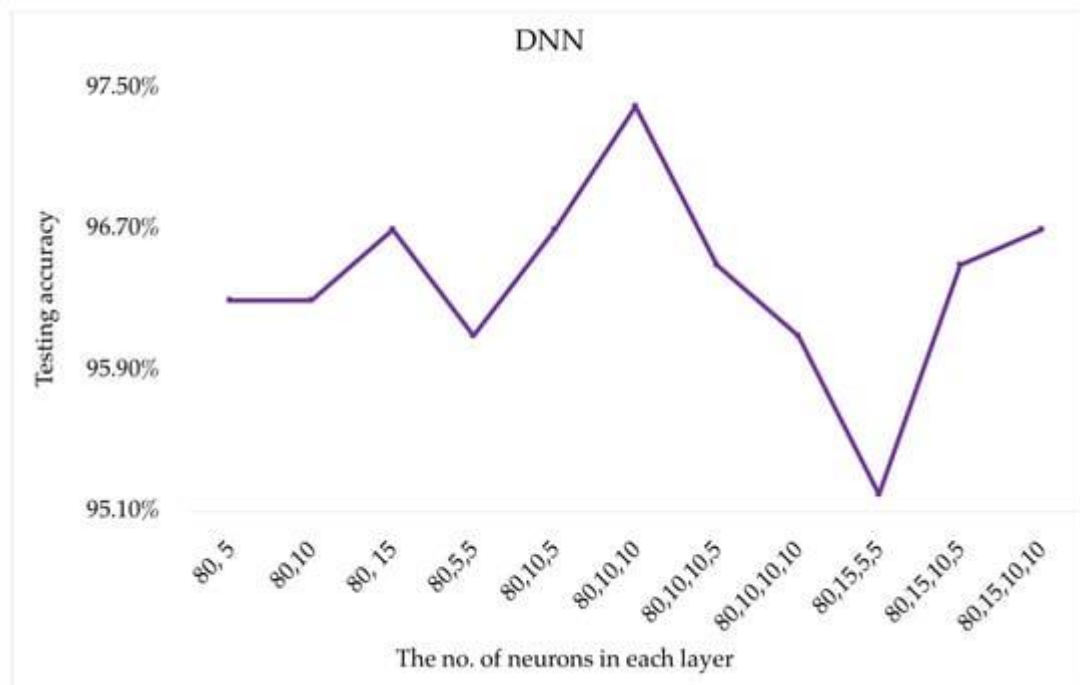
**Figure 11.** The testing accuracy of SVM associated with C.

What stands out from [Figure 12](#) is that the optimal result of 0.972 is provided by the relu and 80 neurons configuration. We can see a higher performance reported by the tanh and relu compared to identity.



**Figure 12.** The testing accuracy of SNN responses to the three types of activation function and the number of neurons.

From [Table A4](#), [Table A5](#) and [Table A6](#), we already know that the 80 neurons provide the optimal performance for SNN with the relu activation function. As aforementioned, the DNN features two or more hidden layers compared to the SNN. Therefore, the DNN is tuned by adding more hidden layers to the optimal SNN to see any improvements. [Figure 13](#) presents the optimal capability is 0.974 procured by 3 hidden layers with 80, 10, and 10 neurons in each hidden layer. As can be seen from [Table A5](#) and [Table A7](#), there is a slight rise in the testing accuracy of 0.002 from SNN to DNN. Overfitting refers to the trained model having poor performance on the testing dataset but good performance on the training dataset [22]. There is no significant gap between them the training accuracy and testing accuracy in each model. The model does not encounter an overfitting problem.



**Figure 13.** The testing accuracy of DNN responses to the No. layers, the No. neurons in each layer, and the Relu function.

## **CHAPTER 4**

### **SYSTEM ANALYSIS**

## **4.1 HARDWARE REQUIREMENTS:**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list, especially in case of operating systems. The minimal hardware requirements are as follows,

- System : Intel Core i3|i5
- Hard Disk : 512 GB
- RAM: 8 GB

## **4.2 SOFTWARE REQUIREMENTS**

The software requirements are description of features and functionalities of the target system. Requirements convey the expectations of users from the software product. Software requirement definition involves identifying and documenting the user requirements, business needs, and technical specifications of the software. It also involves defining the system architecture, hardware and software components, and testing and validation requirements.

Language: Python

IDE: Jupyter Notebook

## **4.3 TOOLS**

- Machine Learning Frameworks : scikit-learn
- Data Analysis Libraries: Numpy
- Integrated Development Environments (IDEs):ANACONDA

## 4.4 LIBRARIES USED:

### NUMPY:

NumPy (Numerical Python) is a fundamental library for numerical and scientific computing in Python. It provides support for working with large, multi-dimensional arrays and matrices, along with a collection of high-level mathematical functions to operate on these arrays. NumPy's highly optimized array operations can improve the performance of AI algorithms and mathematical computations within the Gym Tracker. This optimization is crucial for real-time feedback and guidance during workouts. Incorporating AI into a Smart AI Gym Tracker enhances its ability to provide real-time, personalized, and data-driven fitness guidance and support. It allows users to receive insights and recommendations tailored to their specific needs and helps them achieve their fitness goals more effectively. The AI-powered gym tracker can continuously learn and adapt to user behavior, making it a valuable tool for anyone looking to optimize their gym experience.

