**What the Hack: Music Therapy Documentation**

Jordan Tioe, Dani Major, Edwin Kamgang, Cavan Dunkley

**Table of Contents**

1.  **Problem Statement**

------------------------------------------------------------------------------------- 6

1.1. Concept Design

------------------------------------------------------------------------------ 7

2. **Use Cases**

--------------------------------------------------------------------------------- 9

3. **Backlog User Stories**

------------------------------------------------------------------------------------------------- 12

Sprint Cycle 1

------------------------------------------------------------------------------------------ 12

Sprint Cycle 2

------------------------------------------------------------------------------------------ 14

Sprint Cycle 3

------------------------------------------------------------------------------------------ 16

Sprint Cycle 4

------------------------------------------------------------------------------------------ 18

4.  **Prototype**

------------------------------------------------------------------------------------------------- 21

5. **Architectural Design/Doc**

----------------------------------------------------------------------------- 25

Purpose

------------------------------------------------------------------------------------------------- 25

Scope

---------------------------------------------------------------------------------------------------- 25

Architectural representation

-------------------------------------------------------------------------- 25

Logical view

-------------------------------------------------------------------------------------------- 26

Process view

-------------------------------------------------------------------------------------------- 27

Physical view

------------------------------------------------------------------------------------------- 28

Development view

------------------------------------------------------------------------------------- 29

6. **Testing**

----------------------------------------------------------------------------------------------------- 30

Objective/Overview

----------------------------------------------------------------------------------- 30

Testing Gear and Setup

------------------------------------------------------------------------------- 30

Test Cases

---------------------------------------------------------------------------------------------- 31

Test 1: Webcam Access

--------------------------------------------------------------------- 31

Test 2: UI Functionality

--------------------------------------------------------------------- 32

Test 3: Emotion Finder

---------------------------------------------------------------------- 33

Test 4: Prediction Accuracy

---------------------------------------------------------------- 35

Summary and Observations

-------------------------------------------------------------------------- 36

7. **Version Control**

----------------------------------------------------------------------------------------- 37

8.  **User Help Guide**

---------------------------------------------------------------------------------------- 44

Product Name & Model number

---------------------------------------------------------------------------------------- 44

Intended use

-------------------------------------------------------------------------------------------- 44

Disclaimer

------------------------------------------------------------------------------------- 44

Features

------------------------------------------------------------------------------------------------- 45

Installation instructions

------------------------------------------------------------------------------- 45

Use/Operation

----------------------------------------------------------------------------------------- 47

Troubleshooting

--------------------------------------------------------------------------------------- 48

9. **Lessons Learned**

---------------------------------------------------------------------------------------- 49

10. **Glossary**

--------------------------------------------------------------------------------------------------- 50

**Problem Statement**

Every day, people carry themselves with a particular expression on their face depending on their moods and thoughts. Music is made with many emotions and cause many types of emotions in people. It is common for people to listen to certain types of music to cause different types of emotions. There are plenty of songs out there for people to listen to that can change their emotion and facial expression. People should have access to music quickly based off their emotions.

Facial recognition in apps has been popular, especially with the rise of snapchat in the last five years. Snapchat is best known for its facial filters. Snap chat is the is one of the only apps that can pick up on changes in facial expression but does not give the mood of the user.

The problem that our group is trying to solve is that there is an inherent lack of applications that allow users to receive feedback about themselves based off their facial expression. Nor are there any available applications to deliver a song recommendation to improve their mood based off of their facial expression. Through a google search you can find many apps that have facial recognition. When you go deeper into this search it stops there. When looking for a facial expression application you find a lack of. The applications that do exist tend not to work for its intended purpose. When looking up music therapy, the search will be wider than the previous. The issue is with the lack of facial expression apps, there is a close to none when it comes to apps that will give you musical feedback based analyzing your facial expression. Our groups focus is to bring these two ideas that have hardly been done and combine them to be one of the few applications that will give musical feedback based on the analysis of the user’s facial expression. We are providing a way for users, who look to music when feeling down or emotional, a way to multiple kinds of music with the click of a few buttons all based off their own face.

**Concept Idea:**

The idea of the application is to be a web application. When the web application is open, it will access the users web camera built in or plugged in through the USB port. The user can then take a picture using this video feed. There will be a button so that the user can click on the button when they are ready to take a picture of themselves. Once the picture has been taken, the photo should be analyzed for the facial expression of the user. If the program successfully can read the facial expression of the user, it will be able to tell the user one of six emotions. The six emotions the web application will be able to tell the user are Happy, Angry, Fear, Sad, Surprise and Neutral. Not only will the user be told one of six emotions but will be provided a little quote to read. This is just a little motivation to go along with the mood. The user will also be provided a song to match the mood given to the user. The song will be a YouTube hyperlink. It is hyper-linked, so the user can just click to instantly be taken to the song. The song is not intended to be the same song as the user’s mood. If the user is found to be sad, the web application is not going to give a sad song to the user. The song given to the user will be a happy and uplifting song to change the mood of the user. If the user is found to be angry, they will be provided with more songs of tranquility to calm the user down. Neutral can be provided with any song on the playlist. When the user is found to be in fright, they will be provided with more comforting music.

**Use Cases**

Use cases are a way of describing interactions between the user and the system. The following are use case diagrams:

**Use Case #1:**

|  |  |
| --- | --- |
| Use Case Name | System training for user’s face and mood |
| Time | Sprints 1 and 2 |
| Actors/Participants | User |
| Flow of events | 1. User poses in front of camera  2. User takes several pictures of happy and save them as happy  3. User takes several pictures of sad and save them as sad  4. User takes several pictures of neutral and save them as neutral  5. User takes several pictures of angry and save them as angry |
| Entry Condition | Face should be in a normal condition, so machine has higher accuracy |
| Exit Condition | All moods of user are saved |
| Quality Constraints | Picture should be taken in good lighting |

The user interacts with the camera (system), where the user takes a picture with an emotion and saves the picture as that emotion in a jpeg format. Then the user continues on taking the next picture with another emotion, and the same can be done for the rest of the emotions.

