TECHNICAL MANUAL

GymVision

GymVision is a unique application that utilizes computer vision and pose estimation to provide the user with live feedback and analysis while they perform a gym exercise in front of either an android camera or a webcam connected to a windows device. The application will provide feedback in a way that will help the user improve their technique while also ensuring your safety to avoid injury during exercise.

Fawaz Alsafadi – 15380871

Ayman El Gendy – 15395461

Supervisor – Alistair Sutherland

May 15th 2020

CONTENTS

1. INTRODUCTION

1.1 MOTIVATION

As both myself and my project partner are members of a Gym all too often we have observed incorrect form both in ourselves and other Gym goers while performing exercises involving free weights. Majority of people will start off learning weight training techniques by observing their friends or others in the Gym. But more often than not what they see is unsafe which can lead to a number of serious injuries including sprains, fractures and spinal damage. Free weight exercises account for over 90% of weight training-related injuries in the U.S. alone [1]. This is because consultation with a professional training specialist or a personal trainer is a luxury most can't afford or feel that it is unnecessary. Motivated with this knowledge we felt that the development of a mobile and portable platform to help a user assess their technique on the go would benefit not only their health and wellbeing but also suit the modern user looking for feedback and information at the click of a button.

1.2 OVERVIEW

The aim of this project is to produce a platform for users to analyse their technique while performing various exercises using free weights and provide feedback to the user based on their technique. The project consists of two seamlessly integrated applications both on mobile android devices and Windows desktop. The application uses computer vision to analyse the user's technique. We carry out the analysis by using a pre-trained pose estimation model to extract the user's KeyPoints (Limbs) in Realtime and perform a set of mathematical checks to determine the location of the limbs and where they should be to achieve correct technique.

To use the application the user will log on to either the desktop application or the android application, the same credentials will work on both devices. The user then will need to select an exercise to analyse and begin performing the exercise in front of either their webcam or android camera. The applications will then provide live feedback, through audio and text on the screen, to the user based on their performance and will advise them of any incorrect technique and how to remedy it to ensure proper and safe technique. The user will also be able to view a breakdown of all the checks performed and which ones the user passed or failed so that they may know what to correct the next time they are attempting the exercise.

Through the use of GymVision, both experienced and inexperienced gym goers should be able to benefit and improve their technique so as to avoid injury. Due to the application being on both windows desktop and android a user should always have the application available to them whether in the gym or from the comfort of their own home. Safe and correct exercise technique is just a click away.

1.3 BUSINESS CONTEXT

From the standpoint of purely generating income, fitness product producers and various health corporations could run advertisements within the app to target users who are active and health conscious as this would be the target market for the application. However, an application such as GymVision could be utilized far better by potentially setting up devices integrated into the infrastructure of a Gym, for example setting up a booth where a gym goer can go and analyse their technique quickly and efficiently to ensure they are performing the exercise correctly and safely. This could be an alternative to offering personal trainer services which many gyms already do and many members can't afford.

1.4 GLOSSARY

- *GymVision* - A multiplatform application that helps assess technique in the gym.
- *Computer Vision*- A computer science field that deals with how computers can interpret images and videos.
- *Android Studio* - An IDE for developing android applications.
- *Pose estimation* - The use of computer vision methods to detect human body parts in images and videos.
- openPose - A library for using pose estimation.
- *FritzAI* - A library for using pose estimation on Android.
- *Form/Technique* - The correctness/safeness of how a weightlifting exercise is performed.
- *PyQt5* - A python binding for the Qt GUI toolkit.

2. RESEARCH

2.1 POSE ESTIMATION RESEARCH

The app is based on functionality provided by pose-estimation models. Pose estimation refers to computer vision methods that identifies human body parts in images and videos. This is done through a deep learning process using TensorFlow. In pose estimation, inferences are performed on a set of observations on body parts and the spatial dependencies between them. Classic pose estimation was completed using a "pictorial structure framework" which represented humans as a loose collection of parts. These parts were found matching a template to a feature in an image.

Deep learning methods have been used to obtain these observations recently and have improved the accuracy of pose estimation models. This involves convolutional neural network-based regression.

Which model to choose?

We chose openPose for because of its python API, extensive documentation and promising speeds. As python is our preferred language of choice and openPose is arguably the most popular open source pose estimation library, we chose it.

For android, we initially wanted to use a model based on openPose, but after testing we realised that the model was not performing to the level we needed. We decided to switch from openPose to fritzAI as it appeared more accurate.

2.2 RESEARCHING EXERCISES

Weightlifting and going to the gym are important parts of many people's lives and are important in living a healthier lifestyle. However, such exercise often comes with the risk of injury Free weight exercises account for over 90% of weight training-related injuries in the U.S. alone [1].

Through our research we identified which weightlifting exercises left beginners most prone to getting injured. The results were that the highest risk exercises were compound movements such as Squat, Deadlift, Bent-Over Row, Bench Press and Overhead Press. These movements require the use of almost the entire body, which means that good form requires the person to focus on multiple parts of their body throughout a range of motion.

After identifying the types of exercises that we would be targeting, we researched common mistakes in these exercises. Common mistakes in these exercises included incorrect width in grip or stance, incorrect position/depth in the movement, rounded backs and other unique exercise specific checks.

3. GENERAL DESCRIPTION

3.1 SYSTEM FUNCTIONS

The application is designed to allow users to improve their form in the gym. They need an account to login in with. They can create them within the app. The system then lets users choose common gym exercises and analyse their form through the use of computer vision.

Then they will be given feedback that they can use to perform better and safer in the gym.

The system does not perform guidance on how to perform a squat, but rather errors that are performed by beginners in the squat movement. This is an important distinction for the users to properly benefit from the software.

3.2 USER CHARACTERISTICS

The target audience for this application will be beginners in the gym who are looking for alternatives to personal trainers for improving their technique for gym exercise. The first time in a gym can be an intimidating time, especially performing a movement you have never done before. It is important for beginners to understand certain key checks in an exercise before they begin to add weight and progress every week. They should know the basic knowledge of an exercise before having their form analysed, to ensure they are not performing the wrong exercise.