### Features of NumPy

- NumPy has various features including these important ones:
- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

## **MEDIAPIPE:**

MediaPipe is a versatile and open-source framework developed by Google that provides a wide range of pre-trained machine learning models for various tasks related to computer vision, including body and pose tracking. In the context of a Smart AI Gym Tracker, MediaPipe can be used for real-time body tracking, exercise form analysis, and posture correction. MediaPipe's Pose models can accurately detect and track key points on a person's body, such as the position of the head, neck, shoulders, elbows, wrists, hips, knees, and ankles. You can use this functionality to monitor a user's body posture during exercises. This is particularly useful for ensuring proper form in exercises like squats, deadlifts, and push-up. NumPy's fast and efficient array operations can be used for signal processing tasks, such as filtering noisy sensor data (e.g., accelerometers, gyroscopes) to improve accuracy in tracking body movements during workouts.

### Features of Mediapipe:

- Cross-Platform Support
- Pre-Trained Models
- Pose Estimation
- Objectron
- Hand Tracking



## **OPEN CV:**

OpenCV (Open Source Computer Vision Library) can be used in a Smart AI Gym Tracker in several ways to enhance its functionality and capabilities. OpenCV is a versatile open-source computer vision library that provides various tools for image and video analysis, which can be valuable for tracking and analyzing movements, body posture, and more in the context of A gym tracker.

OpenCV can be used for human pose estimation, which involves detecting and tracking key points on the human body, such as joints and limbs. This allows the AI Gym Tracker to assess the user's posture and movement during exercises, providing real-time feedback on form and technique. OpenCV can be utilized to recognize and classify different exercises being performed by the user. This can help the tracker provide exercise-specific advice and track performance on a per-exercise basis.

## **Features of Opencv:**

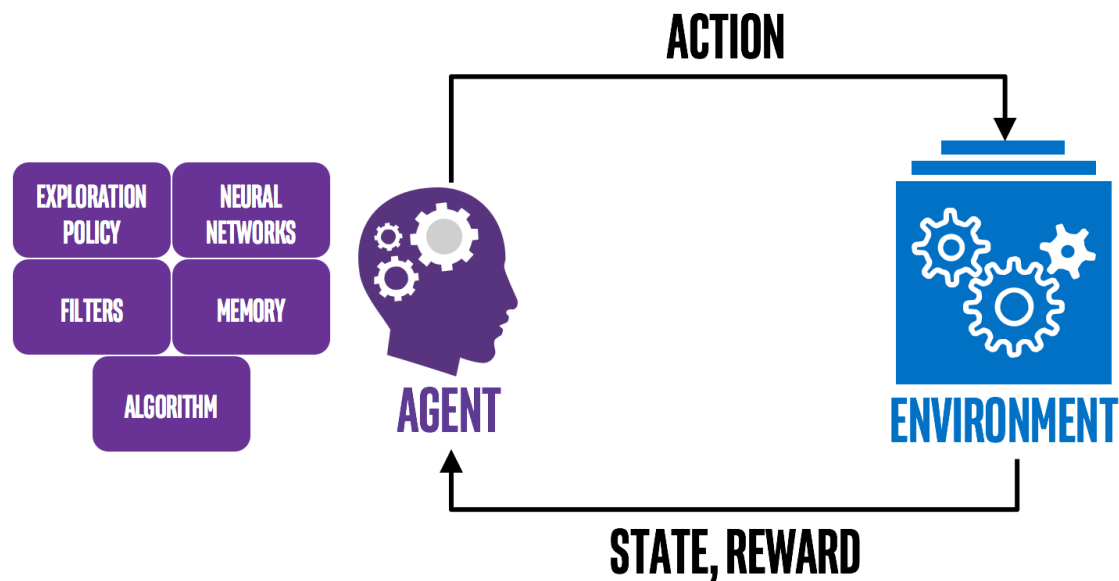
- Machine Learning Support
- Image Stitching
- Blob Analysis
- Extensive Documentation and Community Support
- Open Source and Free

## 4.5 METHODOLOGY

The methodology adopted for **developing an smart AI gym tracker**.

### REINFORCEMENT LEARNING:

Reinforcement Learning (RL) is a type of machine learning algorithm that focuses on training intelligent agents to make sequences of decisions to achieve to goal.



Common usages of reinforcement learning models include the following:

**Real-Time Feedback:** RL can be used to provide real-time feedback to gym-goers. For example, an RL agent can monitor a user's performance during a workout and adjust exercise difficulty or intensity to match their skill level and goals. It can adapt the user's training plan on the fly based on their performance and progress.

**Personalized Workout Plans:** RL algorithms can create personalized workout plans by considering a user's fitness level, preferences, and goals. The agent can continuously adapt the plan as the user's fitness improves or their goals change.

**Optimizing Exercise Intensity:** RL can optimize exercise intensity and rest periods to maximize the effectiveness of a workout. It can learn from the user's performance data and adjust the workout parameters to keep the user challenged without overexerting them.