**Use Case #2:**

|  |  |
| --- | --- |
| Use Case Name | Happy Face Recognition (Also applicable for sad, angry, neutral) |
| Time | Sprints 2 and 3 |
| Actors/Participants | User |
| Flow of events | 1. User poses in front of camera  2. User takes a picture  3. User clicks on “check mood” button  4. User can see analysis of mood in text-field window  5. User can see song recommendation (youtube url)  6. User can click on the song url link |
| Entry Condition | Live picture of a happy face |
| Exit Condition | Mood feedback and song recommendation |
| Quality Constraints | Accuracy of training datasets |

The user interacts with the camera, the “check mood” button, and the mood analysis display. After the user takes a picture, the user finds out what mood he/she has. Then the user can see the mood feedback and song recommendation.

**Use Case #3:**

|  |  |
| --- | --- |
| Use Case Name | Have the program take the still picture and analyze the picture for the face of the user |
| Time | Sprints 2 and 3 |
| Actors/Participants | User |
| Flow of events | 1.Open webcam/application will open the webcam for you  2. Webcam will be opened in video capture set up  3.User will click button/keyboard input to take still image  4. Image is converted into a jpeg image  5.Program takes in the jpeg image and scans the picture for the face that is supposed to be detected  6. A circle will appear of the detected face in the image taken by the user |
| Entry Condition | The webcam takes a single frame image |
| Exit Condition | A face is detected within the image |
| Quality Constraints | This whole process should be done with just a few seconds, after the image is taken it should only be a few seconds for the face to be detected |

The user interacts with the camera. Similar to Use Case #1, the picture is taken, saved as a jpeg file, and then the photo is analyzed to determine what mood the user has.

**Use Case #4:**

|  |  |
| --- | --- |
| Use Case Name | Assign mood to user based off of facial expression from the list of: happy, sad, fear, anger, surprise, neutral |
| Time | Sprints 2 and 3 |
| Actors/Participants | End user, back-end program, image input |
| Flow of events | 1.Collect data and images of faces expressing the 7 different emotions.  2. Train the program to recognize the features that distinguishes each face and classify them into clusters  3. Setup the program to associate each emotion with a corresponding cluster that consist of different features of the face during the expression of an emotion  4.Train the program to be able to find the correct the emotion on the face of a new image input by identifying the features on the face and finding the cluster with the most similarities  5.Assess the level of accuracy of the prediction by prompting the user for a review  6. The program should output an apologetic phrase to the user if it was unable to find an emotion. eg:Sorry we are unable to detect your emotion |
| Entry Condition | When the program detects a face on the saved user’s image |
| Exit Condition | When the program finds an accurate match for the image’s emotion |
| Quality Constraints | 1.The emotion on the face should be detected within 2 minute after the image is saved.  2. Ideally, the program should detect the correct emotion 7 times out of 10 (Best case scenario) |

The user interacts with the mood checking and analysis, where the user finds out what mood he/she has. This is determined from the training of the program to recognize different emotions based on the accuracies of different datasets.

**Backlog User Stories**

|  |
| --- |
| User Story 1: As a user I want to allow the app to access my webcam, so that I can take a picture. |
| User Story 2: As a user I want to take a picture of my face using my webcam, so that I can evaluate my facial expression. |
| User Story 3: As a user I want my emotions to be read based off of my picture, so that I am aware of my emotions. |
| User Story 4: As a user I want to receive feedback based off of facial expression, so that I can improve my mood. |
| User Story 5: As a user I want to have the ability to configure the number of songs for each mood, so that the song selection process is totally random for each mood. |
| User Story 6: As a user I want to follow a link to a recommended song, so that I can listen to it. |

Disclaimer: During the first two sprint cycles, we used Python Tkinter to set up the camera user interface. However, since Tkinter could not be run as a web application, we had to refactor our application by using HTML and Flask throughout the last two sprint cycles. This was a good lesson learned to do a complete analysis in our subsequence solution approach.

We have a total of 4 Sprint cycles, in which each sprint lasts for 2 weeks.

**Sprint Cycle 1**

During this Sprint Cycle, we worked on User Story 1 and User Story 2.

User Story 1 has a list of tasks to display webcam using OpenCV, start a thread that constantly pools the video sensor for the most recently read frame, set a callback to handle when the window is closed, and allow python to display the webcam.

To satisfy the required tasks for the first User Story, we chose to use Tkinter to build the Graphical User Interfaces because of its robust and independent platform windowing toolkit. This includes the windows for the live webcam and a button to take a picture.



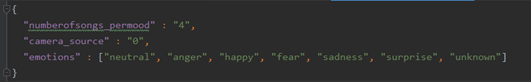
User Story 2 has list of tasks are as follows: research on available implementation options to take a still picture, implement the selected option to capture picture when button is clicked, add a capability to save the picture in jpeg format, and add the capability to save onto a directory for future emotional analysis. To satisfy those requirements, the snapshot button was created for the user to take the photo and save it as a jpeg file, named as ‘myPicture.jpg’.



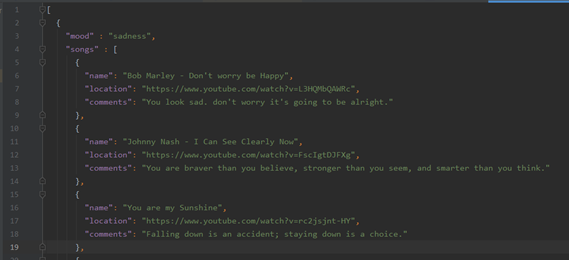
**Sprint Cycle 2**

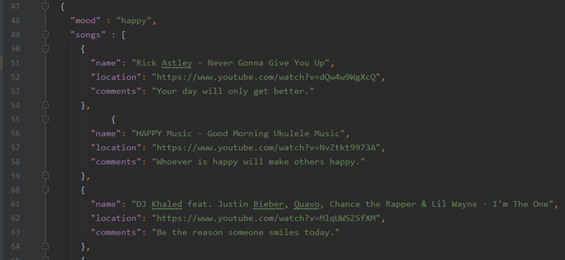
During this Sprint Cycle, we worked on User Story 3, User Story 4, User Story 5, and User Story 6, as well as finishing up the remaining tasks from Sprint Cycle 1.

The task was to allow the user to choose which camera to use if more than one camera is connected. To satisfy the task, we modified the application settings json file to configure the video source, so that we can enable and use any USB camera to take pictures. If the camera source is set to 0 (shown below), then the default camera is viewed from the computer. If it is set to 1, then the camera can be viewed from a USB camera.



