VizFit: Week 8 Milestone Documentation

Sasha Lawson Sam Donaldson
Alana Matheny Noah Buchanan

May 10, 2022

Department of Computer Science and Engineering
University of Arkansas – Fort Smith

Contents

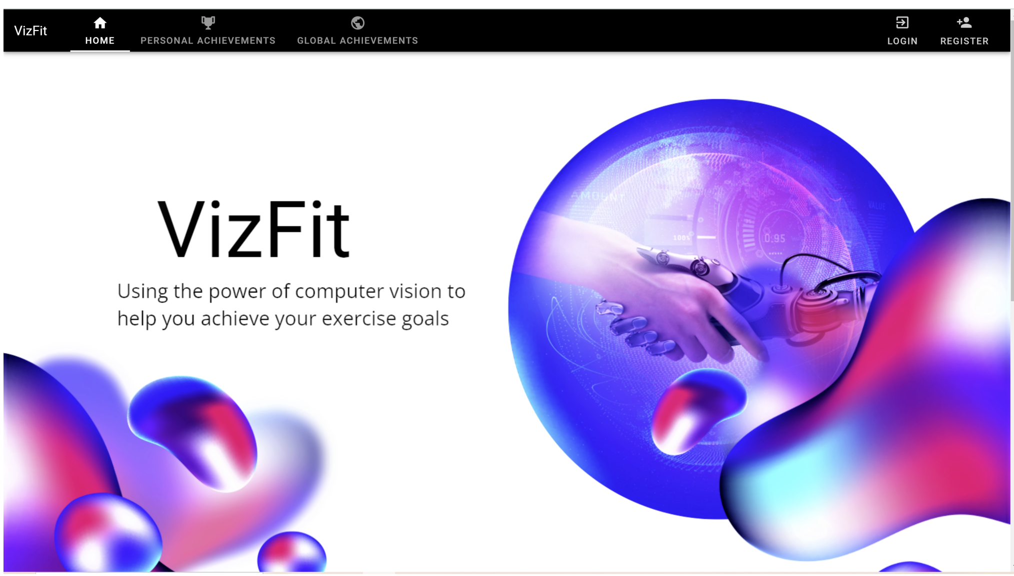
1 Summary Report

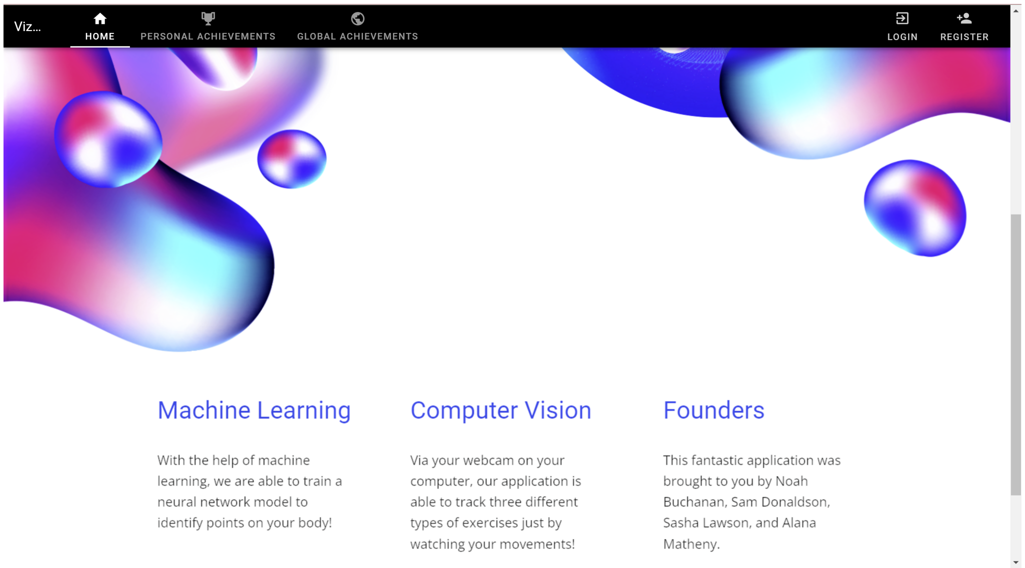
1.1 State of the Project

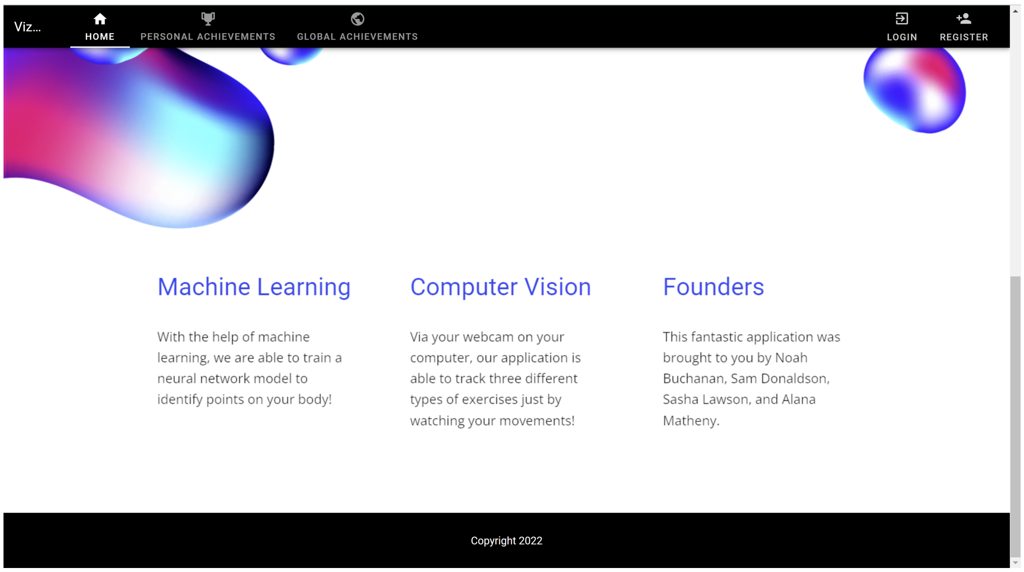
The state of the project so far can be seen through several lenses: front-end Vue application, the trigonometry functionality to determine exercises and, more importantly, the machine learning models.