**Form and Technique Correction:** RL agents can analyze the user's exercise form and provide feedback for corrections. For example, it can monitor weightlifting exercises and advise users on proper posture and technique to prevent injuries.

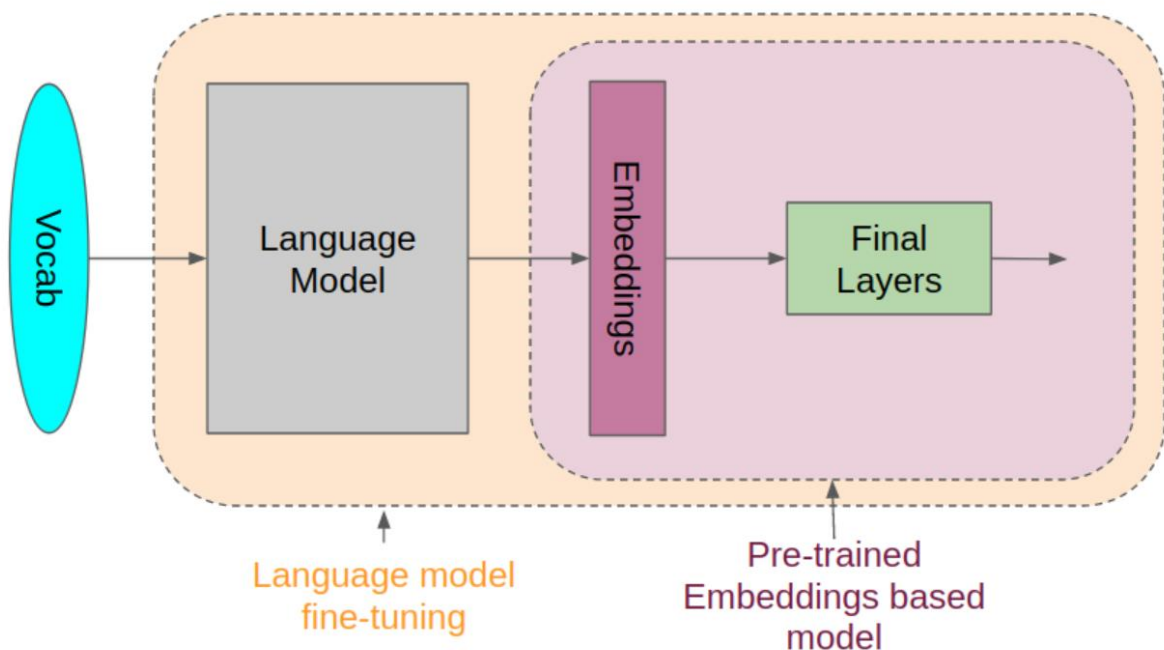
## NATURAL LANGUAGE PROCESSING ALGORITHM

Natural Language Processing (NLP) is a subfield of machine learning that focuses on the interaction between human language and computers. It involves the use of computational techniques to understand, analyze, and generate human language. NLP techniques are widely used in various applications such as sentiment analysis, chatbots, language translation, speech recognition, and text classification, among others.

Common usages of reinforcement learning models include the following:

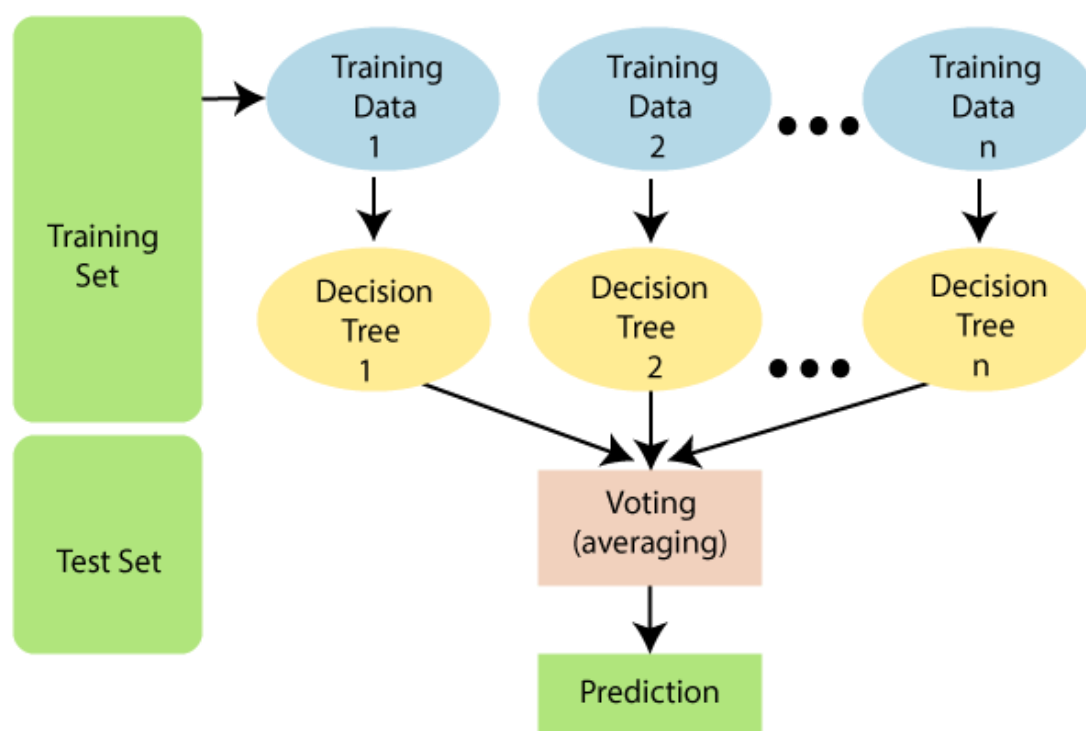
**Information Extraction:** Identifying and extracting specific information from unstructured text, such as extracting names and dates from news articles or medical records.