The tasks for User Story 3 are the following: read input picture then crop face and resize picture to match dataset, predict the picture against the dataset, and pass result of the mood back to the main program. After the picture was saved as ‘myPicture.jpg’, the picture would be cropped, then saved as ‘myPictureCropped.jpg’. The purpose of cropping the faces is to allow the datasets to match up the predicted emotions. We used convolutional neural networks for training the computer to recognize different emotions, which are happy, sad, neutral, anger, surprise, and fear. For User Story 4, the tasks required are that each emotion should have list of corresponding feedbacks, and program the application to randomly display a feedback based on the input. To satisfy the required tasks, we used a song configuration json file to store the list of songs and mood feedbacks associated with each emotion. This allows user to configure songs without having to change the application code. The json file is called after the mood is checked. These tasks also satisfy User Story 5 and 6, in which we modified the song configuration file. In addition, we also have to modify the application settings to add a new variable to configure the number of songs per mood. This also allow user to make application settings without having to change the code.





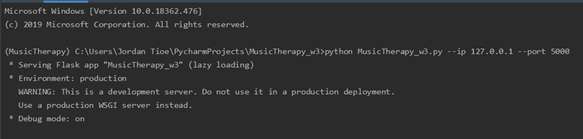
**Sprint Cycle 3**

During this Sprint Cycle, we added User Story 7 and User Story 8.

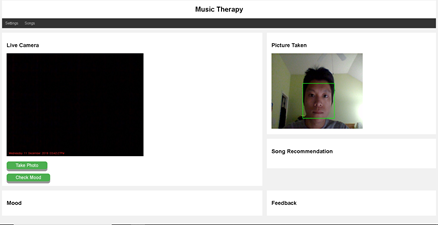
|  |
| --- |
| User Story 7: As a user, I want to be able to access a working web application to take my photo. |
| User Story 8: As a user, I want to use an interface in the browser, so that I can see my photo and receive feedback via the internet. |

As mentioned earlier, since Tkinter could not be used, we would have to refactor our web application by creating a Flask project. This was a setback for the team, because we had to do additional analysis and design of the Graphical User Interface.

The task required for User Story 7 is to implement the web integration for user to take a photo via a browser, in which we would have to create a Flask project, use the Flask project to create a development server, and build a URL using Flask. To run the application, we would need to go to the terminal and type the following command: python MusicTherapy\_w3.py --ip 127.0.0.1 --port 5000. MusicTherapy\_w3 is the name of the main program, and 127.0.0.1 (localhost) is the URL that allows the program to run on the server.



Finally, the tasks required for User Story 8 are the following: Use Flask to create canvas to display video stream, create HTML page to communicate with Flask in Python script, and create working buttons that allow the user to take picture or check mood. To satisfy the requirements, we were able to create the camera user interface successfully, along with the video stream using Flask and HTML. HTML and CSS were both used to create the buttons and the overall layout of the web application.

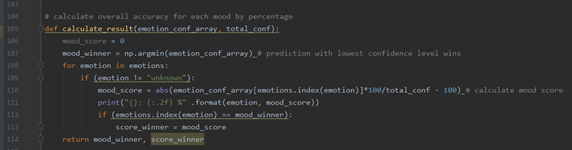


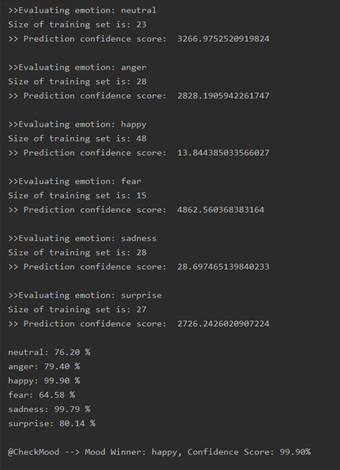
**Sprint Cycle 4**

During Sprint Cycle 4, we put finishing touches on our application, as well as writing the documentation for this project.

|  |
| --- |
| User Story 9: As a user, I want to be able to see of the accuracy of mood prediction for all the six moods defined. |
| User Story 10: As a user, I want to have a full documentation of how the project is built. |

The problem that we ran into regarding the accuracy of datasets was that the expected mood was not always consistently accurate. Therefore, we decided to print out the accuracies for all six different moods, as well as the prediction confidence to compare the different results for each emotion. The emotion with the lowest number returned from the model (confidence) would result in the best accuracy. To calculate the accuracies for each mood, the prediction confidence of a mood is first multiplied by 100, then is divided by the total prediction confidence for all six moods. In addition, subtract 100 again and take the absolute value of the result. The mood with the best accuracy is the mood being displayed on the screen.

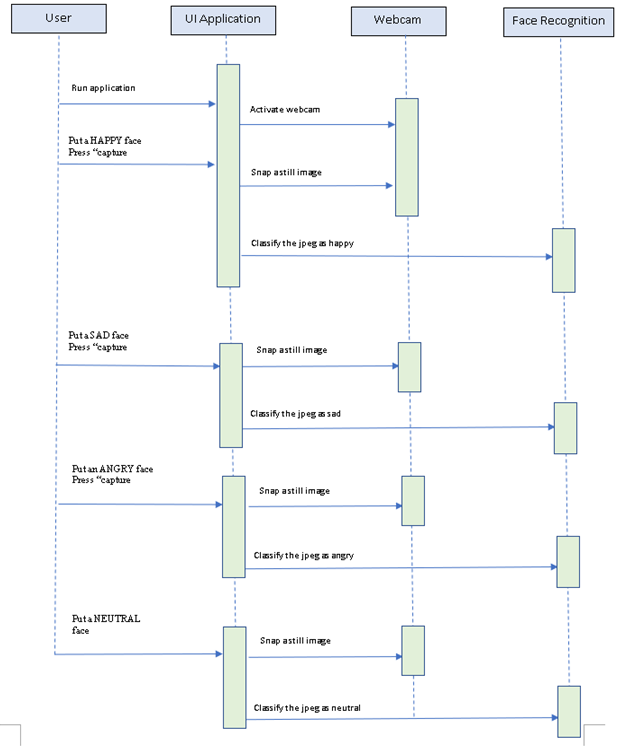




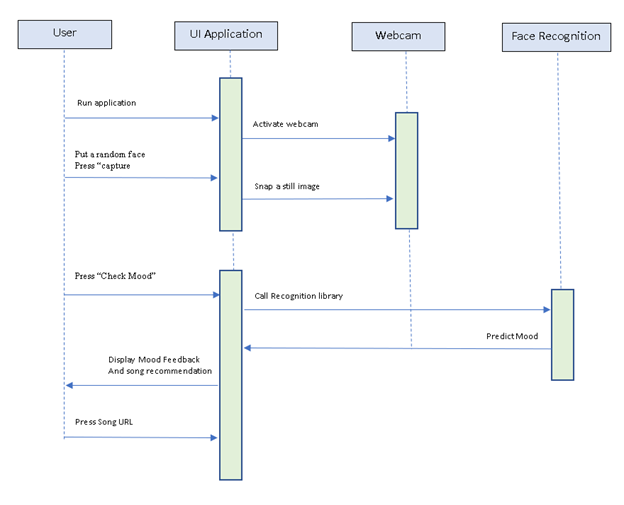
**Prototype (Drawings):**

This section describes the diagrams for the final prototype for this software. The diagrams are sequence diagrams which show interactions between the actors and objects involved as well as between actors themselves for the different functionalities of system and are helpful to have a better understanding of what is going on under the hood.

