Thapar Institute of Engineering and Technology

Software Requirements Specification for

Sports Analytics web API using Deep Learning

Version 1.0 approved

Submitted by: IDE-6:-

Shobhit Sinha: 101815004

Sayam Mittal: 101995013

Priyam Gupta: 101995008

Pranav Singh: 101865011

Ananya Tiwari: 101965022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Basketball Analytics	9/16/20	Initial Release	1.0

1. Introduction

1.1 Purpose

The objective of this SRS is to provide a comprehensive sketch of our software product, its parameters, and its purposes. This document specifies the project's target client and its user interface, hardware, and software requirements. It illustrates how our team and audience recognize the product, that is, Pose Detection and Estimation Software, and its efficacy. Also, it should predict and sort out how this product is hoped to be used to gain a better understanding of the project, outline notions that may be elaborated later, and document approaches that are being considered but may be rejected later as the product advances.

1.2 Intended Audience and Reading Suggestions

This Software Requirements document is intended for sportspersons in general, but specifically it targets basketball players to help them understand where their efforts should be targeted to improve or add some new techniques to it so as to increase the probability of scoring the basket.
The remaining sections of this document provide a general description, including characteristics of the users of this project, the product's hardware, and the functional and data requirements of the product. . General description of the project is discussed in section 2 of this document. It also gives the user viewpoint of product use. Section 3 gives the system features of the product. Further, section 4 tells us about the external interface requirements. While, section 5 informs us about other non-functional requirements such as performance, security and safety requirements.

1.3 Project Scope

In order to construct the shooting prediction model of basketball free throw, we took movies of basketball free throw motions with a full hi-vision video camera. Human body pose detection based on deep learning can be of considerable importance in the surveillance industry. This innovation can be used at sports venues, airports, train stations, and other crowded places to enhance security. Human pose estimation, coupled with other data science algorithms for motion recognition and analysis, and deep neural networks (DNNs) to tackle the ML involved in this, is the perfect combination to help avert and restraining violent situations.

The technology has the potential to remodel Entertainment and Media with compelling AR effects. It helps render and regulate human body parts speedily and accurately, delivering a lifelike illusion.

Figure 1 : Pose estimation skeleton map of man throwing basketball

1.3.1 Statistical Model

The shooting prediction model is a binary prediction as to whether to enter a basket or not. As major binary prediction models there are logistic regression and SVM. The SVM using the kernel method is a nonlinear model which may make high accuracy but cannot calculate the shooting probability.

1.3.2 Future Work

The data of basketball free throw in this experiment were taken from one side only by a web camera, so it was suitable to analyse with 2 dimensional data provided by OpenPose. However analysis of general sports motion requires 3 dimensional data like a tennis or ballet dance, so it is necessary to use 3 dimensional OpenPose or expand 2D data generated by 2D OpenPose to 3D data.

1.4 References

1. CMU-Perceptual-Computing-Lab, https://github.com/CMU-Perceptual-Computing-Lab/openpose

2. MSCOCO key point evaluation metric. http://mscoco.org/dataset/#keypoints-eval

3.https://www.igi-global.com/article/a-study-of-vision-based-human-motion-recognition-and-analysis/160126

4.https://www.sciencedirect.com/science/article/abs/pii/S1077314298907445

5.https://www.kdnuggets.com/2019/06/human-pose-estimation-deep-learning.html

6. https://nanonets.com/blog/human-pose-estimation-2d-guide/

7.https://www.kdnuggets.com/2019/06/human-pose-estimation-deep-learning.html

8. http://www.ee.oulu.fi/~gyzhao/research/gait_recognition.htm

9. https://www.clickworker.com.

10. "MSCOCO keypoint evaluation metric," http://mscoco.org/ dataset/#keypoints-eval.

11. https://arxiv.org/pdf/1812.08008.pdf

12.https://mobidev.biz/blog/human-pose-estimation-ai-personal-fitness-coach

2. Overall Description

2.1 Product Perspective

The product will serve as a system for basketball sports analytics. Physically the system does not require any attached device. The product does require an input file image/video which is used for the analytics which points out the required. For example, the user has the access to upload a file upto 5Mb and the necessary things will be predicted as per output.

Firstly the user will have to register him/her against the normal authentication terminal and once the terminal identifies the card user by matching the cryptographic key generated by the software with the one generated by the card, the welcome message is displayed to the user.

Next the software inside the terminal will check whether the user is registered or not. If the user has been denied access, then the software will send the message “Access Denied” onto the screen. Otherwise an “Access Granted” message will be displayed and the student can then enter the lab and access the resources.

Then the user will have to upload the file against the dashboard terminal which is provided in the left most part and once the video is uploaded it identifies the user via the following mechanism:

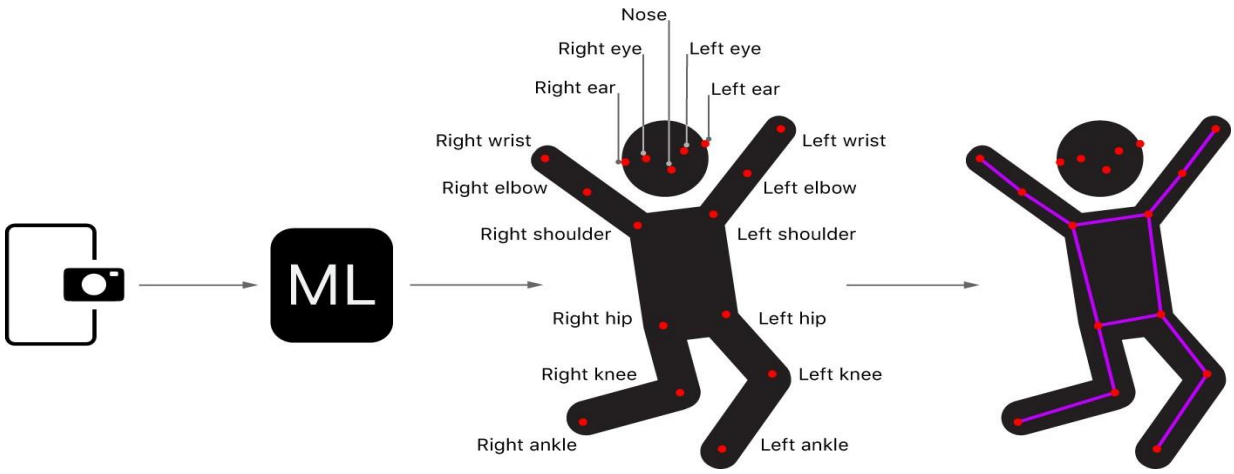


Figure 2: shows how the media is broken for the analytics.