**Topic Modeling:** Identifying topics within a collection of documents. Topic modeling is used in content categorization, organizing large datasets.



## RANDOM FOREST

Random Forest is a powerful machine learning algorithm used for both classification and regression tasks. It belongs to the family of ensemble methods, which means that it combines the predictions of multiple models to improve accuracy and reduce overfitting. The basic idea behind Random Forest is to build : multiple decision trees on random subsets of the training data, and then aggregate their predictions to make the final prediction. Each decision tree is built using a subset of the features, chosen randomly at each split, which helps to reduce correlation between the trees and increase their diversity. During the training phase, each decision tree is grown by recursively splitting the data into smaller subsets based on the values of the features, until the leaf nodes contain data points that belong to the same class or have similar regression targets. The split criteria are chosen to maximize the information gain or decrease the impurity of the data at each split. During the prediction phase, the new input data is passed through each of the decision trees, and the final prediction is made by taking the majority vote (in classification) or the average (in regression) of the individual predictions.



## SVM:

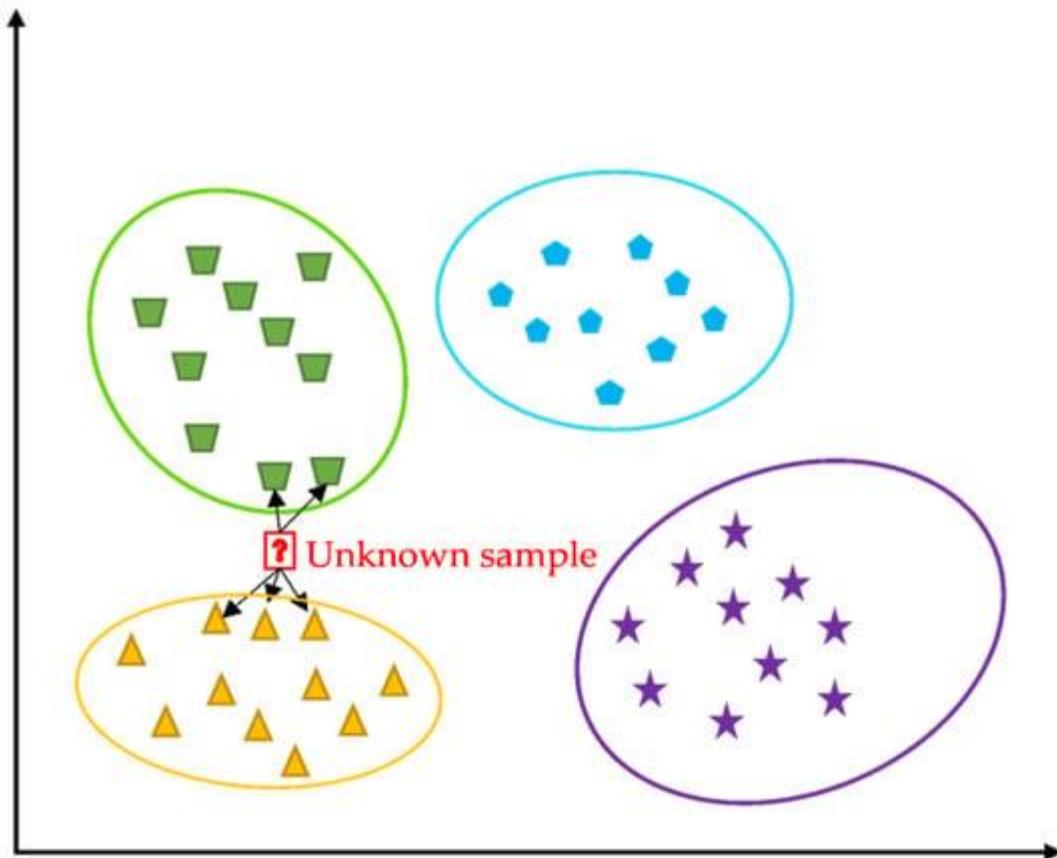
Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well it's best suited for classification. The main objective of the SVM algorithm is to find the optimal hyperplane in an N-dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

