A SMART FITNESS ACTION CORRECTION AND EXERCISE ASSISTANCE SYSTEM BASED ON COMPUTER VISION AND ARTIFICIAL INTELLIGENCE

Jiarui Cai[1], Matthew Ngoi[2]

[1]Northwood High School, 161 Bishop Landing, Irvine, CA, 92620
[2]Computer Science Department, California State Polytechnic University,
Pomona, CA 91768

ABSTRACT

Obesity and a lack of motivation for exercising is considered a major problem in the world because they lower the quality of life for many people. Mobile fitness applications are an emerging solution to this problem because of the unique features that are provided [12]. They are seen as a vital tool to motivate people suffering from obesity. Our solution uses a mobile application and machine learning to detect, track the movements of users in a video and give an analysis of the exercise. It is made up of three essential components: the mobile application which acts as a frontend, the online storage, and the server which hosts our AI model. The program was limited by the accuracy and effectiveness of the AI model. We aim to test the accuracy of the AI model by providing it with five videos with different amounts of repetitions for each of the sample exercises of pushup, pullup, squat, and plank for the experiment, which uses the calculation of the percent error [13]. We resulted in mostly excellent results with few inaccuracies, shown through the average percent error being 2.09%.

KEYWORDS

Motivation, Mobile Applications, Machine Learning, AI Model

1. INTRODUCTION

Obesity and lack of healthy habits are severe problems that the majority of the population in the world struggle with daily. From 2015 to 2016, “47.3% of all American adults were considered as obese” [2]. Obesity can often lead to major health problems such as high blood pressure, coronary heart disease, and strokes [1]. For those who are trying to improve and are just starting their fitness journey, it will be difficult to know where to begin. Uncertainty of fitness objectives, possibility of injury from improper movements, and lack of exercising knowledge are all impediments that a beginner might encounter, which contributes to the unhealthy habits of the majority of the population in the world.

In our experiment, we tested the accuracy of the AI model. To perform the experiment, we provided the AI model with twenty videos in total, to give each exercise – Pushup, Planks, Squats, Pullup – 5 videos. Pushups, which we spent the most time on, were the most accurate, while the exercises with the least accuracy were pullups because each exercise utilizes different body points for detection for body movement since some body parts were more clear for detection

David C. Wyld et al. (Eds): NBIoT, MLCL, NMCO, ARIN, CSITA, ISPR, NATAP-2024
pp. 133-141, 2024. CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.140512

while others can cause inaccuracies. We originally hoped for our percent error for all of the exercises to be less than 5%, but it was close.

To measure the effectiveness of our solution, we looked at alternative methods to creating a fitness application designed to help and motivate people to pursue a healthier lifestyle. When examining Twazon, an alternative fitness application focused on users in Saudi Arabia, it was discovered that it was important to "meet the cultural expectations and standards of its audience". Our application attempts to fulfill the needs of those trying to start a healthier lifestyle by supplying them with basic knowledge and instructions of beginner movesets such as pushups and squats. Other fitness applications have also considered using AR and VR as a way to enhance the quality of service and provide effective and interactive information. They were particularly effective "amongst those with declining cognitive ability such as the elderly". Our solution also provides information through AR to convey analysis on the exercises. There were also alternative motion/body tracking techniques involving markers on the user's body or specialized cameras that can detect depth, however we opted to use a solution based on machine learning and computer vision as it was the least intrusive and most accessible option for users of the application.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Fitness Analyzer

There were a couple of issues regarding the use and application of Mediapipe in the fitness analyzer. One issue was a tendency for Mediapipe to inaccurately detect the positions of body parts in certain frames [3]. This would often result in the display of false error messages and false repetition counts. We resolved this issue by taking the readings of multiple nearby frames, that way single frame errors are not as significant. Another issue relates to the precision of Mediapipe. Body parts that stood relatively still were giving imprecise readings to our fitness analyzer. As such, we added a margin of error to increase the range of accepted values.

