

**FitAI: Home Workout Posture Analysis using Computer Vision**

by

Hilman Zafri bin Mazlan

18000080

Dissertation submitted in partial fulfilment of  
the requirements for the  
Bachelor of Information Technology (Hons)  
(BIT)

JANUARY 2022

Supervised by  
Ts. Jale Ahmad

Universiti Teknologi PETRONAS  
32610 Seri Iskandar  
Perak Darul Ridzuan

**CERTIFICATION OF APPROVAL**

**FitAI: Home Workout Posture Analysis using Computer Vision**

by

Hilman Zafri bin Mazlan

18000080

A project dissertation submitted to the  
Information Technology Programme  
Universiti Teknologi PETRONAS  
in partial fulfilment of the requirement for the  
BACHELOR OF INFORMATION TECHNOLOGY (Hons)  
(BIT)

Approved by,



---

(Ts. Jale Ahmad)

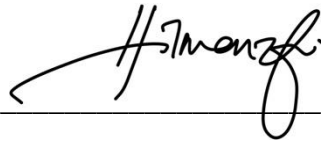
UNIVERSITI TEKNOLOGI PETRONAS

SERI ISKANDAR, PERAK

January 2022

## **CERTIFICATE OF ORIGINALITY**

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

A handwritten signature in black ink, appearing to read 'Hilman Zafri Bin Mazlan', written over a horizontal line.

**HILMAN ZAFRI BIN MAZLAN**

## **ABSTRACT**

Home-based workouts are known to have significant health and fitness benefits, but it could be dangerous if not performed with proper posture. This could be due to a lack of sufficient training and bad habits, or the inability to afford a personal trainer to fix the individual's posture. With the current Artificial Intelligence technology, we are now able to achieve workout analysis by using computer vision approach without any involvement of personal trainer. This report discussed the approach of keypoint detection, human pose estimation and pose analysis that will be utilized to produce the proposed project through Rapid Application Development (RAD) methodology, which will monitor these procedures phase by phase. Pretrained model from MediaPipe was chosen as the main human keypoint detection, and various machine learning algorithms such as Linear Regression and Random Forest were studied to find the suitable algorithm to be implemented in detecting and analyze home workout postures. Furthermore, this research also studies on geometrical approach to determining proper and improper posture for analysis. These approaches were then compared to see which will produce the best results and user experience. Since machine learning approach does not produce the best result due to lack of training data and variety of other factors, the geometrical approach was chosen as the final approach before implementing it into desktop application.

## ACKNOWLEDGEMENTS

*In the name of Allah, the Most Gracious and the Most Merciful.*

Alhamdulillah, all praise and thanks to Allah for the given strength and His blessing to me while working on this dissertation as part of Bachelor of Information Technology requirements.

First and foremost, I would like to thank my supervisor, Ts. Jale Ahmad, who, despite being extremely busy with his duties and work, but still willing to make an effort to guide and support me at the right time. His supervision, encouragement, and support throughout the project has allow me to move one step ahead than the others.

Special appreciation goes to Dr. Manzoor Ahmed Hashmani and Dr. Said Jadid Abdulkadir (FYP1 and FYP2 coordinator) for their tireless efforts in providing details and information about Final Year Project guidelines. Their contributions allow us final year students to stay alert on the submission deadlines and any other important information.

Finally, I would like to take this opportunity to extend my gratitude to both my family and friends for their given supports, both mentally and physically, throughout the completion of this project. They have given me the courage and strength to face any challenges to may arise during my final semester at Universiti Teknologi PETRONAS.

## ABBREVIATIONS AND NOMENCLATURES

HPE	Human Pose Estimation
CNN	Convolutional Neural Network
COCO	Common Objects in Context
LSP	Leeds Sports Poses
RAD	Rapid Application Development
GUI	Graphical User Interface
GPU	Graphics Processing Units
RGB	Red-Green-Blue
BGR	Blue-Green-Red
NaN	Not a Number

## TABLE OF CONTENTS

<b>CERTIFICATION OF APPROVAL .....</b>	<b>ii</b>
<b>CERTIFICATE OF ORIGINALITY.....</b>	<b>iii</b>
<b>ABSTRACT .....</b>	<b>iv</b>
<b>ACKNOWLEDGEMENTS.....</b>	<b>v</b>
<b>ABBREVIATIONS AND NOMENCLATURES .....</b>	<b>vi</b>
<b>CHAPTER 1: INTRODUCTION.....</b>	<b>1</b>
<b>1.1 Background of Study .....</b>	<b>1</b>
<b>1.2 Problem Statement.....</b>	<b>3</b>
<b>1.3 Objectives.....</b>	<b>3</b>
<b>1.4 Scope of Study .....</b>	<b>4</b>
<b>CHAPTER 2: LITERATURE REVIEW.....</b>	<b>5</b>
<b>CHAPTER 3: METHODOLOGY .....</b>	<b>11</b>
<b>3.1 Define Project Requirements .....</b>	<b>12</b>
<b>3.2 User Design .....</b>	<b>17</b>
<b>3.3 Graphical User Interface (GUI) Prototype.....</b>	<b>18</b>
<b>3.4 Testing phase .....</b>	<b>19</b>
<b>3.5 Implementation phase.....</b>	<b>19</b>
<b>3.6 Gantt Chart .....</b>	<b>20</b>
<b>CHAPTER 4: RESULTS AND DISCUSSION .....</b>	<b>21</b>
<b>4.1 Human Pose Estimation .....</b>	<b>21</b>
<b>4.2 Posture Analysis Methods .....</b>	<b>22</b>
<b>4.2.1 Machine Learning Approach.....</b>	<b>23</b>
<b>4.2.2 Geometrical Approach.....</b>	<b>34</b>
<b>4.3 Desktop Application.....</b>	<b>38</b>

<b>CHAPTER 5: CONCLUSION AND RECOMMENDATION .....</b>	<b>41</b>
<b>REFERENCES.....</b>	<b>43</b>

## LIST OF FIGURES

Figure 1 Alignment of two sample sequences using DTW algorithm.....	7
Figure 2 Rapid Application Development Process Flow .....	11
Figure 3 Keypoints location in COCO image dataset.....	13
Figure 4 Example of LSP Sports Dataset.....	13
Figure 5 Keypoints location on LSP Dataset .....	14
Figure 6 Apple Macbook Pro .....	14
Figure 7 Python programming language.....	15
Figure 8 OpenCV library .....	15
Figure 9 MediaPipe library .....	15
Figure 10 PyQt5 library .....	15
Figure 11 Jupyter Notebook.....	16
Figure 12 PyCharm IDE .....	16
Figure 13 Flowchart of project workflow .....	17
Figure 14 Main Page GUI Prototype .....	18
Figure 15 Feedback Page GUI Prototype .....	18
Figure 16 Project Gantt Chart .....	20
Figure 17 MediaPipe Pose Landmarks .....	22
Figure 18 Process flow of Posture Analysis .....	22
Figure 19 Proposed Machine Learning pipeline .....	23
Figure 20 Code for Human Joints Coordinate Extraction.....	24
Figure 21 Bicep Curl Proper Posture: Down .....	25
Figure 22 Bicep Curl Proper Posture: Up .....	25
Figure 23 Bicep Curl Improper Posture: Forward .....	25
Figure 24 Bicep Curl Improper Posture: Backward.....	25
Figure 25 Raw data .....	26
Figure 26 Cleaned data.....	26



Figure 27 Sample dataset .....	27
Figure 28 Confusion Matrix of Logistic Regression .....	29
Figure 29 Classification Report of Logistic Regression .....	29
Figure 30 Confusion Matrix of Ridge Regression .....	30
Figure 31 Classification Report of Ridge Regression.....	30
Figure 32 Confusion Matrix of Random Forest.....	31
Figure 33 Classification Report of Random Forest.....	31
Figure 34 Confusion Matrix of Gradient Boost .....	32
Figure 35 Classification Report of Gradient Boost.....	32
Figure 36 Correct Prediction using Ridge Regression.....	33
Figure 37 Incorrect Prediction using Ridge Regression .....	34
Figure 38 Process flow of geometrical approach.....	35
Figure 39 Angle of three joints .....	35
Figure 40 Geometrical calculation formula .....	35
Figure 41 Example of angle in degrees of three joints.....	36
Figure 42 Angle of three joints: Bicep forward .....	37
Figure 43 Angle of three joints: Bicep backward .....	37
Figure 44 Main Page of FitAI Desktop Application.....	40
Figure 45 Feedback Page of FitAI Desktop Application.....	40

## LIST OF TABLES

Table 1 Breakdown of human pose estimation and pose comparison method .....	7
Table 2 Comparison between results of different Machine Learning Algorithms.....	32

## **CHAPTER 1: INTRODUCTION**

### **1.1 Background of Study**

Since the national movement control order imposed by the government of Malaysia, people have been looking for ways to exercise without having to go outside. Therefore, doing home workouts has become a common choice among Malaysians. In addition, movement control order had caused closure of business activities, public places, and daily workout activities at the gym, resulting in various serious psychological issues and major health concerns (Kaur et al., 2020). There are several benefits of doing workout such as reducing the risk of cardiovascular diseases and metabolic syndrome, controlling body weight, strengthen bones and muscle, and improving mental health (Gulam, 2016). However, doing workout alone without having someone to evaluate their posture can be dangerous if done incorrectly. This is due to the fact that exercising regularly without maintaining proper posture can result in severe injuries to the muscles or ligaments. Imbo (2018) stated that standing incorrectly can cause the bones and muscle to line up incorrectly and obstructing the flow of the sciatic nerve. Such blockage can result in sciatica, a condition that causes pain in the back of the thigh, calves, and feet. This leads to the necessity of having a trainer to supervise the exercise session and correct the individual's posture. Since not everyone has access to personal trainer for various reason, such as unable to go outside due to lockdown or unable to afford a personal trainer, an artificial intelligence-based application could be used to identify the workout poses and provide personalized feedback to assist in improving their exercise posture.

This project aims to develop software that focuses on assisting people in properly performing workout such as bicep curl and sit-up using Artificial Intelligence with Computer Vision technique. The goal of this project is to help prevent injuries and improve the quality of individual's workout by providing personalised feedback on their posture using only a computer and a webcam. Artificial Intelligence has always been discussed nowadays due to its ability to assist human with their daily activities. The capabilities of Artificial Intelligence giving humans hope to develop machines with human intelligence. One of the common Artificial Intelligence

technologies is called Computer Vision, where machines can recognise photographs and videos in the same way that humans do.

Computer vision is a branch of Artificial Intelligence that aims to teach computers to see, identify, and interpret images in the same way that humans do, and then produce appropriate outputs. In other words, it combines a computer with human intelligence and sensibilities. Image classification, image localization, object detection, segmentation, and keypoint detection are some of the tasks included in computer vision.

In this project, the author will focus on Human Pose Estimation (HPE) which is also part of computer vision task that determines the pose of a human in an image or video. HPE can also be characterised as the challenges in determining a camera's position and orientation in reference to a specific person. This is performed by detecting, locating, and tracking the number of keypoints on a given person's image. HPE is classified into two categories which are 2-dimensional and 3-dimensional pose estimation. 2-dimensional pose estimation simply predicts the location of a specific keypoint in a 2-dimensional space relative to the image, whereas 3-dimensional pose estimation converts a 2-dimensional object to a 3-dimensional object by adding a z-dimension to the prediction. There are numerous approaches for achieving human pose estimation; however, Convolutional Neural Network (CNN) will be the most suitable deep learning architecture in achieving this due to its specialty in image analysis.

Because deep learning is known to be capable of performing supervised and unsupervised data on an image, its use would make this research more efficient and effective. This research will use the CNN-based deep learning algorithm to estimate human poses and various machine learning models for posture analysis in order to provide users with personalised feedback and correct their posture.