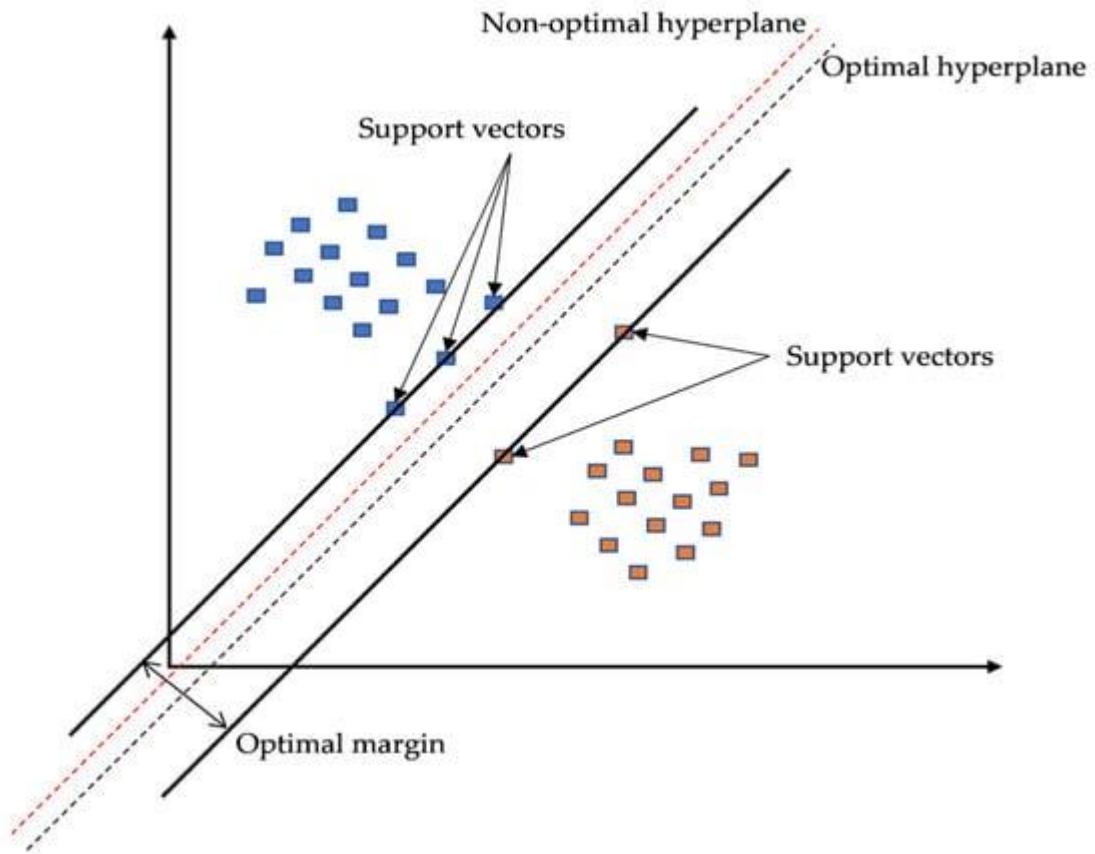
Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog.

KNN, in pattern recognition, classifies samples based on the distance to the closest training samples in the feature map exemplified in [Figure 5](#). It is noticeable that two axes are used to reflect two attributes of the sample in [Figure 5](#). The meaning of the axis is also applicable in [Figure 6](#) and [Figure 7](#). An unknown sample is assigned to one of the groups according to a majority vote of its k nearest neighbors. As can be seen in [Figure 5](#), the k nearest neighbors is set to 5 (the k must be a positive integer and can be tuned to obtain the optimal performance), which implies that the five nearest points are identified by measuring the Euclidean distance between two points, and the unknown point is compared to the 5 points to

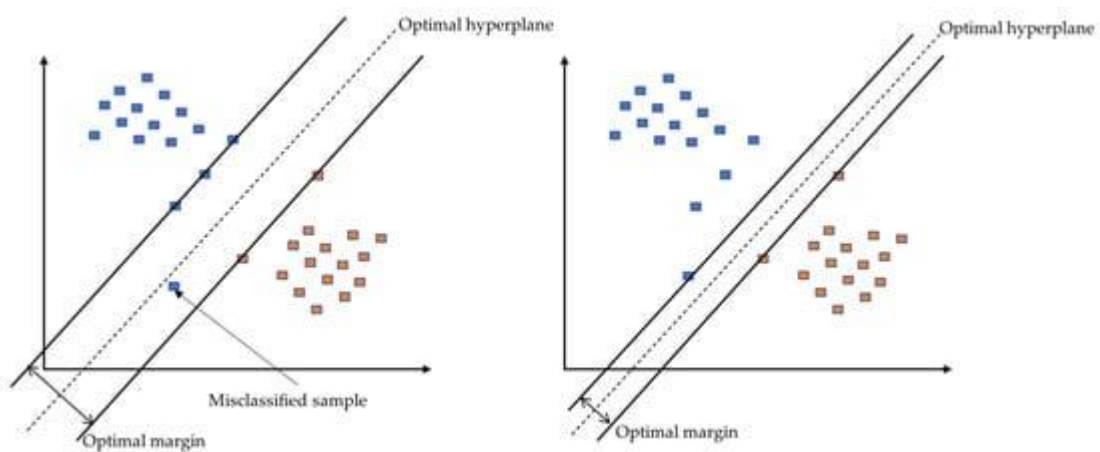
determine which group the unknown point is allocated to. During the five closest samples, the majority are yellow samples, so that the unknown sample is classified to the yellow group.



