University of Technology, Sydney
Faculty of Engineering and Information Technology

Capstone Project

Basketball Artificial Intelligence Application Group Project Members

Student ID	Name	Major
12590941	Lachlan leslie	Software
12582678	Sebastian Southern	Software
12610782	Benjamin Gillespie	Software

Supervisor: Min Xu

Abstract

Basketball Artificial Intelligence Application – (6cp)

Sebastian Southern - 41030

Supervisor: Dr Xu

Major: Software Engineering

Problem Statement:

Basketball shooting coaching methods are not specific enough as they allow room for human error

The following report outlines a mobile application that allows users to upload videos of their basketball shots where the video is processed into a series of frame by frame key points of the user. These keypoints are then used as input into a series of algorithms that compares the player's pose and movement through their shooting motion against predefined threshold values Based on the output of these algorithms, feedback is generated that the user can make use of to improve their shooting form.

By developing a mobile application, we are able to provide an extremely accessible platform for users to make use of in their daily training routine to continuously improve themselves over time. Furthermore aside from mobile phones being extremely accessible to users now, as the mobile application is cross platform (works for both iOS and Android), we have provided the best opportunity for a user with any device to make use of the product.

Acknowledgements

I would like to say thank you to all of my friends and family who made my experience whilst designing and building this extremely enjoyable, particularly during these extremely difficult and troubling times.

To all those who provided me the possibility to complete this report. A special gratitude I give to our final year project manager, Dr Xu, whose contribution in stimulating suggestions and encouragement, helped me to coordinate my project especially in writing this report.

Furthermore I would like to give special thanks to my teammate, Mr Gillespie and Mr Leslie, who helped to engineer the project into fruition. I would further like to say thank you to Shankar, S., Suresh, R., Talasila, V., & Sridhar, V. for their work on “Performance measurement and analysis of shooting form of basketball players using a wearable IoT system” that was an inspiration for a lot of our work.

Table of Contents

1. Introduction

In basketball, the action of shooting is thought to be, one of, if not the most important skill within the sport. Players will often train themselves in this skill in a variety of ways including practising shooting consistently, logging their shots, filming themselves, watching others play, getting coaching alongside a variety of other methods. The key to improvement in this, like most other skills, is in repetition and awareness of weak areas and knowing what to improve on. A problem that exists with all of these elements of training is that a significant part of what makes them work is humans doing the right thing consistently and making conscious, well-informed decisions of what they need to do to improve.

Throughout 2020, a team consisting of myself, Benjamin Gillepsie and Lachlan Leslie, explored how to improve the training methods for basketball players to improve consistency of player's shooting and their overall skill within the sport. This began by exploring current solutions to training methods within basketball and how these methods have changed over time, with technology becoming a key driver behind these changes. This was then followed with ideating on innovative ways to provide an improved way of training players to shoot a basketball. After coming to a decision on the best way we could provide a strong learning experience for players, we then designed and finally implemented this product.

The main goal of the product is to analyse video footage of basketball players taking shots and provide feedback on how they can improve their shots. This will work by having a video recorded of a player taking a shot which they will then upload. That video footage will then be processed using pose recognition software to capture key points of a person's body from the video footage which can then be analysed. The analysis is then used to determine the quality of the player's shot where we can then provide feedback on the player's from whilst taking the shot and providing suggestions on how they can improve.

To achieve all of this analysis we used a combination of the OpenPose library and custom made services. These services included a pose recognition service to utilise the OpenPose library, a mobile service to allow for a user interface to interact with the rest of the system. An analytics service that would employ our algorithms and decipher the results of the pose recognition service.

We will also talk about the challenges we faced during this project such as the restrictions we had placed upon us by the COVID-19 pandemic and how we adapted to them to be able to collaborate on this project and get the results we needed.

2. Literature Review

2.1 Analysis of Basketball Player Field Goal Shooting Postures for Play Motion Correction Using Kinect Sensor

Given a video has been captured of a player shooting a shot and image recognition software and pose recognition software has been used to process the video. The team will need to understand how to process that data to determine whether the motion used in the shot followed good practices or whether it can be improved. Research from the paper “Analysis of Basketball Player Field Goal Shooting Postures for Play Motion Correction Using Kinect Sensor” (Hsia et al., 2015), discusses the separation of movement into three phases of the shot to make it easier to analyse the motion. This was split into three stages, “pre-shot”, “mid-shot”, and “post-shot”.

It further explains that using the y-index of the hip keypoint the shot phase a player is in can be determined. This allows them to analyse the alignment of each of the body key points in relation to the phase of the shot the player is in. For example, when the Y coordinate in the middle of the hip is at its lowest point, the frame can be defined as the “before_index” shot. When the y-coordinate in the left palm is at its highest, this can be defined as the “mid_index” shot. The final frame for each shot is called the “after_index” shot. These indices are used as indicators for what phase a player is in and are used as reference points to help identify patterns in a player's shooting habits.

In order to then understand what constitutes an effective shot, this analysis would need to be done on a large set of videos containing successful shots as well as missed shots to capture what the expected motion should be. For this the use of video motion (manual marking) could be done, however this would require a lot of time and require the process of taking the captured video, separating it into photographs and marking annotations on each pose. Motion capture analysis could also be performed which would utilise sensors on limbs to acquire the data and more effectively collect the data. It is all possible that the team uses pose detection software to capture the keypoints and although not as effective as motion-capture analysis with sensors, is at a point right now where it will provide enough viable data to use to capture these shots.

2.1.2 Body Shots: Analyzing Shooting Styles in the NBA using Body Pose

In 2017 Panna Felson, author of “Body Shots: Analyzing Shooting Styles in the NBA using Body Pose” (Felson, 2017) performed an analysis of basketball and in particular “jump shots” and how players were able to make them. Concluding her research, Felson found that in order to quantify what led to a “perfect” 3 point shot she needed “Objective measures required trajectories of players’ body pose” (Felson, 2017), not

what was considered a “beautiful jump shot” (Felson, 2017) by experts or spectators. She decided to map the skeleton to 17 attributes that helped her describe the movements of a player when they make a jump and to determine what led to a successful attempt. Further to this, Felson also took game context into account, determining that she needed the system to be able to decide if a shot was “open”, a shot without a defender, or if it was “tough”, a shot where the player was being guarded. After gathering all of this data and deciphering the correlation between the two sets of variables, she created a generalised formula that would attribute to a higher chance of success when taking a jump shot.

By doing all of this Felson has provided a set of criteria to be able to teach an AI system to be able to understand footage of jump shots from the 3 point range using body pose recognition and use that understanding to analyse a shot and determine how to correct it for a higher chance of success. While this system is an amazing step in the right direction to creating a system to be able to analyse and create the perfect jump shot it doesn’t take into account all the factors required to be able to determine a full model of the perfect shot. With the addition of further game context such as, more information about the pass and where it came from plus further contact on the defender and how well they performed, to Felson’s model would create a more perfect understanding of the jump shot.

2.2 Prediction of Basketball Free Throw Shooting by OpenPose

The paper “Predictions of Basketball Free Throw Shooting by Openpose” (Nakai et al., 2019), explores the use of OpenPose in a practical purpose commenting that “OpenPose is a convenient and practical generator of posture data”. It then goes onto further discuss how predictions of free throws were done using pose data but then explains how they identified whether a player was in the initial shooting position or end position based on when the knees were bent the most being the start of a free throw and the time when the hands were the highest to be the end of the free throw. This seems like quite an effective way of determining the player’s stage of shooting, however, it is assumed that players will be bending their knees for this and this may not be the case for some players, particularly novice players to the game.

The paper then goes into further detail explaining how they interpreted the posture of a player shooting identifying the main key points to look at being the knee width and height from before and after, the arm height as well as the angle of the elbows where the player would have a correct posture with a 90degree angle in the before position. Using these details, they were able to identify the probability of a player making a shot before they took it which is quite interesting and potentially something that could be looked at as part of the team’s project. They do explain however, that for the

experiments, all the video footage was taken from one side using a web camera discussing the challenges that would be faced in using multiple angles and depths which shares a similar understanding to the team's belief.

3. Solution

In order to understand what the optimal way to provide users with correct feedback on their basketball shots, the team needed to conduct a high degree of research into understanding all of the movements that composed an effective basketball shot . The core movements of an excellent basketball shot were broken down and include the following 3 stages. The stages have been identified to be a "preparation stage", "execution stage" and "follow-through stage". Within each stage, the posture and movements of a player were able to be broken down through the use of keypoint detection software to identify key body parts and joints. These can then be used during analysis.

3.1 Preparation Stage

The preparation stage is the initial stage of the shot. It is necessary to analyse this part of a player's shot because this phase sets up a player to make a successful shot due to the nature of all the movements involved. Throughout this stage , the player will want to "align the midline of their body to the hoop so that their gaze, the ball and the front of the hoop are in a straight line" (Vickers, 2007). As a player's shot generally begins off of a series of key movements to make it closer to the hoop, balance is a general problem that all players must overcome and become extremely good at inorder to improve their shot accuracy as well as consistency.

"Squaring up" within basketball is used to refer to the ideal stance of the player during the preparation stage. This term reflects the pose of a player where they become aligned with the hoop. This further consists of an alignment of the player with their shoulders and feet pointing towards the hoop whilst also maintaining a high level of stability. The player should aim to have their feet "slightly less than shoulder width" (Knudson, 1993) with their shooting foot slightly more fowards than their back foot allowing the player to maintain more stability throughout the shot. By beginning with this, the player is set up for the rest of the stages of the shot as it decreases the motion required by the entire body whilst taking the shot.

3.2 Execution Stage

The execution stage is the stage of the shot that follows the preparation phase and consists of the player's movements up until the ball just leaves the maximum height of a player's jump. Due to the movement throughout this stage, the player needs to make use of a large number of different muscle groups in order to perform this part of the shot well including the wrist, elbow, shoulder, ankle and knee muscles.

Figure 1

3.3 Follow-through Stage

The "follow through stage" is the last part of the player's shot which essentially finalises the player's motion of the shot being taken. Although it is after the actual execution stage where the player has already taken the shot, it is still an important part of the shot for ensuring the player has optimised the shot. The actions that are involved in this stage are a series of small but critical movements including correctly controlling their shooting hand, control over the fingertips, aligning elbows correctly, releasing the ball as well as the direction of the ball towards the basketball, and the arc of the ball. The player should aim to ensure their fingers, wrist and arm all point towards the basket at a ~60 degree angle when performing the follow through technique (Wissel, 2014)

Figure 2

4. Solution

Below is a detailed design discussion of how the backend systems that enable the product to function. The overall architecture is discussed in detail which is then followed by the backend services.

4.1 Architecture

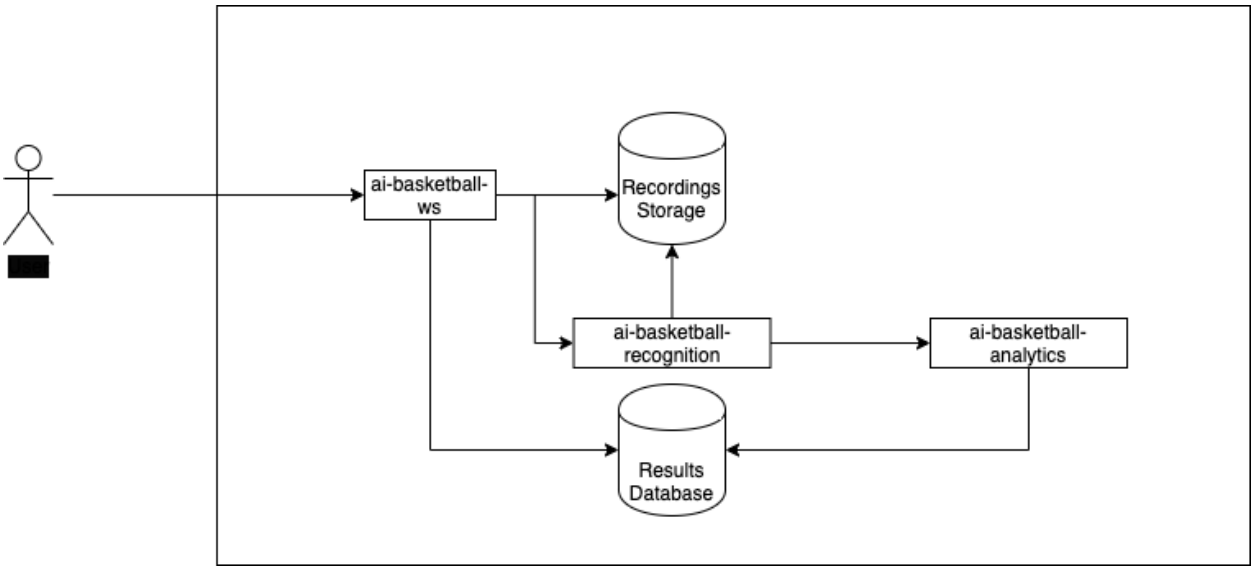


Figure 3

4.1.1 Explanation of architecture

As seen in Figure 3, the architecture displayed provides a simplistic design for taking in uploaded videos from the user which is persisted in a database, then transformed by the Pose Recognition Service and finally analysed by the Analysis Web Server. Once the analysis has been performed, feedback is then persisted into the database where the user will be able to view this from the application.

The design makes use of a microservices architecture where each application has one primary responsibility, each being described further in detail below in section 4.2. This brings in similarities to a 3-tier architecture consisting of a frontend, a webservice and a database, however by splitting the responsibilities into different applications we gain a few advantages. These advantages include:

4.1.1.1 Scalability

Because we are splitting these services, we can take advantage of horizontal scaling (increasing the number of instances of applications running in parallel) instead of using vertical scaling (increasing the CPU/Memory of the host running the application). If we were to run this in a cloud we could scale on demand to meet our needs, ie scale up the instances of an application when we have peak usage, and scale down when there is very little usage which would improve cost efficiency of hosting the application. Furthermore, the most resource intensive application is our pose recognition service as it needs quite a good GPU to run (in our case we were running with a Nvidia GTX 1070). However, as the other applications do not have the same GPU requirements, it would be unnecessary and cost inefficient to host these applications on servers running with the same specs.

4.1.1.2 Availability

If this application was being consumed by a large number of users, a high uptime would be critical and making sure our users are not being impacted by hardware/network/etc failure can not be understated. It is well known that if something can go wrong it will go wrong and by making dividing the applications we can more easily handle failure if a single instance of one application goes down without impacting the user*.

* It is to be noted that this is not currently handled by the applications in its current state but can be quite easily implemented if work on this was continued.

4.1.2 3rd Party Services

Our architecture depends on a variety of other applications/services in order to function including Postgres and S3

4.1.2.1 Postgres

Postgres is an open source object-relational database and is the database that has been chosen to be used as part of this architecture. Alternatives that were discussed were MySQL and Mongodb, however, familiarity with Postgresql made it the more obvious choice over mysql, and the document model of mongodb was not seen as a necessity for our database.

4.1.2.2 Simple Storage Service (S3)

As our architecture relies on videos being uploaded and analysed, we needed an object storage solution to persist these videos where we could then download them from our ai-basketball-recognition service. AWS S3 was chosen as the most appropriate solution to use for this due to a variety of factors including its low cost, backups, easy to use SDK as well as its extremely high uptime SLAs (99.999%).

4.1.3 Architectural Improvements:

As the applications are all communicating with each other using the HTTP protocol it does create some challenges particularly when it comes to error handling. An example of this being, if the pose recognition service died whilst it was processing a video. As we currently respond to the client once the request from the ai-basketball-ws service is sent to the ai-basketball-recognition service and the response from that service just notifies that it has started processing the video, it creates a scenario where an error could happen and a video would be lost in transit. One way to get around this would be by making use of an event-driven architecture and making use of a messaging service such as AWS Simple Notification Service (SNS) or Apache Kafka. By adopting this, rather than the services communicating with each other over HTTP, they could instead produce messages to a topic where a message broker will handle it, and when the service responsible for handling the next step of the process is ready to handle the next request it can then consume the message, process it, and produce it to continue the flow of events. This enables services to only process messages when they are able to (ie are not already maxed out on CPU/memory) and if an application is restarting, once it has fully restarted it can then pick up the message from the broker preventing the possibility of messages being lost.

4.2 Backend Services Detailed Solution

4.2.1 AI Basketball Web Server

Github Repo: https://github.com/Jahmilli/ai-basketball-ws

Description:

The ai-basketball-ws service is the entrypoint for requests within the product. It provides a REST API with a number of routes that the client can make requests in order to upload videos or get information about previously uploaded videos. This server communicates directly with S3 to upload videos to, postgres to persist and get data from as well as the ai-basketball-recognition service to continue with the process of analysing videos.