2.2. Communication Between Server And Application

Our solution involves communication between the mobile app that collects the user's information and exercise and the server which contains the analyzer. In order to provide communication, the mobile application can send different requests or tell the server to do different tasks based on unique links or routes [4]. There are several routes or links that the server possesses. Firstly, "/" corresponds to the home page which returns a string that verifies the app successfully connected to the server. Second, "/exercisevideo" analyzes the video from the request and uploads it to the database, which then returns a public link. In this route, the application sends the video in a request from the phone's gallery or camera. Lastly, "/exerciselist" feeds the mobile app with the list of available exercises in the analyzer. This is done upon the start up of the application.

2.3. The Firebase Storage

The fitness application uploads all analyzed results to the Firebase storage to create a temporary public link to the video that can be accessed from the app via network. The video in the Firebase storage will be replaced with a new result as soon as another analyze request is sent to the server. Users can have the option to save the video on their device locally before this occurs. Videos uploaded to the server can be selected from a recent recording from the device's camera or a previously recorded video saved in the device's storage. To correctly access the storage on different platforms, different builds of the application were created using Flutter.

3. SOLUTION

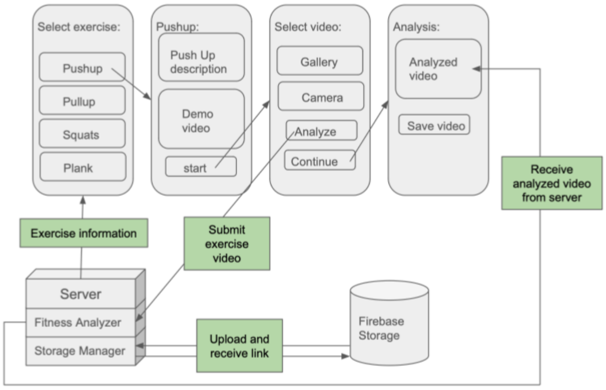


Figure 1. Overview of the solution

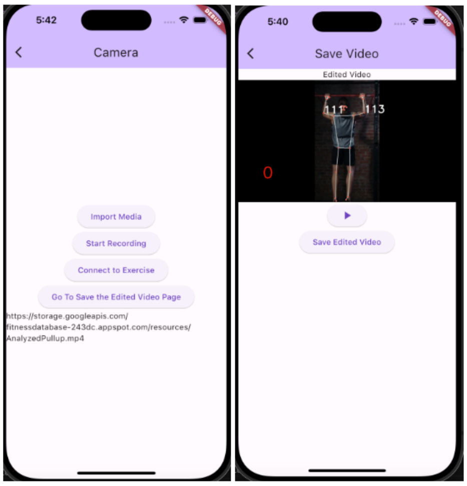
Our fitness program solution consists of three core components: the mobile application, the exercise analyzer, and the storage. To use the solution, users will download an application on their mobile device. As the user opens up the application, they will be able to view the list of exercises in which they can select. After they click on the specific exercise they would like to perform, they will be given a brief description about the specific exercise and the demonstration video on how it is correctly performed. Based on the demonstration video, the user will perform the specific exercise in the same perspective. When the user is prepared for the specific exercise, they are able to either import previous recordings or use the camera to record a new video. Users will be able to submit the video to the exercise analyzer hosted on a remote server.

The exercise analyzer receives both the video and the exercise being performed. It uses object detection to recognize parts of the athlete's body from a given image [8]. With that, the system is able to track the movements and position of the athlete. Each exercise has its own predefined set of restrictions that serve to detect any errors in the movement. Once the video has been analyzed, it is saved locally and uploaded to a remote storage online.

Videos uploaded to the remote storage can be accessed through a public link. This link is returned by the server in its response to the mobile application, where it can be displayed and saved on the device.