## **1.2 Problem Statement**

Even though workout exercises have significant health and fitness benefits, but it could be dangerous if not performed with proper posture. A common observation shows that most people find it very challenging to perform any home workout effectively while maintaining a proper posture. For example, an observation by Duncan (2019) throughout the session at MOD Fitness still shows that many students make mistakes when executing workouts such as plank, lunge, and push ups. Doing exercise incorrectly on a regular basis may result in significant long-term injuries. It is reported that prolonged poor posture may result in body discomfort and had a detrimental impact on the musculoskeletal system, caused localized muscular fatigue, and may compromise physical function and level of abilities (Ahmad and Kim, 2018). This could be due to a lack of sufficient training and bad habits, or the inability to afford a personal trainer to fix the individual's posture. Additionally, the lack of computer vision-based technology in correcting an individual's workout posture could also be the problem. There are various workout applications available, such as Nike Training Club and Freeletics, but none of them use computer vision technology to monitor and analyze user's workout posture.

## **1.3 Objectives**

The goal of this project is to develop an Artificial Intelligence-based system that assists people in performing home workouts such as bicep curl and sit-up by combining computer vision technology with human pose estimation technique, along with deep learning and machine learning approaches. Following are the objectives of this project in order to achieve the mentioned goals:

- To investigate how computer vision can assist in detecting human exercise posture and incorrect posture.
- To develop an artificial intelligence-based software that uses camera to detect user's workout posture and provides personalized feedback on improving their exercise posture.

- To help preventing injuries and breaking the bad habits of exercising with incorrect posture.
- To evaluate the effectiveness of implementing this software in individual's health.

## **1.4 Scope of Study**

This section will explain the project research study and specify the criteria for home workout posture analysis. The study and researched area are divided into three sections: the system's target user, the required tools and technologies, datasets, and platform.

### **1. Target Users**

Target users for this home workout posture analysis using computer vision will be UTP students that frequently perform workouts such as bicep curl and sit-up without a personal trainer. Even though everyone in the world able to utilise this system, but due to time constraints and the capabilities of a single novice developer, it is not yet suitable to release this system globally.

### **2. Tools and Technologies**

This project will be using CNN-based deep learning algorithm, various machine learning algorithm such as Linear Regression and Random Forest, and MediaPipe library from Google to achieve human pose estimation. This system will be built using author's personal laptop, MacBook Pro with the specification of M1 Chip and 8 Gigabytes of RAM. It will also implement the laptop's internal webcam to capture real-time video.

### **3. Datasets**

The datasets used for this project will be from open-source collection that is publicly available such as COCO and LSP datasets. The deep learning model will be trained with thousands of images of different people to ensure that it can deliver the desired result with high accuracy of workout pose estimation.

## CHAPTER 2: LITERATURE REVIEW

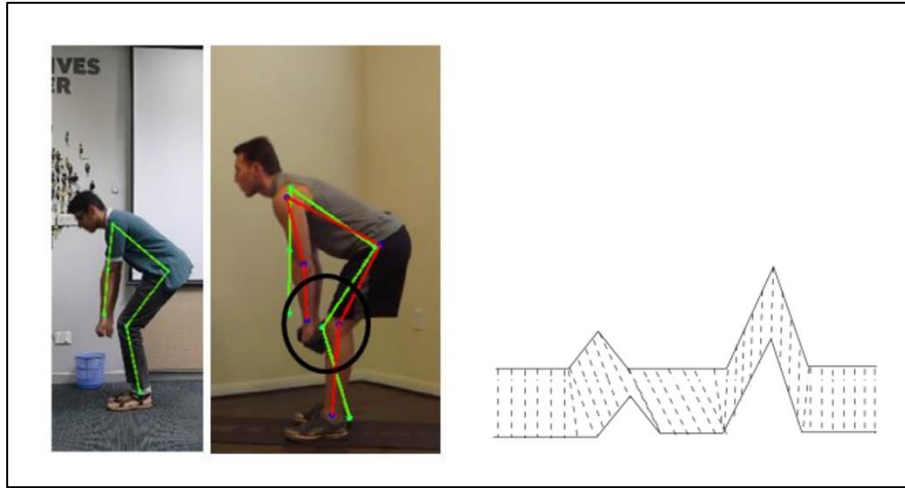
In this project research, the author decided to focus his attention towards two particular task of computer vision, which are achieving human pose estimation and posture comparison in terms of video frames. The following publications and research paper were reviewed to study their development method that can help to achieve this project's goal.

Human pose estimation has been studied extensively as a part of computer vision and used in varieties of application, such as workout posture analysis. However, achieving human pose estimation is quite a challenging problem in the computer vision field. Automatically detecting a person's pose in an image is a tough issue since it is dependent on several factors such as image scale and resolution, change of lighting, cluttered background, clothing variations, surroundings, and human interaction with the surroundings (Sigal, 2020). However, Yamakawa et al. (2020) from Waseda University in Tokyo, Japan proposed a method for improving the accuracy of human pose estimation in videos. Technically, predicted human pose is a set of time series data; so, human pose estimation can be accomplished more accurately by applying time series correlation. To improve detection accuracy, they integrated CNN-based model with multiple objects tracking framework. This means that undetected or incorrectly detected body joints will be interpolated using information from previous and following frames.

There are in fact many other ways to achieve human pose estimation in video frames. For pose detection, Steven and Richard use a pretrained model from the opensource library called OpenPose. OpenPose uses vectors that encode the position and orientation of limbs as a new way of pose estimation. The model is made up of multiple stage CNN with two branches which are learning the confidence mapping of a keypoints on an image and learning the part of affinity fields (Cao et. al, 2017). Mahendran (2021) proposed an approach that makes use of another opensource library known as PoseNet. PoseNet is another real-time pose detection approach that works on both single and multiple human detection. It was trained on Google's MobileNet Architecture, which is similarly a CNN-based architecture used mostly for image analysis. MobileNet architecture was also recognised to be light weighted than any

other architecture since it uses depth wise separable convolution to deepen the network while reducing parameter, computation cost, and increasing accuracy (Agrawal, 2021). Whereas Jin et. al (2015) proposed another method for estimating human posture which is by using Microsoft Kinect sensor. Microsoft Kinect has their own usage of machine learning or deep learning algorithm that captures from the Infrared emitter. However, according to Jin et al. (2015), there is one limitation when using the tool, which is that the user must stay between 1.8 to 2.5 metres of the Kinect sensor in order for it to work effectively.

The achievement of human pose estimation alone will not meet the project's goal since the system must be able to provide feedback to the end-user. For instance, Chen et al. (2020) established the angle between the upper arm vector and the torso vector in their study on bicep curl workout analysis. They measure the minimum angle achieved between the upper arm and forearm by implementing geometry evaluation, and if the angle exceeds 35 degrees, it shows that the user rotates the arm excessively, while less than 70 degrees indicates that the user is not curling the weight all the way up. This method was also implemented by Chen et. al (2019), where they calculated the angle of each joint while performing a plank exercise and determining which angle was appropriate when performing the exercise. They further claim that in order to obtain a reliable angle from the proper perspective, two and three dimensions of these angles must be identified, which can be accomplished using Least Square approximation approach. Nagarkoti et. al (2019) takes advantage of time series analysis using Dynamic Time Warping (DTW) algorithm in order to compare the posture of two people. DTW requires the definition of a similarity measure between any two points/items in a pair of sequences and produces a mapping of closest matching point or item pairs (Zhang, 2020). The similarity of two matching points indicates that the user is completing the exercise with proper posture, whilst the difference indicates that the user's posture is poor.



*Figure 1 Alignment of two sample sequences using DTW algorithm*

According to these literatures, it has been found that there are many ways and techniques to achieve human pose estimation and pose comparison between two people. For human pose estimation, CNN-based architecture is the common framework to be, as CNN algorithm specialize for image and video recognition. CNN works by extracting features from the images through input layer, output layer, and hidden layer. Even though Kinect has been used in human pose estimation, it is important to be noted that the device was released back in 2010 where very little information about the quality of the data were obtained. Under a paper of Khoshelham (2012), he discovered that the random error of depth measurement grows with increasing distance to the Kinect sensor, ranging from a few millimetres to nearly four centimetres at the sensor's maximum range. He also discovered that the device's depth resolution diminishes quadratically with increasing distance from the sensor.

The table below shows a tabular breakdown of the method used to achieve human pose estimate and pose comparison.

*Table 1 Breakdown of human pose estimation and pose comparison method*

<b>Author</b>	<b>Title</b>	<b>Method for Human Pose Estimation</b>	<b>Method for Pose Comparison</b>
<b>Kothari, S. (2020)</b>	Yoga Pose Classification Using Deep Learning	Deep learning method: Multilayer Perceptron, Long Short-Term Memory,	



		Convolutional Neural Network and machine learning method: Support Vector Machine	
<b>Ribera, S., M. (2020)</b>	Computer Vision Analysis of the body-pose similarity from two different subjects with the aim of the correct development of physical exercises.	PyTorch model that uses Convolutional Neural Network approach	Percentage of Detected Joint
<b>Chen, S. &amp; Yang, R., R. (2020)</b>	Pose Trainer: Correcting Exercise Posture using Pose Estimation	OpenPose model that uses Convolutional Neural Network approach	Angle differences using geometry evaluation, and machine learning evaluation: Dynamic Time Warping
<b>Chen, Y., Chen, Y., Tu, Z. (2019)</b>	Fitness Done Right: A Real-Time Intelligent Personal Trainer For Exercise Correction	Two-branch Convolutional Neural Network	Vector distance, weighted Euclidean distance, and weighted angle distance
<b>Nagarkoti, A., Teotia, R., Mahale, A., K., &amp; Das, P., K. (2019)</b>	Realtime Indoor Workout Analysis Using Machine Learning & Computer Vision	Two-branch Convolutional Neural Network	Dynamic Time Warping
<b>Nishani, E., &amp; Cico, B. (2017)</b>	Computer Vision Approaches based on Deep Learning and Neural Networks: Deep Neural Networks for Video Analysis of	CoordinateNet and HeatmapNet that uses Convolutional Neural Network approach	

	Human Pose Estimation		
<b>Dsouza, G., Maurya, D., &amp; Patel, A. (2020)</b>	Smart gym trainer using Human pose estimation	Convolutional Neural Network	
<b>Jin, X., Yao, Y., Jiang, Q., Huang, X., Zhang, J., Zhang, X., &amp; Zhang, K. (2015)</b>	Virtual Personal Trainer via the Kinect Sensor	Microsoft Kinect Sensor	Action Classification, Dynamic Space-time Warping, and Fitness Score
<b>Anilkumar, A., Athulya, K., T., Sajan, S., &amp; Sreeja K., A. (2021)</b>	Pose Estimated Yoga Monitoring System	MediaPipe model that uses Convolutional Neural Network approach	Angle differences of each joint
<b>Yang, L., Li, Y., Zeng, D., &amp; Wang, D. (2021)</b>	Human Exercise Posture Analysis based on Pose Estimation	OpenPose model that uses Convolutional Neural Network approach	Angle differences of each joint
<b>Sonwani, N., &amp; Pegwar, A. (2020)</b>	Auto_Fit: Workout Tracking using Pose Estimation and DNN	PoseNet model that use Deep Convolutional Neural Network approach and Deep Neural Network	Binary-Cross Entropy
<b>Alatiah, T., &amp; Chen, C. (2020)</b>	Recognizing Exercises and Counting Repetitions in Real Time	OpenPose model that uses Convolutional Neural Network approach	
<b>Mahendran, N. (2021)</b>	Deep Learning for Fitness	PoseNet model that use Convolutional Neural Network approach and Deep Neural Network	Difference between slope in body parts
<b>Yadav, S., K., Singh, A., Gupta, A., &amp;</b>	Real-time Yoga recognition using deep learning	Convolutional Neural Network and Long Short-Term Memory	