GET "/v1/video"

Expected Query Parameters:
userId: string

Responses:

- Status code: 200
  Body: Video[] (See Section 4.3 for schema)
- Status code: 400
  statusMessage: "Bad Request"

Sequence Diagram:

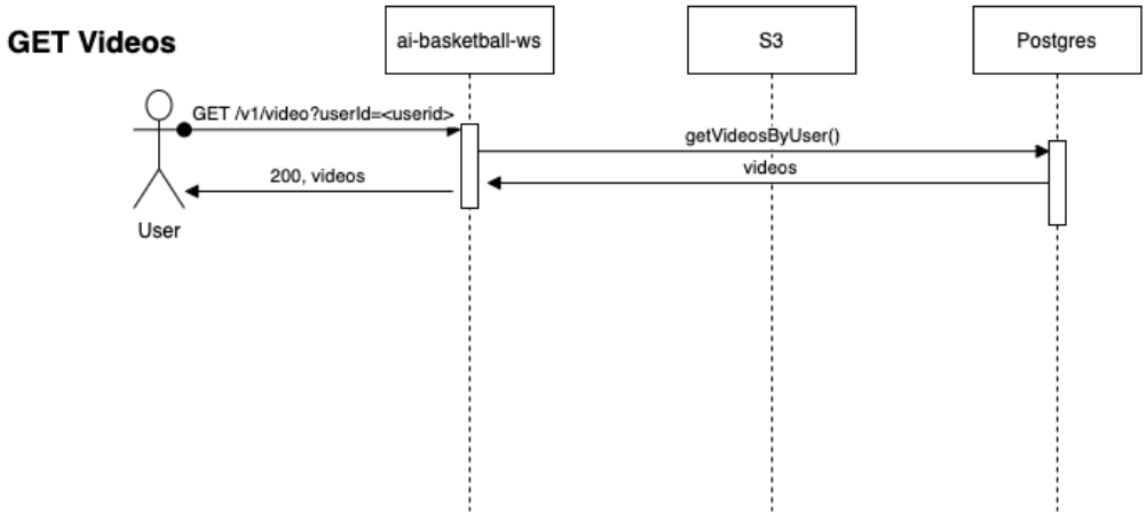


Figure 4

Handler Description:
This is used to retrieve an array of videos from the database for the provided userId.
If there are no videos in the database an empty array will be returned.
If the user does not exist, a 404 will be returned.
If “userId” is not passed in as a query parameter, the server will respond with 400 “Bad request”.

POST “/v1/video/create”

Expected Body Parameters:
userId: string;
name: string;
Description: string;
angleOfShot: string;
typeOfShot: string;
uploadedTimestamp: string;

Responses:

- Status code: 200
  video: Video[]
- Status code: 400
  Status message: “Bad Request”

Sequence Diagram:

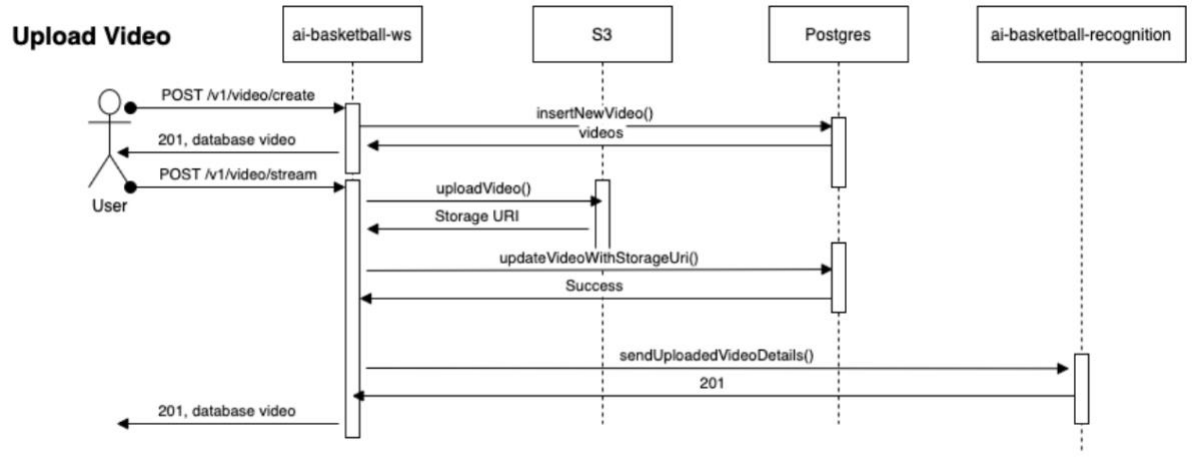


Figure 5

Handler Description:
This is used to create a “Video” entry in the database using the data provided by the user from the application. This is the first part of the 2 step process for uploading a video and is necessary to perform here as we need to create a unique identifier which will be used as the video name that is stored in S3. Once the database entry is saved a successful response is returned back to the client.

POST “/v1/video/stream”

Expected Body Parameters:
userId: string;
name: string;
Description: string;
angleOfShot: string;
typeOfShot: string;
uploadedTimestamp: string;

Responses:

- Status code: 200
  Body, Video[]
- Status code: 400
  Status message: “Bad Request”

Handler Description:
This route receives a request as a “multipart upload” which enables streaming a video to the web server. By using a multipart upload type of request we can fragment the video into different “parts” of a set size which is necessary as the file sizes can be quite large and posting a single file would create issues. The client sends the request with a field containing the id of the video entry in the database. Once the server receives the request it then creates a stream to S3 allowing us to stream the video from the client to an S3 bucket. Once an end of stream event is emitted from the S3 client, we close the stream and save the URI of the object in S3 to Postgres. This is then finalised by sending a request to the pose recognition service and responding back to the client with a successful message alerting the user that the video has been uploaded.

4.2.2 AI Basketball Recognition Server

The AI basketball recognition server was developed by my fellow group member Mr. Leslie. In order to view the details surrounding the work, please see Appendix B.

4.2.3 AI Basketball Analytics Server

The AI basketball recognition server was developed by my fellow group member Mr. Gillepsie. In order to view the details surrounding the work, please see Appendix C.

4.3 Database Design

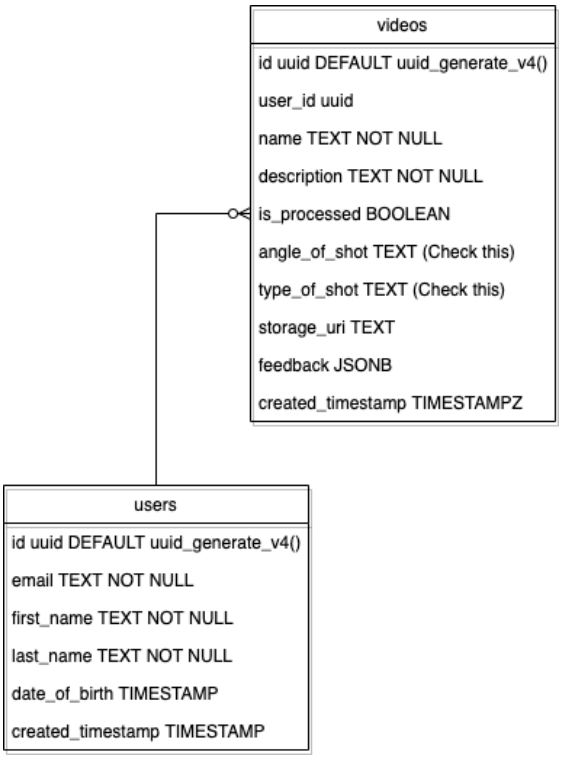


Figure 6

As the application functionality is quite simple at the moment, we were able to develop this with a relatively simple database relationship. As seen in Figure 6, for each user we have a one to many relationship of videos. Each user will be uniquely identifiable and not share their list of videos with any other user, although future versions of the program may make functionality like this available to provide a more interactive community for our users. Each video will also be uniquely identified and have a “storage_uri” to be able to retrieve the video from the S3 bucket by other services.

5. Mobile Application Development

A critical part of putting together this product was building a user interface to allow users to access the analysis easily. The idea of putting together a web application was discussed due to the familiarity that the team had with building websites. However, given the use case and the need for accessing a camera it was not deemed to be the most effective choice and so the idea of a mobile application was agreed upon by the team. Furthermore, as the team had both Android devices and iOS devices, and were keen on using the application after the completion of the project, we set out our own requirement of making sure that it was built with support to run on both types of devices.

Although we were more familiar with developing Android applications using Android Studio, the team had a strong background in Javascript and Typescript as well as React, a framework developed by Facebook for building Web applications. With this knowledge, we decided React Native, a framework similar to React, however, used to develop cross platform mobile applications would be the best choice to develop the mobile application with.

5.1 Requirements

Before beginning development of the application, the team discussed a series of requirements around the mobile application to ensure we all had a common understanding of what we would be attempting to develop, these included the following:

- Must be cross platform
- Must support users and user creation
- Must allow users to select the type of shot the player is taking (ie “3 point shot”, “Free throw” etc)
- Must allow users to select the angle of the shot the player is taking (“Behind the player”, “In front of the player”, “Side on”, etc)
- Must allow users to record a video for up to 8 seconds
- Must allow users to submit videos to be processed
- Must allow users to view all the videos they have submitted
- Must allow users to view feedback on the videos they have submitted

It is to be noted that from the above requirements, all features were able to be created aside from user creation where challenges with the authentication were faced. All of these requirements were captured in Trello where the team was able to track progress of the work each individual was doing throughout the lifetime of project development, see Appendix D. Furthermore, the team practiced made sure to follow Agile practices throughout the sprints of the project where we held standups as well as retrospectives to better understand our progress and our thoughts on what we felt we were doing right as well as what we were doing wrong. Practicing Agile, also meant that some of the requirements were added in as well as removed during the project development where tasks such as only allow the users to record the video for up to 8 seconds was added in the later stages of development.

5.2 Designing the Application

After the requirements were completed, the team needed some basic designs that really captured the requirements in visual form. The team used Balsamiq, an online wireframe editor to generate some low-fidelity wireframes that created a common understanding of what the team wanted to build and would provide an extremely good starting point for development, see Appendix E.

5.3 Implementation

The implementation of the mobile application was quite smooth for the most part, where development was fairly standard as a majority of the features for the user interface were purely visual with screens displaying data retrieved from the ai-basketball-ws service. However, by using Typescript, the team was able to make use of the static typing provided by the language which made development much faster due to it catching out any errors in the types used for functions as well as data and so a combination of these enabled this streamlined development. See Appendix F for a selection of the screenshots from the application displaying the general user experience a user would have.

6. Methodology

The main shooting technique can be separated into three different stages; preparation, execution and follow through. Within each stage of the shot, a basketball player will position themselves in order to make an optimal shot. The preparation stage focuses on “squaring up” before shooting the ball. The execution stage is then about the actual execution of the shot focusing on the alignment of the shooting arm between the preparation stage and shooting the ball. The follow through stage is the final part where the player releases the ball. All of these stages can be broken down into threshold values to determine the quality of a player’s shot.

6.1 Stages of the jump shot

This product brings together the shooting postures of a basketball player within the different stages of the shot as mentioned above. Therefore, the differentiation and identification of each stage from within a video is essential to accurately analyse the posture of a player through the motion of shooting a basketball and is displayed below in Figure 7. Based on this, each stage of the shot should therefore be identifiable via the

key points generated from Openpose, which then allows us to track the y-coordinates of the middle of the hip (Ymidhip) as well as the right or left wrist ($Ywrist_R$ or $Ywrist_L$).

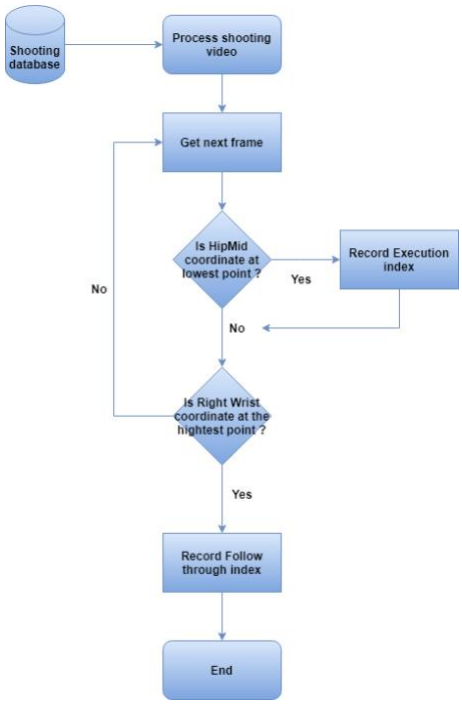


Figure 7

When the y-coordinate of the middle-hip reaches the highest pixel value (lowest point in motion), the frame at this point is recorded and defined as the ‘eIndex’ also known as the “execution index”. When the y-coordinate of the shooting hand, which is either the left or right wrist ($Ywrist_L$ or $Ywrist_R$) reaches the lowest pixel value (highest point in motion), the frame at this point is then recorded and set as the ‘fIndex’ also known as the “follow through index”. From the ‘eIndex,’ including the last 10 frames that lead to it; the interval between them is seen as the preparation phase and known as phase P. All the frames between the eIndex and the fIndex are then understood to be the execution phase and are referred to as phase E. From the ‘fIndex’ including 20 frames proceeding it; the interval between them is the follow through phase and is referred to as phase F. All of the phases as well as the differences between them throughout the motion of performing a shot is shown in figures 8 and 9.

Figure 8

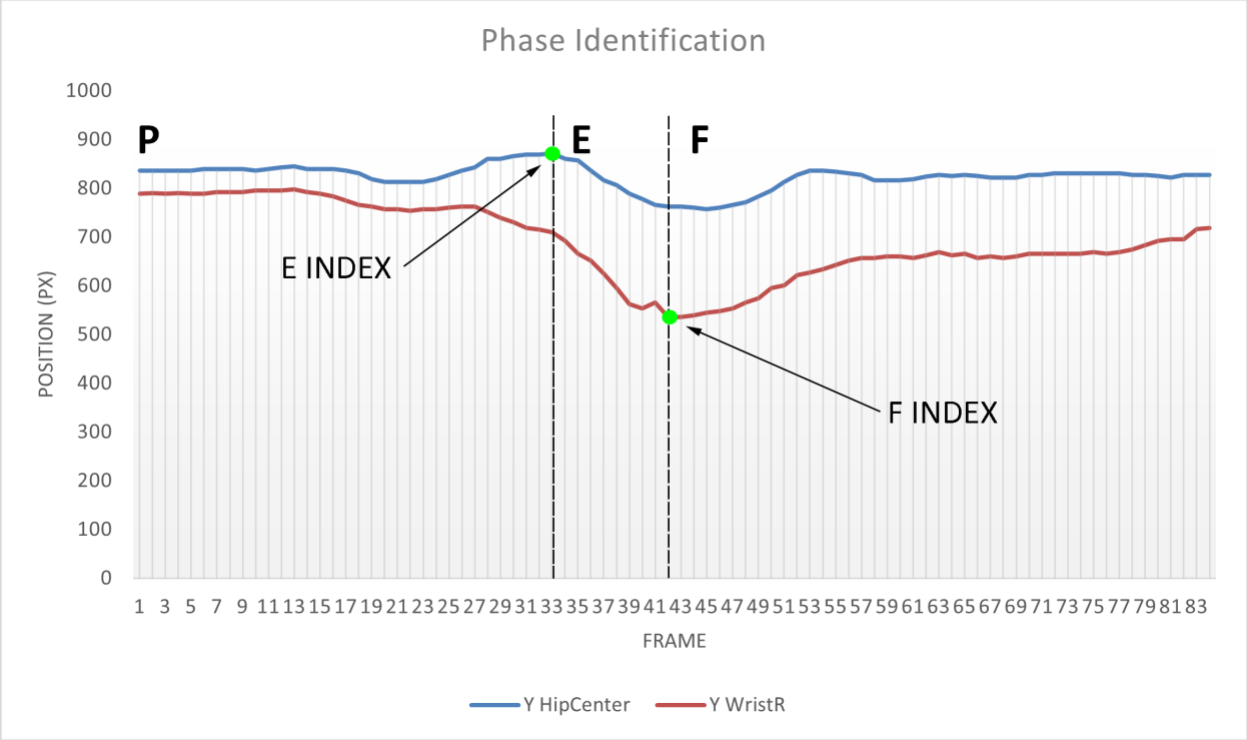


Figure 9

$Y \rightarrow 0$, keypoint is in an upward motion

$Smallest\ Y$ =keypoint reached peak of motion

$Y \rightarrow Resolution\ width$, keypoint is in a downward motion

$Highest\ Y$ =keypoint reached bottom of motion

6.2 Algorithms

Preparation phase

The posture of the athlete examined during this phase is the alignment between the feet and shoulders during the motion of the square up as shown in figure X. The open pose technology estimates the coordinates of the right and left shoulder; RShoulder($Yshoulder_R$, $Xshoulder_R$) & LShoulder($Yshoulder_L$, $Xshoulder_L$) as well as the right and left feet through the ankles; RAnkle($Yankle_R$, $Xankle_R$) & LAnkle($Yankle_L$, $Xankle_L$). Using the distance between two points equation, the distance between the athlete's shoulders and distance between feet is obtained; equations (1) & (2). From this, the difference between shoulders and feet is calculated from equation (3).