When the user clicks the analytics button it generates the analytics of the system with the prediction of basket, the probability message is displayed to the user on the screen.
Here the model is based on SSD:

Single Shot MultiBox Detector:

SSD is designed for object detection in real-time. Faster R-CNN uses a region proposal network to create boundary boxes and utilizes those boxes to classify objects. While it is considered the start-of-the-art in accuracy, the whole process runs at 7 frames per second. Far below what a real-time processing needs. SSD speeds up the process by eliminating the need of the region proposal network. To recover the drop in accuracy, SSD applies a few improvements including multi-scale features and default boxes.

The SSD object detection composes of 2 parts:

- Extract feature maps, and
- Apply convolution filters to detect objects.

SSD uses **VGG16** to extract feature maps. Then it detects objects using the **Conv4_3** layer. For illustration, we draw the Conv4_3 to be 8 × 8 spatially (it should be 38 × 38). For each cell (also called location), it makes 4 object predictions.

Each prediction consists of a boundary box and 21 scores for each class (one extra class for no object), and we pick the highest score as the class for the bounded object. Conv4_3 makes a total of 38 × 38 × 4 predictions: four predictions per cell regardless of the depth of the feature maps. As expected, many predictions contain no object. SSD reserves a class “0” to indicate it has no objects.

2.2 Product functions:

The product should be able to perform the following operations:
1. It must be able to authenticate the user and manage the access against the values stored in the database.
2. It must be able to access the video status by the database and the videos uploaded.
3. The software must be able to update the video access privileges onto a particular user's credentials and the database where the privileges themselves will be modifiable only by the system administrators (or some authorized staff members).
4. The software must be able to determine whether the shooting can result in a basket or not a basket for a particular image/video.
5. Should also successfully analyze the pose of the player and track the basketball

2.3 User Class

- **Web Api:**
  We create a web API based in flask where user can input the image or video they want to analyze then the program does the shot and pose analysis.
  It Counts shooting attempts and missing, scoring shots from the input video. Detection keypoints of keypoints is done in different colors each having a specific meaning. OpenPose is implemented to calculate the angle of elbow and knee during shooting. Release angle and release time are calculated by all the data collected from shot analysis and pose analysis. Detection will be shown on the image. The confidence and the coordinate of the detection will be listed.

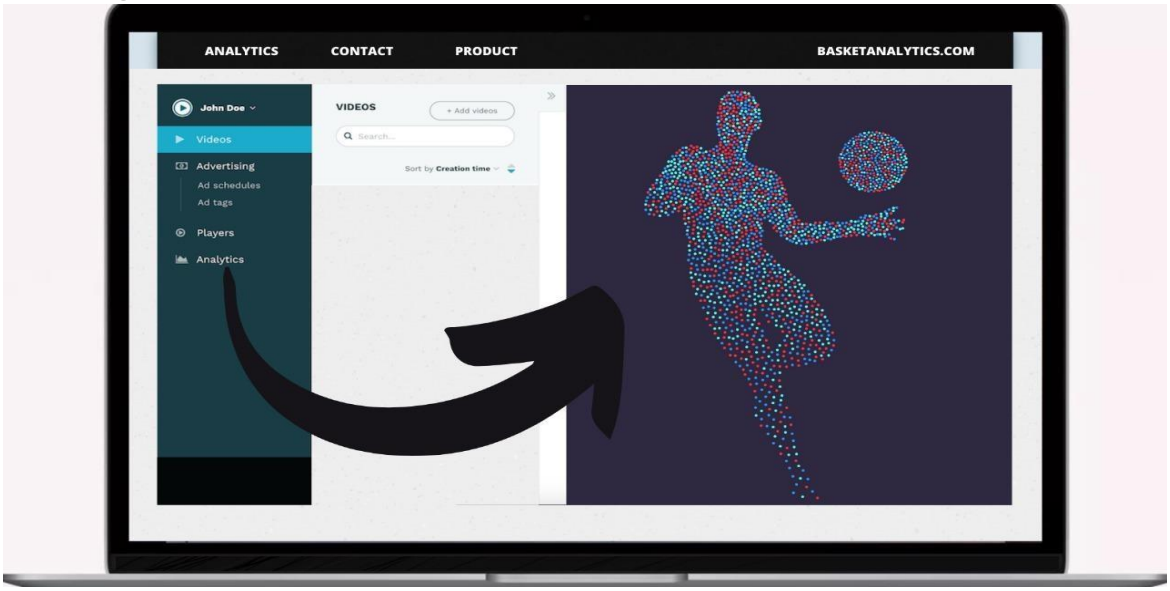


Fig 3: Tentative web API

- **Pose Estimation:**
  OpenPose is a pose estimation library which estimates pose using VGG19 neural network with part affinity vector. This is trained on COCO dataset and has been originally implemented in caffe. We convert it to tensorflow 2.0 so that keras deep learning models can be used with it.