- **Front-End:** Using Vue.js, the front-end of the application has been started. As of now a home, personal achievements, and global achievements page has been created, but not completed. Two drop-down forms, login and register, have also been created in which the two can be accessed on any of the three web pages previously stated. The next steps in this web page development will be to add more analytic charts and other statistical charts for the user to view, as well as add a 'users' web page where a user can edit their profile. The website will also be reviewed and refined in how it is presented. Below are the in-progress web pages along with the login and register forms.

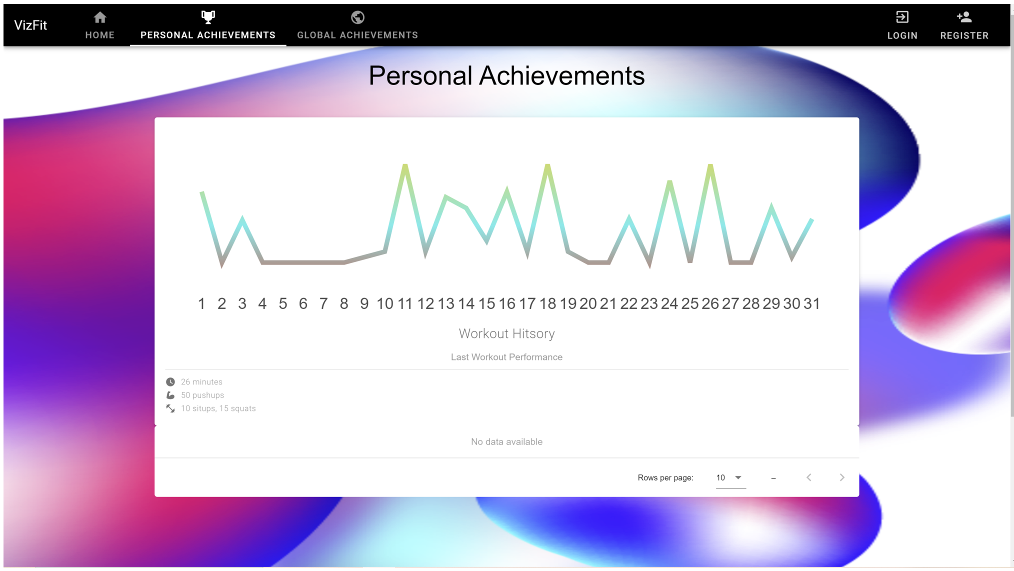
    – Home web page:

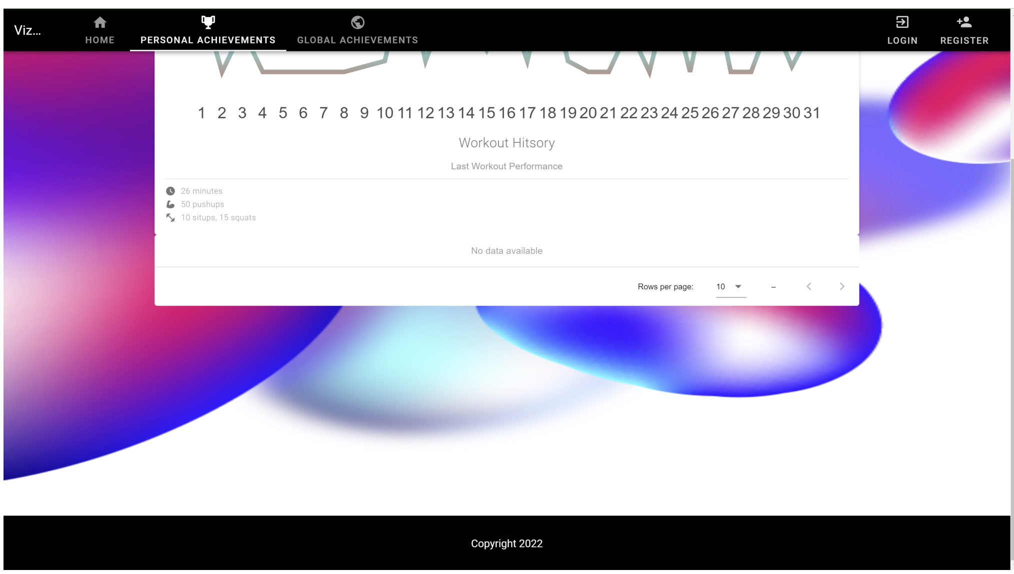




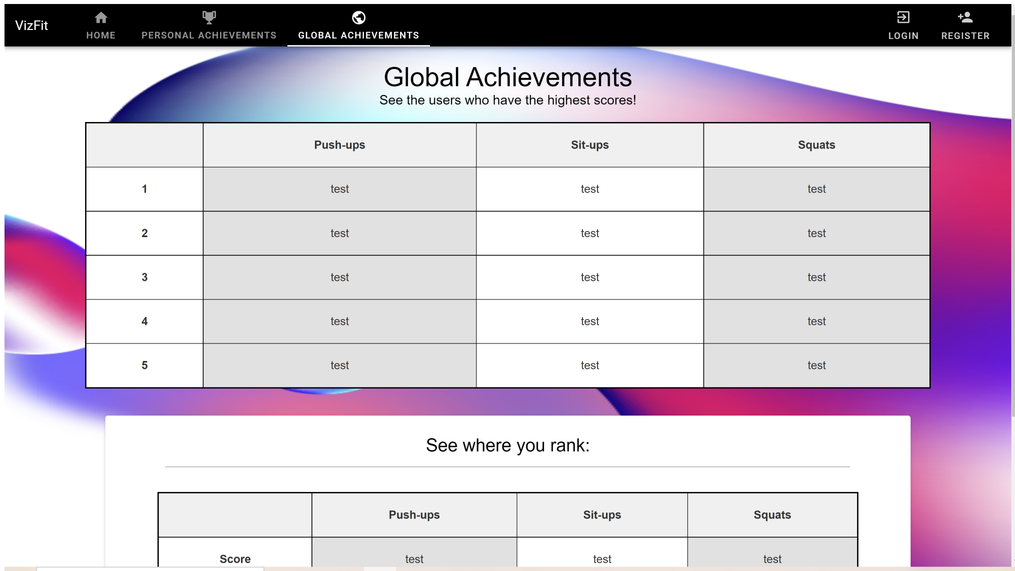


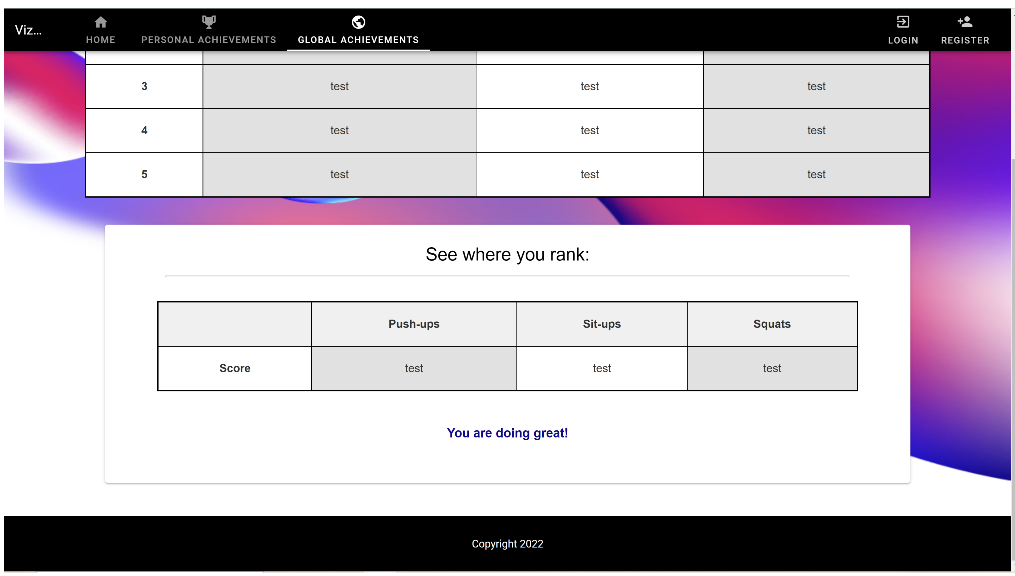
– Personal achievements web page:



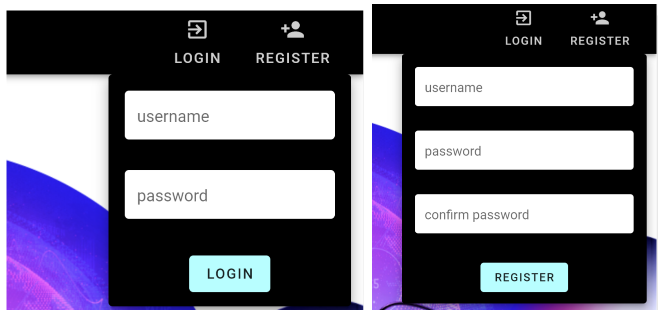


– Global achievements web page:



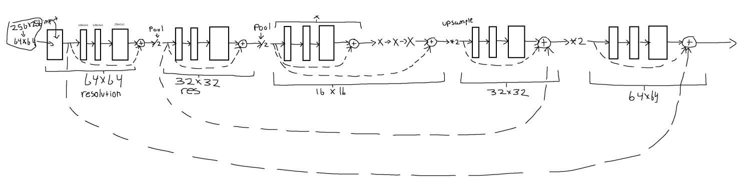


– Login and register forms:



- **Pose Estimation Models:** In this phase of the project a heavy focus was put into creating the team's custom wireframe model. Three model variations were made with varying results. Unfortunately, none have produced results at the level the team find satisfactory. In addition to these three models, a pre-built bound box model was found and tested in order to help better clean the image data being passed into the models.
  - Pre-Built Bound Box Model: This pre-built model uses Python's OpenCV and the Coco configuration and weights to form bounding boxes around various objects. In the case of this project, the focus is on identifying a person or people in a picture or given frame. The purpose of this is to clean the image dataset so that background noise is reduced by just focusing on what is inside the bound box i.e. a person. A cleaner dataset will allow the neural network model to more efficiently train; therefore, produce a better wireframe output. [1] [2] [3]

  - Custom Wireframe Model 1: This iteration of our model was our first attempt at creating and training it ourselves. The architecture consisted of a single simple hourglass module. With almost entirely unchanged data. The results were unsatisfactory given that the model was our first shot. Keep in mind that this illustration was still using a dense layer prediction after the convolutions. Below is a rudimentary sketch of what our model architecture looks like.

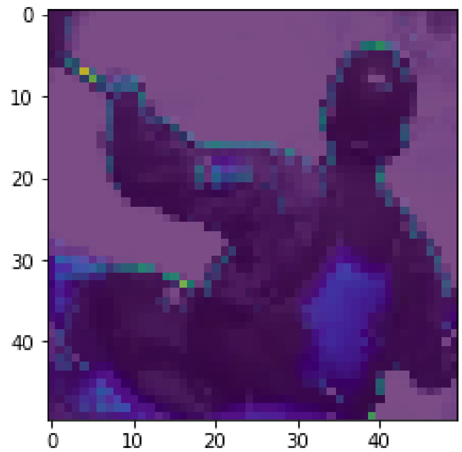
    

    The failures of this model was not as deterministic in its cause as we would have hoped. However, given the nature of the data we trained it on, we felt that we still had room to improve, which brings us to our second custom wireframe model.
  - Custom Wireframe Model 2: In this model architecture we attempted to augment the data according to best practice techniques in the research we have been using. We first tried cropping the images around the annotated person in each image with a 20 pixel buffer around the edges. At this time we were still using the same architecture and found no improvements it seemed. Further research led us to creating heat maps instead of

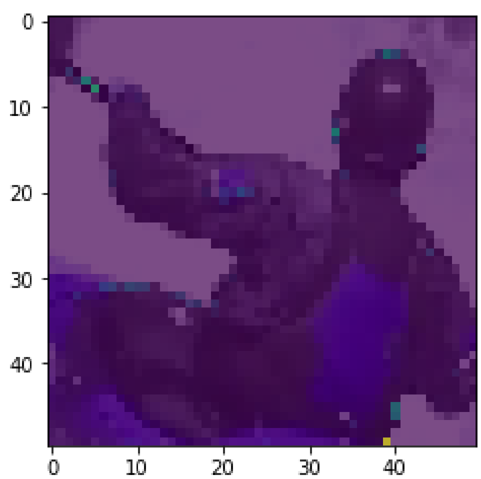
direct key point regression prediction from dense layers. Each key point is one 256 x 256 image with a single pixel with a value of 255 while the rest is 0, the image then undergoes Gaussian blurring and we have our first key point heat map. This is done for all 16 key points for each image. As a result of the heat map ground truths that we now have, the output of the model must match. We dropped our dense layer predictions and went with 2 convolution layers of output sizes 16x1x1 each to attempt to predict our 16 key points using heat maps. Below are some results from this models attempted predictions.

```
plt.imshow(imgn, cmap='jet', alpha=0.3)
```

```
t[199]: <matplotlib.image.AxesImage at 0x1dc180d4400>
```