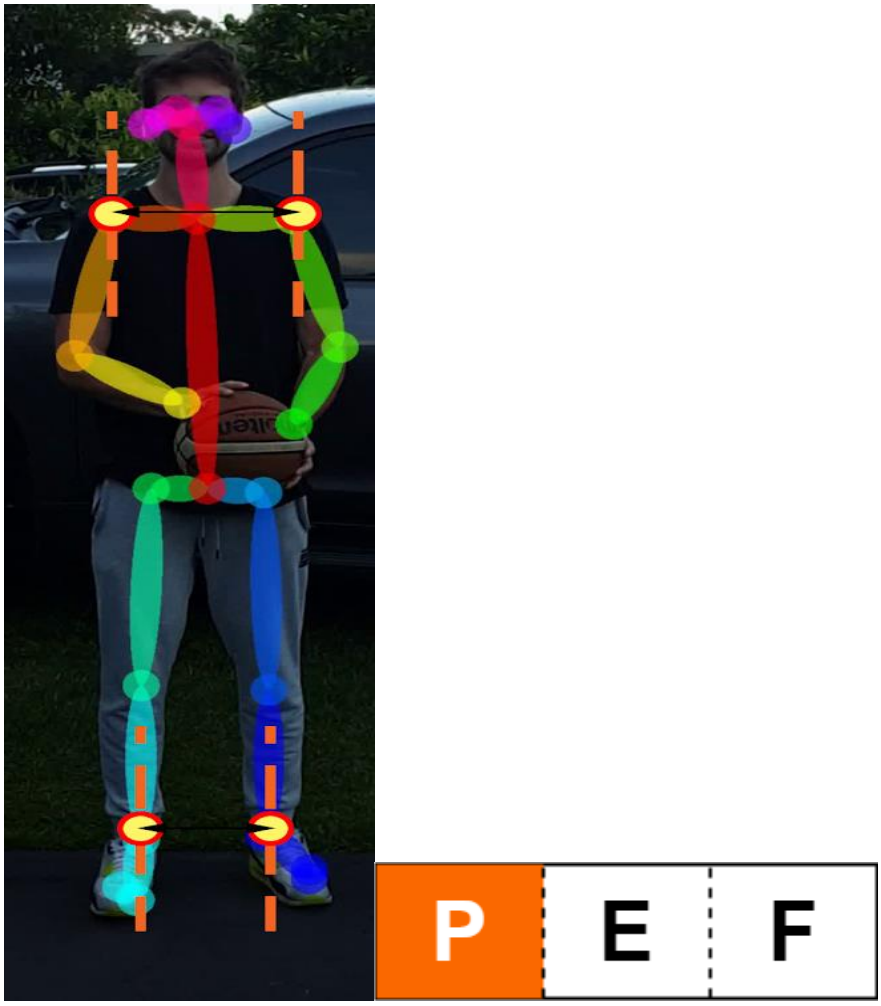


Figure 10

$$D_{RShoulder\ LShoulder} = \sqrt{(Xshoulder_R - Xshoulder_L)^2 + (Yshoulder_R - Yshoulder_L)^2} \quad (1)$$

$$D_{RAnkle\ LAnkle} = \sqrt{(Xankle_R - Xankle_L)^2 + (Yankle_R - Yankle_L)^2} \quad (2)$$

$$D_{SF} = |D_{RShoulder\ LShoulder} - D_{RAnkle\ LAnkle}| \quad (3)$$

Using the D_{SF} values from the 10 frames leading up to the 'eIndex' or execution index, the average D_{SF} is calculated to find the average alignment between the shoulders and feet during the preparation phase. If the avg D_{SF} is less than or equal to the multiAxisThreshold value, equation (4), alignment between the shoulders and feet is verified.

$$D_{SF} \leq Threshold_{MultiAxis} \quad (4)$$

Execution phase

The AI basketball recognition server was developed by my fellow group member Mr. Leslie. In order to view the details surrounding the work, please see Appendix G.

Follow through phase

The AI basketball recognition server was developed by my fellow group member Mr. Gillepsie. In order to view the details surrounding the work, please see Appendix H.

Challenges

There were a number of challenges that were faced by the team throughout the lifetime of this project, including challenges that impacted the team's ability to develop as well as test the work done as well as challenges that were an inherent part of the problem we were trying to solve as well as through the solution that was being implemented.

Challenges impacting the team

COVID-19

COVID-19 has posed to be extremely challenging putting everyone in far more difficult and abnormal situations. The team has faced many difficulties with this and social distancing, particularly with going out and testing the software that has been developed due to the lockdowns that have been in place, as well as the less willing people are to meet in person. This has resulted in the tests taken with the software limited to just the team and a select number of close friends/family. However, the team is extremely keen on testing out the application with players from basketball teams at a later date in order to get more relevant and spread feedback in order to improve it.

Insufficient access to hardware

As Openpose performs extremely intensive computations, it requires quite a good graphics processing unit (GPU) in order to run which is common for most machine learning software. This created a massive bottleneck within the team as only one member had a graphics card capable of running Openpose who had a Nvidia GTX 1070. However, early on the team did look for alternative ways to get around this which

included looking into other keypoint detection software such as Detectron2 a Facebook developed image processing toolkit. However, when it came to running it in CPU mode, it took roughly ~5 minutes to process a 2 second clip using a 2019 IMac and so was believed to be unusable for testing. An alternative was then to run the processing in AWS using the Elastic Cloud Compute (EC2) service to run virtual machines that were running with Nvidia GPUs. Yet this created a problem of cost where we had to pay to run these services and so the team developed with a strong dependency on Ben's computer throughout the span of the project.

Challenges inherent to the solution

Lighting of the video

One of the challenges that the team was aware would be faced with the solution we were building was recording with proper lighting as the quality of the video recordings needed to be quite good in order for the keypoint detection of Openpose to be usable. In particular, filming during the dark was believed to be the most significant problem we would face with this and so the team determined that allowing the application to access the camera's flash would be a suitable short term fix to this aside from providing some information on the application alerting the user of optimal conditions to record in. However, the team was unable to develop this feature in time, it remains in the backlog of tasks for any future development to be done.

Angles

Ensuring the algorithms are able to handle different angles that video is recorded from was also considered as one of the most significant problems that would be faced with the solution. The problem with this was that different angles would result in the difference in keypoint movement being quite different, even if the player's movement was almost identical in each recording. To get around this, the player is provided a set of options when interacting with the application prior to recording the video that allows them to choose the angle they are filming the player from. This is then passed as metadata when the video is uploaded and can be accessed during analysis.

Detecting type of shot

As the feedback for each type of shot can be quite varying, it is important that what type of shot the player is making is considered during analysis. Similar to the current work around for handling angles, the user is presented with a set of options from the mobile application where they can choose what type of shot the player will be making which is

then passed in as metadata and can then be accessed during analysis. Although we don't currently handle analysing different types of shots, it is believed that with this data, we would just need to generate a new set of threshold values for each type of shot and then compare the shot to the relevant thresholds.

Experimental Results

Setting of thresholds

- Talk about how thresholds are based off shots from a particular high level/skill shooter
- Adhering to these thresholds would teach you to shoot in the same style/form as him.
    - Using the same methodology, this could be repeatable for any player's jump shot form/style
- Actual thresholds in the application for a more casual pick up are much less strict
    - Probably not necessary to state what those values are but we could make something up

MultiAxisThreshold is set at 15 px
SingleAxisThreshold is set at 20 px
CosineThreshold is set at 165 degrees

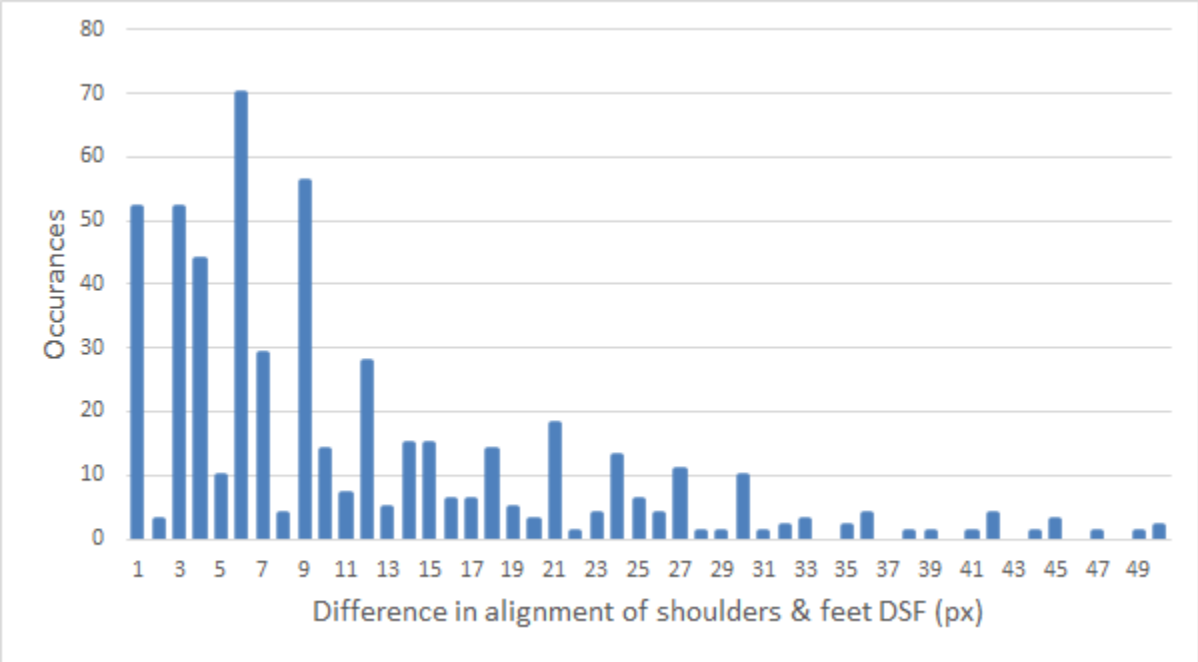


Figure 11

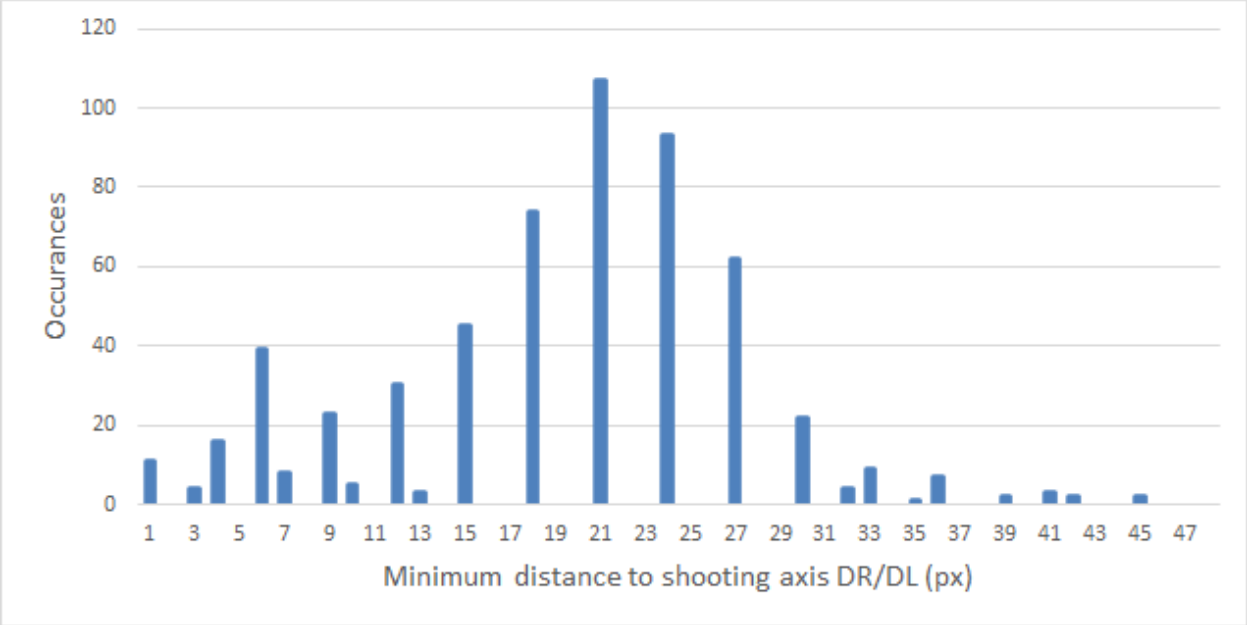


Figure 12

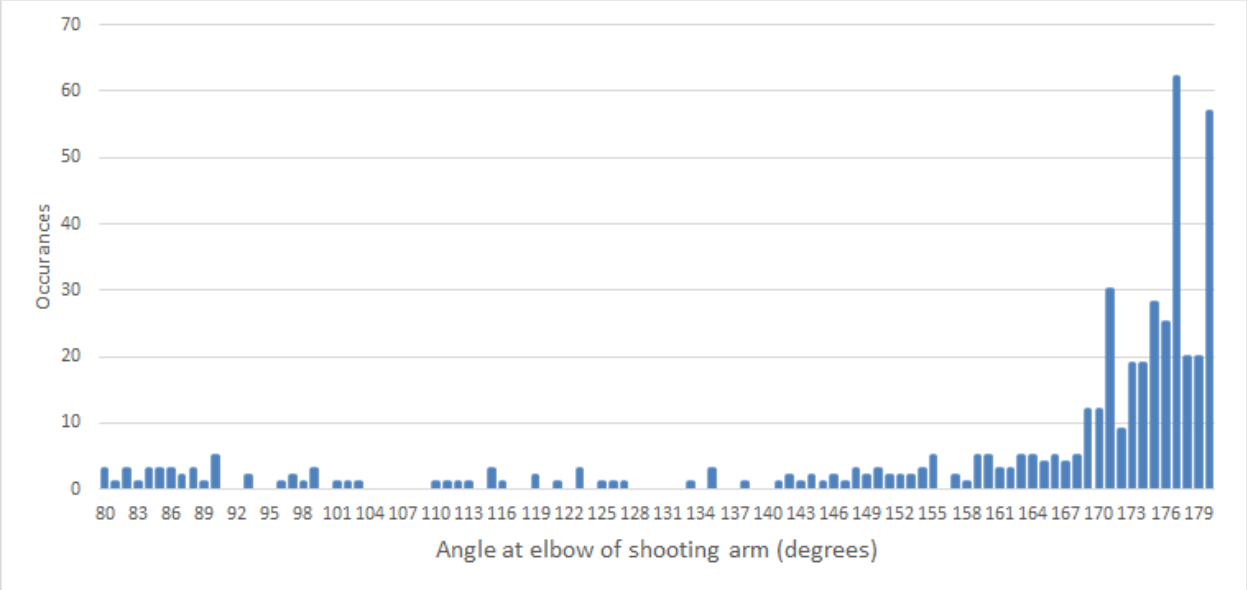


Figure 13

Feedback produced

Conclusion

Basketball is an extremely popular sport in many countries with a massive following behind it. It is consistently seeing an increase in people both watching and playing the sport and for players wanting to succeed within the sport and play in the highest leagues possible, they need access to the best methods of training. Our team has developed an application that provides an easily accessible means of providing advanced training methods for players wanting to improve their shooting within the sport of basketball. This application has the capability to see further improvement to not only improve the whole application as a whole but improve capabilities behind the analysis and feedback with an easily adaptable backend. This project has been an extremely exciting and inspiring journey from start to finish allowing the team to put forward a variety of thoughts and improvements into building something that we had never considered before allowing all of the skills that we have learned throughout our years at UTS to come together to research, design and develop a new way to provide better coaching abilities to basketball players.