**Figure 5.** KNN classification example for  $k = 5$ . Four different shapes point to four different groups.



**Figure 6.** SVM classification example for two-group problems.



**Figure 7.** An example of the impact of low  $C$  (left) and high  $C$  (right).

SVM was first introduced for 2-group classification problems [19]. The binary input vectors are mapped to a feature map, and then the SVM is used to determine the hyperplane that can separate the binary input vectors seen from [Figure 6](#). The support vectors characterize the most significant margin between the two groups. Many hyperplanes can separate the two classes, such as the two hyperplanes shown in [Figure 6](#). SVM strives to find an optimal hyperplane that is the most far away (optimal margin) from any samples in the feature space. If the misclassified samples cannot be avoided, a constant  $C$  is introduced to define the tolerance of the misclassified samples. From [Figure 7](#) (left), a low value of  $C$  ignores the misclassified sample to achieve a large optimal margin. In contrast, the right of [Figure 7](#) provides a small optimal margin due to the immense value of  $C$  not allowing an error. The extension of SVM from binary classification to multiclass classification has been well developed and assessed in [20]. An artificial neural network (ANN) is inspired by the human brain structure [21], which is the fundamental reason it has intelligence. [Figure 8](#) provides a typical example of SNN with only one hidden layer between the input and output layers.  $W$ ,  $b$ , and  $z$  are the weights, biases, and the output of the hidden layer; the activation function denotes tanh, relu, or identity; and  $X$  and  $y$  are the input and output vectors, respectively. DNN shares the characteristics with SNN but more than one hidden layer.



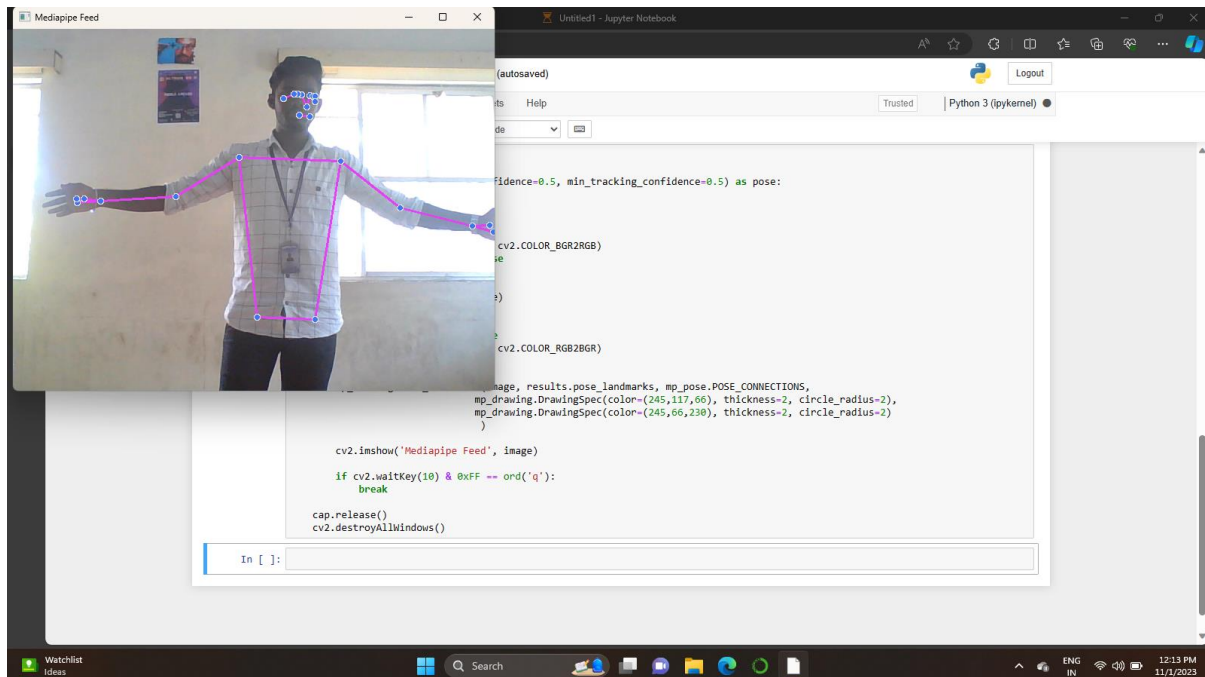
## 4.6 RESULTS:

### DETERMINING JOINTS

- Determining joints in a Smart AI Gym Tracker involves a process known as human pose estimation.
- Human pose estimation aims to identify and track key points, or joints, on the human body in images or video frames.
- This capability is essential for tracking body movements and postures during gym workouts.
- To determine joints in a Smart AI Gym Tracker, you can use various approaches and technologies, including computer vision techniques and machine learning models.