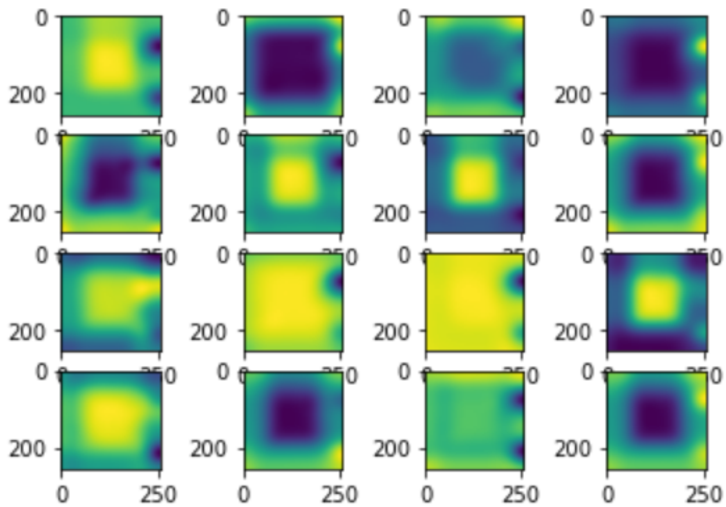


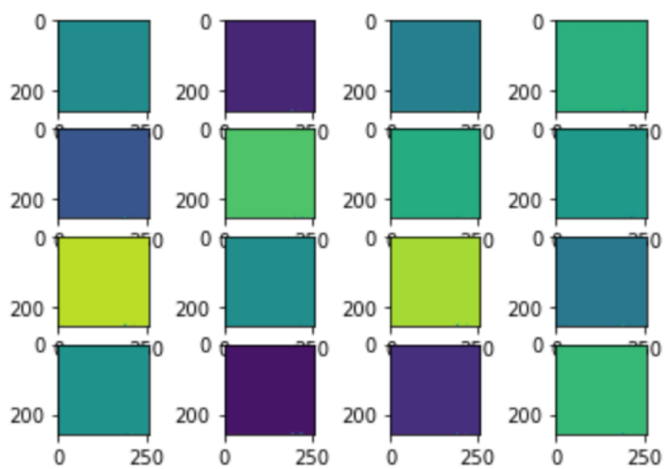

```
t[215]: <matplotlib.image.AxesImage at 0x1dc18de35b0>
```



As you can see the results were once again unsatisfactory. The first image shows our prediction with little to no training on a smaller scale and smaller resolution data set. The following is the same model trained further on the aforementioned data set. The results appear to disappear as the model trains further. This has been a reoccurring problem we have been consistently dealing with.

– Custom Wireframe Model 3: Our current model we are attempting to create has had further data augmentation and model architecture has been changed. More skip connections were added and we attempted to implement Leaky ReLU as well as changing the architecture of our skip modules to implement a conv − > add − > batch normalization − > Leaky ReLU setup to try and fix a possible vanishing gradient problem. Below are the results of this model.





Once again our models performance only decreases with time. The first image is from only about four hours of training. The following is after nearly a day and a half of training. Which leads us to our next course of action. We will change the architecture or perhaps find a new approach if we do not see results soon as well as augment the data further to making rotations and scaling of images randomly to add an aspect of rotation and scale invariance.