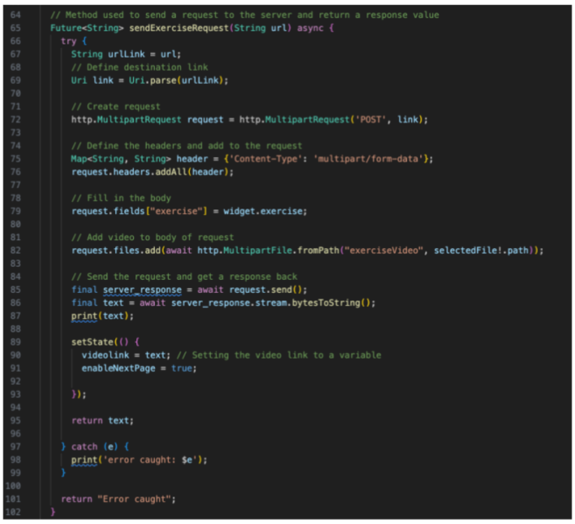
Once a response from the server is received, the application displays a new page that includes the analyzed video along with an option for the user to export it.

One core component of our solution is the mobile application. It was developed using Flutter allowing us to build an application on Android and iOS platforms. The mobile application allows for the user to record videos in a convenient way and communicate with the exercise analyzer. The application also provides the users with access to a simple to use interface.



https://storage.googleapis.com/fitnessdatabase-243dc.appspot.com/resources/AnalyzedPullup.mp4

Figure 2. Screenshot of the app

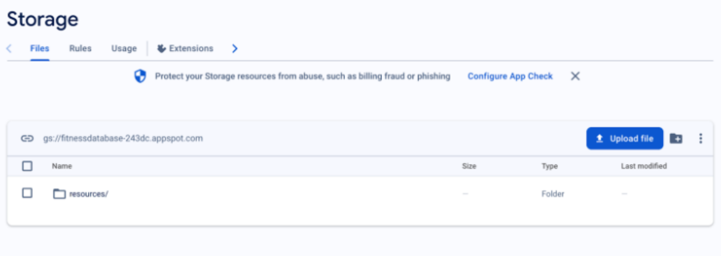


```
// Method used to send a request to the server and return a response value
Future<String> sendExerciseRequest(String url) async {
  try {
    String urlLink = url;
    // Define destination link
    Uri link = Uri.parse(urlLink);

    // Create request
    http.MultipartRequest request = http.MultipartRequest('POST', link);

    // Define the headers and add to the request
    Map<String, String> header = {'Content-Type': 'multipart/form-data'};
    request.headers.addAll(header);

    // Fill in the body
    request.fields["exercise"] = widget.exercise;

    // Add video to body of request
    request.files.add(await http.MultipartFile.fromPath("exerciseVideo", selectedFile!.path));

    // Send the request and get a response back
    final server_response = await request.send();
    final text = await server_response.stream.bytesToString();
    print(text);

    setState(() {
      videolink = text; // Setting the video link to a variable
      enableNextPage = true;

    });

    return text;

  } catch (e) {
    print('error caught: $e');
  }

  return "Error caught";
}
```

Figure 3. Screenshot of code 1

The screenshot above is a method used to send a request to the exercise analyzer hosted in the server. The method takes in a single String parameter which it uses as the request's destination. The method then creates a header for the request specifying the type of data being sent. And it also stores the exercise name and video containing the user's exercise set in the body of the request [5]. Once the method sends out the request, it waits for a response and extracts the link from the body as a String and stores it in the variable "text". This response is a link that leads to the analyzed video which can be accessed publicly. The setState() method is used to update the page's state variable "videolink" with the text received from the server. Once the link is received and the variable state is updated, the page renders the video on screen for the user to view and download.

Another core component of our solution is the online storage that is utilized. It was created using Firebase allowing us to store the analyzed video [6]. The online storage allows for the user to store their analyzed videos for long term access. The storage can provide a link that the user can access to both view and download their analyzed videos.