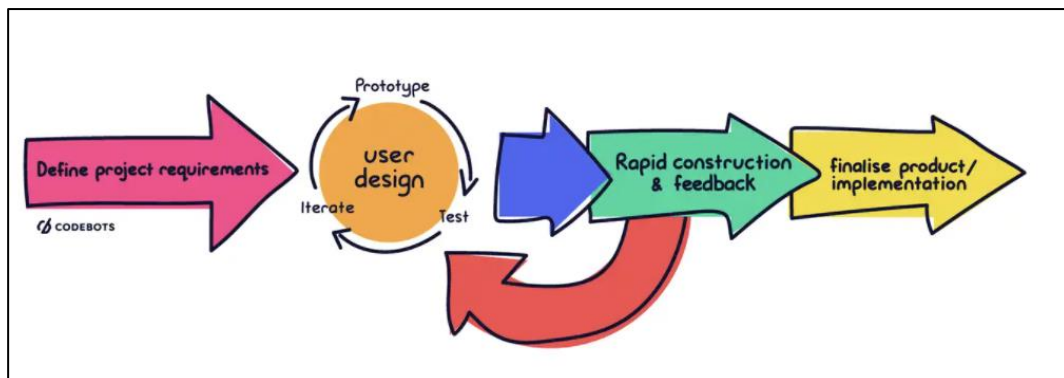
<b>Raheja, J., L. (2019)</b>			
<b>Bhambure, S., Lawande, S., Upasani, R., &amp; Kundargi, J. (2021)</b>	Yog-Assist	OpenPose model that uses Convolutional Neural Network approach	

## CHAPTER 3: METHODOLOGY

Rapid Application Development (RAD) method is chosen in this project research to achieve the proposed aims and goals in developing FitAI: Home Workout Posture Analysis using Computer Vision. This methodology focuses on rapid application development through multiple iterations and continuous feedback from the end-user. This is important because it breaks up into several phases and involve collaboration with the end-user, allowing this project to be enhanced at every stage.

The RAD cycle consists of four stages which can be defined as follows:

1. Define project requirements
2. Prototype
3. Rapid construction and gather feedback
4. Finalise product and implementation



*Figure 2 Rapid Application Development Process Flow*

Some other key benefits of applying RAD cycle in this project are:

- Developer have greater flexibility and adaptability, allowing them to make rapid adjustments during the development process.
- Early system integration and frequent iterations of feedback accelerates project execution.
- Better risk management since testing occurs during each iteration, allowing end-users to quickly identify and discuss errors.

### **3.1 Define Project Requirements**

The project requirements are mentioned in the objectives section earlier, which is to investigate on how computer vision can assist in detecting human exercise posture and incorrect posture. To achieve this, the author must first achieve human pose estimation by using deep learning with CNN-based architecture as it is the common framework that specialized in image processing. The system should then be able to provide personalized feedback on their workout posture by detecting improper posture using various approaches. Machine learning and geometrical approaches are studied to find which approach is capable of achieving the author's main goal. The selected approaches will be then implemented into a desktop application following the best practices of designing user interface for better user experience.

This methodology was also implemented by Grandel et. al (2020), where they separate the process of achieving human pose estimation and pose comparison on different stages. This is because both of this stage requires different datasets to be trained with deep learning or machine learning model, as creating the algorithm together would possibly reduce the amount of model accuracy, hence delaying project's implementation phase.

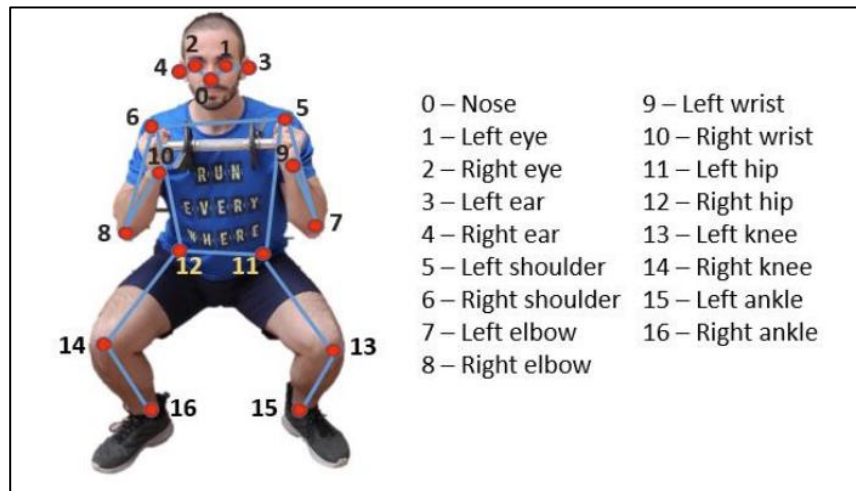
The requirements of these two stages will be broken down into datasets, hardware tools, programming language, and other associated software used for this project. The following are the required items for each category:

#### **1. Dataset for Human Pose Estimation**

##### **a. Common Objects in Context (COCO) Dataset**

COCO is a large-scale open source dataset with over 330 thousands of images in which more than 200 thousands images are labelled. This dataset contains tagged images with features such as Object Classification, Object Segmentation and Keypoint Detection.

Since this project will be focusing on keypoint detection of human joints, COCO uses a total of 17 keypoints as shown in figure 3. These keypoints includes the detection nose, eye, shoulder, elbow, wrist, and so on.



*Figure 3 Keypoints location in COCO image dataset*

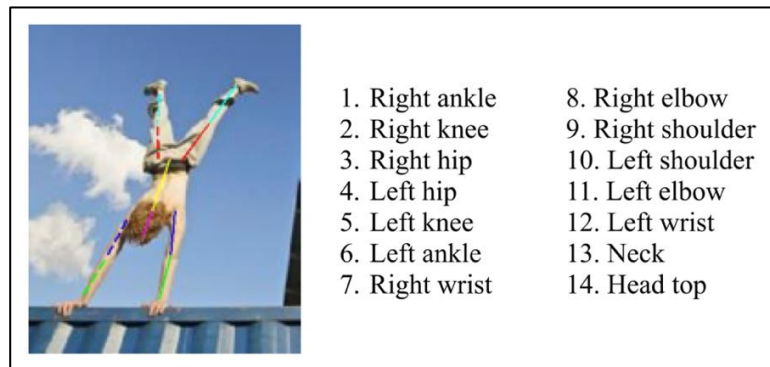
#### **b. Leeds Sports Pose (LSP) Dataset**

The COCO dataset may contain all of the common objects around us, but too many objects can cause our model to become overwhelmed and misrepresent the posture it is intended to predict. As a result, LSP may be beneficial for this project since it has 2000 images of athletes gathered from the photography website, Flickr. Figure 4 illustrates some of the sports represented in the dataset.



*Figure 4 Example of LSP Sports Dataset*

This dataset only has 14 keypoint locations, which is less than the COCO Dataset. Figure 5 illustrates the ordering of the keypoints:



*Figure 5 Keypoints location on LSP Dataset*

## 2. Dataset for Pose Comparison

### a. Proper posture and improper posture

This dataset can be obtained after the author has finally achieve building human pose estimation using CNN-based architecture. The expected dataset will be in a form of x, y, and z coordinates of human joints in an image.

## 3. Hardware tools

- a. MacBook Pro. Specification: M1 Chipset, 8GB RAM, 720p internal camera.



*Figure 6 Apple MacBook Pro*

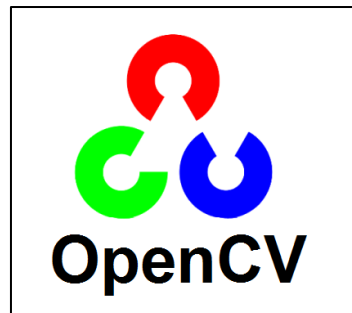
#### 4. Programming Language & Libraries

a. Python



*Figure 7 Python programming language*

b. OpenCV



*Figure 8 OpenCV library*

c. MediaPipe



*Figure 9 MediaPipe library*

d. PyQt5

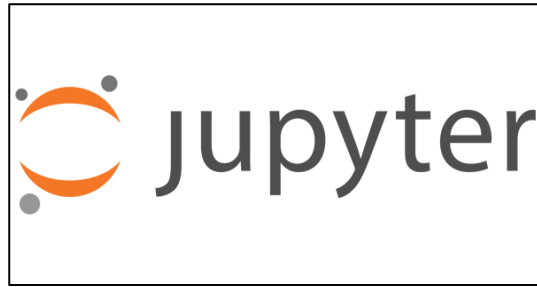


*Figure 10 PyQt5 library*



## 5. Software

### a. Jupyter Notebook



*Figure 11 Jupyter Notebook*

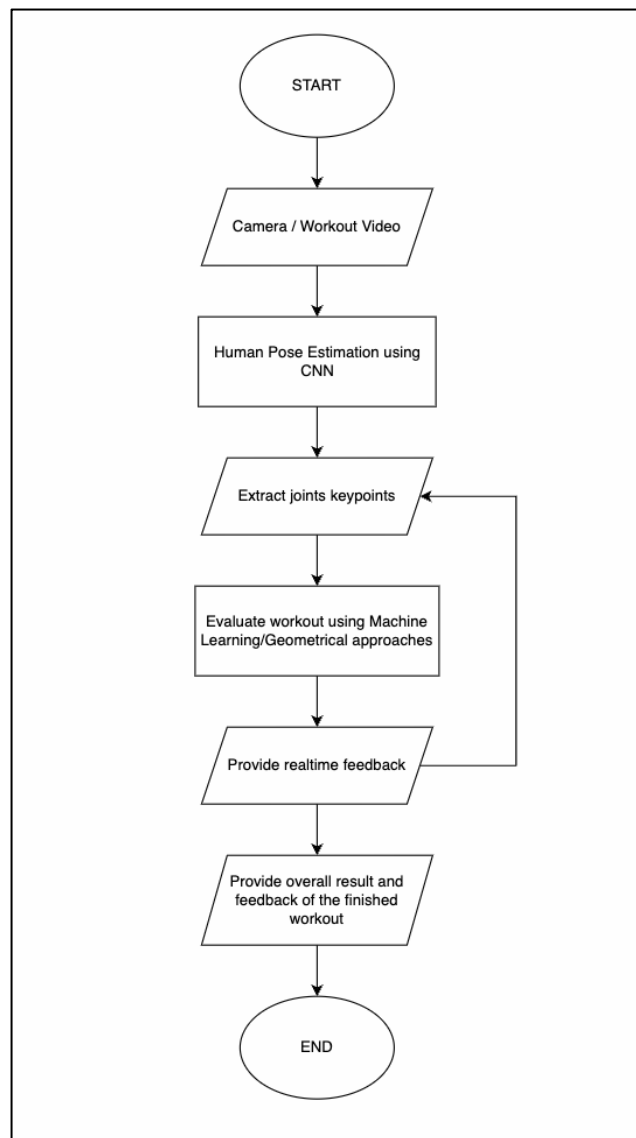
### b. PyCharm



*Figure 12 PyCharm IDE*

### 3.2 User Design

The following phase will be the process of designing the flow of this project. The goal is to rapidly create a workable design that can be demonstrated to the end-user. Rapid design encourages user interaction, testing, and frequent feedback on the design. Doing flowchart can be useful in visualizing the project workflow. The next step is to prototype the Graphical User Interface (GUI) of the system.