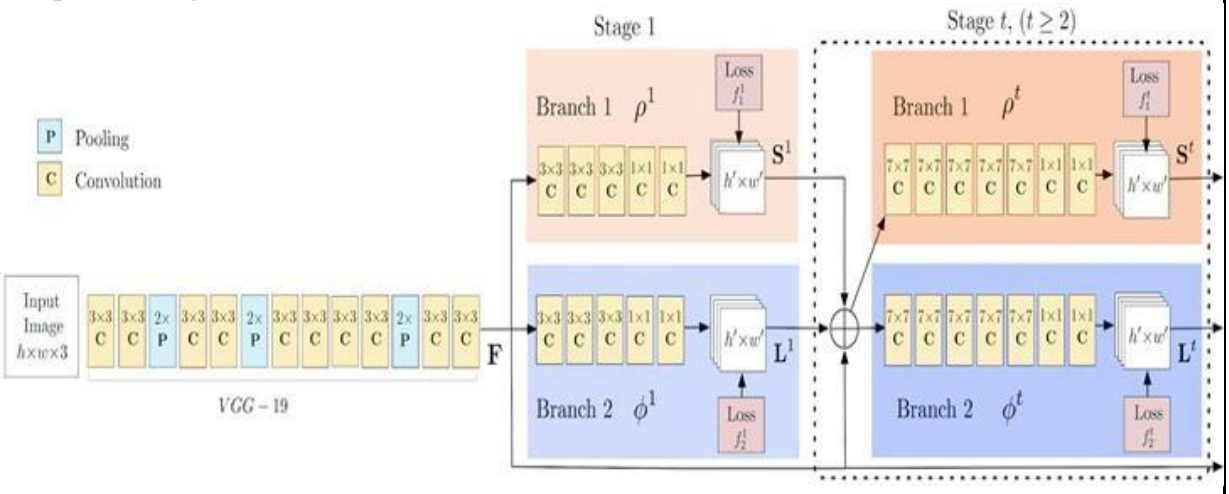


Figure 4 : OpenPose model architecture

The OpenPose network first extracts features from an image using the first few layers (VGG-19 in the above flowchart). The features are then fed into two parallel branches of convolutional layers. The first branch predicts a set of 18 confidence maps, with each map representing a particular part of the human pose skeleton. The second branch predicts a set of 38 Part Affinity Fields (PAFs) which represents the degree of association between parts.
Successive stages are used to refine the predictions made by each branch. Using the part confidence maps, bipartite graphs are formed between pairs of parts (as shown in the above image). Using the PAF values, weaker links in the bipartite graphs are pruned. Through the above steps, human pose skeletons can be estimated and assigned to every person in the image. We currently use the 17 keypoint based system.

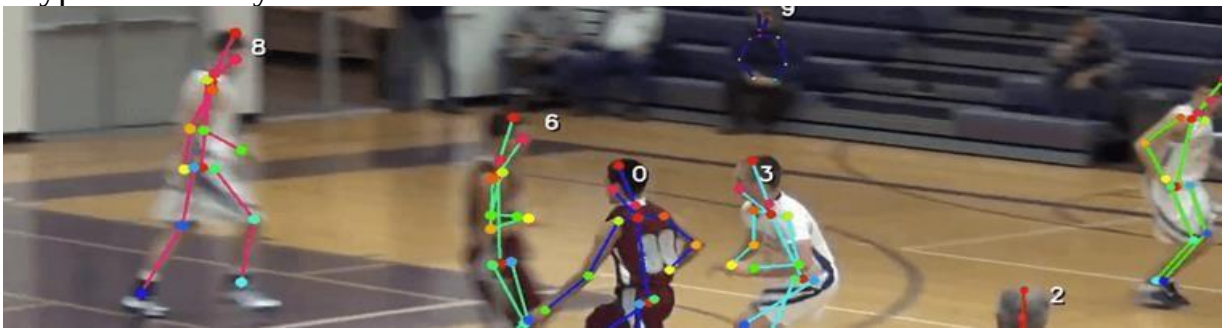


Figure 5 : OpenPose pose estimation working on live video feed

- **Detection and Analysis:**

For this part we use SSD neural network which is one of the fastest for real time detection of objects. SSD300 achieves 74.3% mAP at 59 FPS while SSD500 achieves 76.9% mAP at 22 FPS, which outperforms Faster R-CNN (73.2% mAP at 7 FPS) and YOLOv1 (63.4% mAP at 45 FPS). To have more accurate detection, different layers of feature maps are also going through a small 3×3 convolution for object detection.

Instead of using all the negative examples, we sort them using the highest confidence loss for each default box and pick the top ones so that the ratio between the negatives and positives is at most 3:1. This can lead to faster optimization and a more stable training.

The base network is VGG16 and pre-trained using ILSVRC classification dataset. **FC6 and FC7 are changed to convolution layers as Conv6 and Conv7** Furthermore, **FC6 and FC7 use Atrous convolution** (a.k.a Hole algorithm or dilated convolution) instead of conventional convolution. And **pool5 is changed from 2×2-s2 to 3×3-s1**.

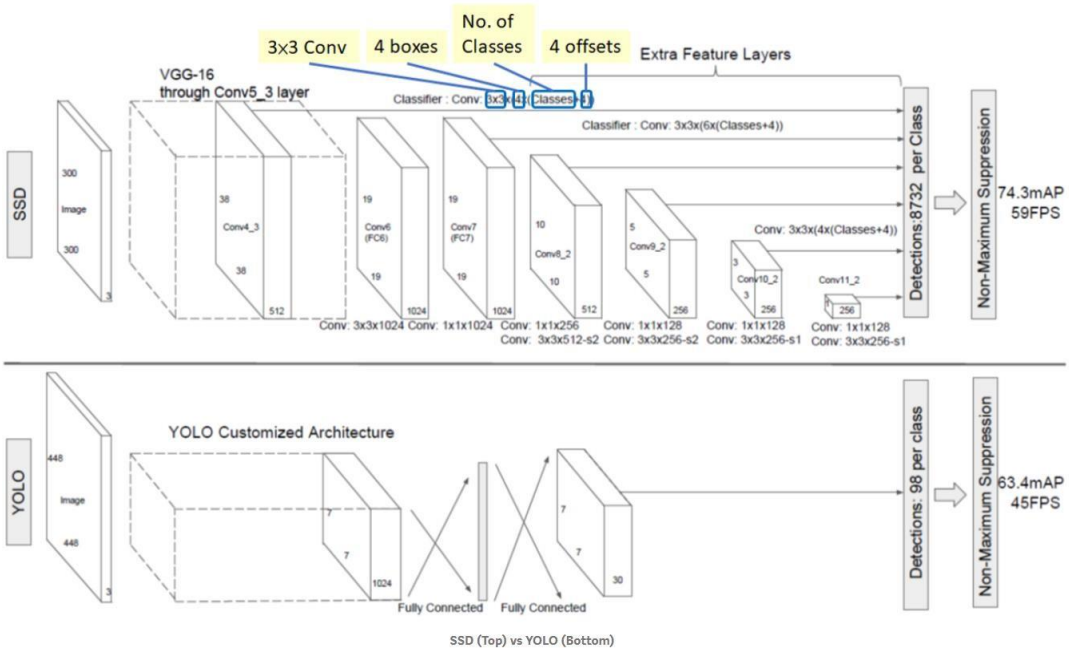


Figure 6 : Architecture of SSD deep learning model

2.4 User Characteristics:

The goal is to design the software for a selected group of individuals or team managers.
Our goal is to develop software that should be easy to use for all types of users. While designing the software one can assume that user type has the following characteristics:

- The user is computer-literate and has little or no difficulty in using a smart card to access information such as room status.

In order to use the system it is not required that a user beware of the internal working of a smart card but he/she is expected to know how to access the results in the analytics column.