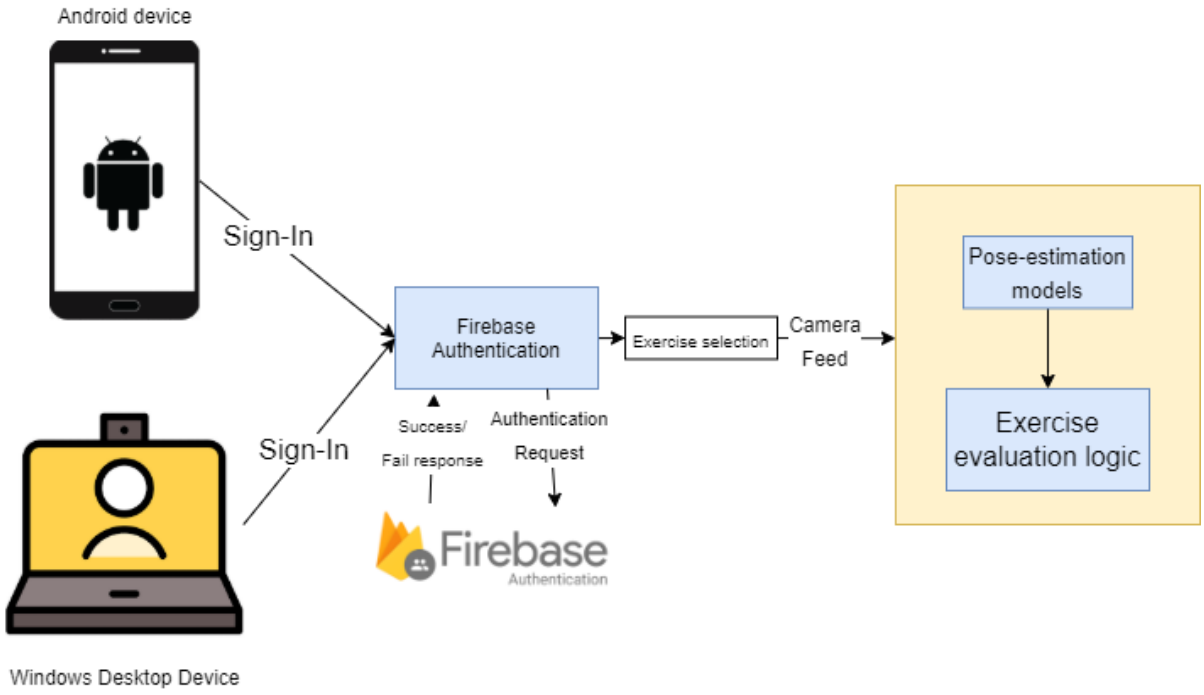
Beginners make progress in the gym quite quickly and this often leads them to attempting exercises with heavier weights, this could lead to injury. Our application should be used by people who want to move from novice to intermediate skill level in the gym, while staying safe and learning common mistakes made for certain exercises. Users who are looking to learn advanced bodybuilding techniques and form will not benefit from our application.

Users should either have a modern android phone capable of running the software, or a desktop/laptop with a camera and NVidia card. The average user does not require any advanced technology skills, they only need to understand how to download and install applications.

4. SYSTEM ARCHITECTURE

4.1 GENERAL SYSTEM ARCHITECTURE

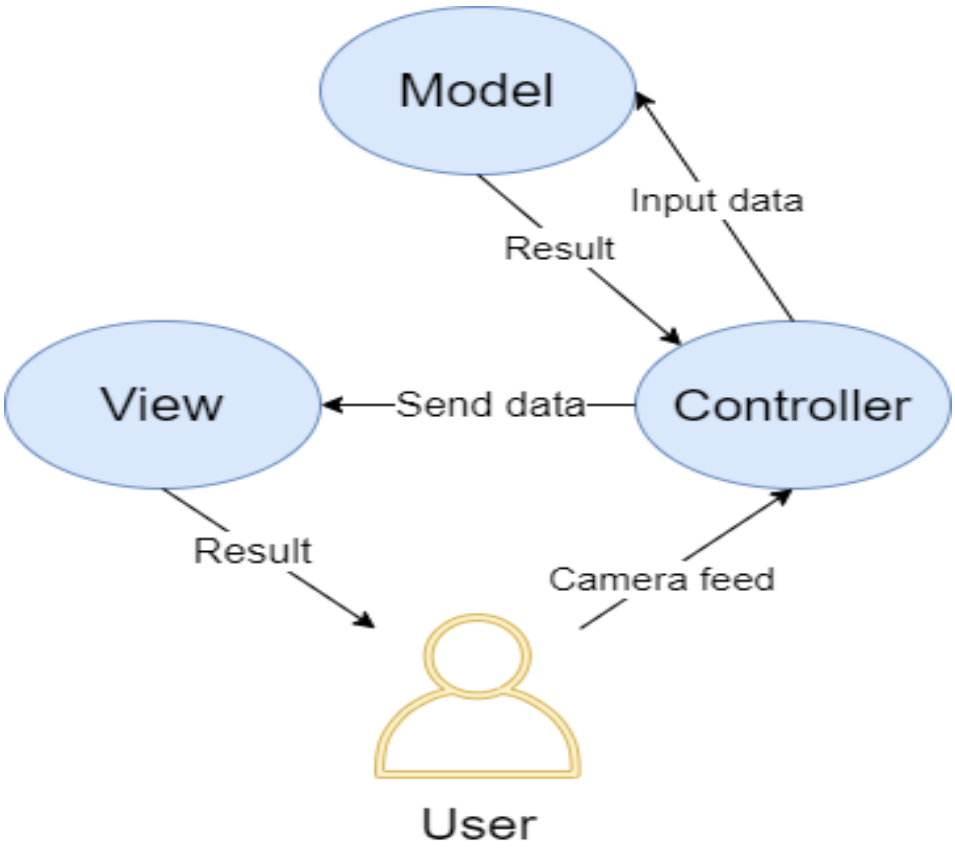
Figure 1



The GymVision application is built for both Android devices and Windows desktop and the same functionality is shared across both devices. The android implementation is in Java with XML for designing the user interface and application layouts. The windows implementation is in python and uses PyQt5 for the user interface. The application requires users to authenticate using their email and password. The application will then contact firebase to authenticate the user or to allow the user to create an account. The same account will work for both the android application as well as the desktop application. So, the user is able to switch seamlessly between the two devices. Once the user has successfully signed into the application. They will be able to navigate to the exercise selection screen through the use of the navigation drawer on android, on windows the first screen will be the exercise selection screen. Here the user will be able to select an exercise to evaluate and stand in front of the camera to begin. The application will take the live video feed and input the frames into the pose estimation model for keypoint prediction and detection. The application will use the key point X and Y coordinates in order to analyse the users technique and provide feedback.

4.2 MODEL VIEW CONTROLLER

Figure 2



The purpose of the diagram in figure 2 is to illustrate the relationship between the primary components of the system. Breaking the application into the selections as shown in the diagram is useful for visualizing the core functions and enables clearer understanding of the various components.