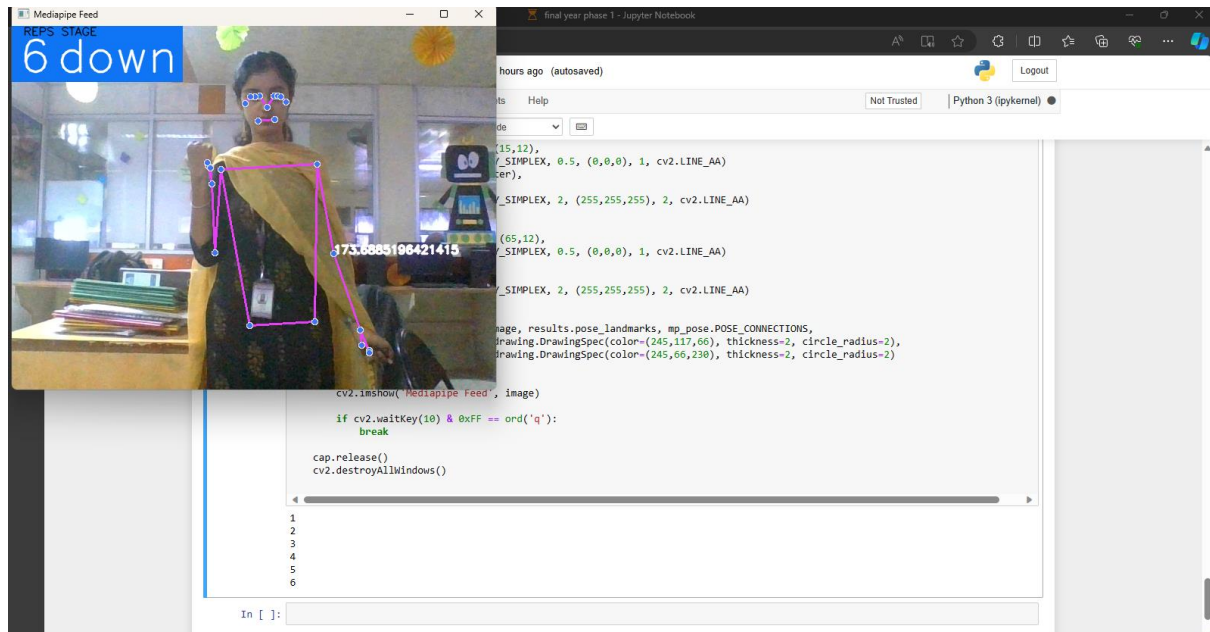
## DETERMINING JOINTS(FULL BODY)



## AI POWERED GYM TRACKER:

- AI can analyze user data, including fitness goals, previous workout performance, and body metrics, to generate personalized workout plans.
- These plans can adapt over time to help users reach their fitness goals more efficiently.
- AI can offer real-time feedback during workouts, assessing factors like exercise form, pace, and effort.
- It can provide audio or visual cues to help users maintain proper technique and intensity.
- AI can offer personalized motivational messages and reminders to keep users engaged and motivated to continue their fitness journey.
- Injury Prevention: AI can monitor user performance for signs of overtraining or improper form and provide warnings to reduce the risk of injuries.

## OVERALL OUTPUT:



- From the above image we have tracked the body movements using mediapipe we are calculating the angles of the body and predicting the output.
- Our project aim is to predict the movements of the exercises and to calculate the performance.
- Also using this engine we can count the number of pushups taken by a person in a realtime.
- Hence it is user friendly and very simple to use.

## **CHAPTER 5**

## **CONCLUSION**

## **5.1 CONCLUSION:**

The smart AI gym tracker represents a groundbreaking advancement in the field of fitness technology. With the integration of artificial intelligence and sophisticated sensor technologies, it offers a comprehensive solution for individuals looking to achieve their fitness goals more effectively and efficiently. The smart AI gym tracker has transformed the way individuals approach fitness and exercise. It empowers users with the tools, guidance, and motivation needed to achieve their fitness goals and make tangible improvements in their physical well-being. With its continuous learning and adaptability, it promises a bright future in the world of fitness technology, offering users a powerful and dynamic fitness companion that supports their journey to a healthier lifestyle.

## **5.2 FUTURE & SCOPE:**

- Enhancement to meet the evolving needs and expectations of users.
- Here are some potential areas for future work and improvement in the smart AI gym tracker: Advanced Exercise Recognition, Multi-Modal Sensing, Enhanced Data Analytics, Personalized Nutrition Guidance, AI Coaching and Virtual Trainers.
- Future work for a smart AI gym tracker involves a commitment to innovation and user-centric design, with a focus on leveraging AI and technology to support users in achieving their fitness and wellness goals.
- The ongoing development and enhancement of the tracker will empower individuals to lead healthier and more active lives.