*Figure 13 Flowchart of project workflow*

### 3.3 Graphical User Interface (GUI) Prototype

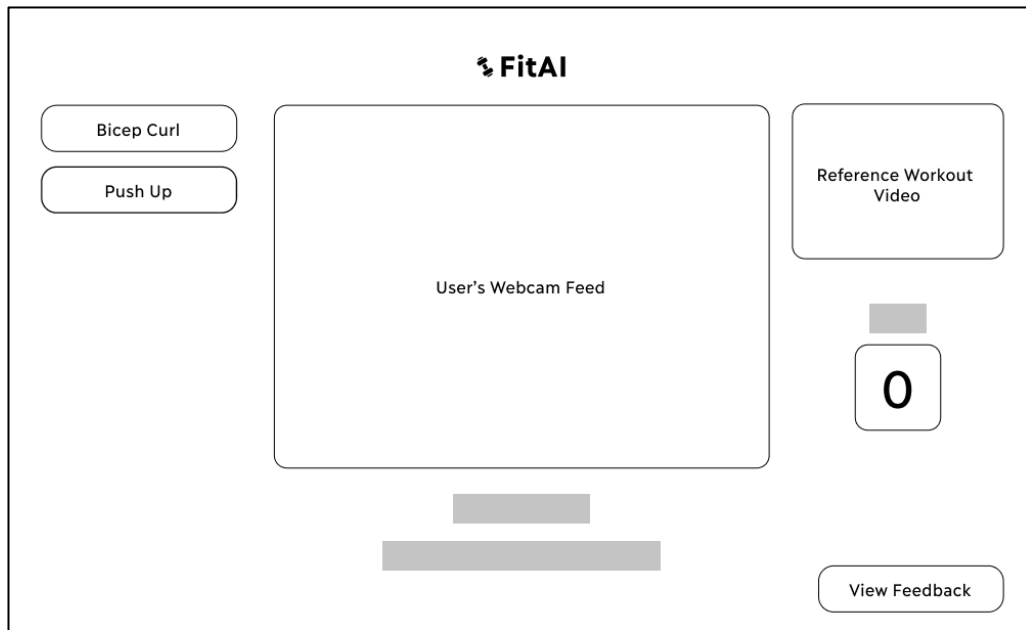


Figure 14 Main Page GUI Prototype

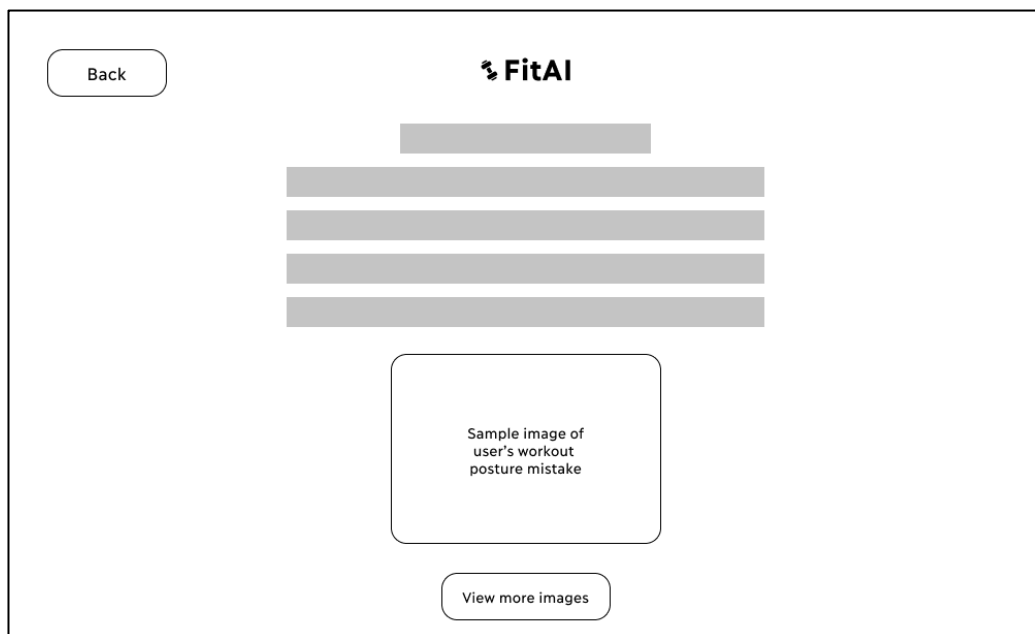


Figure 15 Feedback Page GUI Prototype

### **3.4 Testing phase**

The testing phase will determine whether the implemented model, which to achieve human pose estimation and pose comparison can be performed smoothly. Errors and bugs are also to be expected during this phase, given it is only a testing process and not yet an implementation. As a result, these errors should be documented so that we can analyse and fix them before moving on to the implementation phase.

### **3.5 Implementation phase**

This project is ready to be implemented after the testing phase is completed using the previously mentioned software and tools. To ensure a seamless development process, the system must adhere to the previously stated RAD technique and be built with the design created during the prototyping phase. The system's target user will then be able to interact with the system by start recording or uploading their workout videos to analyse posture and expect personalised feedback to fix their posture.

### 3.6 Gantt Chart

Figure 16 shows the Gantt chart of all the processes involved in FitAI: Home Workout Posture Analysis using Computer Vision project. This project is currently in Phase 2, and the next phase will begin in the following semester, January 2022.

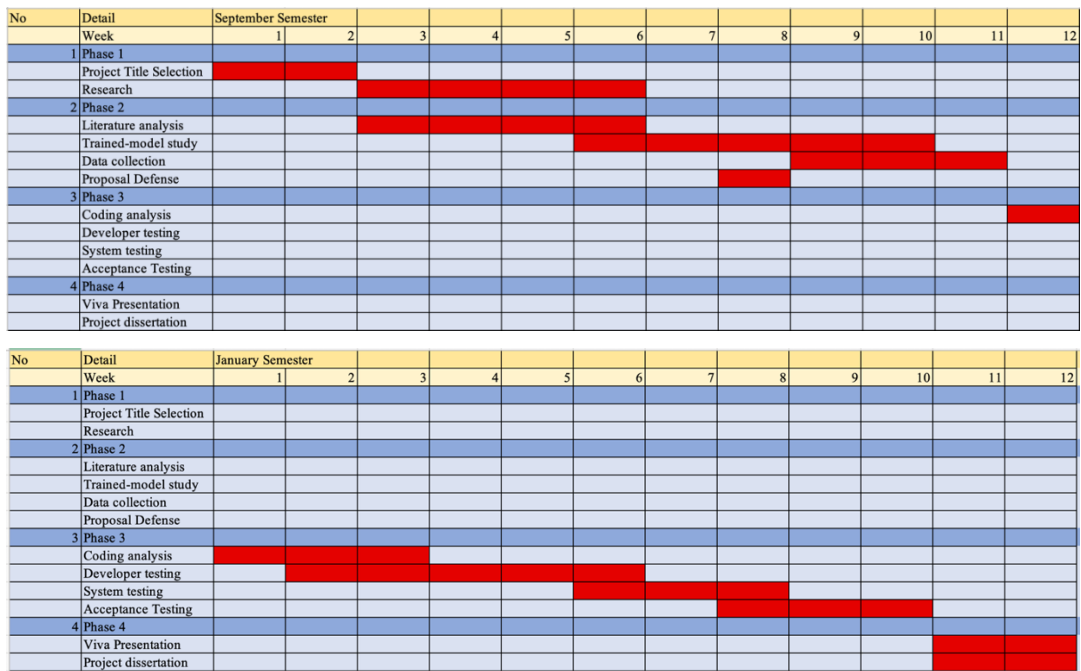


Figure 16 Project Gantt Chart

## CHAPTER 4: RESULTS AND DISCUSSION

### 4.1 Human Pose Estimation

According to the literature review in previous chapter, the common method to estimate human posture is by using convolutional neural network (CNN) for labelling RGB images. CNN is well-known for its excellent approach to image classification and object identification problems. However, implementing CNN from scratch would be challenging for the author because it is critical to include at least tens of thousands of images of people with various illustrated body joints. Furthermore, training on such a large volume of data necessitates a significant amount of computational power. The most resource-intensive task for any neural network is the training phase of a deep learning model. Therefore, much better Graphics Processing Units (GPU) are required, as powerful GPUs are optimised for training artificial intelligence and deep learning algorithms due to their ability to process multiple computations at once.

As an alternative, the author's initial plan of implementing CNN will have to be changed to implementing any open-source pre-trained model available. After experimenting with several state-of-the-art pose estimators, MediaPipe appears to be the most user-friendly and straightforward to implement until production. The other reason why MediaPipe is chosen because of its performance. MediaPipe's speed is achieved thanks to the use of GPU acceleration and multi-threading.

In order to use MediaPipe, it is important to obtain data from user's camera. Therefore, OpenCV is used to feed the data for MediaPipe pretrained model for computer vision tasks. OpenCV is a python library that is widely used for image analysis, image processing, image detection and recognition, and so on. There will be some minor tweaks to make the solutions work, as MediaPipe only allows Red-Green-Blue (RGB) images to detect human keypoints accurately. Meanwhile, OpenCV's default colour input is Blue-Green-Red (BGR). Therefore, it is important to convert BGR data into RGB data to allow MediaPipe's machine learning model to be able to detect human keypoints accurately. In order to obtain human keypoint data, MediaPipe converts it into x, y, and z coordinates, which are

normalised to [0.0, 1.0] by the image width and height. These data can be very useful for posture analysis as it allows to calculate the difference of angles of joints between proper postures and improper postures and then analyse it. It can also be useful for posture classification which will be explain further in the next chapter. Figure 17 shows the current topology of MediaPipe’s library, which present 33 human body keypoints.

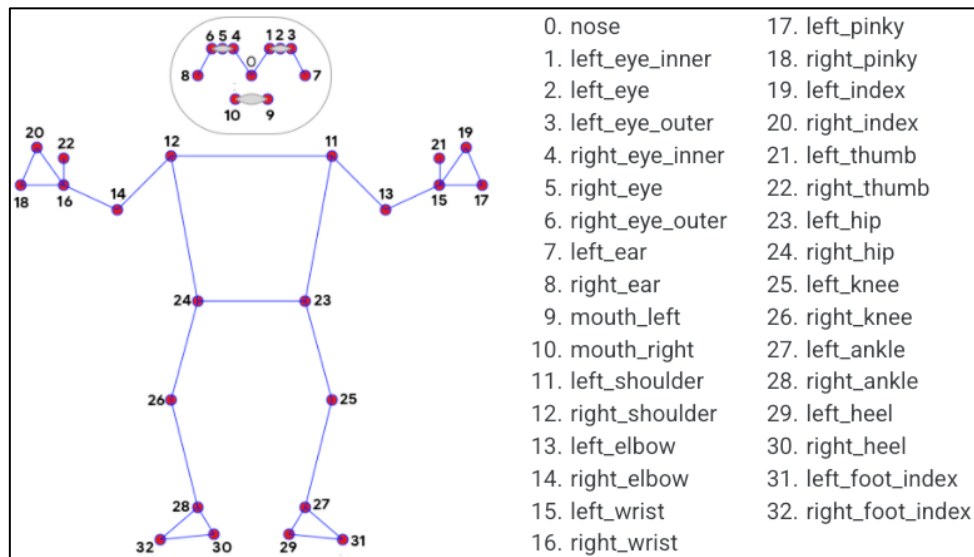


Figure 17 MediaPipe Pose Landmarks

## 4.2 Posture Analysis Methods

Home workout posture analysis can be accomplished using two methods: Machine Learning Algorithm and Geometrical Approach. The author will use these two approaches and compare which one produces the best results and user experience.