Controller - This component of the system acts as a liaison between the model and the view and handles the input data of the user and passes it on to the model. For our purpose the controller takes the input of the camera feed from the user and dissects the video feed into frames that can be then passed into the pose estimation model. The controller then receives the results of the pose estimation prediction and analyses the data to determine whether the user has passed or failed specific checks that evaluate the users technique.

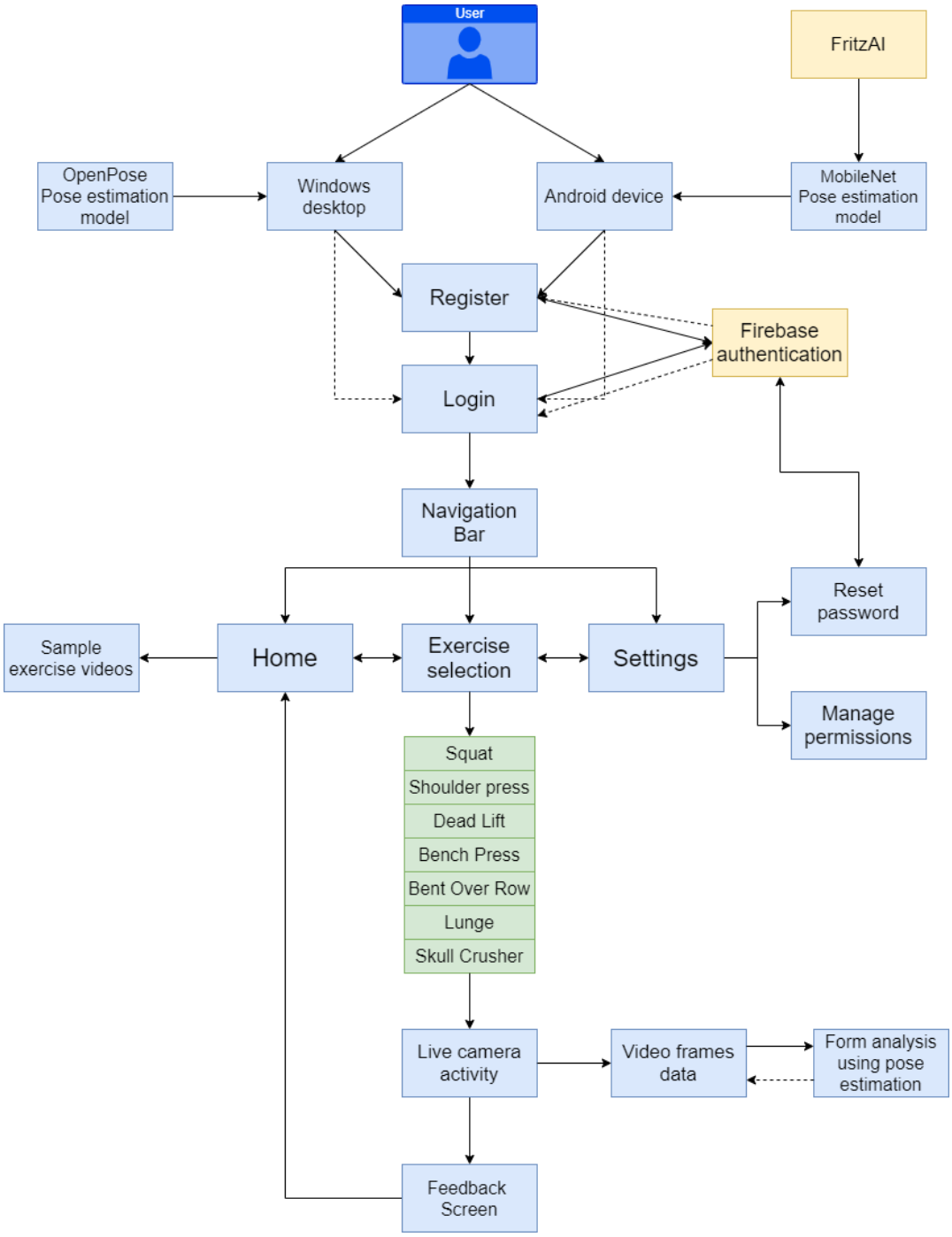
Model - The model is responsible for carrying out the underlying logic of the application for this system the model receives the frame data and runs it through the pose estimation model for prediction of users Keypoints (Limbs). The model then passes the result of the prediction which includes the name of each keypoint found and the X and Y coordinates back to the controller.

View - The view component handles the interaction between the user and the system. The view receives the results of the technique analysis logic from the controller and displays detailed feedback to the user to help the user correct their form. The feedback is provided via text and audio.

5. DESIGN

5.1 HIGH-LEVEL DESIGN

Figure 3



The diagram in figure 3 illustrates the high-level design of the application. As shown in the diagram will be required to register or login if they already have an account to access the application. On initial logins there is a tutorial screen that appears giving the user guidance on the main aspects of

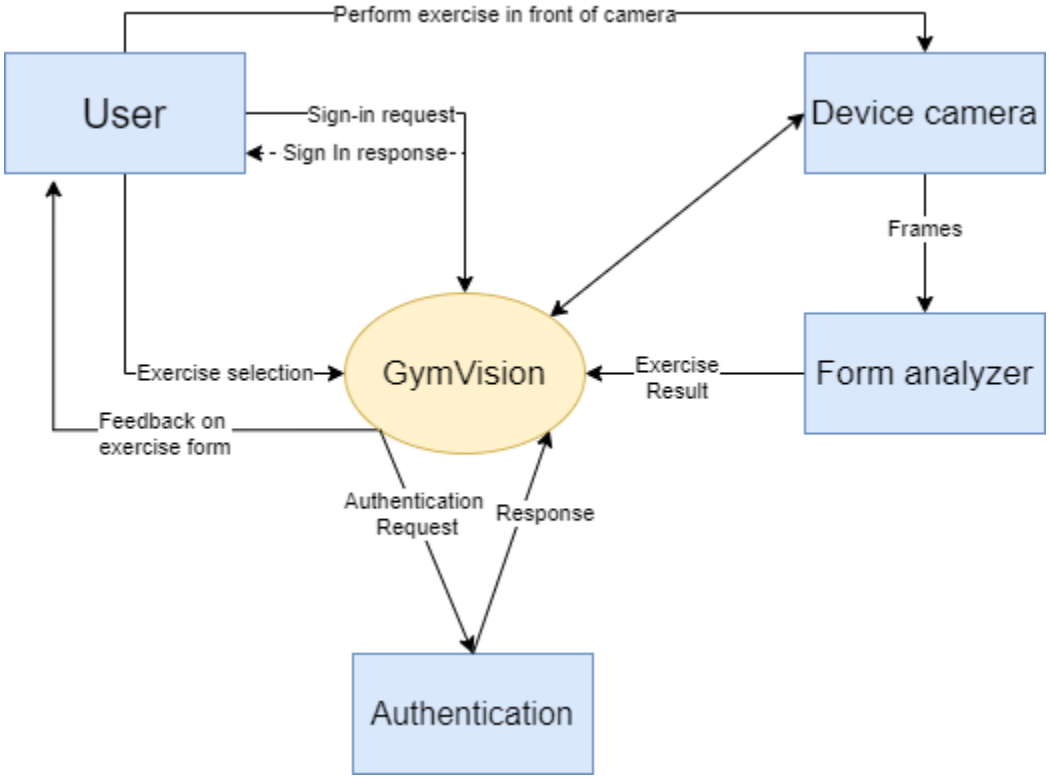
the application and how to use it. This is essential for new users as the application is quite unique and some users will need the tutorial to fully utilize the app. The user is then able to navigate to the Home screen, exercise selection screen or settings.