- **Exercise/Angle Detection Functionality:** The team's current approach to how the application will detect and ensure correct exercising is the use of angles and trigonometry functions. Given the points generated by the wireframe model trig functions can be created to determine the angles at certain locations on the body. Certain angle ranges at the correct locations can help determine the exercise and how well that exercise is being done. Currently, the application is checking three exercises: squats, sit-ups, and push-ups.
  - Squats Recognition: Currently, this very primitive squats recognition method looks at three keypoints on the right side of the body. The first points is at the torso, the second is at the knee, and the last point is right at the angle. The only angle being checked currently is the angle formed when the knee bends during a squat. If the angle less than or equal to 50° then this is considered to be an appropriate squat level; whereas, anything greater than 170° is considering starting standing position. In future implementations of the squat recognition method, a check will be unitized to ensure the individual's heels stay on the ground.
  - Sit-Ups Recognition: This initial version of the sit-up recognition method takes into account three keypoints on the left side of the body. One at the shoulder, another at the torso, and the last at the knee. As the sit-up motion is taken the angle produced at the torso between the leg and the upper body is measured. If the angle is less than 80° then that is considered a completed sit-up. If the angle is greater than 140° then that is the starting position where the individual has their back on the ground. Like before, more checks will be implemented as the exercise is more defined through trial and error testing. One such check is to ensure that the legs are not too far out or too close to the body.
  - Push-Ups Recognition: Like the methods before, this is in it's first phase of production. Additionally, it also initially focuses on just one angle using three points on the right side of the body. Point one being at the hand, point two being at the elbow, and point three being at the shoulder. The angle in question is formed at the elbow when the individual pushes up or goes down. If the angle is less than 60° the individual is on the ground. If the angles is greater than 170° the individual is "pushed up". Unlike the other methods, the push-up exercise has a lot more involved to it. For example, keeping your legs straight, what sort of elbow bends should be allowed, and how far apart are one's hands when on the ground. These factors and more will be considered and implemented if seen fit.
  - Issues and Moving Forward: Right now each exercise only takes into account one angle and one side of the body. Obviously these were initial test-runs. The next steps are to add

the other side of the body and add more angle checks to each exercise detection method. The more angle checks there are the more likely the correct exercise will be picked. Another current issue, especially present in the sit-up method, is that the wireframe on occasion has a difficult time sticking to the individual in the frame while they are preforming their exercise. This ultimately throws off the angles and can falsely add to the score. However, this issue should be solved with the introduction of more angle checks to each exercise method. Finally, as users test the application the strictness of the angle range will be changed as the user base gives their inputs.

1.2 Tasks Completed / In Progress

To summarize, the following are completed tasks.

- Week 8 milestone presentation and documentation
- Bounding boxes
- Chose the exercises for application
- Chose the functionality of how to detect exercises/angles
- Most initial front-end website pages

To summarize, the following are tasks in progress.

- Working on the pose estimation model
- Improve exercise/angle detection functionality
- Connecting API to the wanted aspects of the front-end
- Reviewing, adding to, and revising the front-end website

2 Technical Report

2.1 Individual Accomplishments & Performance

- Alana:
  - Contributed 'Front-End', 'Tasks Completed / In Progress', 'Front-End Deliverables', and 'References' to the milestone documentation
  - Contributed 'Progress to Date', 'Next Steps', and in general to the milestone presentation
  - Worked on the front-end website
  - Researched exercise datasets
- Noah:
  - Made multiple augmentations to the data set
  - Experimented extensively with different model architectures and data formats

  - Research on best practices to find new approaches where ours fall short
  - Assisted in the creation of the Milestone documentation and presentation

- Sam:
  -

- Sasha:
  - Found and tested Pre-Built Bounding Box Model
  - Created and tested Squats Recognition Method
  - Created and tested Sit-Ups Recognition Method
  - Created and tested Push-Ups Recognition Method
  - Milestone documentation and presentations contributions

2.2 Front-End Deliverables

- Home page
- Personal achievements page
- Global achievements page
- Login form
- Register form

2.3 Pose Estimation Model Deliverables

- Data augmentation python files
- Finished model architecture, unfinished with training

2.4 Exercise Recognition Deliverables

- Squats Recognition Method
- Sit-Ups Recognition Method
- Push-Ups Recognition Method

3 References

[1] M. W. Robotics and AI, "Object detection opencv python — easy and fast (2020)," Aug 2020. [Online]. Available: https://www.youtube.com/watch?v=HXDD7-EnGBY

[2] "Pose," 2020. [Online]. Available: https://google.github.io/mediapipe/solutions/pose.html

[3] A. Newell, K. Yang, and A. Arbor, "Stacked hourglass networks for human pose estimation." [Online]. Available: https://arxiv.org/pdf/1603.06937v2.pdf