Appendices

Appendix A: Communication Log

Project Title:	**Basketball Artificial Intelligence Group Project**		
Student Name:	Lachlan Leslie Benjamin Gillespie Sebastian Southern	**Supervisor Name:**	Min Xu
Date	**Event**	**Topic of Communication**	**Outcome**
3/4/2020	Email	Request for meeting to discuss project context	Meeting setup for discussion of project
3/4/2020	Online Meeting	Project context meeting	Discuss initial context of the project and direction we wished to take it

8/5/2020	Online Meeting	Discuss project management plan	Agreed on project management plan requirements
7/5/2020	Email	Discussion on requirements of group project and individual submissions	Agree on individual submissions with same content excluding sections
29/5/2020	Online Meeting	Get feedback on project management plan	Feedback on project management plan
3/8/2020	Online meeting	Weekly meeting to discuss project progress	Decision to continue use of openpose
6/8/2020	Online Meeting	Discuss final project direction	Feedback on project direction
10/8/2020	Online Meeting	Weekly meeting to discuss project progress	Final architecture decisions
17/8/2020	Online Meeting	Weekly meeting to discuss project progress	Code review of application code
24/8/2020	Online Meeting	Weekly meeting to discuss project progress	Code review of pose service
31/8/2020	Online Meeting	Weekly meeting to discuss project progress	Architecture review and alterations based on changed requirements
7/9/2020	Online Meeting	Weekly meeting to discuss project progress	Preparation phase algorithms review

14/9/2020	Online Meeting	Weekly meeting to discuss project progress	Mid shot phase algorithm review
21/9/2020	Online Meeting	Weekly meeting to discuss project progress	Post shot phase algorithm review
28/9/2020	Online Meeting	Weekly meeting to discuss project progress	Progress update on pose and analysis services code
5/10/2020	Online Meeting	Weekly meeting to discuss project progress	Review of analysis service code
12/10/2020	Online Meeting	Weekly meeting to discuss project progress	Review of new Application code
19/10/2020	Online Meeting	Weekly meeting to discuss project progress	Review of pose service code
26/10/2020	Online Meeting	Weekly meeting to discuss project progress	Review of experimental results
28/10/2020	Online Meeting	Project discussions + final report structure	Advice on project direction

Appendix B: Pose Recognition Server Implementation

ai-basketball-recognition

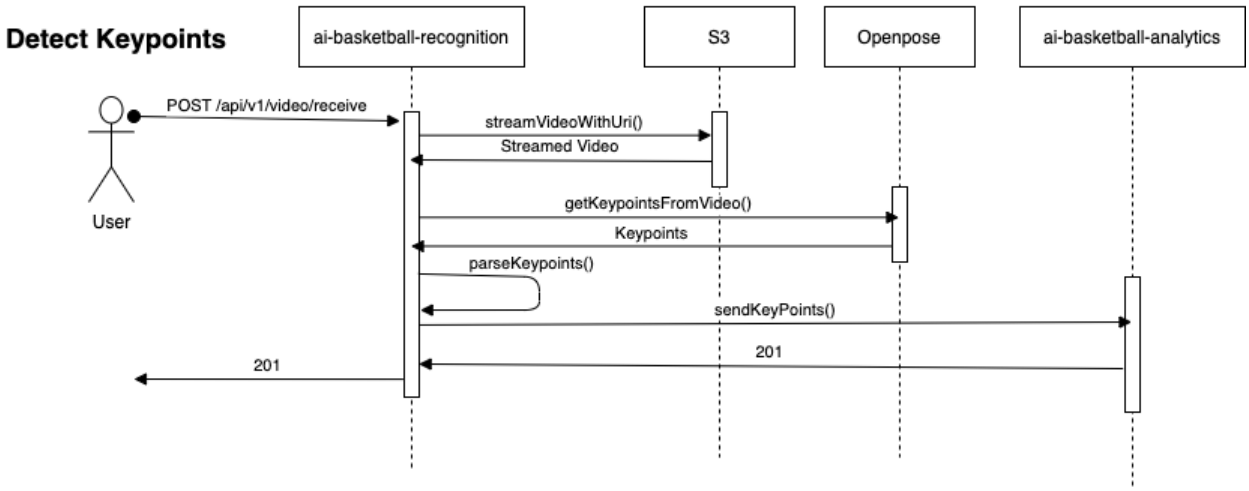
Github repo: https://github.com/lachlanleslie1995/ai-basketball-pose-recognition

Description:
The ai-basketball-recognition service is responsible for the processing of videos to generate key points. It communicates directly with S3 and contains the Openpose binary to generate the keypoints and then sends them to ai-basketball-analytics.

POST “/api/v1/video/receive”

Expected Body Parameters:
id: string;
videoUri: string;

Responses:
Status code: 201



Description:
This is responsible for downloading a video from S3 using the provided S3 URI and then processing it using Openpose. Openpose will then output a series of keypoints in format where we then parse those keypoints into an object containing keys where each key is a specific point of the body containing an array of key points for each frame of the video. We then send those key points onto the analytics service for the final part of the analysis to occur.

Appendix C: Analytics Server Implementation

ai-basketball-analytics

Github repo: https://github.com/Jahmilli/ai-basketball-analytics

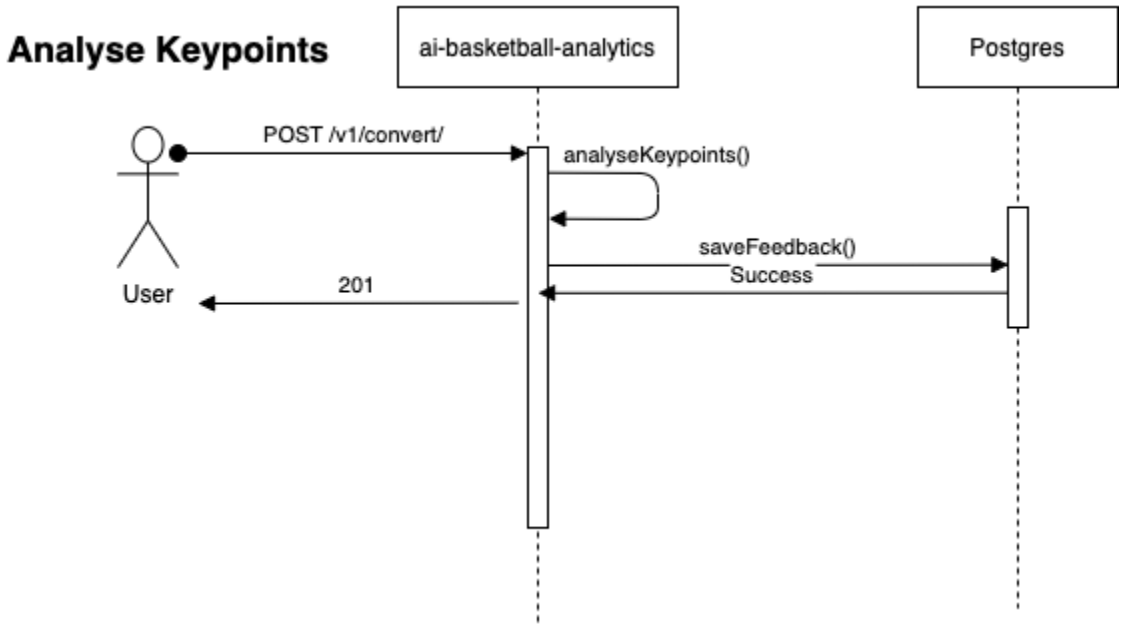
Description:
The ai-basketball-analytics service is responsible for analysing videos and persisting the feedback into the database.

POST “/v1/convert”

Expected Body Parameters:
id: string;
keypoints: Object; // This contains all the keypoints for the processed video

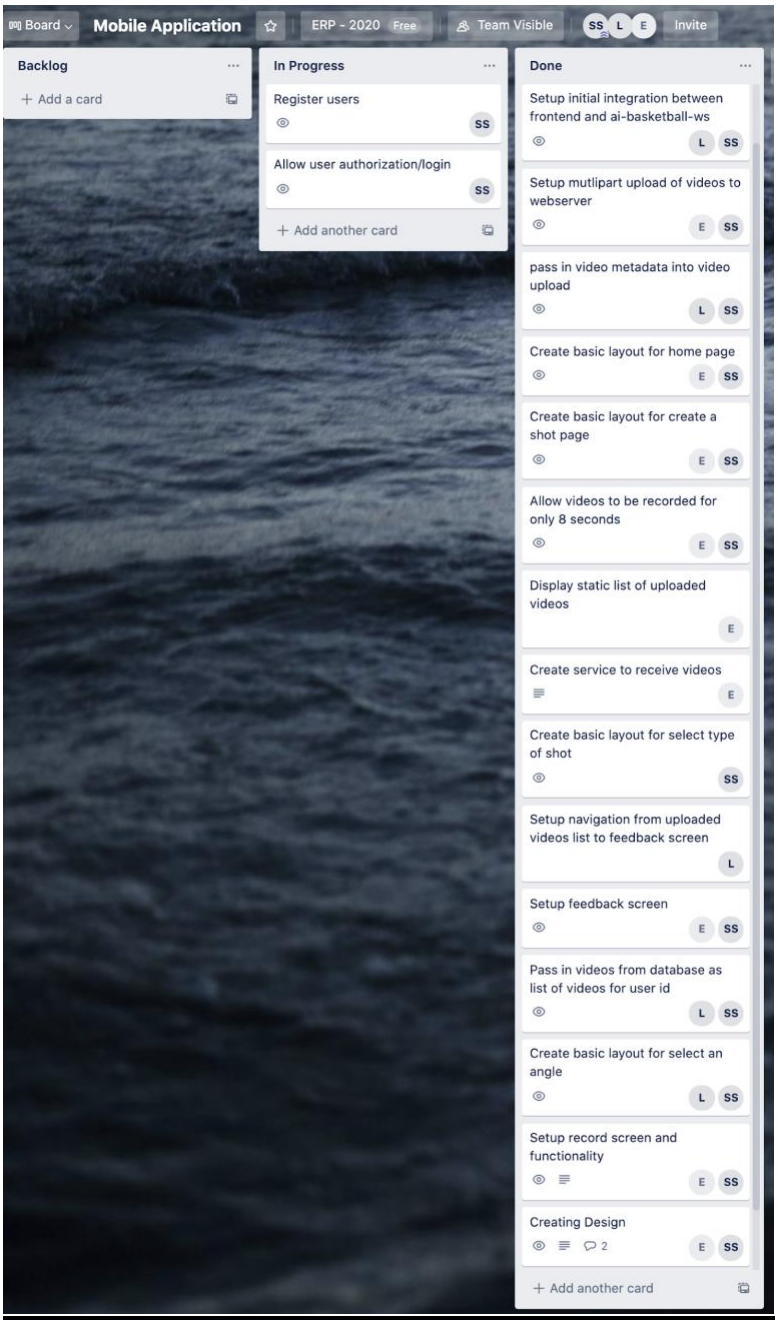
Responses:

- Status code: 201

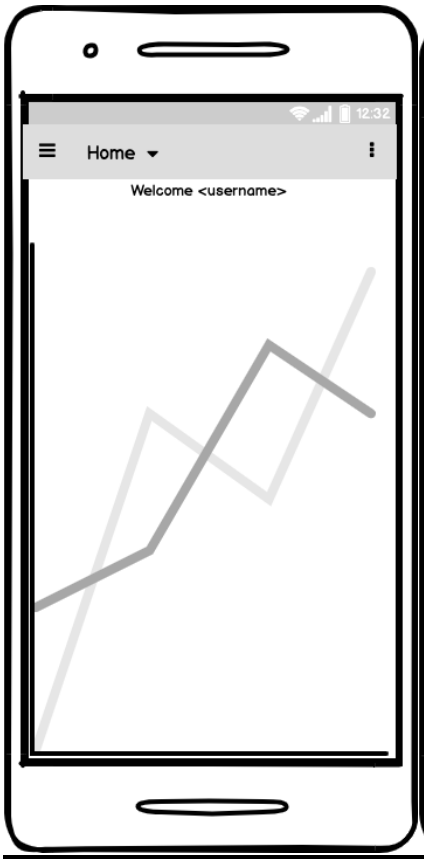


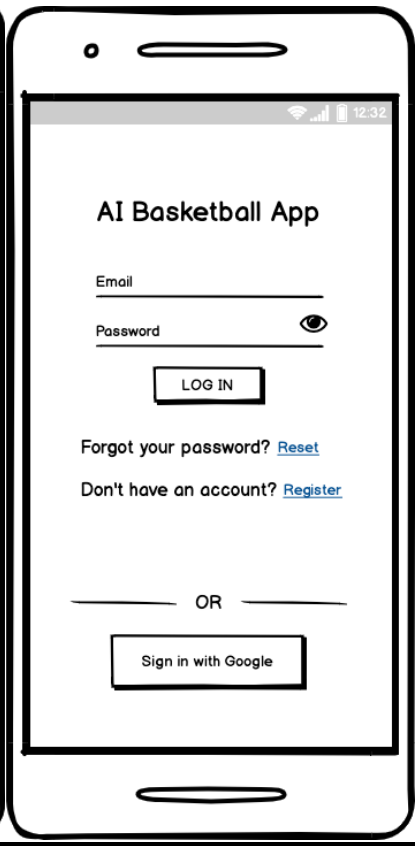
Description:
This is responsible for handling all of the analytics of the key points that are sent in the request to this server using the algorithms described in section . Following the analysis of the keypoints, feedback is generated and then persisted into the database. Once this has been persisted in the database, the client is now able to view the feedback for the analysed video.

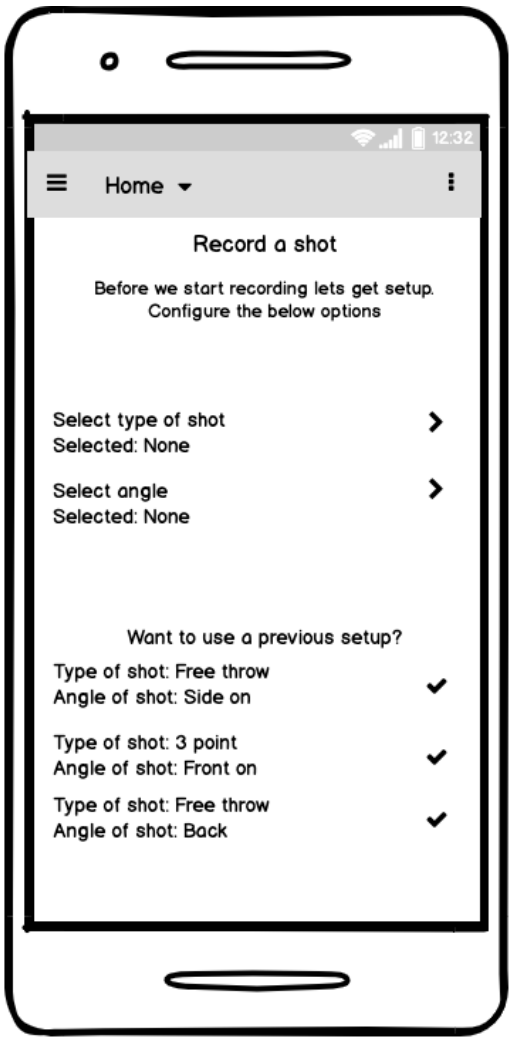
Appendix D: Mobile Application Trello Board