- Home Screen
    - There is a video tutorial showing the user how to use the application to evaluate their technique for an exercise such as the squat.
- Settings screen
    - On the settings screen the user can manage permissions and reset their password.
- Exercise screen
    - On this screen the user can select one of the 7 exercises shown in figure 3.

One the user has made an exercise selection; they will be taken to the camera activity. Here the user should place the device on a steady surface and stand back 2 meters away from their android camera or webcam and begin performing the exercise. This will send the users live video feed to the technique analysis component which will analyse the users technique and provide live concise feedback via text or audio. Once the user has completed their exercise they will be taken to the feedback screen where they will be able to review their performance and get detailed feedback on each aspect of their exercise and whether they passed or failed the checks for that particular exercise.

5.2 SYSTEM CONTEXT DIAGRAM

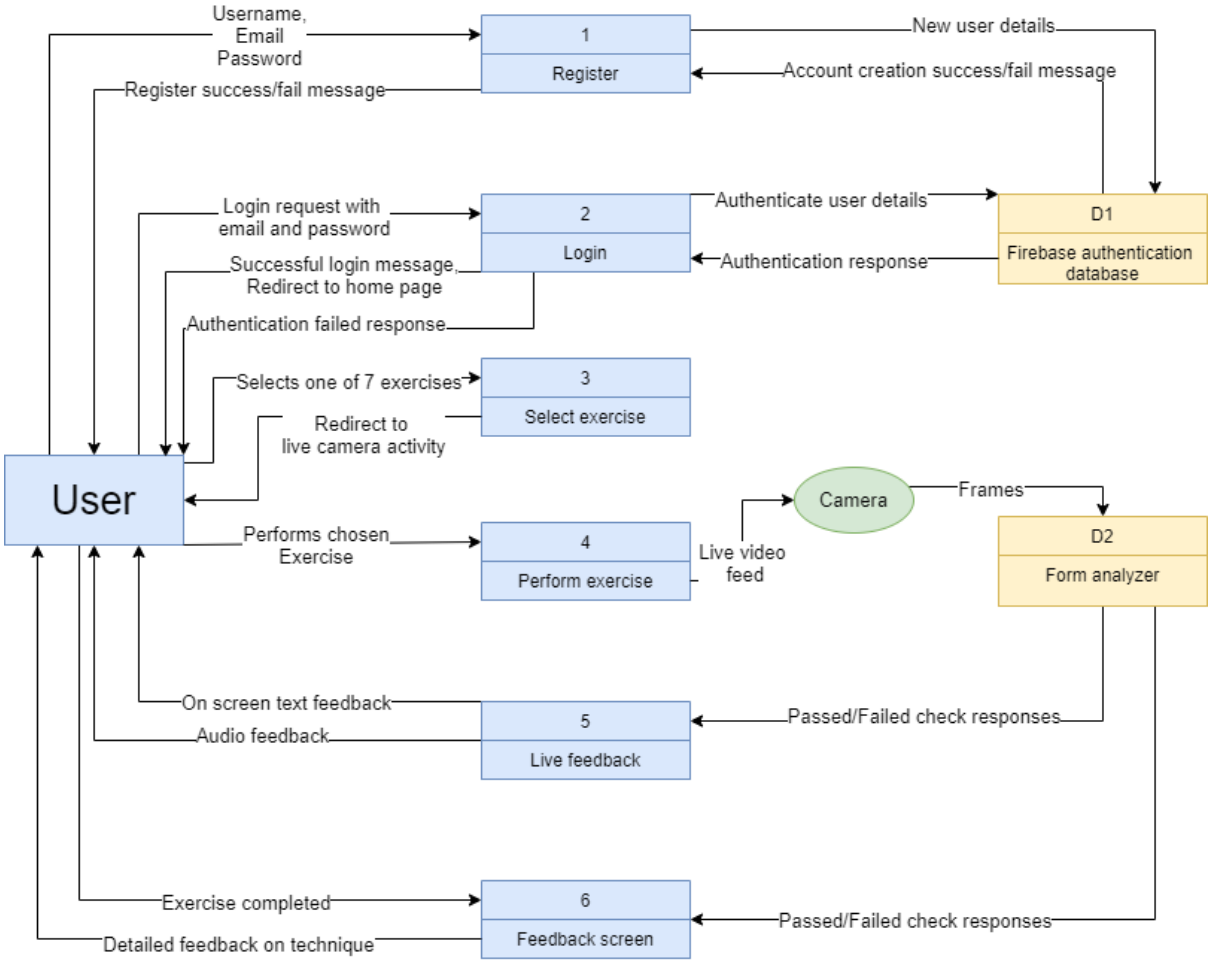
Figure 4



The diagram in figure 4 illustrates the relationship between the system and external entities and the various contexts of the system. The diagram illustrates both the windows and android implementations of GymVision. The diagram shows the key relationships between the different components that are key to the functionality of the system.