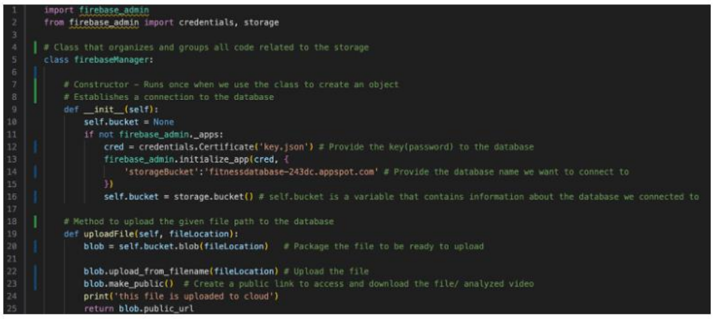


Protect your Storage resources from abuse, such as billing fraud or phishing Configure App Check

gs://fitnessdatabase-243dc.appspot.com

□	Name	Size	Type	Last modified
□	resources/	–	Folder	–

Figure 4. Screenshot of storage



```
import firebase_admin
from firebase_admin import credentials, storage

# Class that organizes and groups all code related to the storage
class firebaseManager:

    # Constructor - Runs once when we use the class to create an object
    # Establishes a connection to the database
    def __init__(self):
        self.bucket = None
        if not firebase_admin._apps:
            cred = credentials.Certificate('key.json') # Provide the key(password) to the database
            firebase_admin.initialize_app(cred, {
                'storageBucket':'fitnessdatabase-243dc.appspot.com' # Provide the database name we want to connect to
            })
            self.bucket = storage.bucket() # self.bucket is a variable that contains information about the database we connected to

    # Method to upload the given file path to the database
    def uploadFile(self, fileLocation):
        blob = self.bucket.blob(fileLocation)   # Package the file to be ready to upload

        blob.upload_from_filename(fileLocation) # Upload the file
        blob.make_public()  # Create a public link to access and download the file/ analyzed video
        print('this file is uploaded to cloud')
        return blob.public_url
```

Figure 5. Screenshot of code 2

The screenshot above is a class used to organize all of the code that is relevant to the storage component of the application. The class includes a constructor to help connect to the storage and a method used to easily upload files into the storage. The constructor runs only once and when the object is first created from the class. Using the provided JSON object as a key and the database name, the constructor establishes a connection to the storage. The variable “self.bucket” provides a handle to the storage and allows the rest of the program to perform storage operations by using Firebase’s API. The method “uploadFile()" uploads the given file path to the storage. It accepts the address of the file through the variable “fileLocation” , then uses a blob to prepare it for upload, and it eventually creates a public link that users can utilize to access and download the analyzed video. This link is returned by the method and used throughout the rest of the application to access the analyzed video.

The third core component of our software solution is the fitness analyzer that is hosted on our server. The fitness analyzer utilizes the library OpenCV for media handling and image processing. Individual exercises use Mediapipe for object recognition to track body parts. The fitness analyzer defines different rules and conditions depending on the exercises into separate classes.

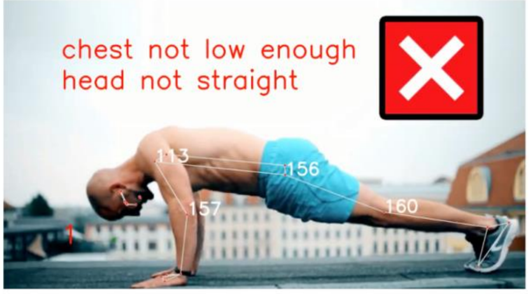
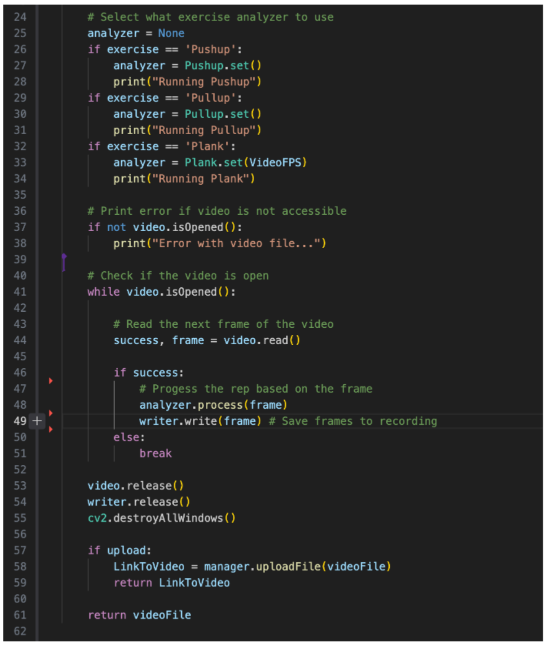