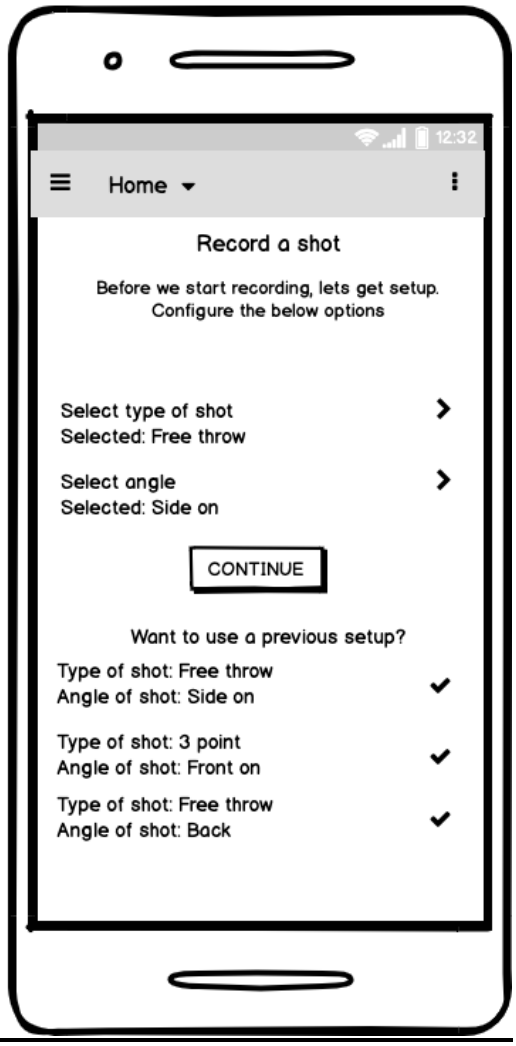


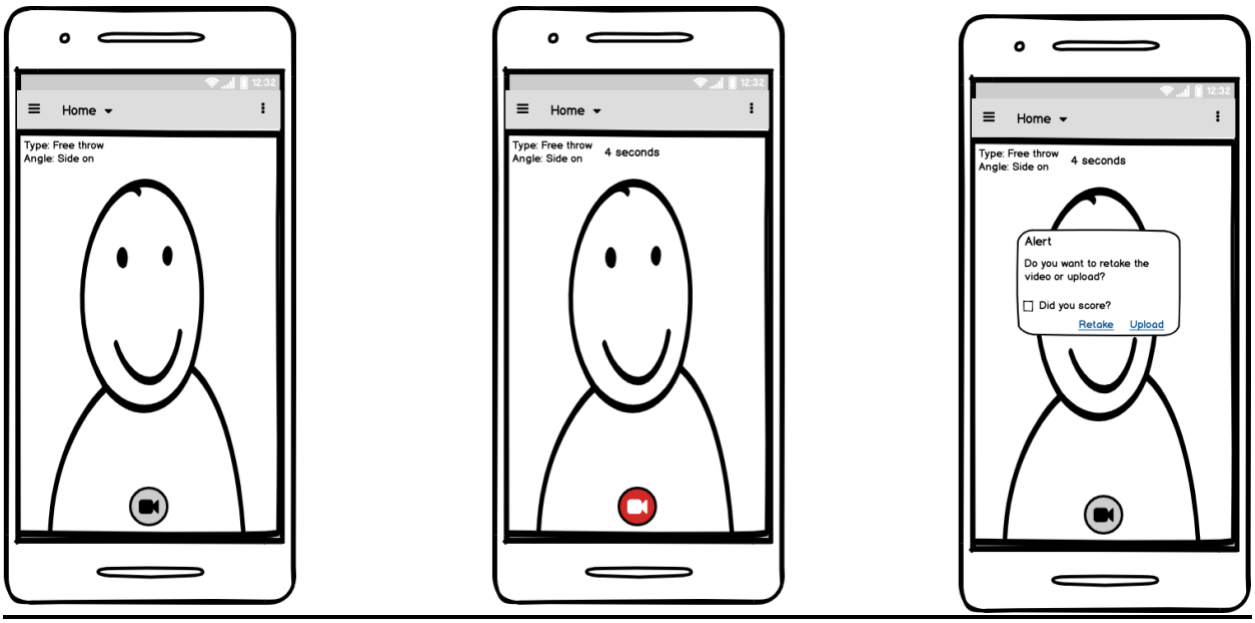
Appendix E: User Interface Low Fidelity Wireframes

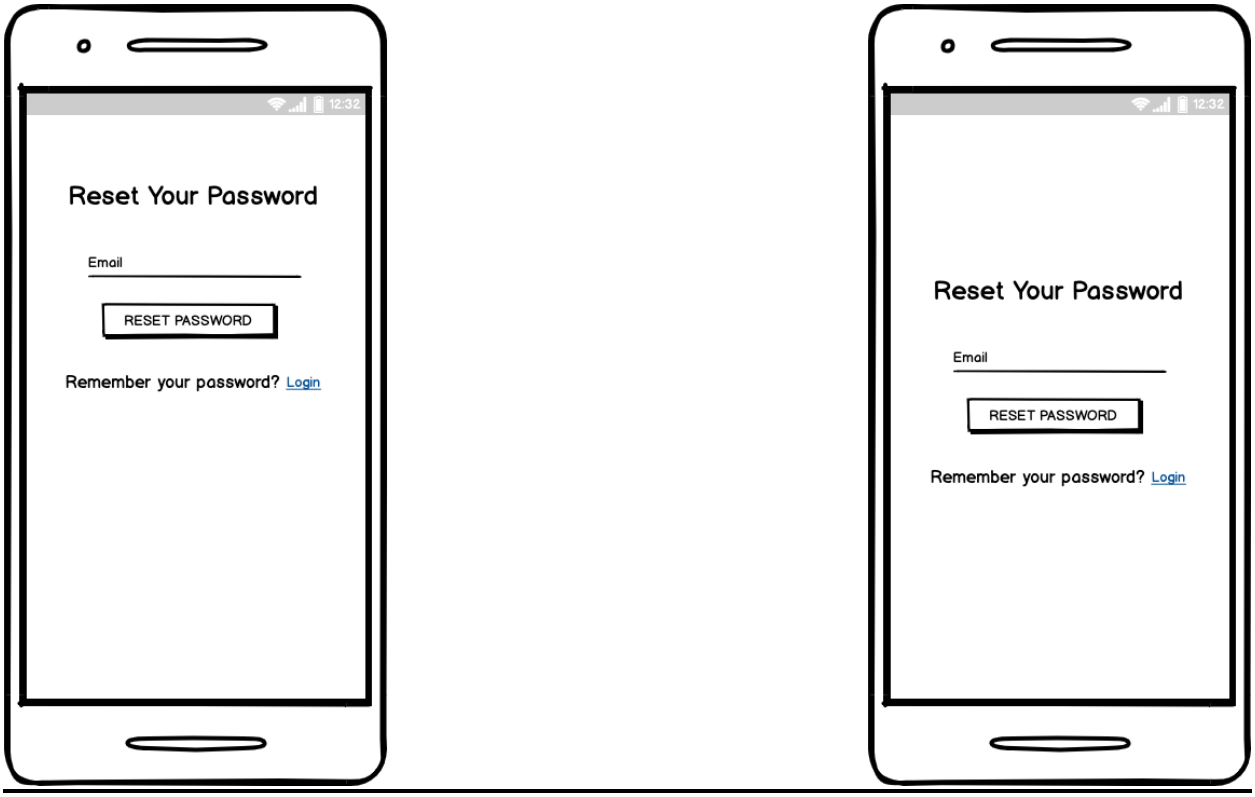


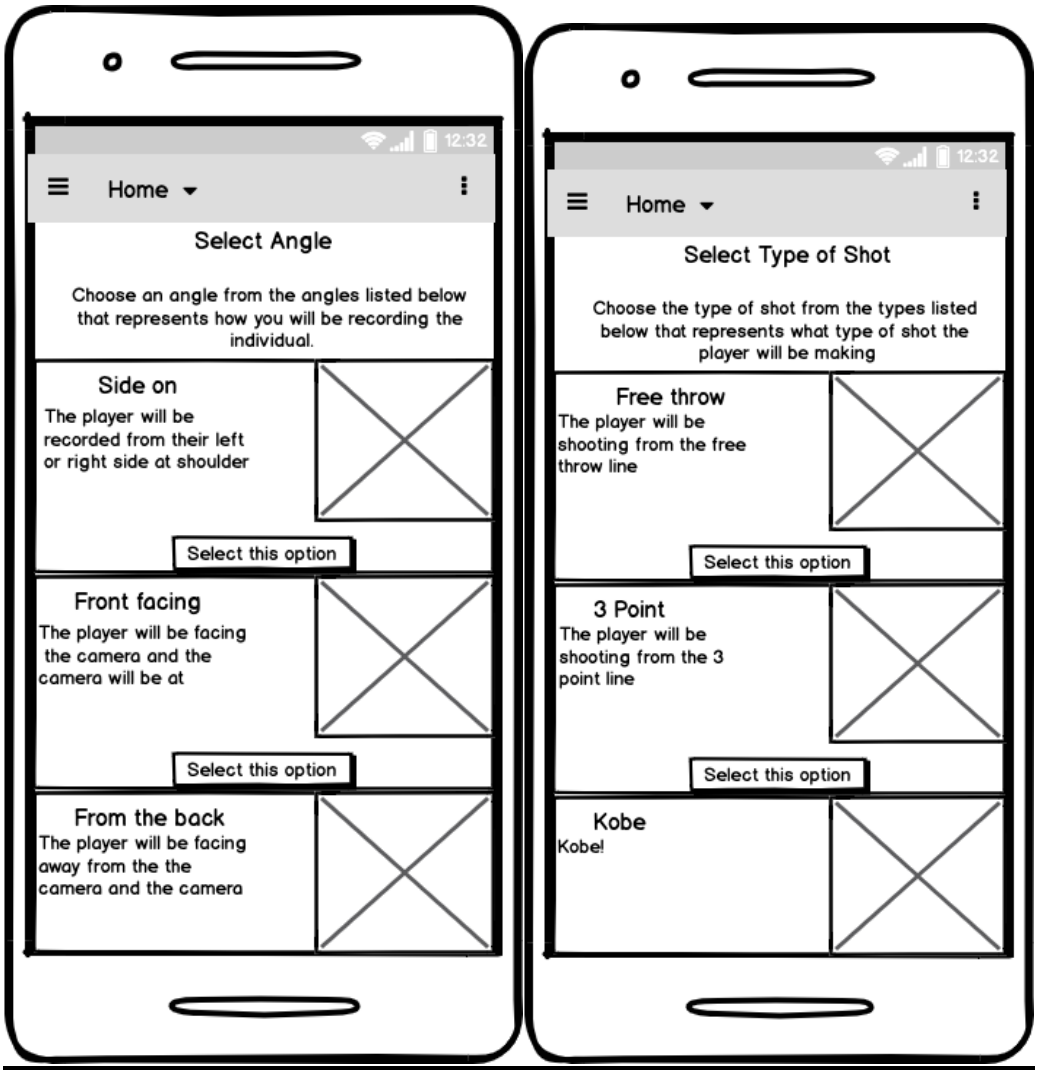


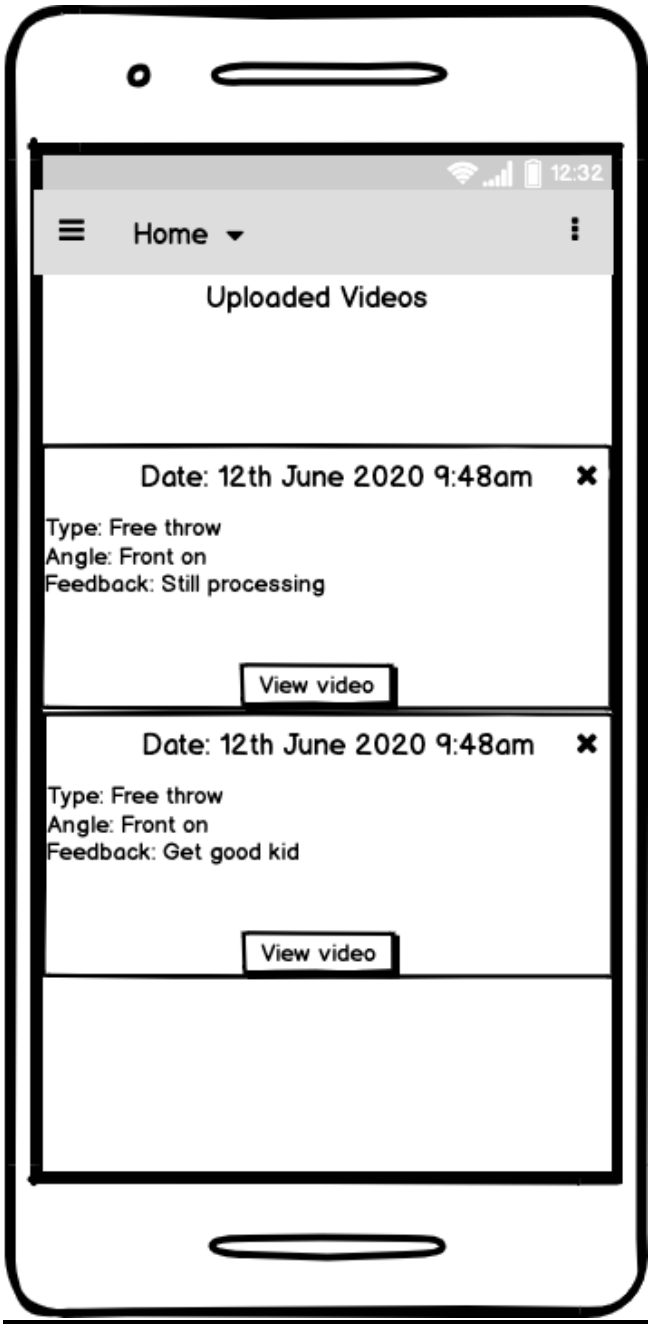

12:32
Home
Record a shot
Before we start recording lets get setup.
Configure the below options
Select type of shot
Selected: None
Select angle
Selected: None
Want to use a previous setup?
Type of shot: Free throw
Angle of shot: Side on
Type of shot: 3 point
Angle of shot: Front on
Type of shot: Free throw
Angle of shot: Back


12:32
Home
Record a shot
Before we start recording, lets get setup.
Configure the below options
Select type of shot
Selected: Free throw
Select angle
Selected: Side on
CONTINUE
Want to use a previous setup?
Type of shot: Free throw
Angle of shot: Side on
Type of shot: 3 point
Angle of shot: Front on
Type of shot: Free throw
Angle of shot: Back

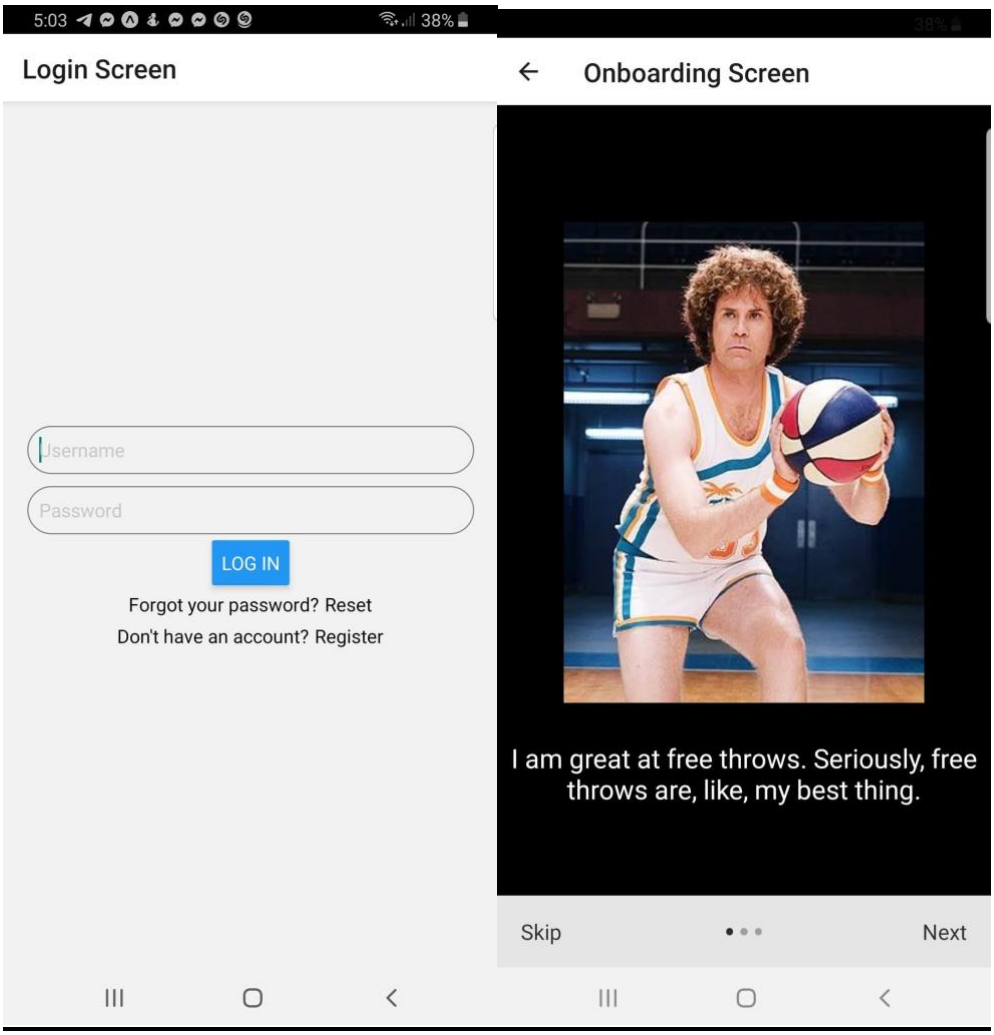

Home
Type: Free throw
Angle: Side on
Home
Type: Free throw
Angle: Side on
4 seconds
Home
Type: Free throw
Angle: Side on
4 seconds
Alert
Do you want to retake the video or upload?
Did you score?
Retake
Upload