5.3 DATA FLOW DIAGRAM

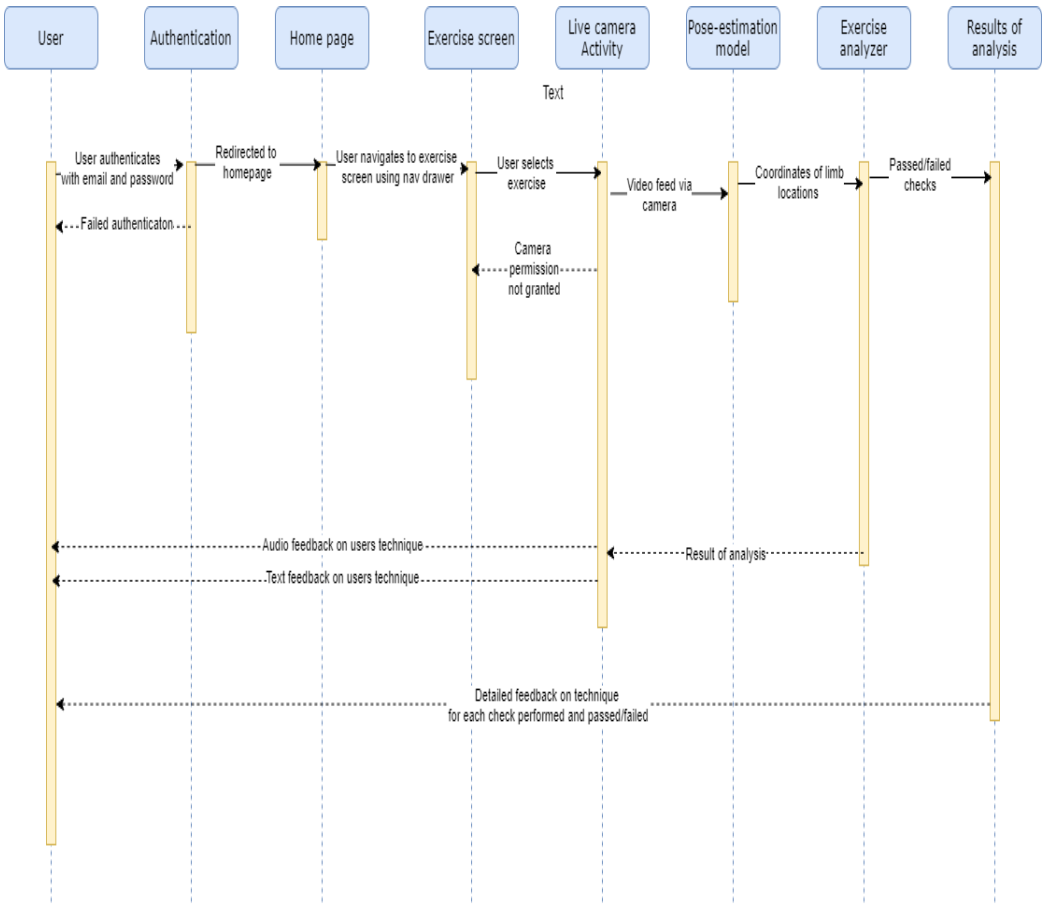
Figure 5



The diagram in figure 5 shows the flow of information through the GymVision application. It's useful for illustrating the flow of data from its origin to its destination and ultimately back to the user after all operations have been carried out. The user is the focus of the diagram and all the interactions and data the user provides is illustrated. For example, for registration and login the user provides username, email and password which is then passed to firebase authentication service and success or fail response is relayed back to the user. For exercise technique analysis the user selects one of the 7 exercises and is then bright to the live camera activity. From here the user performs the exercise in front of the camera and the camera activity relays the live video frames to the form analysis component. The results of the analysis are then passed back via live audio/text feedback and to the feedback summary screen so the user can review their technique after completion of the exercise.

5.4 SEQUENCE DIAGRAM

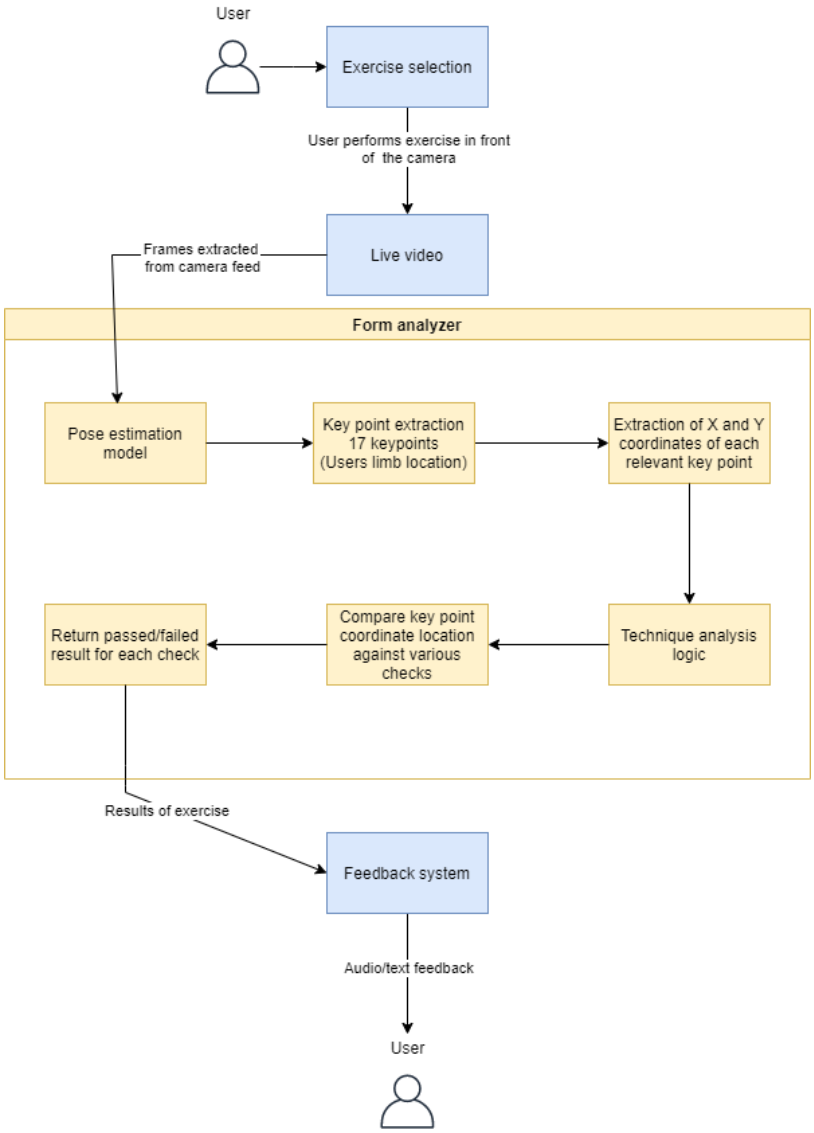
Figure 6



The diagram in figure 6 illustrates the sequence of events that are triggered from the start of the users interaction with the application to when the user has received feedback on their technique for a specific exercise and all the steps in between. The diagram highlights the flow of steps and the liveness of each activity the user performs.