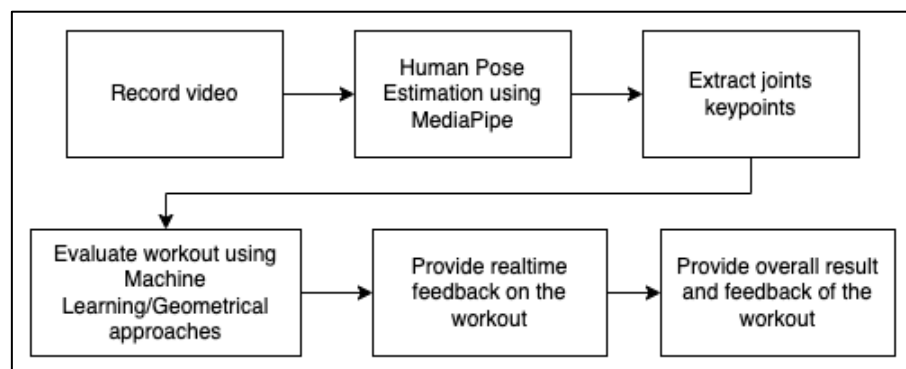
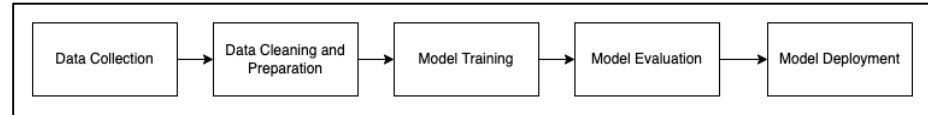


Figure 18 Process flow of Posture Analysis

### 4.2.1 Machine Learning Approach

The machine learning pipeline process will be as figure below:



*Figure 19 Proposed Machine Learning pipeline*

#### Data Collection

To build the machine learning model, the author must collect data on workout postures in the form of images or videos. However, due to the increasing case of Covid-19, scheduling an appointment with a certified workout trainer and obtain data from them is challenging. As an alternative, proper bicep curl and sit-up workout postures were studied and identified through online platforms such as Fitness blogs and YouTube. The author then invites students who currently stays in UTP campus to contribute to the data collection of workout postures. As of early March 2022, three people had agreed to participate in these workouts and had agreed to be recorded for the data collection purposes.

The workout is recorded in MOV format with a total file size of 1GB and consists of two different workouts, bicep curl and sit-up, with proper and improper posture categories. Due to time constraints, these individuals were only required to perform 20 repetitions of each workout, which included 10 proper postures and 10 incorrect postures.

These videos must be fed into a computer vision program that is designed to extract all joints coordinates and store it in CSV file format. Figure 20 shows the full code of the program for human joints extraction in a form of x, y, and z coordinates.



```

import mediapipe as mp
import cv2
import numpy as np

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose

import csv
import os

num_coords = 33

landmarks = ['class']
for val in range(1, num_coords+1):
    landmarks += ['x{}'.format(val), 'y{}'.format(val), 'z{}'.format(val), 'v{}'.format(val)]

with open('coords.csv', mode='w', newline='') as f:
    csv_writer = csv.writer(f, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)
    csv_writer.writerow(landmarks)

```

```

class_name = ""
cap = cv2.VideoCapture("Video/Hilman0.mov")
# 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 from BGR to RGB. Mediapipe accepts RGB. OpenCV uses BGR format.
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        image.flags.writeable = False

        # Make detection with mediapipe model
        results = pose.process(image)

        # Recolor back from RGB to BGR. OpenCV only accepts BGR format.
        image.flags.writeable = True
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

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

            pose_row = list(np.array([[landmark.x, landmark.y, landmark.z, landmark.visibility] for landmark in landmarks]))
            row = pose_row
            row.insert(0, class_name)

            with open('coords.csv', mode='a', newline='') as f:
                csv_writer = csv.writer(f, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)
                csv_writer.writerow(row)

            # Get coordinates
            shoulder = (landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].x, landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].y, landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].z)
            elbow = (landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].x, landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].y, landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].z)
            wrist = (landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].x, landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].y, landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value].z)

            shoulder_coordinate = tuple(np.multiply(shoulder, [1080, 720]).astype(int))
            elbow_coordinate = tuple(np.multiply(elbow, [1080, 720]).astype(int))
            wrist_coordinate = tuple(np.multiply(wrist, [1080, 720]).astype(int))

            cv2.line(image, shoulder_coordinate, elbow_coordinate, (255, 255, 255), 3)
            cv2.line(image, wrist_coordinate, elbow_coordinate, (255, 255, 255), 3)
            cv2.circle(image, shoulder_coordinate, 20, (17, 217, 34), cv2.FILLED)
            cv2.circle(image, elbow_coordinate, 20, (17, 217, 34), cv2.FILLED)
            cv2.circle(image, wrist_coordinate, 20, (17, 217, 34), cv2.FILLED)
            cv2.circle(image, shoulder_coordinate, 15, (0, 0, 0), 3)
            cv2.circle(image, elbow_coordinate, 15, (0, 0, 0), 3)
            cv2.circle(image, wrist_coordinate, 15, (0, 0, 0), 3)
        except:
            pass

        cv2.imshow('Mediapipe Feed', image)

        k = cv2.waitKey(1)
        if k == 120:
            class_name = ""
        if k == 119:
            class_name = "Bicep Down"
        if k == 101:
            class_name = "Bicep Up"
        if k == 115:
            class_name = "Bicep Forward"
        if k == 100:
            class_name = "Bicep Backward"
        if cv2.waitKey(10) & 0xFF == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

```

*Figure 20 Code for Human Joints Coordinate Extraction*

While extracting data, the programme will also draw the joints in the shoulder, elbow, and wrist for better visualisation and observation.



*Figure 21 Bicep Curl Proper Posture: Down*



*Figure 22 Bicep Curl Proper Posture: Up*



*Figure 23 Bicep Curl Improper Posture: Forward*



*Figure 24 Bicep Curl Improper Posture: Backward*

## **Data Cleaning and Preparation**

Handling missing value is an important part of data cleaning and preparation. The extracted data may be filled with Not a Number (NaN) values, but these values were purposefully left as empty because they do not provide any useful information, such as incorrectly detected joints, unrelated postures, and so on. Therefore, these values must be dropped immediately before being fitted into various machine learning algorithms for future evaluation. Instead of manually removing each of those NaN values, it can be easily done with a popular data analysis tool called Pandas using “dropna” function.

	class	x1	y1	z1	v1	x2	y2	z2	v2	x3	...
0	NaN	0.646326	0.093126	-0.762514	0.999541	0.654877	0.047631	-0.733119	0.998520	0.660929	...
1	NaN	0.609334	0.062254	-0.355626	0.999570	0.609822	0.020864	-0.328198	0.998626	0.612339	...
2	NaN	0.595575	0.056722	-0.370127	0.999595	0.593889	0.016085	-0.345562	0.998718	0.595536	...
3	NaN	0.583862	0.059034	-0.326914	0.999631	0.580601	0.018613	-0.301308	0.998834	0.582206	...
4	NaN	0.581075	0.060247	-0.259024	0.999653	0.574972	0.020182	-0.237870	0.998917	0.575595	...
5	NaN	0.577045	0.061682	-0.314242	0.999664	0.573316	0.020838	-0.287918	0.998971	0.574794	...
6	NaN	0.572670	0.062593	-0.277067	0.999680	0.569888	0.022911	-0.254060	0.999032	0.571069	...
7	NaN	0.570170	0.061406	-0.217896	0.999688	0.566917	0.023178	-0.198968	0.999079	0.567933	...
8	NaN	0.565883	0.062104	-0.254606	0.999685	0.563865	0.023395	-0.233205	0.999102	0.565238	...
9	Bicep Down	0.566319	0.062481	-0.258781	0.999664	0.563809	0.023923	-0.235655	0.999085	0.565149	...
10	Bicep Down	0.566792	0.062759	-0.221019	0.999652	0.564401	0.024213	-0.202950	0.999085	0.565583	...
11	Bicep Down	0.563492	0.063009	-0.246001	0.999668	0.561225	0.024378	-0.226396	0.999137	0.562614	...
12	Bicep Down	0.561856	0.063079	-0.240953	0.999676	0.560207	0.024468	-0.220182	0.999172	0.561714	...
13	Bicep Down	0.560788	0.063122	-0.255947	0.999684	0.558903	0.024776	-0.234829	0.999203	0.560447	...
14	Bicep Down	0.559144	0.063351	-0.265532	0.999688	0.557240	0.025461	-0.244541	0.999223	0.558941	...
15	Bicep Down	0.558697	0.063596	-0.277275	0.999697	0.556808	0.025786	-0.253482	0.999253	0.558605	...
16	Bicep Down	0.558336	0.063757	-0.280720	0.999702	0.556178	0.026012	-0.257171	0.999273	0.557905	...
17	Bicep Down	0.557933	0.064719	-0.276733	0.999703	0.556016	0.026884	-0.252549	0.999284	0.557639	...
18	Bicep Down	0.557241	0.064691	-0.273022	0.999715	0.555384	0.027020	-0.249706	0.999318	0.557025	...
19	Bicep Down	0.556971	0.064758	-0.274448	0.999722	0.555362	0.027161	-0.251482	0.999346	0.557031	...
20	Bicep Down	0.555633	0.064649	-0.273878	0.999726	0.553954	0.027267	-0.251989	0.999365	0.555741	...
21	Bicep Down	0.555873	0.064788	-0.266320	0.999745	0.554352	0.028103	-0.245373	0.999414	0.556115	...
22	NaN	0.636970	0.038683	-0.395010	0.999623	0.634544	-0.014051	-0.369608	0.999552	0.637680	...
23	NaN	0.614580	0.040469	-0.271070	0.999621	0.613096	-0.005508	-0.243565	0.999537	0.614812	...
24	NaN	0.609268	0.051075	-0.187621	0.999578	0.606106	0.006494	-0.170573	0.999435	0.606628	...
25	NaN	0.604341	0.056998	-0.173050	0.999478	0.598633	0.013515	-0.155133	0.999288	0.597998	...
26	NaN	0.608358	0.062355	-0.192453	0.999354	0.599869	0.014811	-0.148571	0.999088	0.599029	...

Figure 25 Raw data

	class	x1	y1	z1	v1	x2	y2	z2	v2	x3	...
9	Bicep Down	0.566319	0.062481	-0.258781	0.999664	0.563809	0.023923	-0.235655	0.999085	0.565149	...
10	Bicep Down	0.566792	0.062759	-0.221019	0.999652	0.564401	0.024213	-0.202950	0.999085	0.565583	...
11	Bicep Down	0.563492	0.063009	-0.246001	0.999668	0.561225	0.024378	-0.226396	0.999137	0.562614	...
12	Bicep Down	0.561856	0.063079	-0.240953	0.999676	0.560207	0.024468	-0.220182	0.999172	0.561714	...
13	Bicep Down	0.560788	0.063122	-0.255947	0.999684	0.558903	0.024776	-0.234829	0.999203	0.560447	...
14	Bicep Down	0.559144	0.063351	-0.265532	0.999688	0.557240	0.025461	-0.244541	0.999223	0.558941	...
15	Bicep Down	0.558697	0.063596	-0.277275	0.999697	0.556808	0.025786	-0.253482	0.999253	0.558605	...
16	Bicep Down	0.558336	0.063757	-0.280720	0.999702	0.556178	0.026012	-0.257171	0.999273	0.557905	...
17	Bicep Down	0.557933	0.064719	-0.276733	0.999703	0.556016	0.026884	-0.252549	0.999284	0.557639	...
18	Bicep Down	0.557241	0.064691	-0.273022	0.999715	0.555384	0.027020	-0.249706	0.999318	0.557025	...
19	Bicep Down	0.556971	0.064758	-0.274448	0.999722	0.555362	0.027161	-0.251482	0.999346	0.557031	...
20	Bicep Down	0.555633	0.064649	-0.273878	0.999726	0.553954	0.027267	-0.251989	0.999365	0.555741	...
21	Bicep Down	0.555873	0.064788	-0.266320	0.999745	0.554352	0.028103	-0.245373	0.999414	0.556115	...
30	Bicep Up	0.582806	0.067493	-0.167083	0.998167	0.578285	0.034041	-0.151733	0.997446	0.578488	...
31	Bicep Up	0.579772	0.071313	-0.116165	0.998092	0.576199	0.038235	-0.102093	0.997383	0.576409	...
32	Bicep Up	0.576881	0.073190	-0.141496	0.998142	0.575699	0.039735	-0.125849	0.997461	0.576496	...
33	Bicep Up	0.575279	0.074087	-0.169057	0.998231	0.574344	0.041002	-0.152248	0.997573	0.575344	...