## APPENDIX:

```
!pip install mediapipe opencv-python
```

```
import mediapipe as mp
```

```
import numpy as np
```

```
mp_drawing = mp.solutions.drawing_utils
```

```
mp_pose = mp.solutions.pose
```

```
# VIDEO FEED
```

```
cap = cv2.VideoCapture(0)
```

```
while cap.isOpened():
```

```
    ret, frame = cap.read()
```

```
    cv2.imshow('Mediapipe Feed', frame)
```

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

```
        break
```

```
cap.release()
```

```
cv2.destroyAllWindows()
```

```
cap = cv2.VideoCapture(0)
```

```
## Setup mediapipe instance
```

```
mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5)
```

```
as pose:
```

```
    while cap.isOpened():
```

```
        ret, frame = cap.read()
```

```
        # 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)

# 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'):
    break

cap.release()
cv2.destroyAllWindows()
mp_drawing.DrawingSpec??

cap = cv2.VideoCapture(0)
## Setup mediapipe instance
with mp_pose.Pose(min_detection_confidence=0.5, min_tracking_confidence=0.5) as pose:
    while cap.isOpened():
        ret, frame = cap.read()

        # 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)

# Extract landmarks
try:
    landmarks = results.pose_landmarks.landmark
    print(landmarks)
except:
    pass

# 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'):
    break

cap.release()
cv2.destroyAllWindows()

len(landmarks)

for lndmrk in mp_pose.PoseLandmark:

```



```

    print(lndmrk)

landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].visibility

landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value]

landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value]

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

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

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

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

shoulder, elbow, wrist

calculate_angle(shoulder, elbow, wrist)

tuple(np.multiply(elbow, [640, 480]).astype(int))

cap = cv2.VideoCapture(0)
## Setup mediapipe instance

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

```

```

while cap.isOpened():
    ret, frame = cap.read()

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)

# Extract landmarks
try:
    landmarks = results.pose_landmarks.landmark

    # Get coordinates
    shoulder = [landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x, landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].y]
    elbow = [landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x, landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]
    wrist = [landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x, landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].y]

    # Calculate angle
    angle = calculate_angle(shoulder, elbow, wrist)

    # Visualize angle
    cv2.putText(image, str(angle),

```

```

        tuple(np.multiply(elbow, [640, 480]).astype(int)),
        cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA
    )

    except:
        pass

    # 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'):
        break
    cap.release()
    cv2.destroyAllWindows()

cap = cv2.VideoCapture(0)

# Curl counter variables
counter = 0
stage = None

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

```

```

ret, frame = cap.read()

# 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)

# Extract landmarks
try:
    landmarks = results.pose_landmarks.landmark

    # Get coordinates
    shoulder = [landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x, landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].y]
    elbow = [landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x, landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]
    wrist = [landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x, landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].y]

    # Calculate angle
    angle = calculate_angle(shoulder, elbow, wrist)

    # Visualize angle
    cv2.putText(image, str(angle),
                tuple(np.multiply(elbow, [640, 480]).astype(int)),

```

```

        cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA
    )

    # Curl counter logic
    if angle > 160:
        stage = "down"
    if angle < 30 and stage == 'down':
        stage = "up"
        counter += 1
        print(counter)

except:
    pass

# Render curl counter
# Setup status box
cv2.rectangle(image, (0,0), (225,73), (245,117,16), -1)

# Rep data
cv2.putText(image, 'REPS', (15,12),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE_AA)
cv2.putText(image, str(counter),
            (10,60),
            cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255), 2, cv2.LINE_AA)

# Stage data
cv2.putText(image, 'STAGE', (65,12),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,0), 1, cv2.LINE_AA)
cv2.putText(image, stage,
            (60,60),
            cv2.FONT_HERSHEY_SIMPLEX, 2, (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'):
    break

```

```

cap.release()
cv2.destroyAllWindows()

```

## REFERENCE:

1. "Composite Fields For Human Pose Estimation" By S Kreiss, L Bertoni, And A Alah, IEEE Conference On Computer Vision And Pattern Recognition Pages 11977–11986, 2019.
2. "Openpose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields " Z Cao, G Hidalgo, T Simon, S-E Wei, Y Sheikh, 2020.
3. "Pose Trainer: Correcting Exercise Posture Using Pose Estimation". By S.Chen, R.R. Yang Department Of CS., Stanford University, 2021.
4. "Personlab: Person Pose Estimation & Instance Segmentation With A Bottom-Up, Part-Based, Geometric Embedding Model" G.Papandreou, T.Zhu, L.-C.Chen, S.Gidaris, J.Tompson, K.Murphy ,2021.
5. P. E. Taylor, G. J. Almeida, J. K. Hodgins, And T. Kanade, "Multi-Label Classification For The Analysis Of Human Motion Quality," In Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc., 2020, Pp. 2214–2218.
6. Grandel Dsouza, Deepak, Mayura "Smart Gym Trainer Using Human Pose Estimation" In 2020 IEEE Conf Of (INOCON)
7. 7. Gourangi Taware , Rohit Agrawal , "AI-Based Workout Assistant And Fitness Guide" Published (First Online): 06-12-2021 Publisher: IJERT
8. Xavier Perez-Sala, Sergio Escalera, Cecilio Angulo and Jordi Gonz'alez, A survey on Model-based approaches for 2D and 3D Visual Human Pose Recovery, *pp. 1424-8220*.
9. Gyeongsik Moon, Ju Yong Chang and Kyoung Mu Lee, "V2v-posenet: Voxel-tovoxel prediction network for accurate 3d hand and human pose estimation from a single depth map", *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
10. Ho-Jun Park, Jang-Woon Baek, Jong-Hwan Kim, "Imagery based Parametric Classification of Correct and Incorrect Motion for Push- up Counter Using OpenPose", IEEE paper, 2020.