2.5 Operating Environment:

The general operating environment includes:
Operating System: Windows, Linux.
Web Technologies:
Front-end: HTML5, CSS, Bootstrap, JavaScript.
Backend: Flask/Django, Postman API, SQL: Postgres or oracle mysql.
Machine Learning modules: Numpy and Pandas
Deep-Learning: tensorflow and keras.

2.6 Design and Implementation Constraints:

With the above mentioned technologies the only issue we might face while training the model as it is a deep neural network so it does require ample amounts of data to be trained for a model to get the required results. Also, Since tensorflow 2.0 is a little unstable and majority of things are deprecated so this can also result in issues.

2.7 Assumptions and Dependencies:

- The software has to be integrated onto the terminal that in turn has a limited capability for API request**.**
- There are no memory requirements.
- The product must have a user friendly interface that is simple enough for the users.
- Response time for loading the software and for processing an output should be no longer than seven seconds.
- A general knowledge of basic computer skills and of basic working of navigating and uploading is required to use the product.
- The image/video should be atleast 360p so to get better output.

3. System Features:

3. 1 Shot Detection:

It can detect whether the shot was missed or made and keep output streaks of baskets. It shows the probability of the ball that it would go in or not.
It will also give users the shot arcs from the shots they did score in green and once they missed in red by tracking the ball or drawing the arc itself from release angle and speed if ball is not their to give them a better graphical understanding of what went wrong.

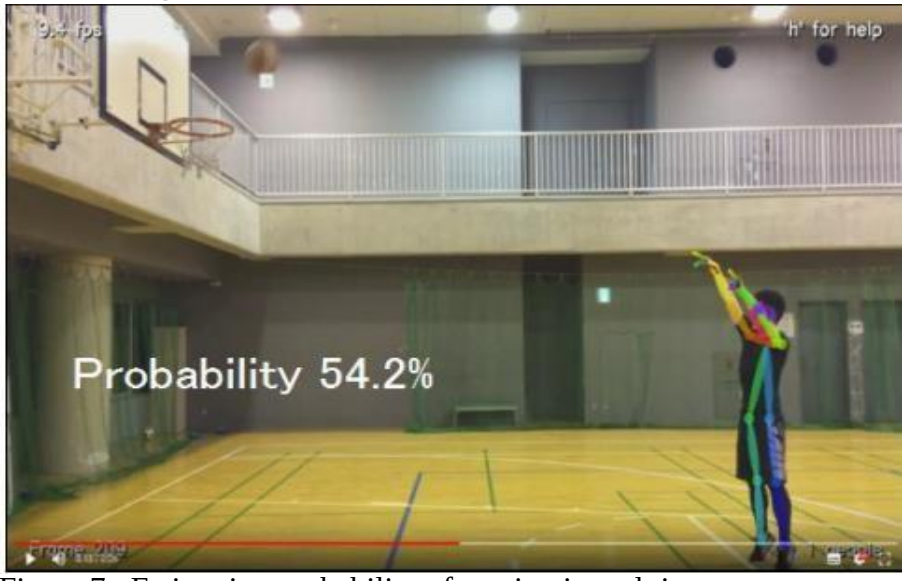


Figure 7 : Estimating probability of scoring in real time

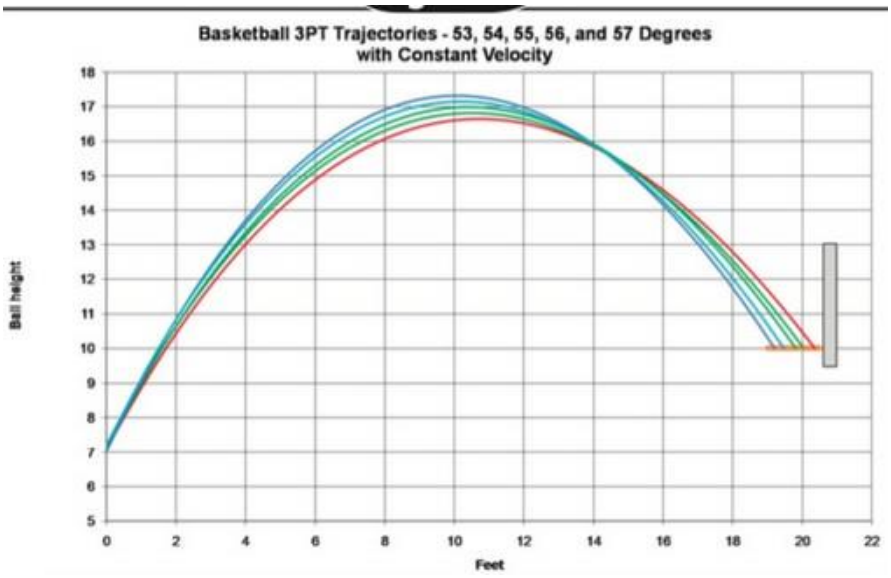


Figure 8 : tracing various ball arcs and color coding successful shots

Attempts	Miss	Score
4	2	2
Streak of 2		

Table 1: depicting the tentative UI

3.2 Pose analysis:

This detects the person shooting the ball and specially the important factors involved in the science of free throws such as the bent of the knee and the elbow. The release time and the release angle of the ball thus tracing the arc it would make from the hoop thus the user can see the probability that he would make the shot or not even if the basketball isn't present helping him improve.

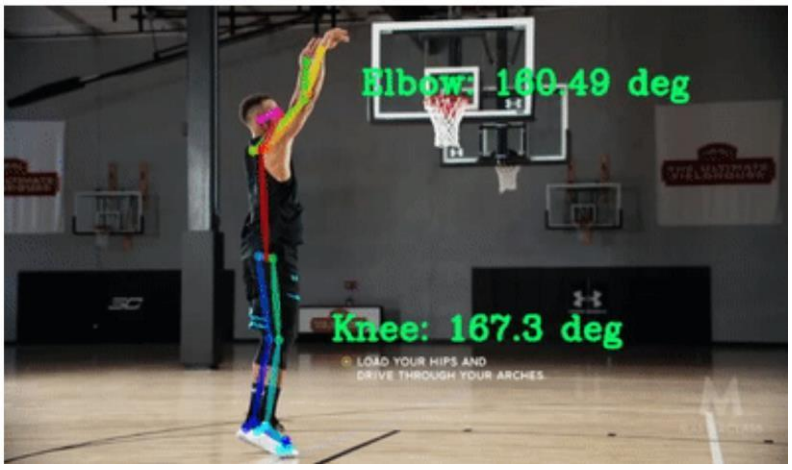


Figure 9: Showing real-time knee and elbow angle for pose analysis

Figure 10: calculating release angle and release time for pose analysis

Elbow angle	Knee angle	Release angle	Release time
61	140	33.61	1.2
Ideal values according to the deep learning model			

Table 2: depicting the tentative UI

4. External Interface Requirements

4.1 User interface

- Front-end software: CSS and HTML
- Back-end software: SQL+, tensorflow

Purpose	Analyzing basketball training sessions
Inputs	User uploaded image or video of their training session
Processing	The video goes through 2 series of deep learning models first is used for 17 key point identification in the human body. The other is used for object detection for court lines, ball, hoop for tracking.
Outputs	The output screen comes in the form of tables, diagrams and charts for the purpose of analytics. the player also sees suggestions and ideal pose analytic

	ratings for improvement. undetectable media is displayed in blue and the user is given suggestions on setting up the device for recording for proper analysis.