Figure 6. Screenshot of detection



```
24      # Select what exercise analyzer to use
25      analyzer = None
26      if exercise == 'Pushup':
27          analyzer = Pushup.set()
28          print("Running Pushup")
29      if exercise == 'Pullup':
30          analyzer = Pullup.set()
31          print("Running Pullup")
32      if exercise == 'Plank':
33          analyzer = Plank.set(VideoFPS)
34          print("Running Plank")
35
36      # Print error if video is not accessible
37      if not video.isOpened():
38          print("Error with video file...")
39
40      # Check if the video is open
41      while video.isOpened():
42
43          # Read the next frame of the video
44          success, frame = video.read()
45
46          if success:
47              # Progess the rep based on the frame
48              analyzer.process(frame)
49              writer.write(frame) # Save frames to recording
50          else:
51              break
52
53      video.release()
54      writer.release()
55      cv2.destroyAllWindows()
56
57      if upload:
58          LinkToVideo = manager.uploadFile(videoFile)
59          return LinkToVideo
60
61      return videoFile
62
```

Figure 7. Screenshot of code 3

The screenshot above is the code within the fitness analyzer method, which takes in three significant arguments: exercise, videoLocation, videoNameAnalyzed. From lines 25 to 34, the program utilizes the method's "exercise" argument to determine which type of exercise is going to be analyzed in the video. It is written using if statements to make the branching decisions. In lines 37 and 38, the program outputs "Error with video file…" if there is an issue opening the video. From lines 41 to 51, a while loop is used to loop through the data in the video. OpenCV edits and analyzes each frame of the imported video based on the different rules and conditions of the selected exercise in our analyzer. From lines 53 to 55, the video and writer is released from memory after the video has been processed. From lines 57 to 61, the program will either return the file location of the analyzed video or upload the video into storage in Firebase and return a public link which can be used to access the video via network.

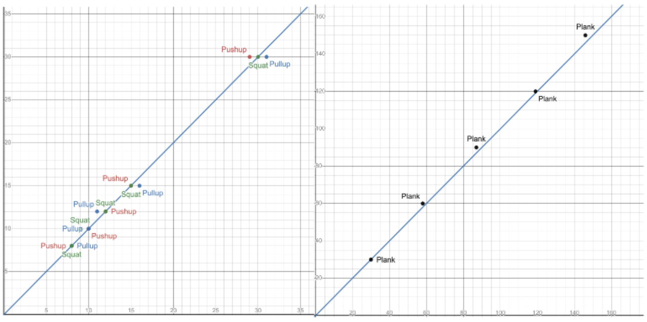
4. EXPERIMENT

In this experiment, we will be testing the accuracy of our exercise analyzers to accurately count the amount of repetitions that occur over a single exercise set. It is essential because the quality of the exercises are partially dependent on the amount of repetitions performed. Finding the correct amount of repetitions will give users a good indicator of their progress.

We are designing an experiment that tests the accuracy of the AI model. To do this, we will gather five videos with differing amounts of repetitions per set for the specific exercises of pushup, pullup, plank, and squat. Then the videos will be analyzed by our model for accuracy. The results from our model will be compared to the expected value from the video. Any differences between the two values will suggest some inaccuracies, whereas finding a matching value from the model will indicate its accuracy. We will use the percent error formula to precisely calculate the percent error for our AI model.