The first diagram represents the steps involved in the use case intended for training the functionality of the software that predicts the mood. The initiator is the user(the software developer) who starts the interaction by running the application then taking pictures through the user interface which is activated by the webcam. The image is saved and sent as output to the last actor the face recognition.This sequence diagram in peculiar in that there are no arrows going backward. This because it describes a one sided interaction that the developers with the system where the goal is to feed the actor "Face recognition" faces expressing different moods and classify them appropriately for better accuracy.



The second sequence is a direct interaction between the common user and the software. It describes the normal course of action when a user uses the software, therefore it is an overall representation of the system as a whole and the main actors. When the user runs the application the user interface is activated by the webcam and displays the output of the webcam. The user then takes a picture with a mouse click on the user interface and the picture is saved. The user then presses the check mood button on the user interface. The UI then calls the recognition library and gives the saved image as input to the face recognition actor. The primary function of the face recognition actor is to match the image to a mood. Upon a successful match it sends the output to the user interface which then displays the recognized mood as well as a mood feedback and a song recommendation as a youtube link. The last interaction involves the user clicking the song url to be directed to a youtube website.



**Architectural Document:**

**Purpose**:

This section is concerned with documenting the architecture of our project. It provides an overall architectural overview of the system, using some UML architectural diagrams and different architectural views to depict different functionalities of the system. It is intended to capture and convey the important architectural decisions which have been made on our system.

**Scope:**

This software architecture document provides an architectural overview of our music therapy software system developed for emotion recognition.

Acronyms and Abbreviations: see glossary

**Representation:**

This document presents the architecture as a series of views; use case view, logical view, process view and deployment view. These are views on an underlying Unified Modeling Language (UML) model.

**Goals and Constraints:**

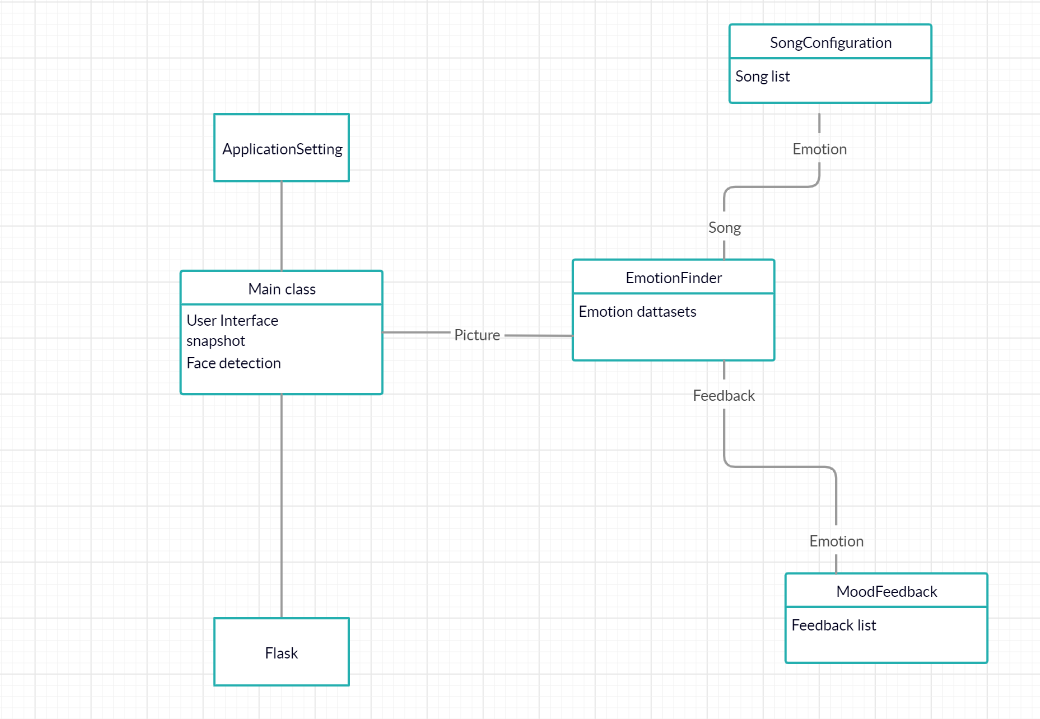
There are several goals and constraints on the architecture of the system. The existing system responsible for saving the pictures need to be secure to make sure no one else from the user can access it. The existing system responsible for emotion analysis has a time constraint and must respond to the user in an adequate amount of time. The interface system must have access the camera and display it's output so it is visible to the user.

**Logical view:**

This is a description of the logical view of the architecture. It describes the most important classes, their organization as subsystems into layers. Also describes the most important use-case realizations, for example, the dynamic aspects of the architecture. A Class diagrams is used to illustrate the relationships between architecturally significant classes and subsystems.