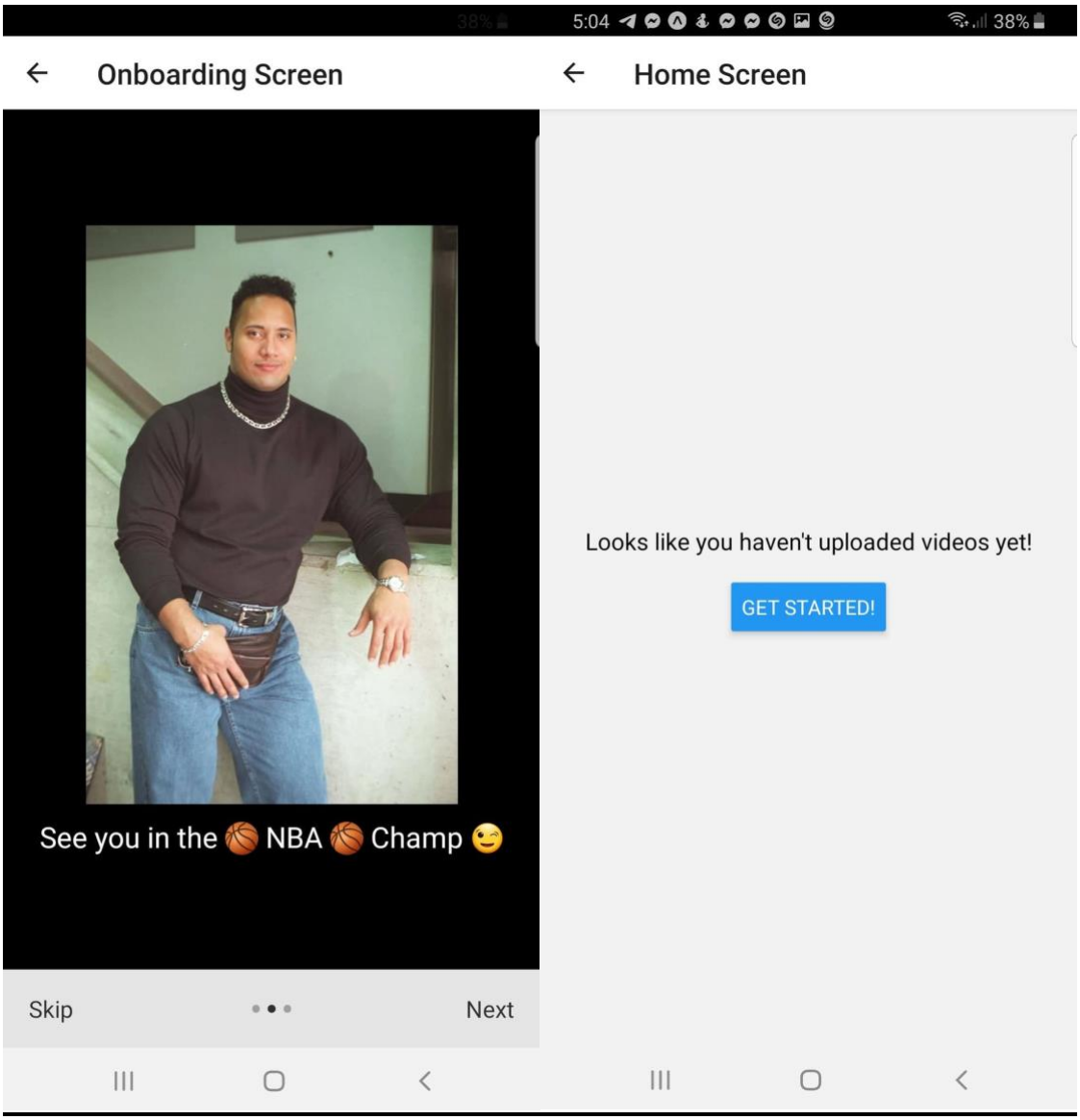

Reset Your Password
Email
RESET PASSWORD
Remember your password? Login
Reset Your Password
Email
RESET PASSWORD
Remember your password? Login

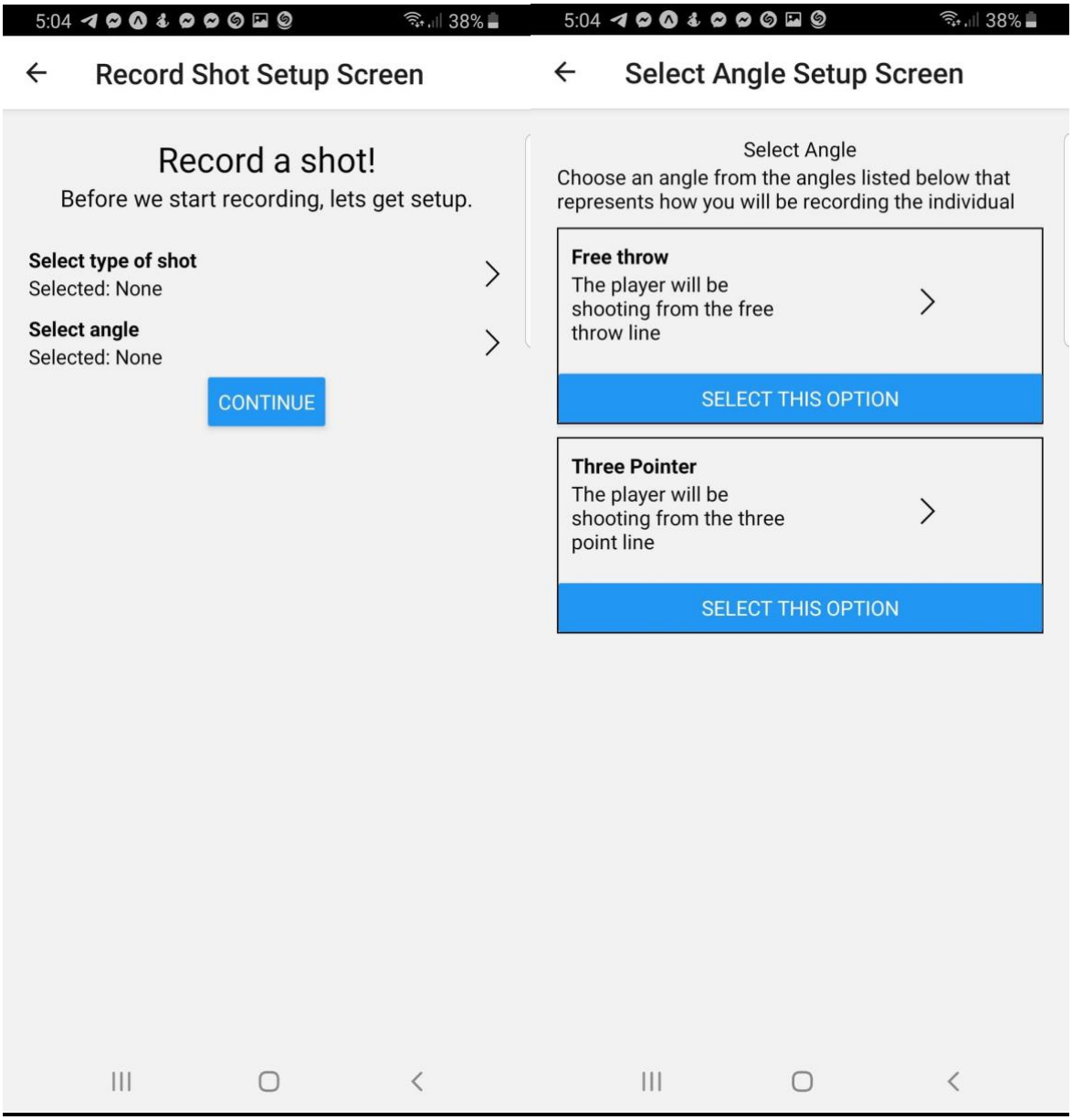

12:32
Home
Select Angle
Choose an angle from the angles listed below that represents how you will be recording the individual.
Side on
The player will be recorded from their left or right side at shoulder
Select this option
Front facing
The player will be facing the camera and the camera will be at
Select this option
From the back
The player will be facing away from the the camera and the camera
12:32
Home
Select Type of Shot
Choose the type of shot from the types listed below that represents what type of shot the player will be making
Free throw
The player will be shooting from the free throw line
Select this option
3 Point
The player will be shooting from the 3 point line
Select this option
Kobe
Kobe!

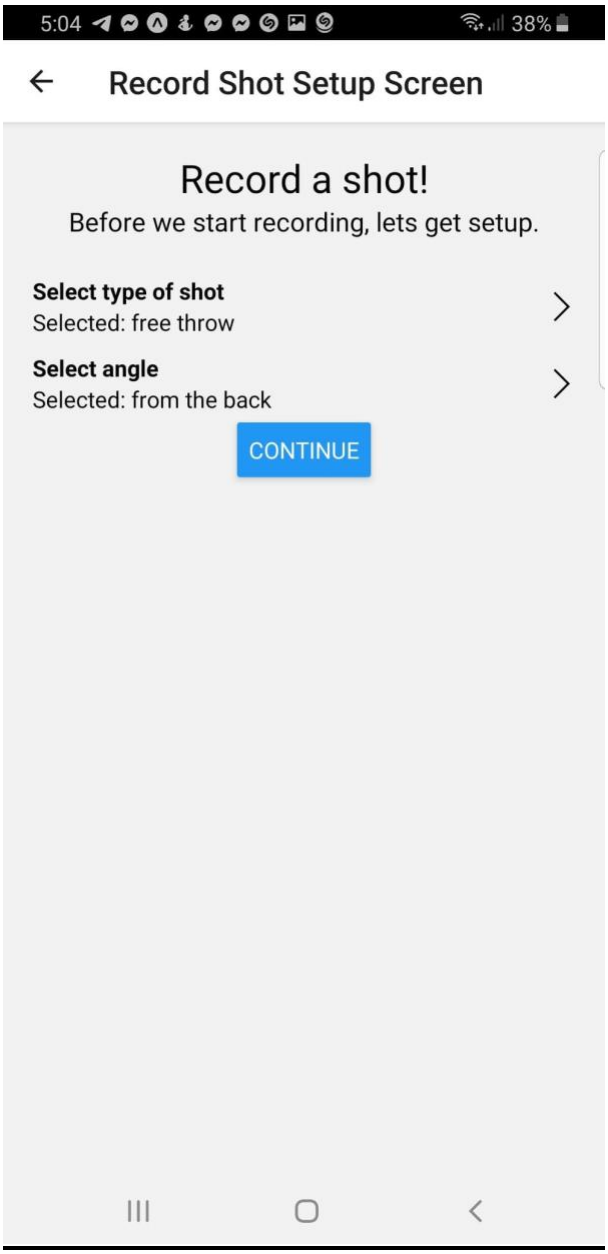

12:32
Home
Uploaded Videos
Date: 12th June 2020 9:48am
Type: Free throw
Angle: Front on
Feedback: Still processing
View video
Date: 12th June 2020 9:48am
Type: Free throw
Angle: Front on
Feedback: Get good kid
View video

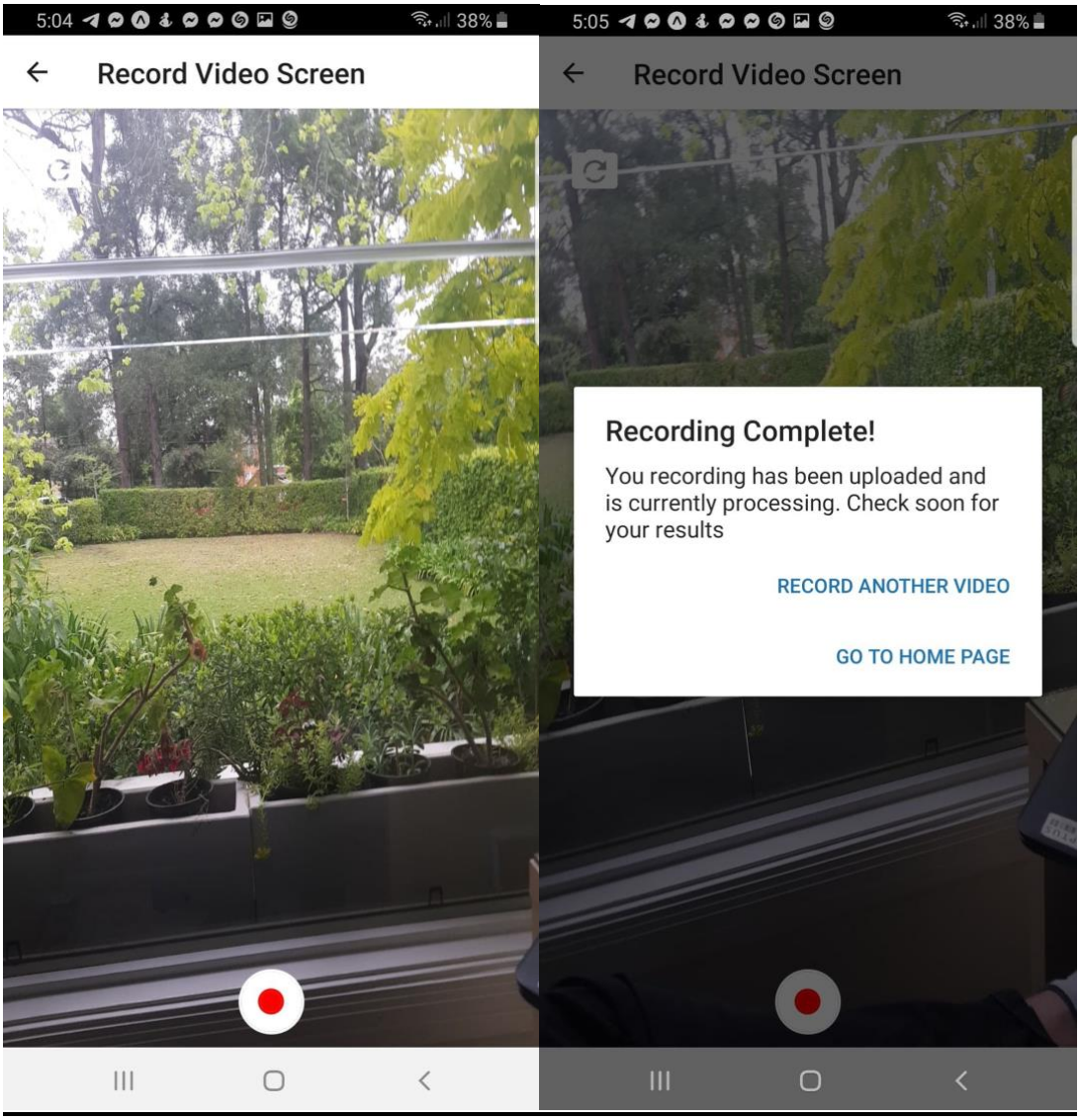
Appendix F: Mobile Application User Interface