Table 3: depicting the tentative workflow

4.2 HARDWARE INTERFACES

- Windows.
- A browser which supports CGI, HTML & Javascript.

4.3 SOFTWARE INTERFACES

Following are the software used for the basketball analysis app online application.

Software used	**Description**
Operating system	We have chosen Windows operating system for its best support and user-friendliness.
Database	To save the training sessions, players records we have chosen SQL+ database.
CSS, HTML	To implement the project we have chosen Vb.Net language for its more interactive support.
Tensorflow	To train deep learning models to analyze the data and calculate the result accordingly

Table 5: Explaining the software stack in the project

4.4 COMMUNICATION INTERFACES

This project supports all types of web browsers. We are employing simple tasks and buttons like upload and analyze so there are simple web and server communication in place.

To speed up the process the api could also be run locally which is beneficial for analysis of large media files.

5. NonFunctional Requirements

5.1 Usability

The system should be easy to use. The user should reach the analysis with few buttons press if possible. Because one of the software's main goal is timesaving.
The system also should be user friendly for users because anyone should be able to use it in their training session. Basketball is one of the most popular sports, so better make it effective and help people improve. It can help your posture get better for excellent free throws even if the ball is not there.

5.2 Reliability

This software will be developed with machine learning, feature engineering and deep learning techniques and combined with web technologies. So, in this step there is no certain reliable percentage that is measurable.

Also, user provided data will be used to compare with result and measure reliability. With recent machine learning techniques, user gained data should be enough for reliability if data is obtained under mentioned conditions like complete visibility of the player and hoop in the frame.

The maintenance period should not be a matter because the reliable version is always run on the server which allow users to access the analysis window. When admins want to update, it take long as upload and update time of executable on server. The users can be reach and use program at any time, so maintenance should not be a big issue.

5.3 Performance:

Calculation time and response time should be as little as possible, because one of the software's features is timesaving. But it still depends on the type of media for example analysis probabilities from the images should be executed well under 30 secs while long training session video or free throws video might take a little longer to run the model on and analyze.

The capacity of servers should be as high as possible. Calculation and response times are very low, and this comes with that there can be so many sessions at the same times. The software is currently used and run locally, but we need to consider global sessions.

1 minute degradation of response time should be acceptable. The certain session limit also acceptable at early stages of development. It can be confirmed to user with “servers are not ready at this time” message.

5.4 Supportability:

The system should require Deep learning, batch, CSS, HTML, Python and flask knowledge to maintenance. If any problem acquire in server side and deep learning methods, it requires code knowledge and deep learning background to solve. Client side problems should be fixed with an update and it also require code knowledge and network knowledge.

6. Other Requirements

Appendix A: Glossary

Definitions:

(i) Computer Vision:

Computer vision is a field of artificial intelligence that trains computers to interpret and understand the visual world. Using digital images from cameras and videos and deep learning models, machines can accurately identify and classify objects — and then react to what they “see.”

(ii) Deep Learning:

Deep learning is an AI function that mimics the workings of the human brain in processing data for use in detecting objects, recognizing speech, translating languages, and making decisions. Deep learning AI is able to learn without human supervision, drawing from data that is both unstructured and unlabeled.

(iii) Augmented Reality:

a technology that superimposes a computer-generated image on a user's view of the real world, thus providing a composite view.

(iv) Human Pose Estimation:

Human Pose Estimation is defined as the problem of localization of human joints (also known as keypoints - elbows, wrists, etc) in images or videos. It is also defined as the search for a specific pose in space of all articulated poses.

(a.) 2D Pose Estimation - Estimate a 2D pose (x,y) coordinates for each joint from a RGB image.
(b.) 3D Pose Estimation - Estimate a 3D pose (x,y,z) coordinates a RGB image.

(v) Data Science:

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data. Data science is related to data mining, machine learning and big data.

(vi) Human Motion Analysis:

Motion analysis of human body parts involves the low-level segmentation of the human body into segments connected by joints and recovers the 3D structure of the human body using its 2D projections over a sequence of images.

(vii) Deep Neural Networks:

A deep neural network (DNN) is an artificial neural network (ANN) with multiple layers between the input and output layers. The DNN finds the correct mathematical manipulation to turn the input into the output, whether it be a linear relationship or a non-linear relationship.

(viii) Machine Learning:

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

(ix) Region-based Convolutional Neural Networks:

Object detection is the process of finding and classifying objects in an image. One deep learning approach, regions with convolutional neural networks (R-**CNN**), combines rectangular region proposals with convolutional neural network features.

Abbreviations Used:

(i) AR: Augmented Reality

(ii) CV: Computer Vision

(iii) DNN: Deep Neural Networks

(iv) ANN: Artificial Neural Networks

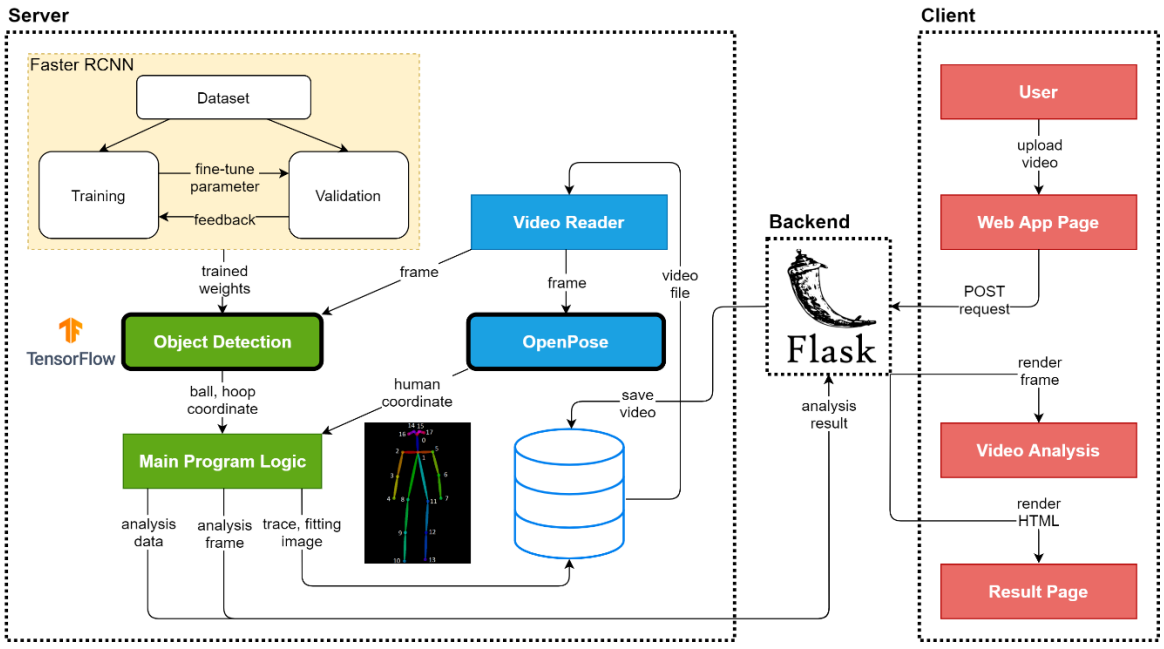
(v) ML: Machine Learning

(vi) AI: Artificial intelligence

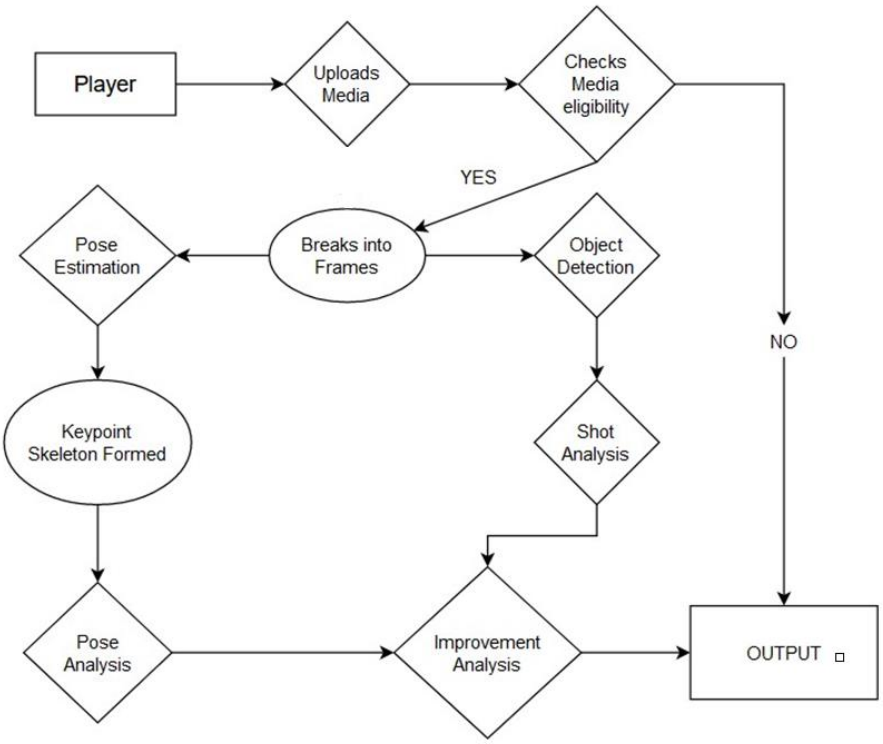
(vii) RCNN: Region-based Convolutional Neural Networks

Appendix B:

Model Diagram:

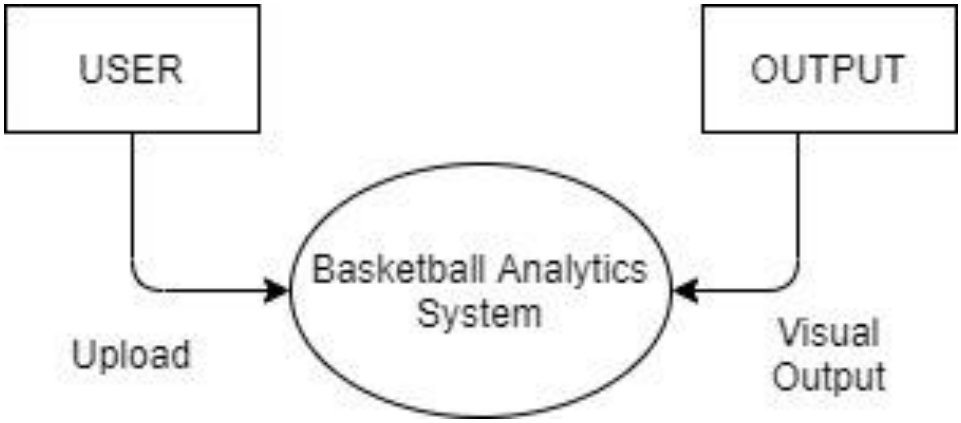


Entity Relationship Diagram:

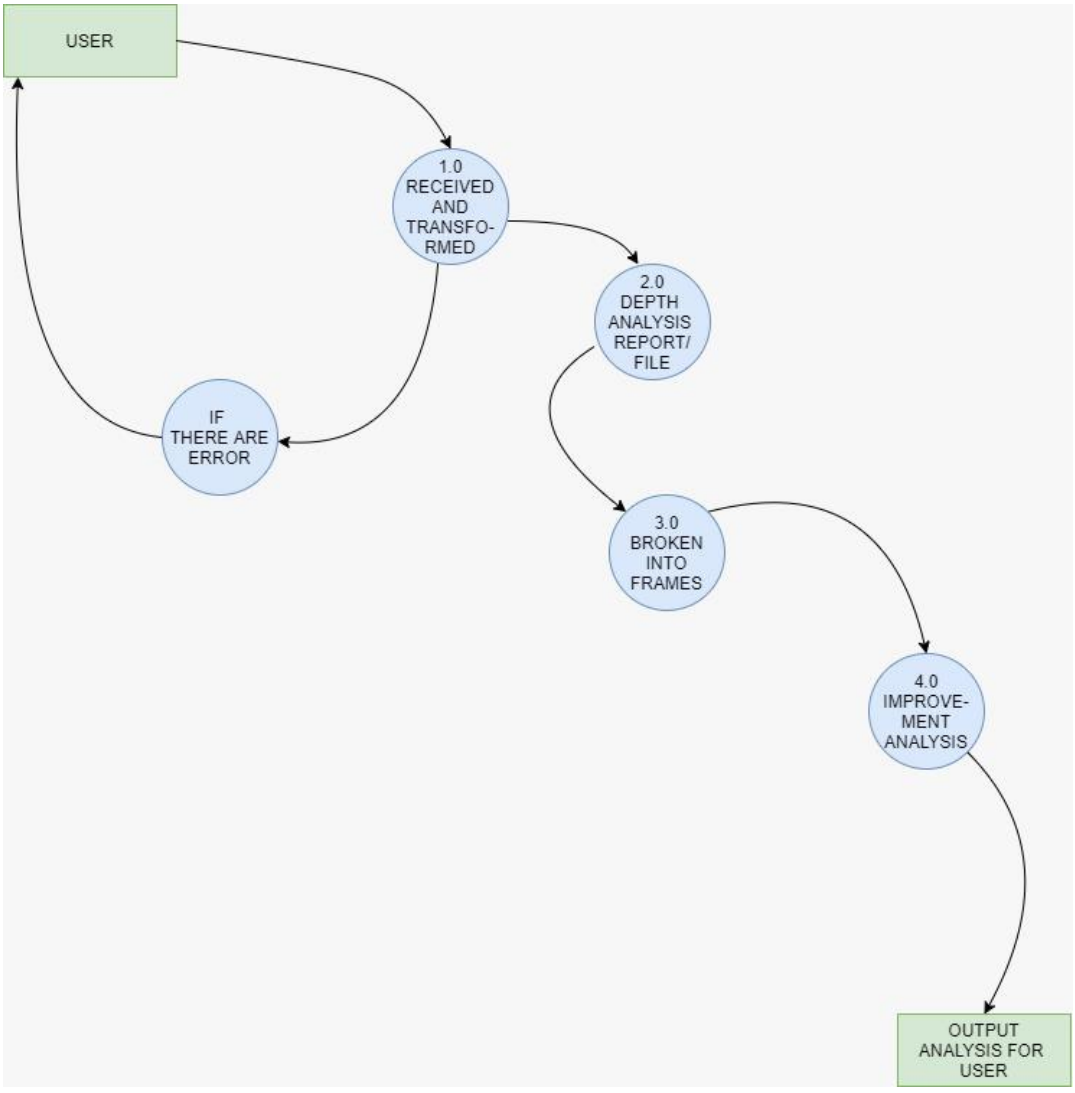


Data Flow Diagram:

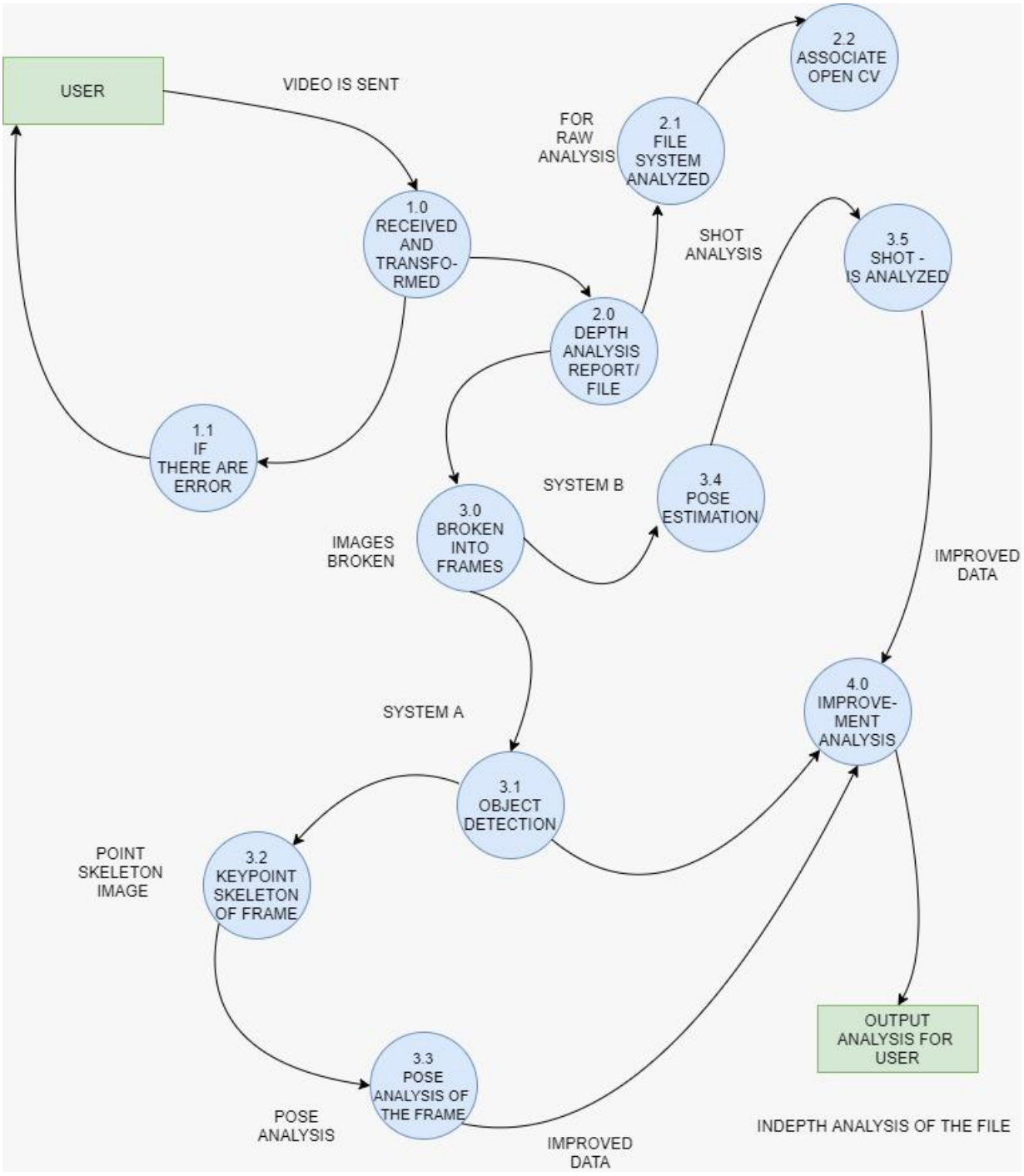
Level 0:



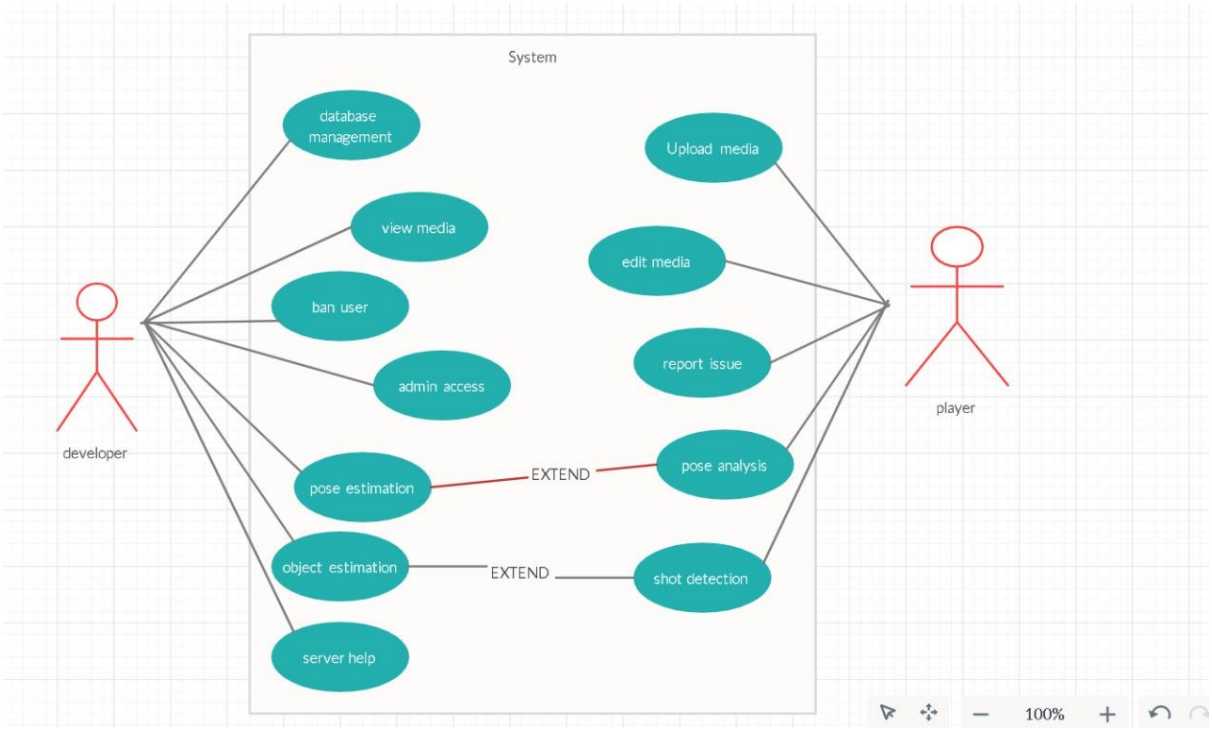
Level 1:



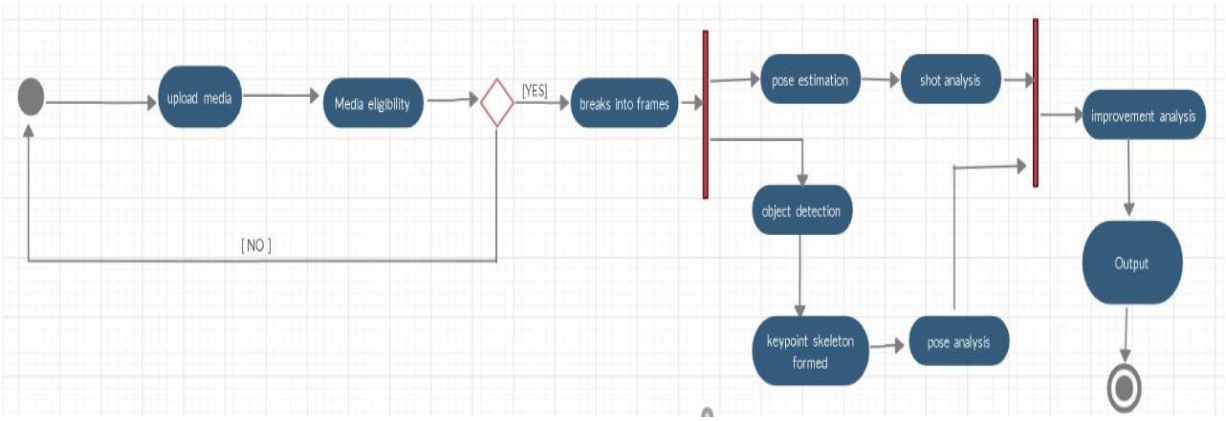
LEVEL 2:


USER
VIDEO IS SENT
1.0 RECEIVED AND TRANSFO-RMED
1.1 IF THERE ARE ERROR
2.0 DEPTH ANALYSIS REPORT/ FILE
FOR RAW ANALYSIS
2.1 FILE SYSTEM ANALYZED
2.2 ASSOCIATE OPEN CV
SHOT ANALYSIS
3.5 SHOT - IS ANALYZED
SYSTEM B
3.4 POSE ESTIMATION
IMAGES BROKEN
3.0 BROKEN INTO FRAMES
IMPROVED DATA
SYSTEM A
4.0 IMPROVE-MENT ANALYSIS
3.1 OBJECT DETECTION
POINT SKELETON IMAGE
3.2 KEYPOINT SKELETON OF FRAME
OUTPUT ANALYSIS FOR USER
3.3 POSE ANALYSIS OF THE FRAME
POSE ANALYSIS
IMPROVED DATA
INDEPTH ANALYSIS OF THE FILE

USE CASE DIAGRAM:



ACTIVITY DIAGRAM:



Appendix C: Issues List

The following are the concerns encountered by the product:

- The limited database available on the internet can pose a problem if any exceptional case, other than what has been fed to the program, is observed.
- The huge amount of database in the backend can make the website considerably slow.
- There are other limitations that we face as students, such as access to high-resolution cameras that this software might need for detection.
- The data of basketball free throw in this experiment were taken from one side only by a web camera, so it was suitable to analyse with 2 dimensional data provided by OpenPose. However analysis of general sports motion requires 3 dimensional data like a tennis or ballet dance, so it is necessary to use 3 dimensional OpenPose or expand 2D data generated by 2D OpenPose to 3D data.