Figure 26 Cleaned data

## Model Training

The machine learning task used in this project is called Supervised Learning. Supervised learning is defined as the type of machine learning that take in data samples, also known as training data, and associated outputs, also known as labels, with each data sample during the model training process. The main objective is to learn a mapping or association between input data samples  $x$  and their corresponding outputs  $y$  based on multiple training data instances. In other words, this method is termed because our model learns on data samples where the desired outputs are already known beforehand. In this example, the desired output is the workout posture class, and the data samples or training data are the  $x$ ,  $y$ ,  $z$ , and visibility joints coordinates.

Therefore, the dataset must be separated into data samples,  $x$ , and corresponding outputs,  $y$ . This can be done by dropping workout posture class values and store it in variable  $x$ . The remaining values will be stored in variable  $y$ . Figure 27 shows the example of these datasets.

X.head()										
	x1	y1	z1	v1	x2	y2	z2	v2	x3	y3 ...
0	0.646326	0.093126	-0.762514	0.999541	0.654877	0.047631	-0.733119	0.998520	0.660929	0.048893 ...
1	0.609334	0.062254	-0.355626	0.999570	0.609822	0.020864	-0.328198	0.998626	0.612339	0.021536 ...
2	0.595575	0.056722	-0.370127	0.999595	0.593889	0.016085	-0.345562	0.998718	0.595536	0.016191 ...
3	0.583862	0.059034	-0.326914	0.999631	0.580601	0.018613	-0.301308	0.998834	0.582206	0.018787 ...
4	0.581075	0.060247	-0.259024	0.999653	0.574972	0.020182	-0.237870	0.998917	0.575595	0.020270 ...

y.head()	
0	Bicep Down
1	Bicep Down
2	Bicep Down
3	Bicep Down
4	Bicep Down
Name: class, dtype: object	

Figure 27 Sample dataset

Before proceeding to training the machine learning model, it is important to split the data into train dataset and test dataset. The reason of splitting the data is that it can produce an unbiased evaluation of later prediction

performance. Splitting into train dataset is used to fit the machine learning model, while splitting into test dataset is used to evaluate the fitted machine learning model. The objective is to estimate the machine learning model's performance on new data that was not used to train the model. This data splitting can easily accomplish using a data science library, scikit-learn, with its `train_test_split()` function. The author chooses a 70/30 split ratio, in which 70 percent of the dataset split into training and 30 percent split into testing.

The datasets can then be trained using various machine learning classification models. The author will use the common classification algorithm such as logistic regression, ridge regression, random forest, and gradient boost. The algorithm with highest accuracy will be chosen to train the classifiers.

### **Model Evaluation**

Classification accuracy, confusion matrix, precision and recall, F1 Score, and many other metrics are generated to assist in determining which machine learning algorithm produces the best and most accurate results.

## a. Logistic Regression

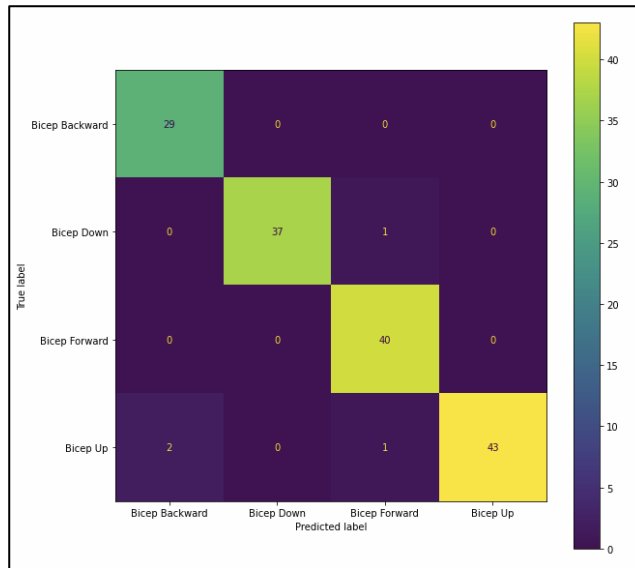


Figure 28 Confusion Matrix of Logistic Regression

Classification Report:				
	precision	recall	f1-score	support
Bicep Backward	0.94	1.00	0.97	29
Bicep Down	1.00	0.97	0.99	38
Bicep Forward	0.95	1.00	0.98	40
Bicep Up	1.00	0.93	0.97	46
accuracy			0.97	153
macro avg	0.97	0.98	0.97	153
weighted avg	0.98	0.97	0.97	153

Figure 29 Classification Report of Logistic Regression

## b. Ridge Regression

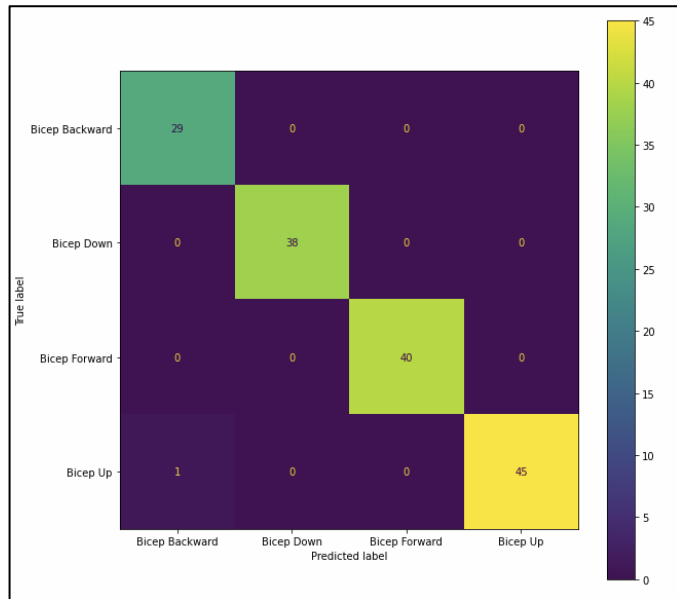


Figure 30 Confusion Matrix of Ridge Regression

Classification Report:				
	precision	recall	f1-score	support
Bicep Backward	0.97	1.00	0.98	29
Bicep Down	1.00	1.00	1.00	38
Bicep Forward	1.00	1.00	1.00	40
Bicep Up	1.00	0.98	0.99	46
accuracy			0.99	153
macro avg	0.99	0.99	0.99	153
weighted avg	0.99	0.99	0.99	153

Figure 31 Classification Report of Ridge Regression

### c. Random Forest

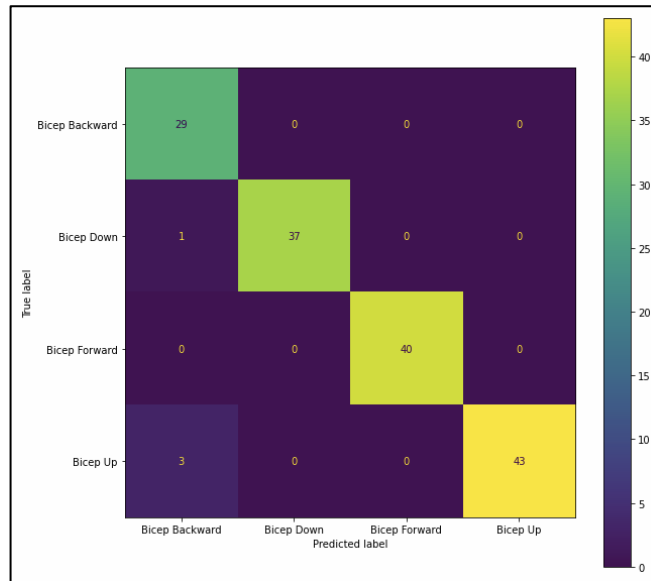


Figure 32 Confusion Matrix of Random Forest

Classification Report:				
	precision	recall	f1-score	support
Bicep Backward	0.88	1.00	0.94	29
Bicep Down	1.00	0.97	0.99	38
Bicep Forward	1.00	1.00	1.00	40
Bicep Up	1.00	0.93	0.97	46
accuracy			0.97	153
macro avg	0.97	0.98	0.97	153
weighted avg	0.98	0.97	0.97	153

Figure 33 Classification Report of Random Forest



#### d. Gradient Boost

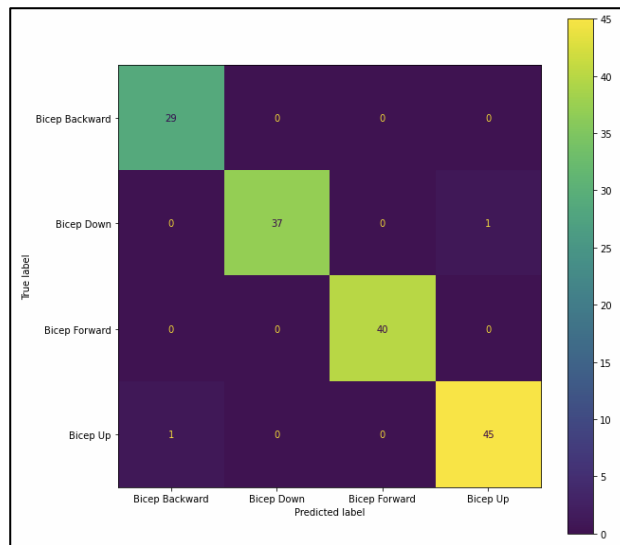


Figure 34 Confusion Matrix of Gradient Boost

Classification Report:				
	precision	recall	f1-score	support
Bicep Backward	0.97	1.00	0.98	29
Bicep Down	1.00	0.97	0.99	38
Bicep Forward	1.00	1.00	1.00	40
Bicep Up	0.98	0.98	0.98	46
accuracy			0.99	153
macro avg	0.99	0.99	0.99	153
weighted avg	0.99	0.99	0.99	153

Figure 35 Classification Report of Gradient Boost

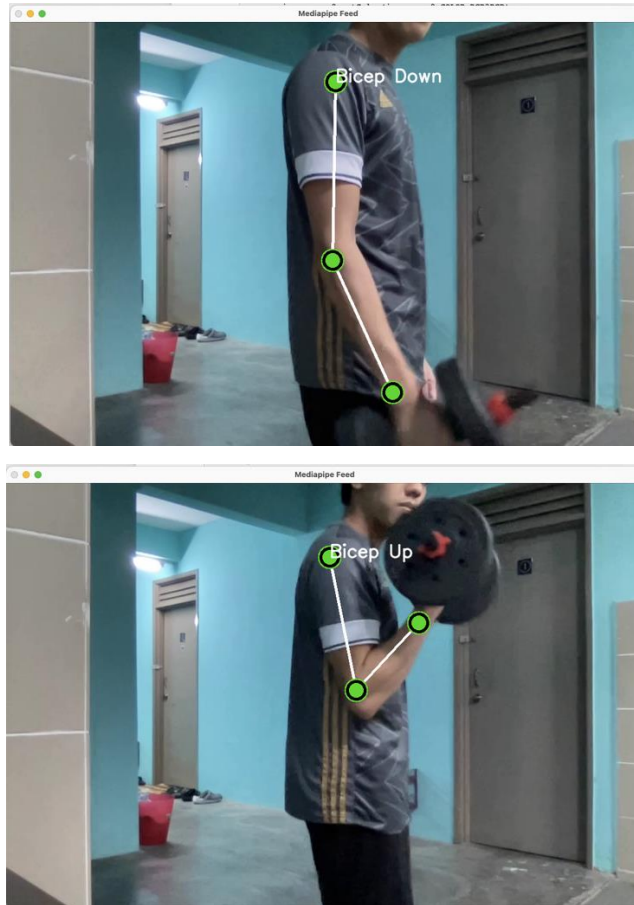
Table 2 Comparison between results of different Machine Learning Algorithms

Machine Learning Algorithm	Accuracy
Logistic Regression	97%
Ridge Regression	99%
Random Forest	97%
Gradient Boost	98%

According to Table 2, Ridge Regression shows the best accuracy among other three algorithms. Therefore, Ridge Regression will be chosen to train the workout classifier.