Exercise	Pushup	Pullup	Squat	Plank
Result of Trial #1: 8 reps/30 s	8	8	8	30
Result of Trial #2: 10 reps/60 s	10	10	9	58
Result of Trial #3: 12 reps/90s	12	11	12	87
Result of Trial #4: 15 reps/120s	15	16	15	119
Result of Trial #5: 30 reps/150s	29	31	30	146
Percent Error for Each Trial	#1: 0% #2: 0% #3: 0% #4: 0% #5: 3.33%	#1: 0% #2: 0% #3: 8.33% #4: 6.67% #5: 3.33%	#1: 0% #2: 10% #3: 0% #4: 0% #5: 0%	#1: 0% #2: 3.33% #3: 3.33% #4: 0.83% #5: 2.67%

Figure 8. Figure of experiment 1



x-axis: Observed Value

y-axis: Expected Value

***Percent error diminishes as you approach the blue line (where y-axis value = x-axis value)**

Figure 9. Figure of experiment 2

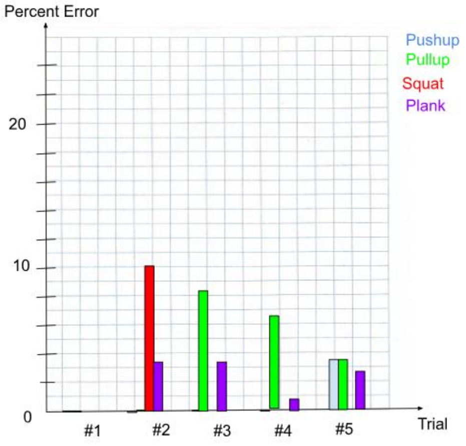


Figure 10. Figure of experiment 3

Pushups were the most accurate because it was able to correctly count the amount of repetitions in 4 of the 5 trials. The next closest exercise was the squats and it was also able to accurately count 4 of the 5 trials. However, pushups had a smaller percent error than squats, making it the best exercise analyzed by the program. Pullups were the most inaccurate because it was not able to accurately calculate the number of repetitions in 3 of the 5 trials. The results were extremely inaccurate with an average percent error of 6.11% from the inaccurate trials. Squats had the most extreme percent error of 10% despite being more consistent and completing more trials than pullup or planks. This value can be read inaccurately because there was not much repetitions required, so each miscalculation was extremely influential to the results. As a result of this, the percent error gets more and more influenced by the amount of errors, the less repetitions are being tested.

5. RELATED WORK

One method that is very similar to our fitness application solution is Twazon, a fitness app based around a Saudi Arabian audience that also attempts to target obesity and incite positive healthy changes. Similar to our application, Twazon hopes to use the unique features of mobile apps and the interactivity they provide to motivate individuals to achieve their fitness goals. The app was tested on seven usability attributes: effectiveness, efficiency, satisfaction, memorability, errors, learnability, and cognitive load. Results showed that “the app failed to meet the cultural expectations and standards of its audience” [9]. For example, many of its users practiced Ramadan, however the application did not provide any diet plans for this practice. The application also scored negatively for satisfaction and cognitive load. The exercise analysis feature in our program aims to make exercising easier and to reduce the amount of work users have to do to start exercising.

Another method currently being explored in fitness applications is the use of augmented reality (AR) or virtual reality (VR) [14]. In the paper written by Ryan Alturki and Valerie Gay, it was presented that “AR and VR mobile applications were particularly useful amongst those with declining cognitive ability such as the elderly” [10]. Both AR and VR provide efficient and effective ways at presenting additional information to the user and can improve the quality of service for many applications. The paper emphasized using AR and VR to four main features: 1) goal settings; 2) monitoring, tracking and feedback; 3) reminders and alerts 4) reward or gamification. Our solution provides some level of AR to demonstrate issues during a video playback [15]. It can also keep track of the amount of reps recorded in a video as a way for users to progress towards their goal.