The logical view of our project has 3 main classes:

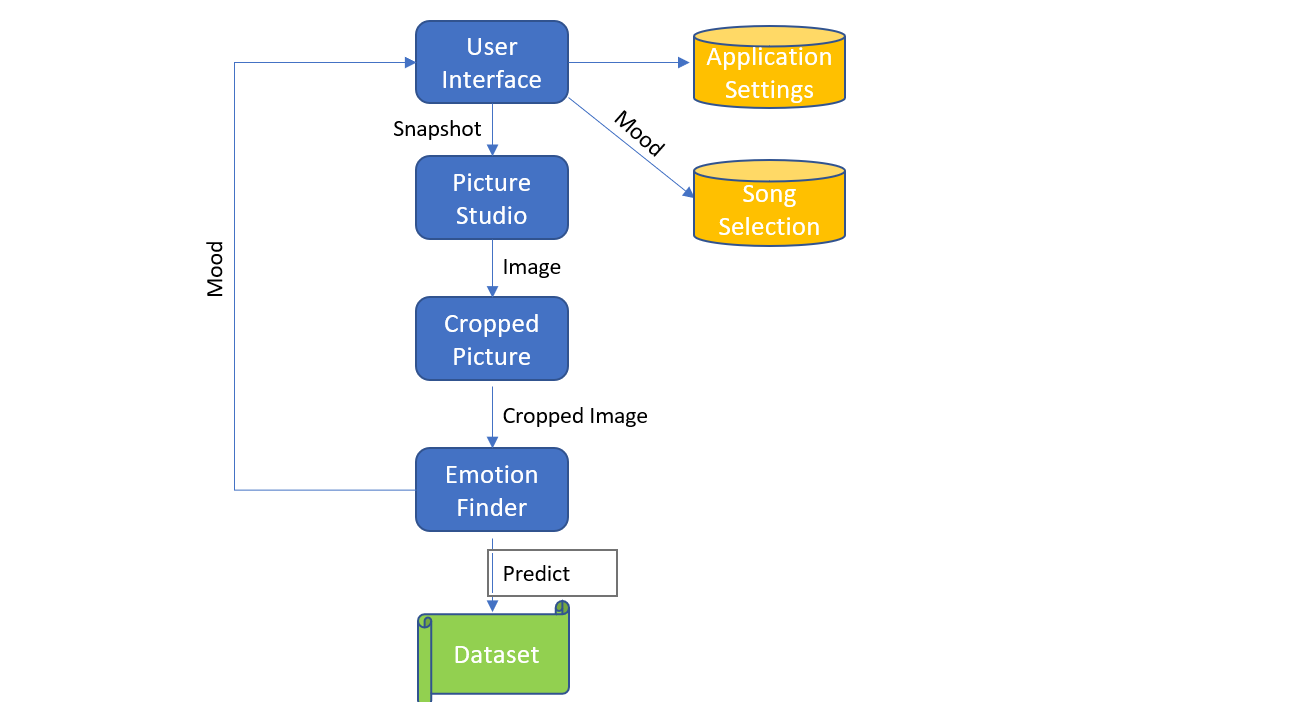
The main class which controls the User Interface ,the webcam access and face detection; the emotion finder responsible for emotion detection; then songConfiguration and moodFeedback interact directly with the emotionFinder class to provide the corresponding input.



**Process view:**

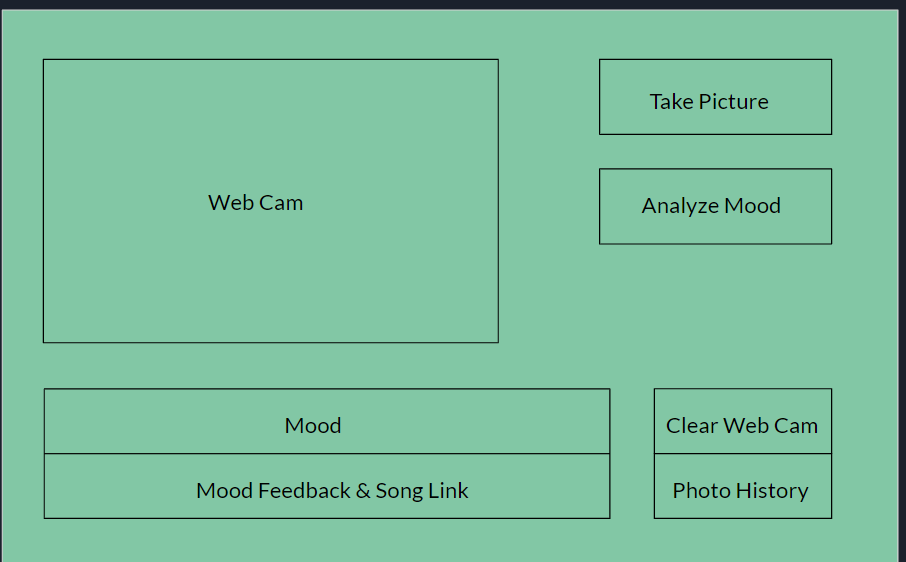
This is a description of the process view of the architecture. Describes the processes involved in the system's execution, their interactions and configurations. Also describes the allocation of objects and classes to tasks.

The following architectural diagram model illustrates the stages of interaction during a session with the user organized as executable processes.