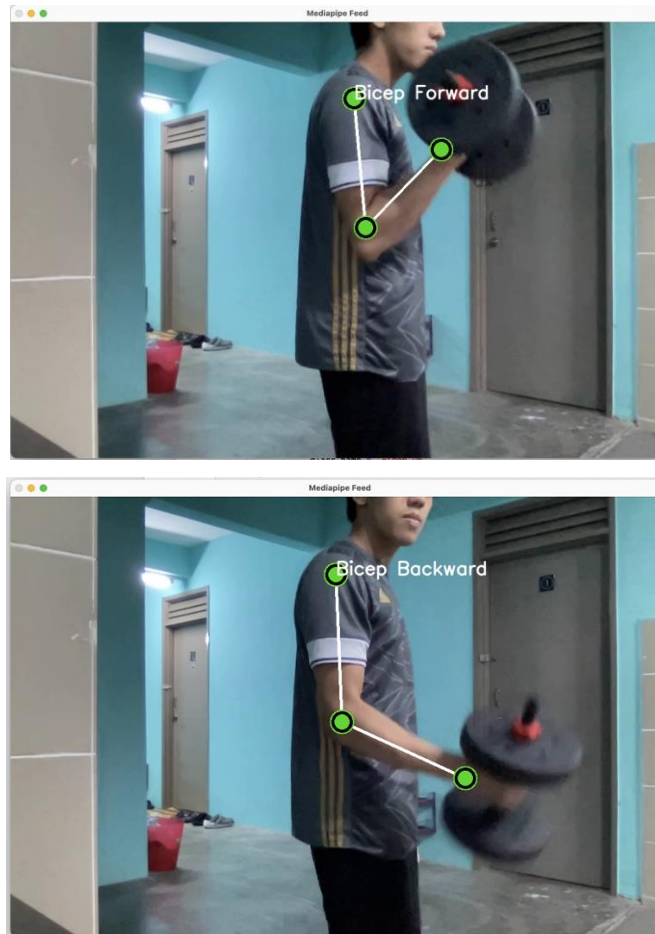
## Model Deployment

### Correct Prediction



*Figure 36 Correct Prediction using Ridge Regression*

## Incorrect Prediction

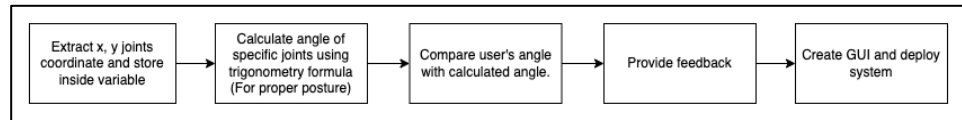


*Figure 37 Incorrect Prediction using Ridge Regression*

Even though the ridge regression shows the highest accuracy, the result after deploying and testing in real-time does not performed perfectly as expected. The model displays many false positive of a certain posture classification which can be seen in figure 37. The author concluded that this occurred as a result of a variety of factors. These factors include a small amount of video data for model training, a different training data environment, a change in lighting, and so on.

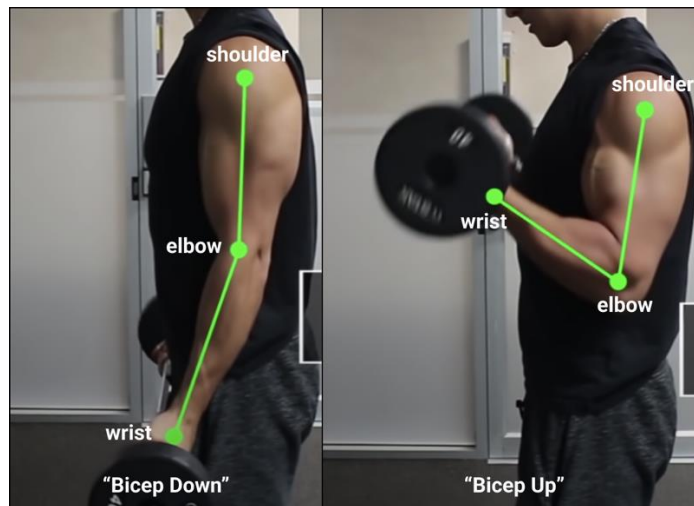
### 4.2.2 Geometrical Approach

Another approach of detecting improper postures during workout is calculating the angles using basic trigonometry. Figure 38 shows the process of developing using geometrical approach.



*Figure 38 Process flow of geometrical approach*

In a bicep curl workout, there will be two different arm position. First, the arm of the user must be kept straight and not move significantly. This position will be classified as “Bicep Down”. Secondly, the forearm and wrist of the user will be brought up until the bicep is fully contracted to show it is a complete bicep curl. In order to calculate this, the angle must be taken between shoulder, elbow, and wrist. Figure 39 illustrates the angle that will be calculated using geometrical approach.



*Figure 39 Angle of three joints*

The position of shoulder, elbow, and wrist are represented by x-coordinate and y-coordinate. These coordinates will be extracted and apply it into a formula to calculate the angle between three points (shoulder, elbow, wrist) in degrees. By doing this, the author can analyse the angle of proper posture when the user is in their start position with the weight down (nearly 180°) and when the weight is lifted (around 25°).

```

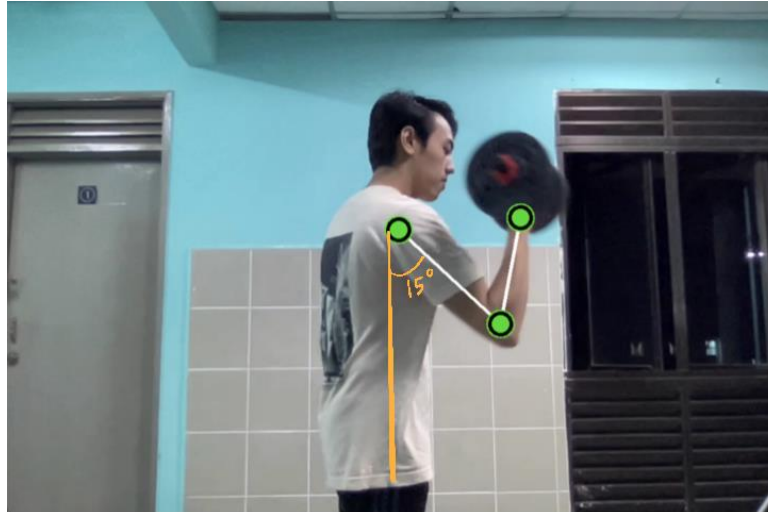
radians = np.arctan2(y3 - y2, x3 - x2) - np.arctan2(y1 - y2, x1 - x2)
angle = np.abs(radians * 180.0 / np.pi)
  
```

*Figure 40 Geometrical calculation formula*



*Figure 41 Example of angle in degrees of three joints*

However, the angle between shoulder, elbow, and wrist is only suitable to calculate the number of repetitions, by counting the stages of “bicep down” and “bicep up”. To determine whether the user is performing the workout with incorrect posture, it is necessary to understand and research the most common bicep curl error that people have been making. According to Herman, the common mistake is hooking weight instead of curling. This means that the movement of their elbows are too excessive either forward or backward and not lock to their side throughout the movement. Therefore, the 3 points that work the best to calculate the angles of improper posture is between elbow, shoulder, and hip. If the user is performing bicep curl correctly by locking their elbows to their side, the angle should be approximately  $0^{\circ}$ . Through the analysis of annotated video data, it is found that if the angle between elbow, shoulder, and hip exceeds  $10^{\circ}$ , the elbow position is too forward. If the angle between those 3 points is less than  $10^{\circ}$ , then the elbow position is too backward. The system will then notify the user by displaying feedback and making suggestions on how to improve their posture using quantified measures and thresholds.



*Figure 42 Angle of three joints: Bicep forward*



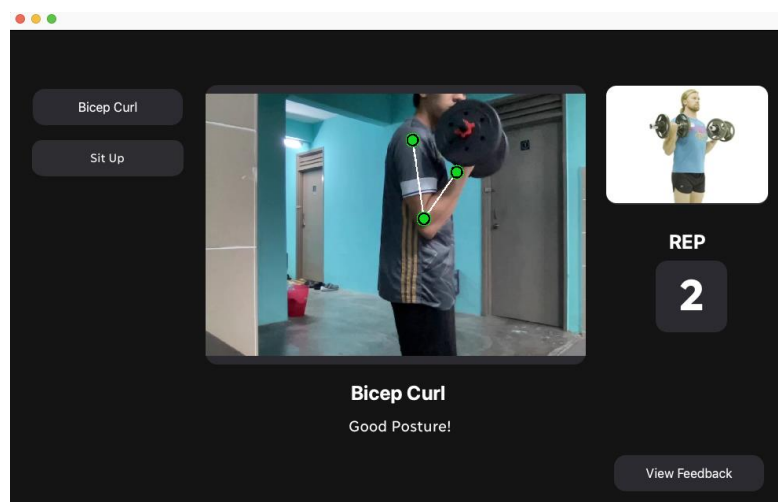
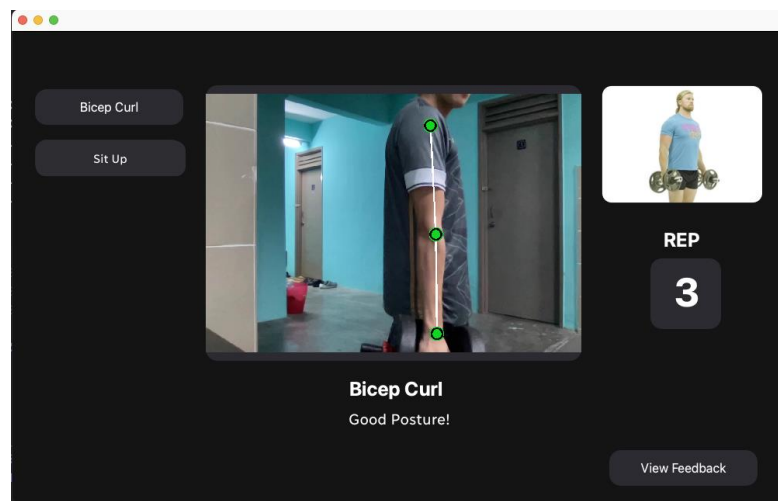
*Figure 43 Angle of three joints: Bicep backward*

As shown in figures above, using geometrical approach performed better than machine learning approach. Through the analysis, the geometric approach classifies all proper posture correctly and almost 90% of improper posture are labelled as improper posture. In addition, geometrical approach uses the most straight forwarded way of detecting improper posture by calculating the difference between angle of proper posture and improper posture in real-time instead of predicting the posture using Machine Learning approach. Therefore, this approach will be implemented in the desktop application which can be seen in the next subchapter.

## 4.3 Desktop Application

### System Design



The following figures shows the main page and feedback page of FitAI desktop application. On the left side buttons, users are able to select either bicep curl or sit up to analyse their workout posture. The system will then display the visualization of relevant joints that associated to the selected workout. Furthermore, the system will display a reference video of a trainer performing the selected workout as well as the number of workout repetitions. Feedback will be presented in the form of instructions and will also be read aloud by a "text-to-speech" generator to alert the user. Finally, once the user has completed the workout, they can view the full analysis of their workout session by clicking on the "View Feedback" button.





Bicep Curl

Sit Up



REP  
**3**



**Bicep Curl**

Position of your elbow is too forward.  
Maintain and lock it to your side.