Aside from using computer vision and machine learning to determine motion and body positions from video, there are also several other alternative methods for motion analysis mentioned in a paper written by Steffi L. Colyer, Murray Evans, Darren Cosker, and Aki Salo. One method utilizes markers adhered onto the user’s body parts which reflect infrared light back to track its position and distance from the camera and light source [11]. However, this can be too invasive and requires too many items to be prepared for an average user. Another method utilizes the use of cameras that can detect depth in its vision to graph and obtain the positions of body parts [11]. Both methods will require special or additional tools but can provide a higher range of accuracy than computer vision. Despite this, the use of computer vision for our fitness application was the most accessible method for motion analysis as it only required the use of software.

6. Conclusions

There were some limitations and improvements that could be made to our application. Some of the limitations in our application are the accuracy of the Artificial Intelligence (AI) and the limited amount of body points [7]. Because the AI model can only function when the entire body is shown, we were limited in how we could position the user in our app. The model was also limited in the amount of body parts which it tracked. This caused the inability for us to detect some poses in the body. For example, we are not able to find out if the user's back was straight because the AI model did not offer a point to track on the middle of the back. Some of the improvements that we can make to our applications is by adding more exercises, adding a new nutrition page, and adding a page for the users to login. Because getting enough nutrition is complementary to exercising, a nutrition page can be essential to increase the efficiency of the user's progress. Also, adding more exercises provides the users with a wider variety of options that they can choose. Lastly, adding a login page for the users can allow them to track their personal statistics and progress as they get more in depth into exercises.

References

[1] GBD 2015 Obesity Collaborators. "Health effects of overweight and obesity in 195 countries over 25 years." New England journal of medicine 377.1 (2017): 13-27.

[2] Brisbois, Tristin D., Anna P. Farmer, and Linda J. McCargar. "Early markers of adult obesity: a review." obesity reviews 13.4 (2012): 347-367.

[3] Bonilla, Diego A., et al. "Exercise selection and common injuries in fitness centers: A systematic integrative review and practical recommendations." International journal of environmental research and public health 19.19 (2022): 12710.

[4] Zolotukhin, Mikhail, et al. "Analysis of HTTP requests for anomaly detection of web attacks." 2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing. IEEE, 2014.

[5] Pauletto, Bruno. "Let's talk training# 1: Sets and repetitions." Strength & Conditioning Journal 7.6 (1985): 67-69.

[6] Kumar, Ashok. Mastering Firebase for Android Development: Build real-time, scalable, and cloud-enabled Android apps with Firebase. Packt Publishing Ltd, 2018.

[7] Santoni de Sio, Filippo, and Giulio Mecacci. "Four responsibility gaps with artificial intelligence: Why they matter and how to address them." Philosophy & Technology 34 (2021): 1057-1084.

[8] Zou, Xinrui. "A review of object detection techniques." 2019 International conference on smart grid and electrical automation (ICSGEA). IEEE, 2019.

[9] Alturki, Ryan, and Valerie Gay. "Usability testing of fitness mobile application: methodology and quantitative results." Comput. Sci. Inf. Technol 7.11 (2017): 97-114.

[10] Alturki, Ryan, and Valerie Gay. "Augmented and virtual reality in mobile fitness applications: a survey." Applications of intelligent technologies in healthcare (2019): 67-75.

[11] Colyer, Steffi L., et al. "A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system." Sports medicine-open 4.1 (2018): 1-15.

[12] Chen, Yu, and Pearl Pu. "HealthyTogether: exploring social incentives for mobile fitness applications." Proceedings of the second international symposium ofchinese chi. 2014.

[13] McLaren, Bruce M. "Extensionally defining principles and cases in ethics: An AI model." Artificial Intelligence 150.1-2 (2003): 145-181.

[14] Anthes, Christoph, et al. "State of the art of virtual reality technology." 2016 IEEE aerospace conference. IEEE, 2016.

[15] Azuma, Ronald T. "A survey of augmented reality." Presence: teleoperators & virtual environments 6.4 (1997): 355-385.