**Physical view:**

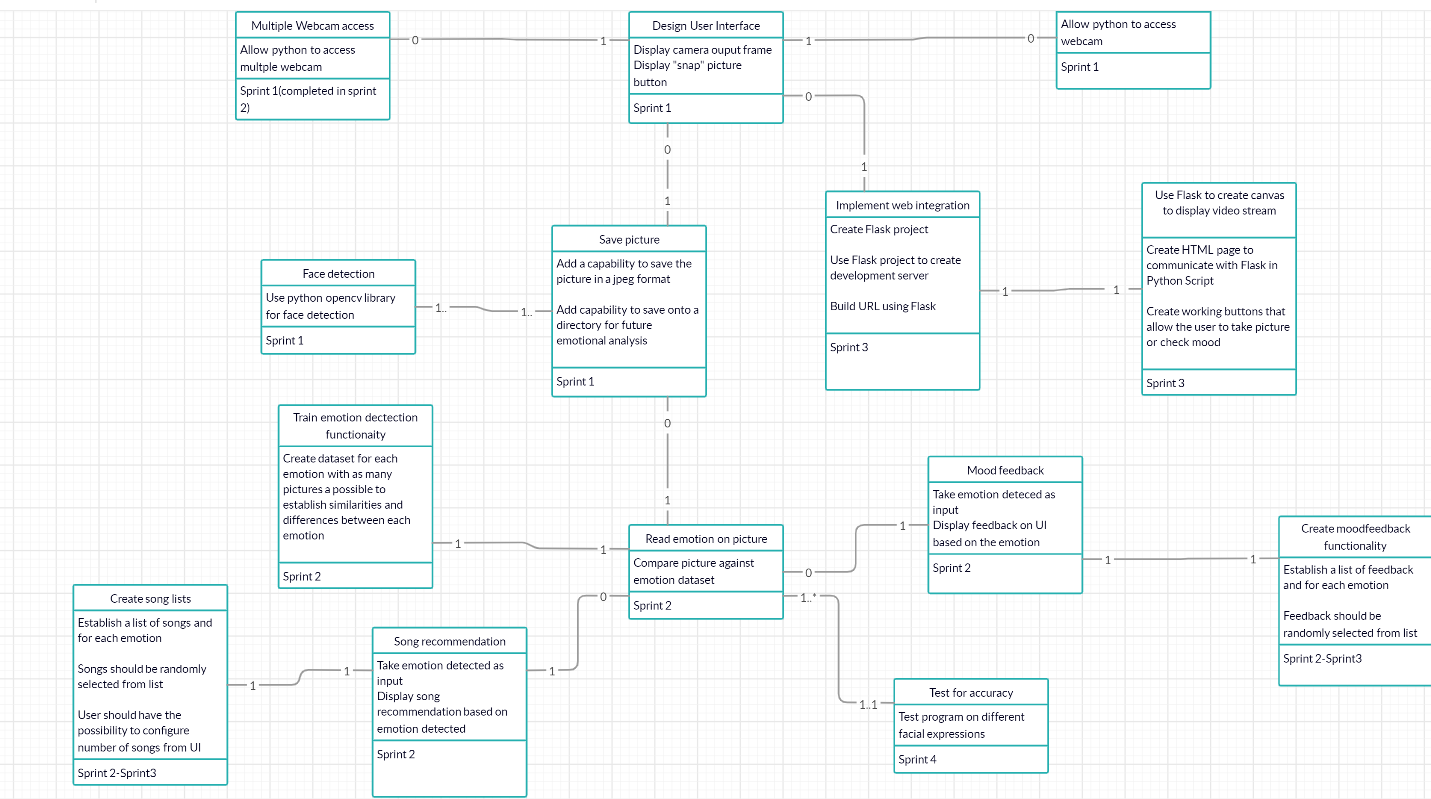
This section is a description of the physical constraints involved in the architecture of our system. A webcam is the only hardware necessary our software.



**Development view:**

PERT chart stands for program evaluation and review technique. It is a diagram used to organize, schedule and coordinate tasks of a project. It gives basic information such as time needed for completion and therefore is a useful tool for facilitating decision making, estimating the pace of the development and the minimum time needed to complete the project. The following PERT chart shows the different development goals of our project at each stage represented as sprint cycles. Dependencies are represented in the numbered lines between the boxes: 0:1 lines represent a one sided interaction where the source system(0) sends the input and the destination

system(1) receives. 1:1 dependencies represented a two sided interaction where both systems send inputs.



**Testing**

**Objective/Overview**

The purpose of the following tests are to validate that the various functionalities of the What the Hack : Music Therapy web application are behaving as needed in order to give the user the optimal experience of using the product.

**Testing Gear and Setup**

The necessary tools needed for testing are the Python Logging library, and an operational web browser. The Python logging file can be configured and set up in the application settings as follows:



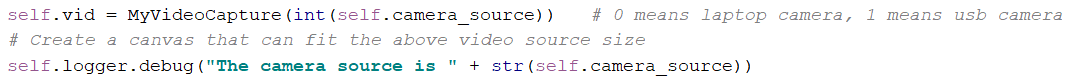
**Test Cases**

**Test 1: Webcam Access**

Objective:

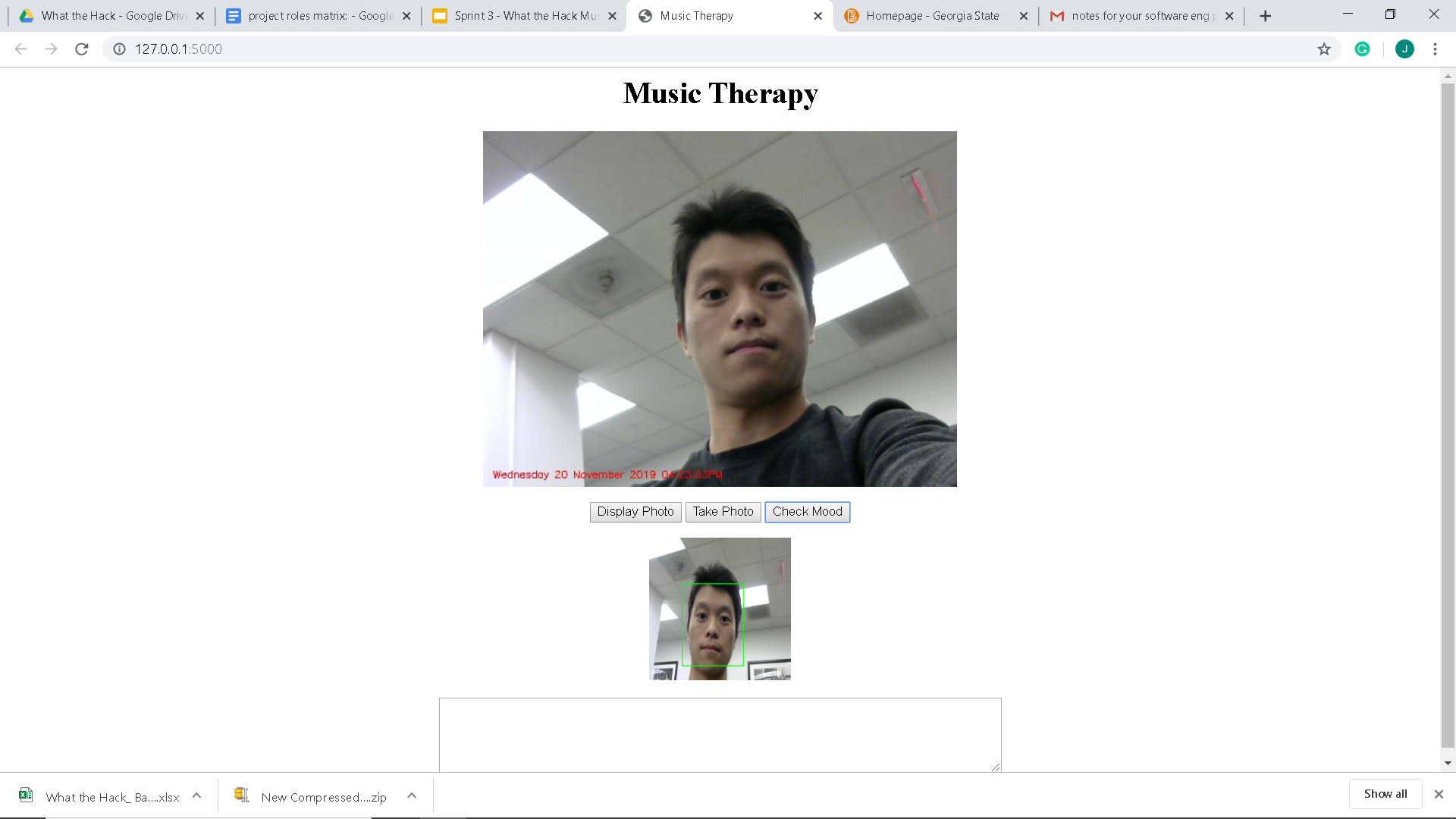
* The goal of this test is the check that the JavaScript has access to the webcam from browser, so that it is able to communicate with the Python Script.