View Feedback

Bicep Curl

Sit Up



REP  
**5**


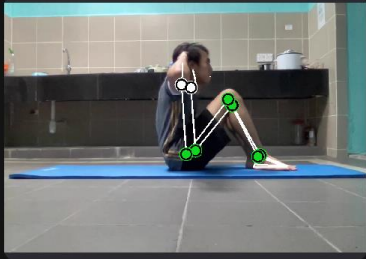
**Bicep Curl**

Position of your elbow is too backward.  
Maintain and lock it to your side.

View Feedback

Bicep Curl

Sit Up



REP  
**5**

**Sit Up**

Good Posture!

View Feedback



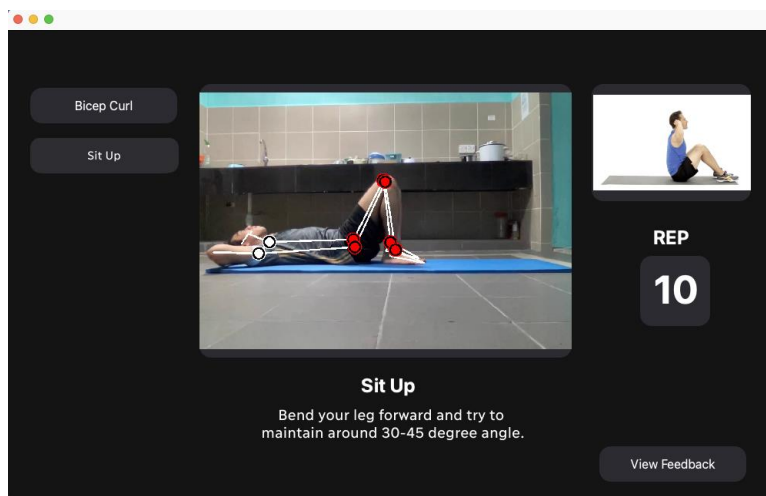
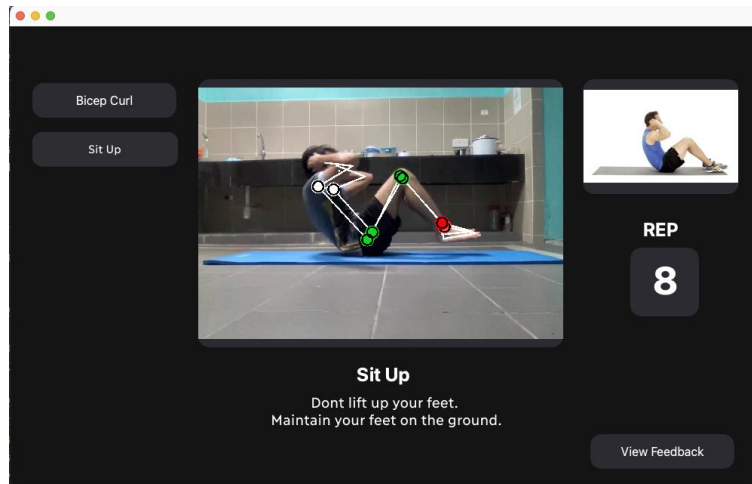


Figure 44 Main Page of FitAI Desktop Application

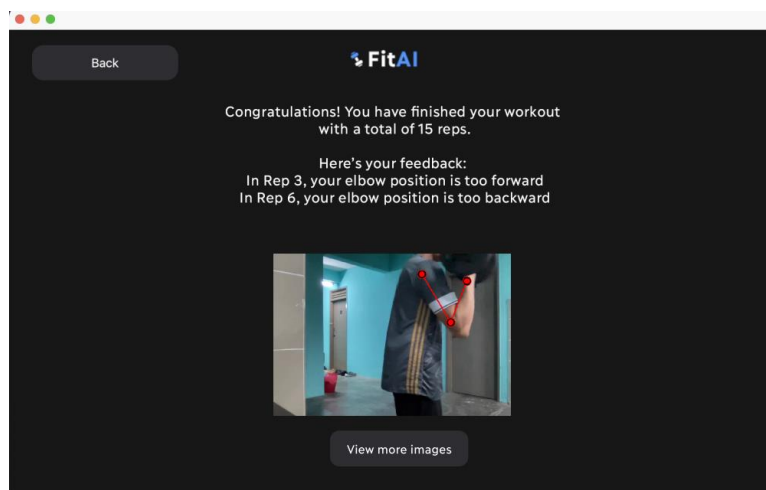


Figure 45 Feedback Page of FitAI Desktop Application

## CHAPTER 5: CONCLUSION AND RECOMMENDATION

Overall, the home workout posture analysis system was developed successfully and met the author's objectives. The system development method is divided into four stages, which are achieving human pose estimation by detecting human joints (keypoints), detecting proper and improper workout posture using machine learning and geometrical approaches, comparing which approaches yield the most accuracy, and finally building a desktop application with a user-friendly interface for better system usability. Initially, various publicly available datasets were chosen, such as COCO and LSP datasets, to achieve human pose estimation using CNN. However, implementing CNN from scratch would be difficult due to the large volume of data, which necessitates a large amount of computational power and a powerful GPU. As an alternative, this system utilizes Google's Mediapipe as the primary keypoint detection model. In terms of detecting workout postures, geometrical approach was chosen as the best approach before implementing it into desktop application.

One notable issue in this project is that the MediaPipe model may overlook human keypoint locations. However, this could be due to lighting, a complex background, multiple people staying in the same video frame, and other factors. The next possible issue that can occur is the accuracy of the angle calculation for proper and improper postures due to different camera angles and position. Throughout the research, the author only tested on one specific camera angle that is parallel to the user. However, this can be fixed in the future iterations by adding a recommended camera angle and position for more accurate results.

The author's inspiration for this project came from observing people who were experiencing mild pain during their workout session, and poor workout posture is one of the factors that contribute to the pain. The author acknowledged that this system has a great potential in fixing this with the current technology advancement of Artificial Intelligence, Computer Vision and Machine Learning. Therefore, this project has allowed people to perform home workouts correctly without any involvement of a personal trainer. However, the performance of current stage of this project would be less perfect due to various factors that cannot be fixed by a single novice programmer in a timespan of two semesters. As a result, finding the closest match of the target

domain would produce a reasonable and acceptable result. Following that, feedback will be documented, and an appropriate improvement strategy will be implemented in the project's future work.

## REFERENCES

- Gulam, A. (2016). Need, Importance and Benefits of exercise in daily life. *International Journal of Physical Education, Sports and Health*. 22, 22-27
- Kaur, H., Singh, T., Arya, Y., K., & Mittal, S. (2020). Physical Fitness and Exercise During the COVID-19 Pandemic: A Qualitative Enquiry, *Frontiers in Psychology*. doi:10.3389/fpsyg.2020.590172
- Imbo, W. (2018). How Poor Posture Affects Your Health and Athletic Performance. Retrieved from <https://boxlifemagazine.com/5193-2/>
- Fintelics. (2021). Computer Vision – What It is And How It Works. Retrieved from <https://fintelics.medium.com/computer-vision-what-it-is-and-how-it-works-c9e1b30a5f2e>
- Duncan, M. (2019). The Top Workout Mistakes And How to Correct Them. Retrieved from <https://camillestyles.com/wellness/top-workout-mistakes-correct/>
- Ahmad I., Kim J. Y. (2018). Assessment of whole body and local muscle fatigue using electromyography and a perceived exertion scale for squat lifting. <https://www.mdpi.com/1660-4601/15/4/784>
- Kothari, S. (2020). Yoga Pose Classification Using Deep Learning. *International Journal of Scientific Research in Computer Science Engineering and Information Technology*. doi:10.32628/CSEIT206623
- Yamakawa, A., Ishikawa, T., & Watanabe, H. (2020). Study on Improvement of Estimation Accuracy in Pose Estimation Model Using Time Series Correlation. *Proceedings of IEEE 9<sup>th</sup> Global Conference on Consumer Electronics*, (pp. 409-412). doi:10.1109/GCCE50665.2020.9291962
- Cao, Z., Hidalgo, G., Simon, T., Wei, S., & Sheikh, Y. (2019). OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. arXiv:1812.08008v2
- Ribera, S., M. (2020). Computer Vision Analysis of the body-pose similarity from two different subjects with the aim of the correct development of physical exercises. Retrieved from <http://hdl.handle.net/2117/331420>
- Chen, S. & Yang, R., R. (2020). Pose Trainer: Correcting Exercise Posture using Pose Estimation. arXiv:2006.11718

- Chen, Y., Chen, Y., Tu, Z. (2019). Fitness Done Right: A Real-Time Intelligent Personal Trainer For Exercise Correction. arXiv:1911.07935v1
- Nagarkoti, A., Teotia, R., Mahale, A., K., & Das, P., K. (2019). Realtime Indoor Workout Analysis Using Machine Learning & Computer Vision. doi:10.1109/EMBC.2019.8856547.
- Nishani, E., & Cico, B. (2017). Computer Vision Approaches based on Deep Learning and Neural Networks: Deep Neural Networks for Video Analysis of Human Pose Estimation. *Proceedings of the 6th Mediterranean Conference on Embedded Computing (MECO)* (pp. 1-4). doi:10.1109/MECO.2017.7977207
- Dsouza, G., Maurya, D., & Patel, A. (2020). Smart gym trainer using Human pose estimation. *Proceedings of IEEE International Conference for Innovation in Technology (INOCON)* (pp. 1-4). doi:10.1109/INOCON50539.2020.9298212
- Jin, X., Yao, Y., Jiang, Q., Huang, X., Zhang, J., Zhang, X., & Zhang, K. (2015). Virtual Personal Trainer via the Kinect Sensor. *Proceedings of IEEE 16th International Conference on Communication Technology (ICCT)*, (pp. 460-463). doi:10.1109/ICCT.2015.7399879.
- Anilkumar, A., Athulya, K., T., Sajan, S., & Sreeja K., A. (2021). Pose Estimated Yoga Monitoring System. *Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS*. doi:10.2139/ssrn.3882498
- Yang, L., Li, Y., Zeng, D., & Wang, D. (2021). Human Exercise Posture Analysis based on Pose Estimation. *Proceedings of IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, (pp. 1715-1719). doi:10.1109/IAEAC50856.2021.9390870
- Sonwani, N., & Pegwar, A. (2020). Auto\_Fit: Workout Tracking using Pose Estimation and DNN. *International Journal of Engineering Applied Sciences and Technology*. <https://www.ijeast.com/papers/167-173,Tesma501,IJEAST.pdf>
- Alatiah, T., & Chen, C. (2020). Recognizing Exercises and Counting Repetitions in Real Time. arXiv:2005.03194
- Mahendran, N. (2021). Deep Learning for Fitness. arXiv:2109.01376
- Yadav, S., K., Singh, A., Gupta, A., & Raheja, J., L. (2019). Real-time Yoga recognition using deep learning. doi:10.1007/s00521-019-04232-7

- Bhambure, S., Lawande, S., Upasani, R., & Kundargi, J. (2021). Yog-Assist. *Proceedings of 4th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*, (pp. 1-6).  
doi:10.1109/ICNTE51185.2021.9487575
- Agrawal, R. (2021). Posture Detection using PoseNet with Real-time Deep Learning project. Retrieved from <https://www.analyticsvidhya.com/blog/2021/09/posture-detection-using-posenet-with-real-time-deep-learning-project/>
- Khoshelham, K., & Elberink, S., O. (2012). Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. *Sensors*, *12*(2), 1437–1454.  
doi:10.3390/s120201437
- Zhang, J. (2020). Dynamic Time Warping. Retrieved from <https://towardsdatascience.com/dynamic-time-warping-3933f25fcdd>
- Herman, S. (2018). 5 Dumbest Dumbbell Bicep Curl Mistakes Sabotaging Your Bicep Growth! Retrieved from <https://muscularstrength.com/article/Five-Dumbest-Dumbbell-Bicep-Curl-Mistakes-Sabotaging-Bicep-Growth>