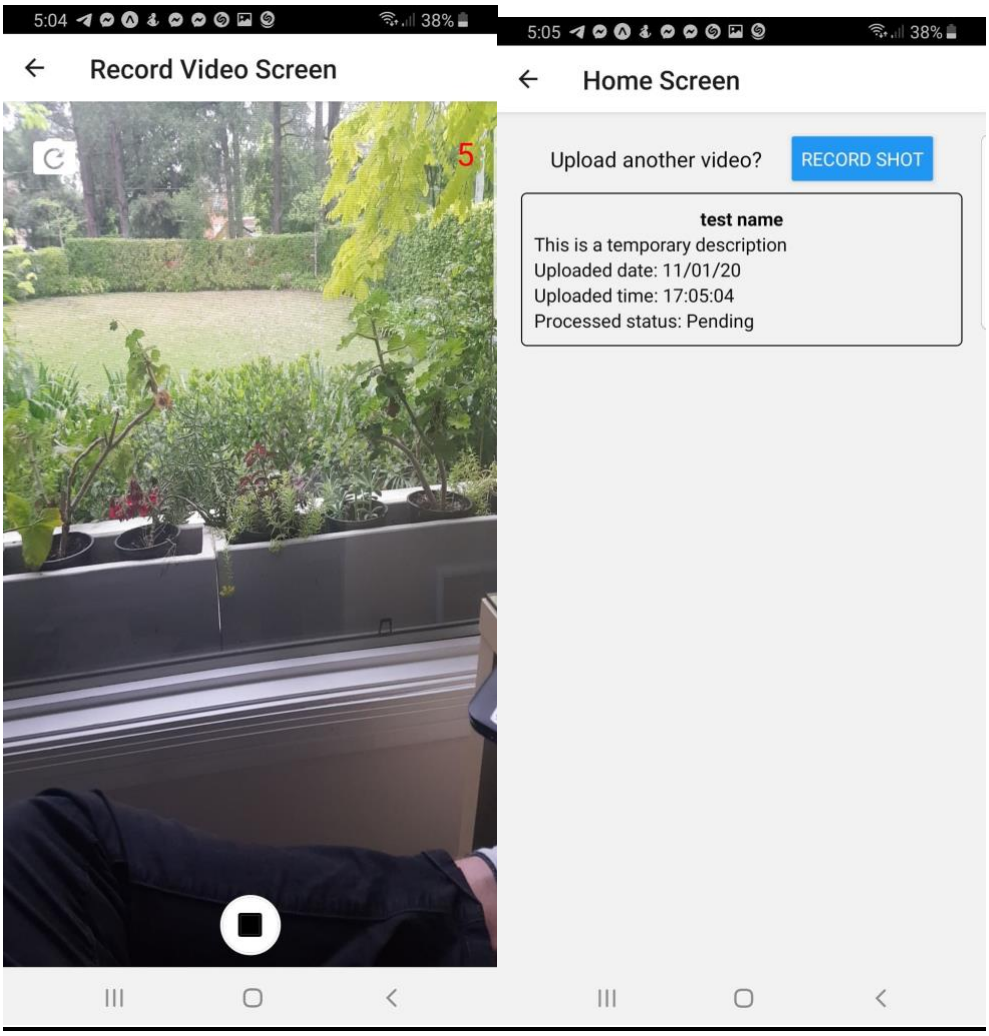


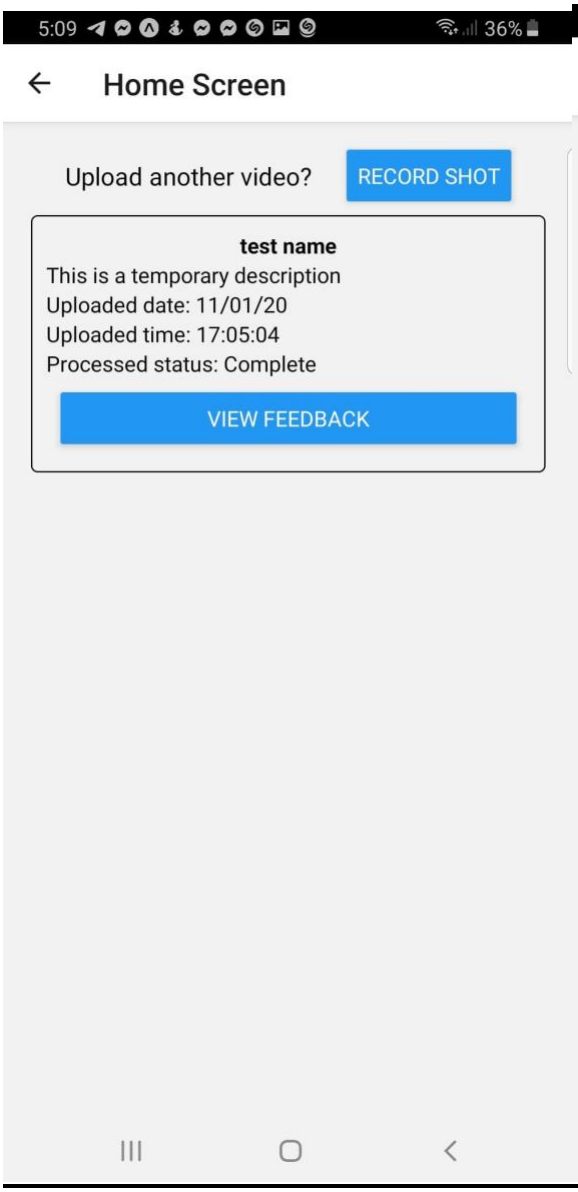

5:04
38%
Onboarding Screen
See you in the 🏀 NBA 🏀 Champ 😉
Skip
Next
Home Screen
Looks like you haven't uploaded videos yet!
GET STARTED!

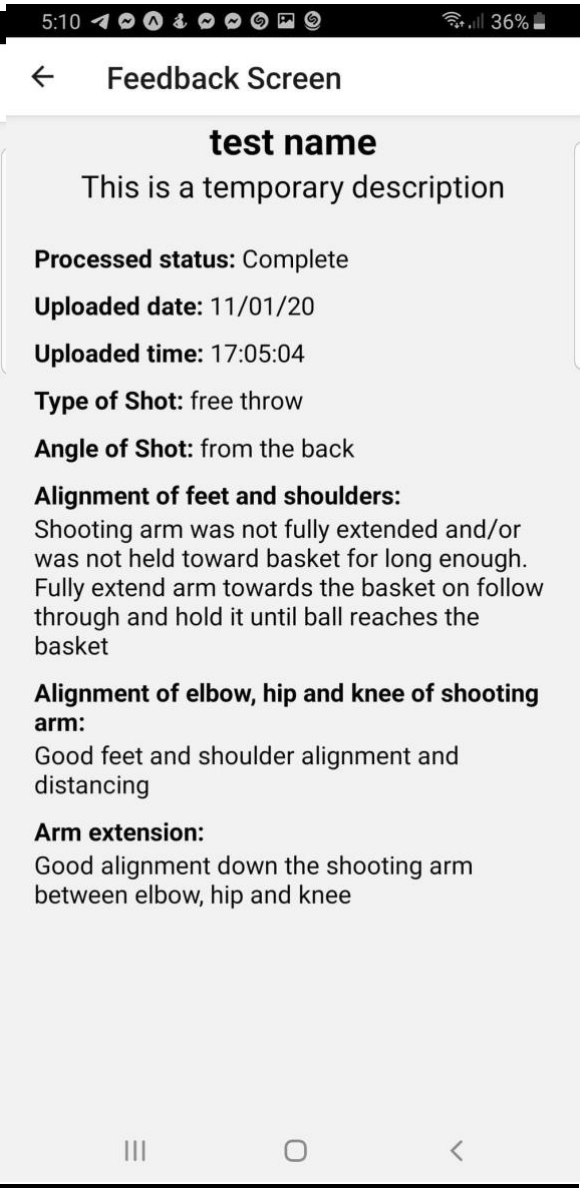

5:04
38%
Record Shot Setup Screen
Record a shot!
Before we start recording, lets get setup.
Select type of shot
Selected: None
Select angle
Selected: None
CONTINUE
5:04
38%
Select Angle Setup Screen
Select Angle
Choose an angle from the angles listed below that represents how you will be recording the individual
Free throw
The player will be shooting from the free throw line
SELECT THIS OPTION
Three Pointer
The player will be shooting from the three point line
SELECT THIS OPTION


5:04
38%
Record Shot Setup Screen
Record a shot!
Before we start recording, lets get setup.
Select type of shot
Selected: free throw
Select angle
Selected: from the back
CONTINUE

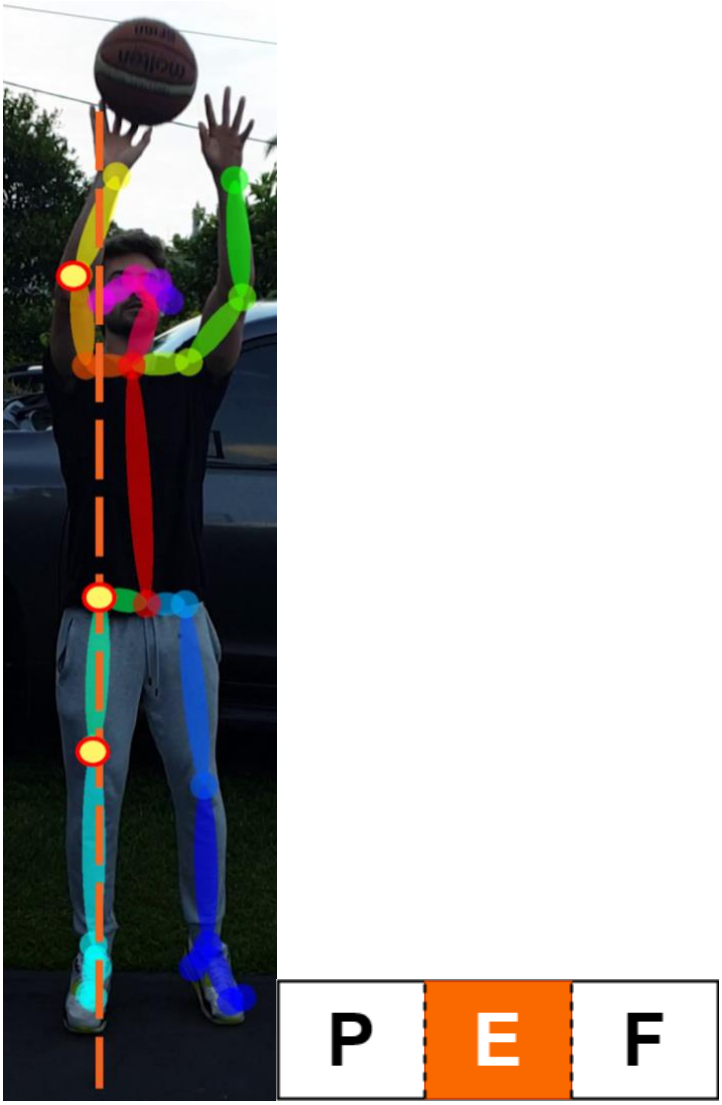

5:04
38%
Record Video Screen
5:05
38%
Record Video Screen
Recording Complete!
You recording has been uploaded and is currently processing. Check soon for your results
RECORD ANOTHER VIDEO
GO TO HOME PAGE


5:04
38%
Record Video Screen
5
5:05
38%
Home Screen
Upload another video?
RECORD SHOT
test name
This is a temporary description
Uploaded date: 11/01/20
Uploaded time: 17:05:04
Processed status: Pending


5:09
36%
Home Screen
Upload another video?
RECORD SHOT
test name
This is a temporary description
Uploaded date: 11/01/20
Uploaded time: 17:05:04
Processed status: Complete
VIEW FEEDBACK


5:10
36%
Feedback Screen
test name
This is a temporary description
Processed status: Complete
Uploaded date: 11/01/20
Uploaded time: 17:05:04
Type of Shot: free throw
Angle of Shot: from the back
Alignment of feet and shoulders:
Shooting arm was not fully extended and/or was not held toward basket for long enough. Fully extend arm towards the basket on follow through and hold it until ball reaches the basket
Alignment of elbow, hip and knee of shooting arm:
Good feet and shoulder alignment and distancing
Arm extension:
Good alignment down the shooting arm between elbow, hip and knee

Appendix G: Execution Phase Images and Algorithms



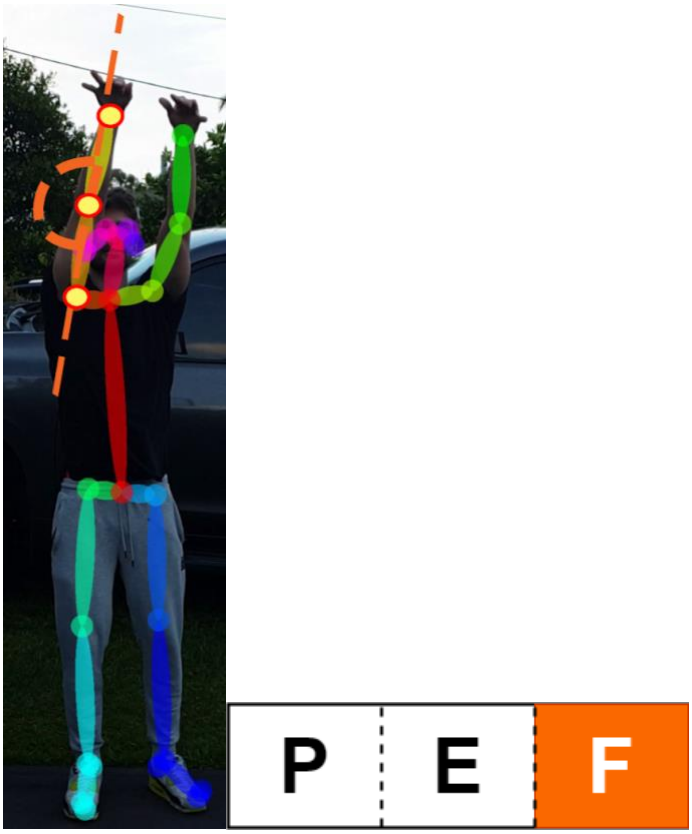
$$D_{EH_R} = \sqrt{(Xelbow_R - Xhip_R)^2} \quad \text{OR} \quad D_{EH_L} = \sqrt{(Xelbow_L - Xhip_L)^2} \quad (5)$$

$$D_{KH_R} = \sqrt{(Xknee_R - Xhip_R)^2} \quad \text{OR} \quad D_{KH_L} = \sqrt{(Xknee_L - Xhip_L)^2} \quad (6)$$

$$D_R = D_{EH_R} + D_{KH_R} \quad \text{OR} \quad D_L = D_{EH_L} + D_{KH_L} \quad (7)$$

$$D_R \leq Threshold_{\ SingleAxis} \quad \text{OR} \quad D_L \leq Threshold_{\ SingleAxis} \quad (8)$$

Appendix H: Execution Phase Images and Algorithms



$$\underline{E_R S_R} = < Ywrist_R - Yelbow_R, Xwrist_R - Xelbow_R > = < U_1, U_2 >$$
$$\text{OR} \qquad (1)$$
$$\underline{E_L S_L} = < Ywrist_L - Yelbow_L, Xwrist_L - Xelbow_L > = < U_1, U_2 >$$

$$\underline{E_R S_R} = < Yshoulder_R - Yelbow_R, Xshoulder_R - Xelbow_R > = < V_1, V_2 >$$
$$\text{OR} \qquad (2)$$
$$\underline{E_L S_L} = < Yshoulder_L - Yelbow_L, Xshoulder_L - Xelbow_L > = < V_1, V_2 >$$

$$\underline{E_R W_R} \cdot \underline{E_R S_R} = U_1 V_1 + U_2 V_2 \qquad \text{OR} \qquad E_L W_L \cdot E_L S_L = U_1 V_1 + U_2 V_2 \qquad (3)$$

$$\left|\left|\underline{E_R W_R}\right|\right| = \sqrt{U_1^2 + V_1^2} \quad \text{OR} \quad \left|\left|\underline{E_L W_L}\right|\right| = \sqrt{U_1^2 + V_1^2} \tag{4}$$

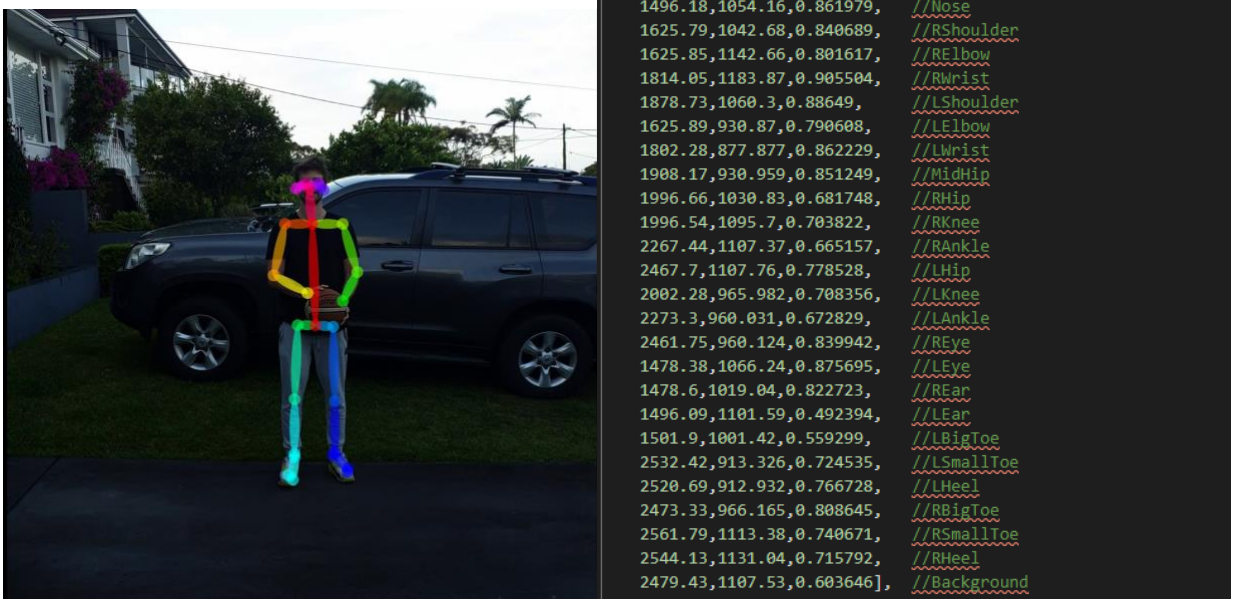
$$\left|\left|\underline{E_R S_R}\right|\right| = \sqrt{U_2^2 + V_2^2} \quad \text{OR} \quad \left|\left|\underline{E_L S_L}\right|\right| = \sqrt{U_2^2 + V_2^2} \tag{5}$$

$$cos\theta = \frac{\underline{E_R W_R} \cdot \underline{E_R S_R}}{\left|\left|\underline{E_R W_R}\right|\right|\left|\left|\underline{E_R S_R}\right|\right|} \quad \text{OR} \quad cos\theta = \frac{\underline{E_L W_L} \cdot \underline{E_L S_L}}{\left|\left|\underline{E_L W_L}\right|\right|\left|\left|\underline{E_L S_L}\right|\right|} \tag{6}$$

$$\theta \geq Threshold_{cosine} \tag{7}$$

Appendix I: Example Pose Keypoint Estimation

Output video with keypoint data:



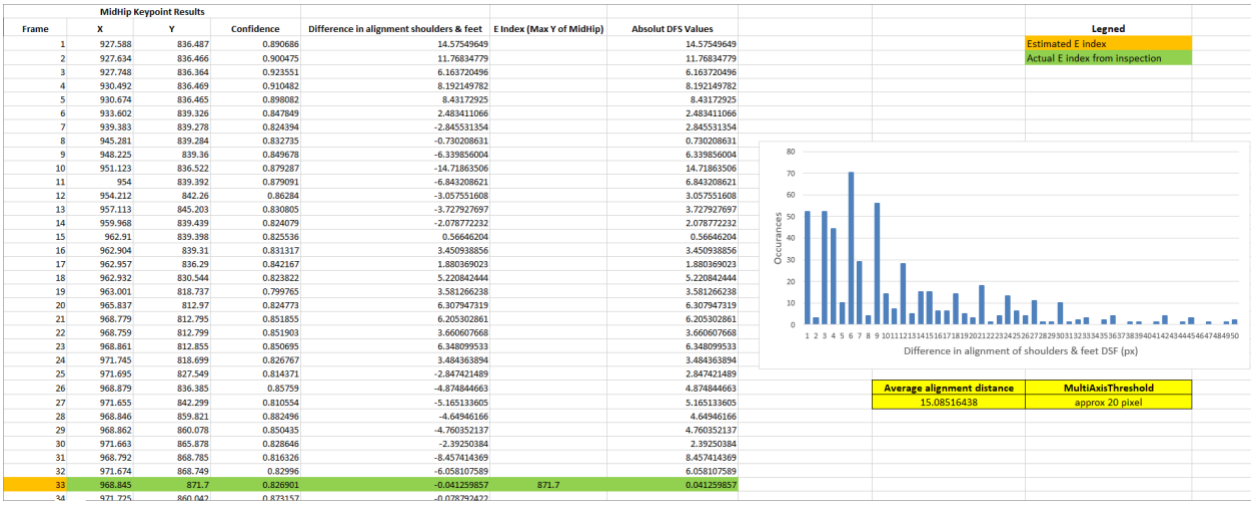

```
{"version":1.3,"people":[{"person_id":[-1],"pose_keypoints_2d":[
    1496.18,1054.16,0.861979,   //Nose
    1625.79,1042.68,0.840689,   //RShoulder
    1625.85,1142.66,0.801617,   //RElbow
    1814.05,1183.87,0.905504,   //RWrist
    1878.73,1060.3,0.88649,     //LShoulder
    1625.89,930.87,0.790608,    //LElbow
    1802.28,877.877,0.862229,   //LWrist
    1908.17,930.959,0.851249,   //MidHip
    1996.66,1030.83,0.681748,   //RHip
    1996.54,1095.7,0.703822,    //RKnee
    2267.44,1107.37,0.665157,   //RAnkle
    2467.7,1107.76,0.778528,    //LHip
    2002.28,965.982,0.708356,   //LKnee
    2273.3,960.031,0.672829,    //LAnkle
    2461.75,960.124,0.839942,   //REye
    1478.38,1066.24,0.875695,   //LEye
    1478.6,1019.04,0.822723,    //REar
    1496.09,1101.59,0.492394,   //LEar
    1501.9,1001.42,0.559299,    //LBigToe
    2532.42,913.326,0.724535,   //LSmallToe
    2520.69,912.932,0.766728,   //LHeel
    2473.33,966.165,0.808645,   //RBigToe
    2561.79,1113.38,0.740671,   //RSmallToe
    2544.13,1131.04,0.715792,   //RHeel
    2479.43,1107.53,0.603646],  //Background
```


Appendix J: Data gathered for threshold setting

MultiAxisThreshold:

https://drive.google.com/file/d/1lNjf8KYqMWjAEgKK1chNutzbAblXqQ-U/view?usp=sharing

(Link to image below for better viewing)



MidHip Keypoint Results

Frame	X	Y	Confidence	Difference in alignment shoulders & feet	E Index (Max Y of MidHip)	Absolut DFS Values
1	927.588	836.487	0.890686	14.57549649		14.57549649
2	927.634	836.466	0.900475	11.76834779		11.76834779
3	927.748	836.364	0.923551	6.163720496		6.163720496
4	930.492	836.469	0.910482	8.192149782		8.192149782
5	930.674	836.465	0.898082	8.43172925		8.43172925
6	933.602	839.326	0.847849	2.483411066		2.483411066
7	939.383	839.278	0.824394	-2.845531354		2.845531354
8	945.281	839.284	0.832735	-0.730208631		0.730208631
9	948.225	839.36	0.849678	-6.339856004		6.339856004
10	951.123	836.522	0.879287	-14.71863506		14.71863506
11	954	839.392	0.879091	-6.843208621		6.843208621
12	954.212	842.26	0.86284	-3.057551608		3.057551608
13	957.113	845.203	0.830805	-3.727927697		3.727927697
14	959.968	839.439	0.824079	-2.078772232		2.078772232
15	962.91	839.398	0.825536	0.56646204		0.56646204
16	962.904	839.31	0.831317	3.450938856		3.450938856
17	962.957	836.29	0.842167	1.880369023		1.880369023
18	962.932	830.544	0.823822	5.220842444		5.220842444
19	963.001	818.737	0.799765	3.581266238		3.581266238
20	965.837	812.97	0.824773	6.307947319		6.307947319
21	968.779	812.795	0.851855	6.205302861		6.205302861
22	968.759	812.799	0.851903	3.660607668		3.660607668
23	968.861	812.855	0.850695	6.348099533		6.348099533
24	971.745	818.699	0.826767	3.484363894		3.484363894
25	971.695	827.549	0.814371	-2.847421489		2.847421489
26	968.879	836.385	0.85759	-4.874844663		4.874844663
27	971.655	842.299	0.810554	-5.165133605		5.165133605
28	968.846	859.821	0.882496	-4.64946166		4.64946166
29	968.862	860.078	0.850435	-4.760352137		4.760352137
30	971.663	865.878	0.828646	-2.39250384		2.39250384
31	968.792	868.785	0.816326	-8.457414369		8.457414369
32	971.674	868.749	0.82996	-6.058107589		6.058107589
33	968.845	871.7	0.826901	-0.041259857	871.7	0.041259857
34	971.725	860.042	0.873157	-0.078792422		

Legned

Estimated E Index

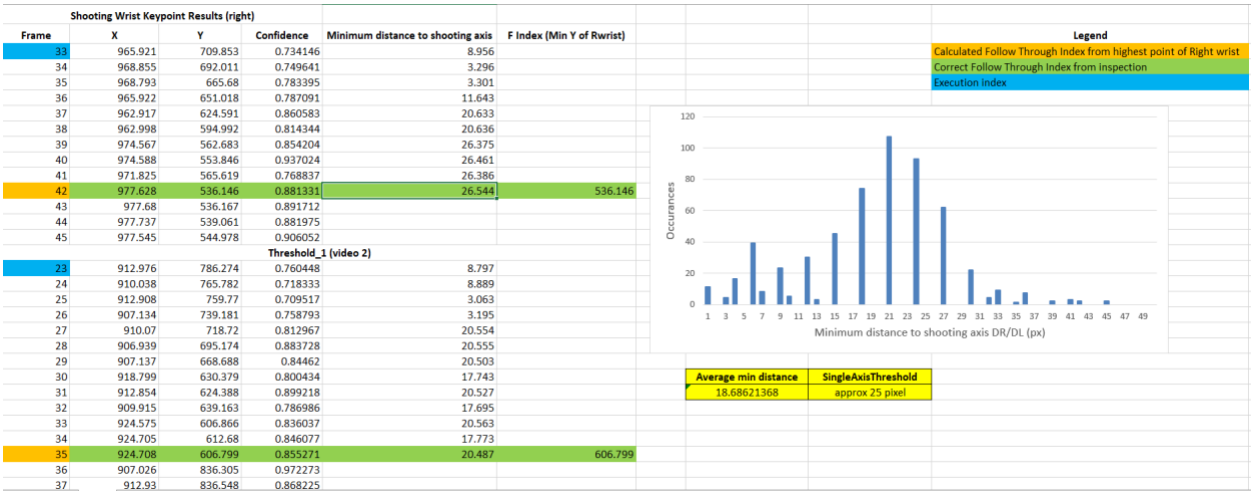
Actual E index from inspection

Average alignment distance	MultiAxisThreshold
15.08516438	approx 20 pixel

SingleAxisThreshold:

https://drive.google.com/file/d/1jOoayIiUZCcZPSVPGvNaoVyQqiOIUHoR/view?usp=sharing

(Link to image below for better viewing)

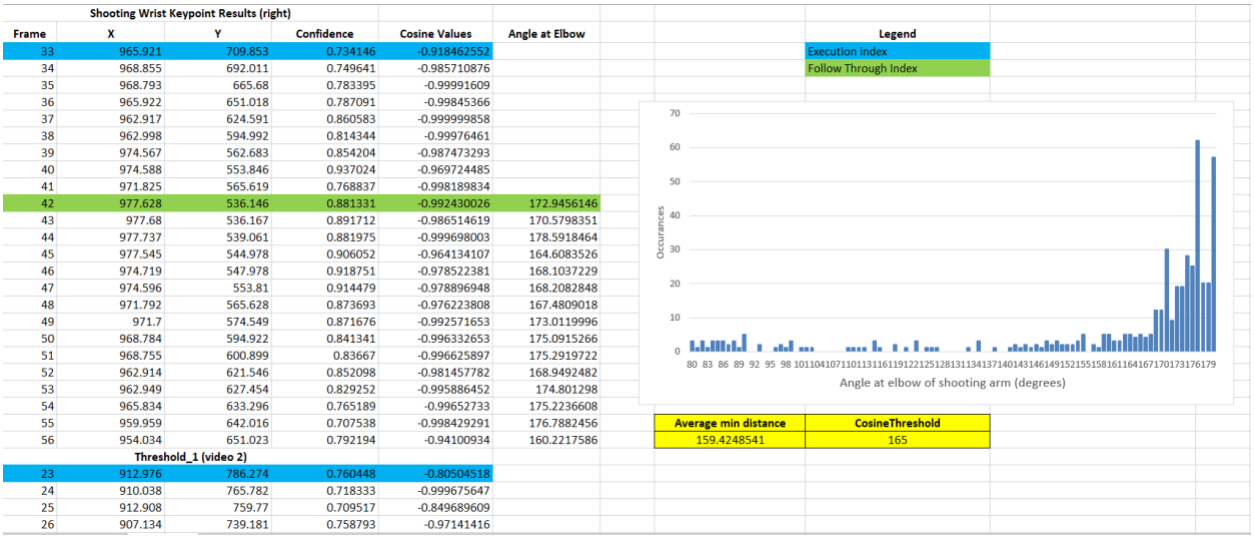


Shooting Wrist Keypoint Results (right)					
Frame	X	Y	Confidence	Minimum distance to shooting axis	F Index (Min Y of Rwrist)
33	965.921	709.853	0.734146	8.956	
34	968.855	692.011	0.749641	3.296	
35	968.793	665.68	0.783395	3.301	
36	965.922	651.018	0.787091	11.643	
37	962.917	624.591	0.860583	20.633	
38	962.998	594.992	0.814344	20.636	
39	974.567	562.683	0.854204	26.375	
40	974.588	553.846	0.937024	26.461	
41	971.825	565.619	0.768837	26.386	
42	977.628	536.146	0.881331	26.544	536.146
43	977.68	536.167	0.891712		
44	977.737	539.061	0.881975		
45	977.545	544.978	0.906052		
Threshold_1 (video 2)					
23	912.976	786.274	0.760448	8.797	
24	910.038	765.782	0.718333	8.889	
25	912.908	759.77	0.709517	3.063	
26	907.134	739.181	0.758793	3.195	
27	910.07	718.72	0.812967	20.554	
28	906.939	695.174	0.883728	20.555	
29	907.137	668.688	0.84462	20.503	
30	918.799	630.379	0.800434	17.743	
31	912.854	624.388	0.899218	20.527	
32	909.915	639.163	0.786986	17.695	
33	924.575	606.866	0.836037	20.563	
34	924.705	612.68	0.846077	17.773	
35	924.708	606.799	0.855271	20.487	606.799
36	907.026	836.305	0.972273		
37	912.93	836.548	0.868225		

CosineThreshold:

https://drive.google.com/file/d/1tHfw9XRtv2S_dYFDRIb-_2HBqa6T3dHR/view?usp=sharing

(Link to image below for better viewing)



Shooting Wrist Keypoint Results (right)					
Frame	X	Y	Confidence	Cosine Values	Angle at Elbow
33	965.921	709.853	0.734146	-0.918462552	
34	968.855	692.011	0.749641	-0.985710876	
35	968.793	665.68	0.783395	-0.99991609	
36	965.922	651.018	0.787091	-0.99845366	
37	962.917	624.591	0.860583	-0.999999858	
38	962.998	594.992	0.814344	-0.99976461	
39	974.567	562.683	0.854204	-0.987473293	
40	974.588	553.846	0.937024	-0.969724485	
41	971.825	565.619	0.768837	-0.998189834	
42	977.628	536.146	0.881331	-0.992430026	172.9456146
43	977.68	536.167	0.891712	-0.986514619	170.5798351
44	977.737	539.061	0.881975	-0.999698003	178.5918464
45	977.545	544.978	0.906052	-0.964134107	164.6083526
46	974.719	547.978	0.918751	-0.978522381	168.1037229
47	974.596	553.81	0.914479	-0.978896948	168.2082848
48	971.792	565.628	0.873693	-0.976223808	167.4809018
49	971.7	574.549	0.871676	-0.992571653	173.0119996
50	968.784	594.922	0.841341	-0.996332653	175.0915266
51	968.755	600.899	0.83667	-0.996625897	175.2919722
52	962.914	621.546	0.852098	-0.981457782	168.9492482
53	962.949	627.454	0.829252	-0.995886452	174.801298
54	965.834	633.296	0.765189	-0.99652733	175.2236608
55	959.959	642.016	0.707538	-0.998429291	176.7882456
56	954.034	651.023	0.792194	-0.94100934	160.2217586
Threshold_1 (video 2)					
23	912.976	786.274	0.760448	-0.80504518	
24	910.038	765.782	0.718333	-0.999675647	
25	912.908	759.77	0.709517	-0.849689609	
26	907.134	739.181	0.758793	-0.97141416	

References

Vickers, J. N. (2007). *Perception, cognition, and decision training: The quiet eye in action*. Human Kinetics.

Knudson, D. (1993). Biomechanics of the basketball jump shot—Six key teaching points. *Journal of Physical Education,* Recreation & Dance, 64(2), 67-73.

Sport NZ. (2010, March 30). *Introduction to Biomechanics.* Module 5, p 20. - https://www.yumpu.com/en/document/read/26518873/module-5-introduction-to-biomechanics-sport-new-zealand

(Below is just copy of references from my Research proposal - Ben)

1. Shankar, S., Suresh, R., Talasila, V., & Sridhar, V. (2018). Performance measurement and analysis of shooting form of basketball players using a wearable IoT system. 2018 IEEE 8th International Advance Computing Conference (IACC), 26–32.

2. Nacsport: video analysis better than ever 2020, Nacsport.com. viewed 20 May 2020, <https://www.nacsport.com/index.php?lc=en-gb>.

3. Hawk-Eye 2020, Hawkeyeinnovations.com. viewed 20 May 2020, <https://www.hawkeyeinnovations.com/>.

4. Stats Perform Team Performance Solutions 2020, Stats Perform. viewed 20 May 2020, <https://www.statsperform.com/team-performance/>.

5. Hsia, C., Chien, C., Hsu, H., Chang, Y. and Chiang, J. 2015, Analyses of basketball player field goal shooting postures for player motion correction using kinect sensor.

6. Piette, J., Anand, S., Zhang, K., & Piette, J. (2010). Scoring and Shooting Abilities of NBA Players. Journal of Quantitative Analysis in Sports, 6(1), 1194–1194.

7. Felsen, P. 2017, “Body Shots”: Analyzing Shooting Styles in the NBA using Body Pose.

8. Guastello, S., & Rieke, M. (1994). Computer-based test interpretations as expert systems: Validity and viewpoints from artificial intelligence theory. Computers in Human Behavior, 10(4), 435–455.