5.5 BACK END ARCHITECTURE DIAGRAM

Figure 7



The diagram in figure 7 illustrates a closer look at the backend of the application and the events that take place in order to analyse the users technique and provide feedback. The user first must select an exercise they wish to perform and improve their technique for. The application will then start the live camera activity, the user should then stand in front of the camera and begin the exercise. Meanwhile the camera activity will be passing every frame to the pose estimation model, if a user is present in the frames the model will produce a mapping of the users Keypoints (Users limbs) and pass these Keypoints to the technique analysis code, which will extract the X and Y coordinate of keypoint. The technique analysis component will then compare the users X and Y coordinates to checks set by the application to analyse whether the user achieves correct technique. The results of the technique analysis component are then relayed to the feedback system as pass or fail result and the feedback system interprets these results and provides detailed feedback to the user to allow them to improve their technique next time they attempt the exercise.

6. IMPLEMENTATION

The GymVision application uses pose estimation models to detect the users key points or body parts along with their X and Y coordinates. Using the keypoint coordinates retrieved by the model the application assesses the users technique based on the users Keypoints and their positions relative to each other. GymVision is built on top of pre-trained pose estimation models through the use of openPose [8] for the desktop implementation and smaller model designed for mobile platforms FritzAI [11] for the android implementation. While working on different systems both models provide 17 Keypoints (Nose, Left wrist, Right elbow etc.) to the application to utilize for the technique analysis component.

6.1 LANGUAGE CHOICE

6.1.1 PYTHON

The language we used for the desktop application was Python. openPose provides a Python API that allows us to detect body parts using a pre-trained model. Our programming ability is significantly better in Python than in C, which is why we chose to use openPose with the python API rather than openPose in C.

6.1.2 JAVA

Since our mobile implementation of GymVision was designed for android it was necessary for us to develop the application using Java as it is the official language of android development. While there are other languages that can be used to develop android applications our existing knowledge of java meant that we were more comfortable programming in this language. GymVision was developed on the android studio IDE which is the official android development tool provided by Google.

6.2 USER INTERFACE DESIGN

6.2.1 ANDROID

For the design of the user interface and various screens of the android application XML language was used. While XML and the built-in android user interface tools provided a lot of flexibility and freedom when designing the UI, we also imported various libraries online to further make our application more appealing and ensure the experience for any user was seamless. A library provided by Airbnb Lottie Files [12] allowed us to import beautiful animations into the application to design our user interface. During user testing our users noticed these extra design features and reacted positively.

6.2.2 WINDOWS

The desktop version was designed using pyqt5, which is a binding of GUI toolkit Qt. Pyqt5 was simple and straightforward and allowed for easily customisable layouts. Through the use of pyqt5 designer we are able to view a layout before fully implementing it, this meant that we could determine our preferred style before committing to an implementation. We kept the design to very closely resemble the design used in android (e.g. same colour schemes and button formats). The design was more lightweight than android.

6.3 AUTHENTICATION

In order for users to access the application they must first register for an account and then login into the app. Users require an email and password to register. The authentication portion of the application is configured through Google's firebase authentication system. The authentication is hosted on the same server for both the android and desktop implementation meaning that a user can log in to both versions using the same account. Below is sample code showing the implementation of user registration and user login for both android and desktop. If the user forgets their password or would like to change it this is also possible and the user will receive an email with a link to change their password.

6.3.1 ANDROID CODE SAMPLE

Figure 8

```
private void CreateUserAccount(String email, final String name, String password) {
    mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener( activity: this, task -> {
        if (task.isSuccessful()) {
            updateUsr(name, mAuth.getCurrentUser());
            showMessage("Account created Successfully");

        } else {
            showMessage("Account was not created" + task.getException().getMessage());
        }
    });
}
```

Figure 9

```
private void signIn(String mail, String password) {
    mAuth.signInWithEmailAndPassword(mail, password)
            .addOnCompleteListener(task -> {
                if (task.isSuccessful()) {
                    sharedPrefEditor.putString("email", mail).commit();
                    updateUI();
                } else {
                    loginBtnProgress.setVisibility(View.INVISIBLE);
                    loginBtn.setVisibility(View.VISIBLE);
                    showMessage(task.getException().getMessage());
                }
            });
}
```

6.3.2 WINDOWS CODE SAMPLE

Figure 10

```
def createAcc(self,username,password,homeScreen):

    try:

        if username == "" or password == "":

        else:
            self.auth.create_user_with_email_and_password (username,password)
            homeScreen.setCurrentIndex(0)
    except:
        pass
```

Figure 11

```
def login(self,username,password,homeScreen):
    try:
        self.auth.sign_in_with_email_and_password(username,password)
        homeScreen.setCurrentIndex(1)
    except:
        pass
```

6.4 EXERCISE SELECTION

Once the user has successfully logged in they may navigate to the exercise screen using the navigation drawer. Once on the exercise screen the user may select one of the seven exercises, once an exercise selection has been made the application will pass a bundle of information to the live camera activity which includes the name of the exercise and which checks to perform. This is implemented this way so that we only need one modular live camera activity that can change based on the exercise coming in. This is also the case for the feedback screen, passing the name of the exercise from the start of the selection means that we can also have a modular feedback screen that changes based on the exercise that was selected.

6.4.1 ANDROID CODE SAMPLE

Figure 12

```
case "Squat": {

    bundle.putString(EXERCISE_KEY, "Squat");
    Intent intent = new Intent(context, InferenceActivity.class);
    intent.putExtras(bundle);
    context.startActivity(intent);
    break;
```

6.5 LIST OF EXERCIES AND THEIR CHECKS

6.5.1 SQUAT

The squat exercise has 3 checks.

1. Checks that your hip has gone below your knees in the movement. This is to ensure you are squatting low enough.
2. This checks that your feet are aligned, this is important for correct balancing and helps evenly distribute the weight.
3. This checks that your knees don't go too far forward in the movement.

6.5.2 SHOULDER PRESS

The shoulder press exercise has 2 checks.

1. Checks that your forearm is in a vertical position.
2. Checks that your grip width is not too wide or too narrow. Helping to target the correct muscles for the movement.

6.5.3 DEADLIFT

Dead lift exercise has 2 checks

1. This checks that your grip is in the correct position.
2. Checks that your feet are in the correct stance for the movement.

6.5.4 BENCH PRESS

Bench press exercise has 2 checks

1. Checks that you are bringing the bar low enough to your chest to reach a full range of motion.
2. Checks that your elbows are down low and close to your sides, if they are too high up it can put serious strain on the shoulder.

