# Milestone 3 Report

The main goal of milestone 3 is to work on web development to collect data for our trained models and output results for the users. We have split the team into two groups separately, with one responsible for the mood classification of music and the other for “emotion” detection of the faces.

## Current State for Team "Emotion"

After the model was trained to classify face emotions, we need a way to retrieve data (face of the user) for the model to predict the emotion on user’s face. We first imported OpenCV’s built-in cascade classifier called “haar” to obtain the coordinates of the box that enclosed the face. Then, we used that coordination to crop the face part out from the picture by using cv2.rectangle and converted it into arrays that could be fed into the model. We also saved the structure of the model in a json file and the weight of the model in a h5 file, so when we want to use the model for prediction, we could just load the model back using keras.model built-in function model_from_json. We uploaded a demo ipynb in GitHub to showcase this image processing function.

Next, we worked on front-end web development that focuses on GUI for users to interact with our system. For the input of our system (pictures of faces), we decided to adopt the method of uploading image files instead of using live video capture via webcam, and the reason for that will be introduced later in the paragraph. After the image has uploaded and goes through the ML prediction process, users can know the emotions detected by the system based on the uploaded images and get the recommended music from our system (Spotify Library Music Prediction Sub-System). To implement the ideas above, we used HTML, JavaScript to support our front-end web development. The code has been embedded in the back-end framework (Python Django). After trying several ways of uploading images, we have met the most challenging part for milestone 3, which is that we cannot implement this function only by the front-end system, it needs to call the back-end system and make a POST request to send a file to the backend. Based on this requirement, we used ajax library to build this bridge connecting the front-end and back-end system.

We originally tried to call the camera API embedded into the browser, and users can take a photo by calling the camera API of the browsers. The reason we have abandoned the idea of using video capture via webcam for data collection is due to the difficulty of building HTTP connection. Due to the default privacy and security settings inside chrome or chromium kernel-based browsers, it is impossible to call a camera API when the web is connected using an HTTP prototype. We need to enable this privacy setting in the chrome flag settings page, or we need to build HTTPS connections. However, to build HTTPS connections, an SSL cert is required, and the visitor needs to install this cert manually. Based on these concerns and any security/privacy apprehensions, we decided to abolish this function and ask the user to upload their photo manually via the web page.

## Current State for Team "Music"

For this third milestone, we mainly implemented our web framework using the Python based web framework, Django. We specifically decided to use the Django framework because it allowed us to proceed with our original proposal, which was creating a web app, without having to deal with the troubles of developing our own backend component. We closely followed an online guide of how to deploy our machine learning model using Django. Essentially, we started off by creating a Django server with database models and we were able to add our previously composed machine learning algorithms into them. To actually access the information from our models, we had to create REST API models using the Django REST Framework. One major obstacle we stumbled across in our music model this time was when we attempted to do a random search of 50 audios, audios corresponding to the energetic mood would not display any results. We believed this was because there were fewer energetic songs as compared to other moods; hence, we decided the best way would be to change the model so that it would search for 500 audios whenever searching for energetic songs. The drawback to this was that this would decrease the speed of the model but at least we would have audio corresponding to the energetic mood.

## Conclusion

So far, we are halfway through the web development of our project. For team “emotion”, we have set up the web server for our proposed application and have established the general GUI and visuals on our web page. We have also developed OpenCV code to prepossess the input image, which are made compatible with the model to read for prediction. For team “music”, we developed a song-selecting algorithm that randomly select 50 audios based on the input emotion. In the proposal, it is suggested that we finished our web development at this stage, however, only upon working on it, we realized that it is a huge work. So, we decided to separate the workload on web development and will work on writing music recommending algorithm that connects the emotion team’s prediction system and music team’s song recommending system in milestone 4 along with making demo video and final report writing, putting everything together.

## Work Distribution

Yisheng: Worked on front-end web development (UI, and request for image upload).

David: Wrote the report of team “emotion” and assembled the two team’s reports into a final milestone 3 report, wrote the OpenCV image processing algorithm.

Jacob: Used Django framework to incorporate machine learning algorithms and updated music model.

Jack: Wrote the music portion of the third milestone report