Procedures:

* Place logger in debug mode after webcam config in Python file
* Add breakpoint to line below camera configuration
* Run application to breakpoint

Expected Behavior:

* The app should access the camera from the webpage and display it on the user interface within the user’s browser.



* Logger should confirm backend behavior by giving the number 0 or 1 for the user’s chosen camera to use.

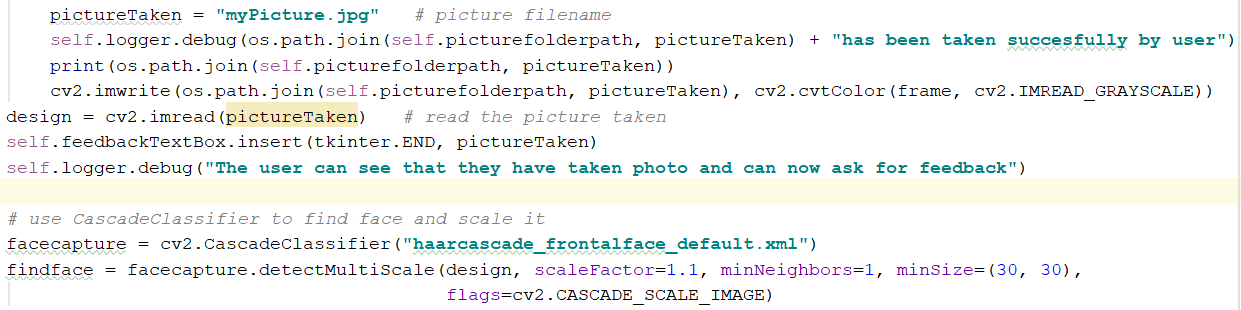
Test Result: **Pass**

**Test 2 : UI Functionality**

Objective:

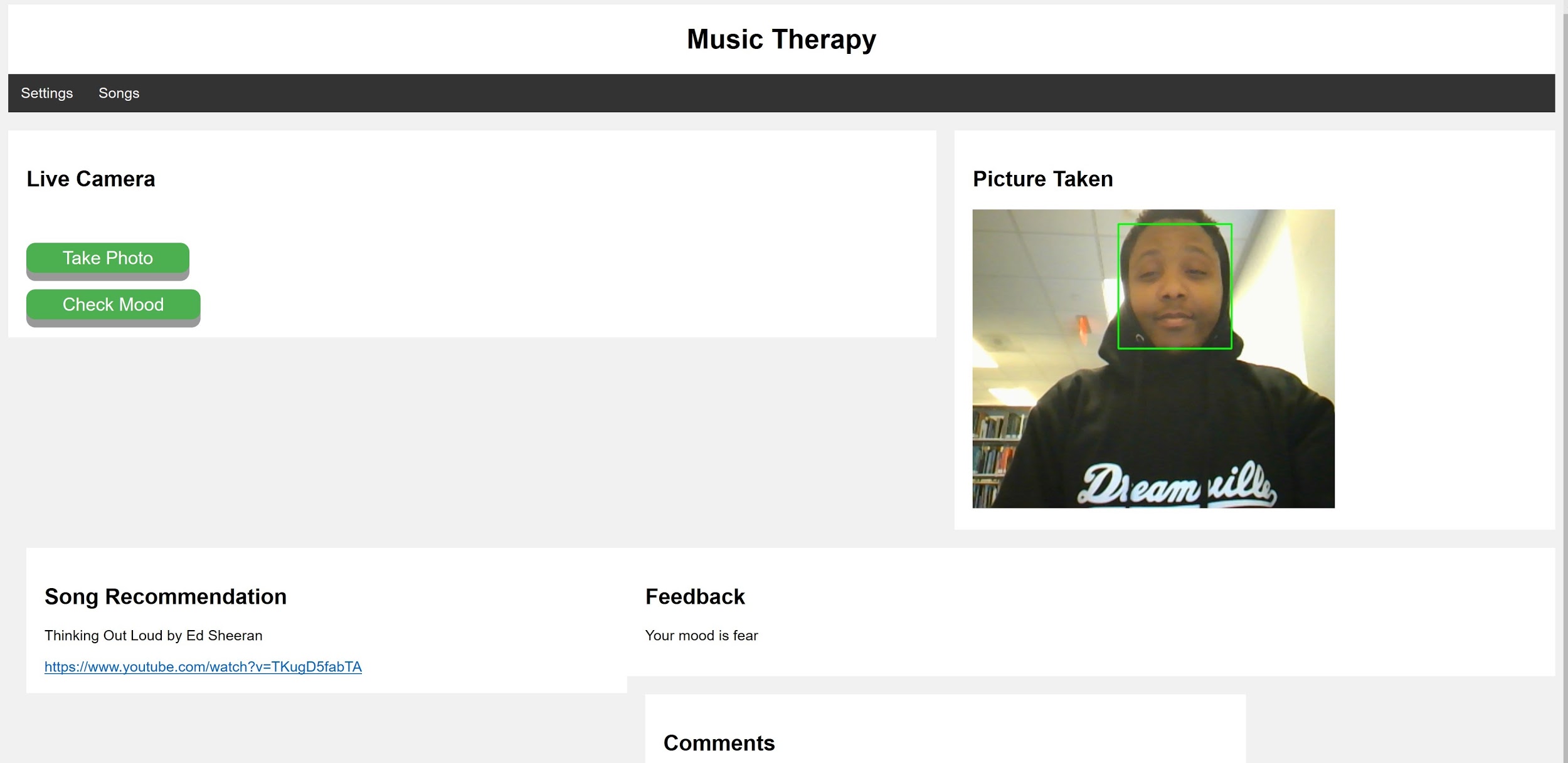
* The goal of this test is to confirm that the buttons on UI work for the user to be able to take a photo, and receive a reading to analyze their facial expression.