6.5.5 LUNGE

Lunge exercise has 4 checks

1. Checks that your front knee does not go past your front toes.
2. Checks that your back knee is going low enough in the movement to fully engage the muscles.
3. Checks that your hip goes to at least front knee height in the movement
4. Checks that you are keeping your back aligned with your hip and not leaning too far forward

6.5.6 SKULL CRUSHER

Skull crusher exercise has 2 checks

1. Checks that your upper arms remain perpendicular to the floor throughout the whole movement.

2. Checks that you are bringing the weights close enough to your forehead to ensure proper activation of the triceps.

6.5.7 BENT-OVER ROW

Bent over row exercise has 2 checks

1. This checks that your grip is in the correct position.
2. Checks that your feet are in the correct stance for the movement.

6.6 TECHNIQUE ANALYSIS

Once on the live camera activity screen the user can begin performing the exercise in live time and receiving live feedback on their technique. The technique analysis component works by comparing the X and Y coordinates extracted from the pose estimation model and using these coordinates we are able to check if a user has passed certain parameters as set in our code. In another section of the technical manual the checks for each of the 7 exercises are outlined.

For better understanding of the technique analysis function of the app here is an example of how the squat technique of the user is evaluated for one of the squat checks.

- In a squat, it is important that the squatter's hip falls just below their knees to achieve perfect technique and avoid injury. In this scenario we determined the Y coordinates for both the knee and the hip Keypoints for each frame in the movement. If the Y coordinate for the hip goes below the knee coordinate in the movement, we determined that the check was passed, and relevant feedback was given. However, if the Y coordinate of the hip never went lower than the Y coordinate of the knee, we determined that the user never went low enough in their movement and thus failed this check.

6.6.1 ANDROID CODE SAMPLE

Figure 13

```
public static Integer evaluateHipBelowKnee(@Nullable Keypoint leftKnee, @Nullable Keypoint leftHip) {
    if (leftKnee == null || leftHip == null) {
        if (leftKnee == null) {
            Log.d( tag: "Keypoint null", msg: "Left Knee");
        }
        if (leftHip == null) {
            Log.d( tag: "Keypoint null", msg: "Left Hip");
        }
        return 0;
    }
    PointF leftKneePos = leftKnee.getPosition();
    PointF hipPos = leftHip.getPosition();

    float kneeY = leftKneePos.y;
    float hipY = hipPos.y;

    return kneeY < hipY+10 ? 1 : 0;
}
```

6.6.2 WINDOWS CODE SAMPLE

Figure 14

```
if hipcheckCount <1:

    if self.hipCheck(human,image,pos):
        hipcheckCount +=1
        hipCheckPassed = True
if hipCheckPassed:
        cv2.putText(image, "Hips went low enough", (5, 70), cv2.FONT_HERSHEY_SIMPLEX, 0.5,(0, 255, ),2)
```

Figure 15

```
def hipCheck(self,human,image,pos): #
    if pos.yLeftKnee <= pos.yLeftHip:
        return True
    return False
```

For the above squat check, we're happy that once a user hip has fallen below their knees that they have successfully passed this check. However, for other checks we found that the user could have correct form for the majority of the exercise but would fail the check if their technique was wrong for 1 frame. We wanted to mitigate this as we believe if the user has correct technique 95% of the time their form is correct. To implement this we set some checks to be in the failed state and only pass if the user performs the correct technique for a certain amount of frames (e.g. 5 frames, frames are read in really fast!) alternatively some checks are set to the pass state and will fail if the users technique is wrong for a predefined amount of frames. For example, if for one frame in the movement the knee goes too far forward but is adjusted to be correct in the next frame this is still correct form.

6.6.3 ANDROID CODE SAMPLE

Figure 16

```
if (squat3 == null || counter_two < 3) {
    exerciseScores.put(ExerciseUtils.SQUAT_KNEES_TOO_FAR_FORWARD, kneesTooFarForwardResult);

    if(kneesTooFarForwardResult == 0){
        counter_two ++; //count all occurences of knees going too far forward

    }


    if (counter_two == 5) {
        this.runOnUiThread(() -> {
            checker3.setText("Knees too forward"); // Sets the visual feedback on the screen
            checker3.setVisibility(View.VISIBLE);

            tts( text: "Bring your knees back, they should be behind your toes");

        });
    }
```

6.6.4 WINDOWS CODE SAMPLE

Figure 17

```
if kneeCheckCount < 5:
    if self.kneeCheck(human,image,pos):
        kneeCheckCount+=1

if kneeCheckCount ==5:
    kneeCheckCount+=1
    kneeCheckFailed = True
if kneeCheckFailed:
    cv2.putText(image, "Knees went too far forward", (5, 90), cv2.FONT_HERSHEY_SIMPLEX, 0.5,(0, 0, 255),2)
```

6.6.5 DETRMINING TECHNIQUE ANALYSIS HEURISTICS

We determined the distance values for the exercises by repeatedly testing the application and adjusting the values in order to align with the research we carried out on the correct technique required for each exercise.

Process

1. Perform an exercise in front of the camera.
2. Label the exercise we are doing as either “good technique” or “bad technique”
3. If exercise was labelled “good technique” and checks fail, we inspect why the check produced a failed result and adjust the values accordingly.
4. If exercise was considered “bad technique” and our model checks passed, as above we inspected the reason for the incorrect pass result and adjusted the values accordingly.
5. Repeated 1-4 until the system outputs the expected results.

6.7 FEEDBACK

Another crucial aspect that we put a lot of thought and time into was the feedback system of the application. Following on the advice of our supervisor Dr Alistair Sutherland stressed the importance of having good feedback to the user in order to give them all the information they need at the appropriate time and appropriate specificity. The feedback given to the user on their technique can be divided into two sections, the real time feedback the user received whilst performing the exercise and the more detailed feedback the user received post exercise.

6.7.1 REAL-TIME FEEDBACK

Audio feedback

While the user is performing the exercise and focusing on their technique we found that it is difficult to look at the devices screen and focus on the exercise at the same time. As such whenever the user fails or passes a check they will be given audio feedback informing them of where they went wrong and how to fix it. This immediate audio feedback ensures that the user can correct their technique quickly to avoid potential injury. This was also critical for the universal accessibility of our application as it allows visually impaired users to also benefit from this application. The implementation of the audio feedback was through the use of text-to-speech libraries.

Figure 18

```
tts( text: "Bring your knees back, they should be behind your toes");
```

Visual feedback

The second mode of real time feedback was textual feedback printed on top of the camera feed. This feedback is short and concise and will only inform the user of the check they failed with more detailed feedback to follow. This feedback is printed to the screen whenever a check passes or fails. Allowing users to see their performance at all times throughout the movement, while also making the app accessible to users with hearing impairment.

