Original Milestone 3 Goals:
ML model should be done training and create accurate bounding boxes. The blur effect should be able to take velocity into account. Work on integrating all students' work as one package should be in progress. Work on presentation should begin.

Progress:
ML model was retrained with different parameters/code and produces more accurate results. Motion tracking component successfully added to the model. Blur intensity takes velocity into account by adjusting a value based on pixel distance between bounding boxes, relative image size, and framerate.

On integration:
Functions concerning the machine learning model, loading the model and running the model on a video to track objects, are complete. They are found in Backend/main_site/tracking.py. These functions allow the web server backend to easily run the model on a video to get the detections, and pass them to the blurring/modification functions.

Functions regarding image processing/blurring are finished and abstracted. The functions have been packaged into a .py file so that it will be easy to use in the backend. Takes as inputs, the tracking boxes given by the model as well as the ids, path to a video, and path to store frames and store new video created.

Testing both tracking and image processing functions from above on the web page side of things is currently underway. Fine tuning and bug fixing are the main hurdles for integration right now.

Presentation/report:
Diagram of components added. Drafts for project and component description are present.

Changes:
File structure changed. Model.pt moved now found under Backend/main_site/Model.pt.
Even though the code for rotation is implemented and works, we no longer plan on using it in the finished product. This is because with our current bounding box implementation, we cannot efficiently or accurately determine the changes in angle for an object between two frames. The model only returns the top left and bottom right coordinates of a detected object's bounding box, making it entirely unreliable to interpret what kind of rotation, if any, happens. For example, an arm at a 45 degree angle that rotates 90 degrees will have the exact same bounding box after rotation, and our model has no way of detecting that as a rotation. However, so long as an uploaded video plays at a decent framerate, large rotations between frames should not be a concern.

Challenges/bottlenecks:
Because we are developing the application for use on a webpage, we only have access to a CPU. This results in the program running slow as we are processing videos. The best we can do to remedy this is to optimize our code to the best of our ability.

We had some technical problems caused by some members working on Windows and others on Linux. We are trying to make the code as cross platform and portable as possible but if there are problems, then try running on Windows as that is what the end product will be based on.

Team Member Progress:
Henry:
Retrained model with new training code and parameters. Produces more accurate results now. Wrote Backend/main_site/tracking.py, which contains the functions that run the model on a video and returns the detections as a list of bounding boxes for each frame, as well as a list of tracking ids for motion tracking. tracking.py also includes example operations. Will proceed to focus on writing the report and making the video.

Nancy:
Used sort to turn object detection into motion tracking. From https://github.com/abewley/sort

William:
Finished motion blur .py file that can be used in backend. Takes tracking boxes and ids from model, as well as paths for videos. Changes video into frames, insert frames where boxes are moving, and recombine frames back into a video. The inserted frames will be affected by motion blur which will detect how boxes move between frames, and add corresponding motion blur. These frames with added motion blur will then be added back between the two original frames.