Figure 19

```
checker3.setText("Knees too forward"); // Sets the visual feedback on the screen
checker3.setVisibility(View.VISIBLE);
```

6.7.2 POST EXERCISE COMPLETION FEEDBACK

In addition to the above feedback once the user has completed the exercise they will be taken to a summary screen, where the user can view detailed feedback on all the checks performed and whether they passed or failed the checks. In this summary screen the feedback goes into more detail and advises the user how to improve their technique for future exercise attempts. Once the user has read the information they may select another exercise and begin again.

7. PROBLEMS AND RESOLUTION

7.1 SINGLE PLATFORM

Problem

Initially we began development for a desktop implementation of GymVision only. This is because desktop devices potentially have much higher CPU and GPU processing power both of which are critical for pose estimation. Desktop screens are also much larger and clearer and allow the user to see the textual feedback much better. However, after researching the topic and consulting with our peers. We realized that target users would also want a portable implementation for use in the gym or the garden etc. where they typically perform their workouts.

Solution

We resolved this issue by developing the application on desktop as well as android platform, this way the user can use our application wherever they may be granted they have an android device. The android platform turned out to be a great success according to the feedback received on our final product from the user testing we carried out.

7.2 DISTANCE FROM CAMERA

Problem

The coordinates of the key points provided by the pose estimation model would slightly vary based on the distance of the user to the camera.

Solution

This issue was resolved by suggesting a distance of approximately 2 meters from the camera. While this is not the most optimal solution, we feel that the user will have to stand roughly 2 meters away from the camera so that their entire body is in the shot due to this we decide to use this solution going forward.

7.3 TESTING AND DETERMNINING CORRECT TECHNIQUE HEURSITICS

Problem

Our initial plan for testing and tweaking the heuristic values of the exercise checks, was that we would test our application on a large variety of users. We planned to oversee and conduct these tests throughout the development process. However due to the restriction imposed on us by the events of Covid-19, we were only able to test the application on ourselves and tweak the app accordingly.

Solution

While this problem was outside of our control by pressing on and thoroughly testing the application we found that the results produced seemed to satisfy the users that took part in our user testing.

7.4 UNDERSTANDING AND USING POSE ESTIMATION

Problem

This task was difficult in the beginning as we didn't have any knowledge of computer vision, pose estimation and training models. The lack of knowledge caused a slow start in our project as we had to study the library, models and the logic behind them, and also learn how to use it with our chosen platforms namely android and desktop.

Solution

Over time throughout the development of our application we slowly gained more knowledge and understood how pose estimation works and how to utilize the resulting data to enable us to carry out our exercise technique evaluation.

7.5 FEATURES MISSED

Problem

In our initial design of the application we had intended to include an option for the user to view history of previous exercises they performed. However, as we approached the deadline we still had not implemented this.

Solution

While attempting to implement this feature we realized that this function was unnecessary as there was little benefit provided to the user by viewing their previous attempts instead we focused on implementing another feature which was suggested by our supervisor. Which was to provide detailed feedback after the user completed the exercise. This is deemed to be the best course of action as the detailed feedback further enhances the users to improve their technique and promotes their safety while performing the exercises which is the core purpose of our application.

7.6 DIFFERENT ANGLES

Problem

Due to a lack of foresight, we failed to realize that certain exercises would need to have checks form multiple angles for example side view of the user and a front view of the user while performing a deadlift.

Solution

To address this issue for each exercise we chose the angle with the most important checks for that exercise. We selected these angles by researching the most likely errors in technique for each exercise that could cause injury. This ensured that our software was helpful but didn't overwhelm the user by making them perform the same exercise at multiple angles.

8. FUTURE WORK

8.1 ADJUSTABILITY FOR EACH USER

For future work in this project we would implement some better solutions for some of the problems we discussed above. For the problem addressing different distances from the camera, we would implement a scaling algorithm that would adjust the values in the exercises according to the distance from the camera and the users body measurements.

8.2 CATER TO MORE ADVANCED USERS

As our app is designed for beginners, it's guidance does not provide a solution to a "perfectly" performed exercise that an advanced bodybuilder would use. The main priority of our application is to ensure the users safety and warn them before an injury could occur. In order to achieve more advanced analysis, we would train a custom mode on thousands of examples of professional level form.

8.3 BROWSER SUPPORT FOR OUR APPLCATION

We would like to implement a docker-ized web app as the main platform for using the applications as there are a lot of dependencies and we would like the software to be easily accessible across multiple platforms. By implementing it into the browser we would significantly increase the portability of the application.

9. RESULTS AND CONCLUSION

In conclusion we are quite happy with the projects final state. We fulfilled majority of requirements we set for the project in the planning phase. At the start of this academic year GymVision was simply a farfetched idea that could allow anyone to analyse their exercise technique while performing free weight exercises from any location they choose using computer vision in Realtime without the need to depend on advice from the internet or a personal trainer. The application we have now serves exactly this purpose, any user regardless of previous experience can user our application on a windows device or android device to analyse their technique in order to achieve better results from the workouts and most importantly avoid serious injury that is often the result of incorrect technique.

Given more time and further opportunity to work on this project we would implement some of the previously mentioned features in the future work section. Most importantly we feel that the major shortcoming of the project was that it did not include purpose-built pose estimation model to analyse the exercise. Given more time we would seek to acquire the large amount of exercise samples of good and bad technique needed to train such a system further improve it.

From a learning perspective, we gained rich knowledge in new areas that we had not studied or come across before such as.

- How pose estimation models are trained.
- How convolutional neural networks work.
- Increased skill in android development and working on a large python project.
- How to develop GUIs in XML and Qt.
- Developing an accessible system.

10. APPENDICES

1. https://dl.acm.org/doi/abs/10.1145/2030112.2030226
2. https://ieeexplore.ieee.org/document/8856547
3. https://link.springer.com/chapter/10.1007/978-981-13-7123-3_69
4. https://www.researchgate.net/publication/277411291_Real-Time_Human_Pose_Estimation_and_Gesture_Recognition_from_Depth_Images_Using_Superpixels_and_SVM_Classifier
5. https://ieeexplore.ieee.org/document/8551559
6. https://www.tandfonline.com/doi/pdf/10.1080/02640411003671212
7. https://stronglifts.com/
8. https://github.com/CMU-Perceptual-Computing-Lab/openpose
9. https://www.healthline.com/health/fitness-exercise/compound-exercises#exercises
10. https://github.com/ildoonet/tf-pose-estimation
11. https://www.fritz.ai/features/pose-estimation.html
12. https://lottiefiles.com/