Procedures:

* Place logger in debug mode inside of snapshot function
  + Log that photo was taken
  + Log that photo was sent to correct path
  + Log that photo is ready to be analyzed

Expected Behavior:

* A picture should be taken and saved to a specified file
* Logger should confirm that Python was able to communicate with webcam





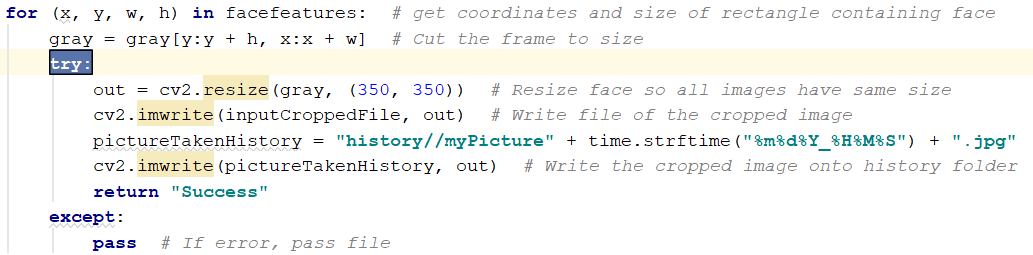
Test Result: **Pass**

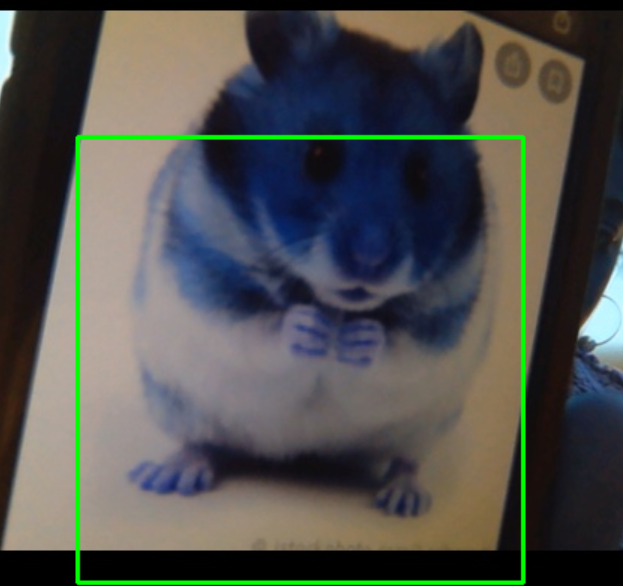
**Test 3: Emotion Finder**

Objective:

* The goal of this test is to check that the app can accurately identify faces on the pictures and does not go further until it does so

Procedure:

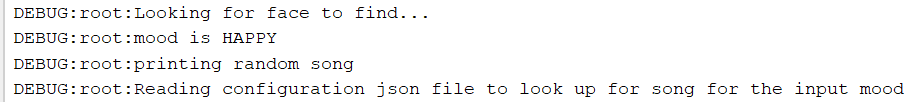
* Python “try/except” with logger set in error mode to get relevant feedback and see exactly where application is not performing as expected
* Try to “confuse” camera by photographing things that are not faces



(Animal faces do not work for the What the Hack: Music Therapy web application.)

Expected Behavior:

* No face should be recognized
* The app should prompt the user to take another picture



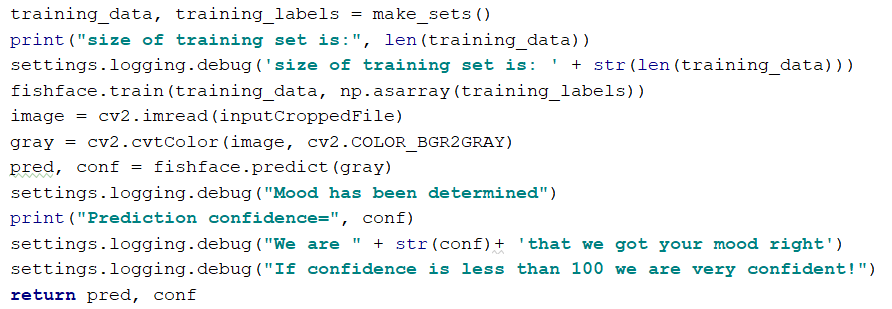
Test Result: **Pass**

**Test 4: Prediction Accuracy**

Objective:

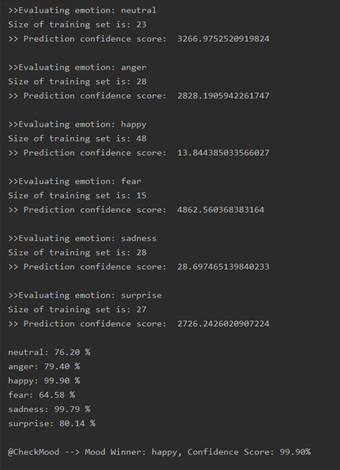
* Test the accuracy of the app to recognize the correct emotions on the face

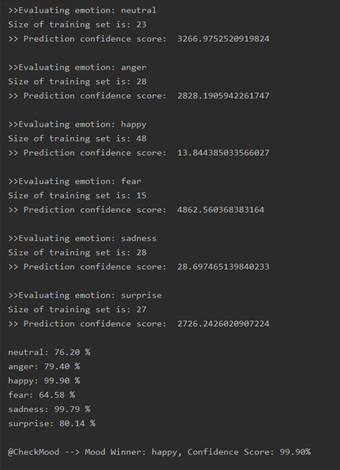
Method of Testing:

* Run app with different faces as input
* Place logger in info mode to return confidence value from CheckMood file

Expected Behavior:

* Maintain values between 800-1200
  + Value of less than 100 is very confident.
  + Value of over 1500 is very low confidence

- The emotion value with the lowest prediction score is the strongest prediction and will be sent to the user.



Test Result: **Pass**

**Summary and Observations**

Through our process of validation testing, we have been able to consistently confirm that all of or necessary functional and non-functional requirements are met in accordance with our project backlog. Test 4: Prediction Accuracy, will be in constant improvement as the data set continues to grow and improve.

**Version Control**

**Music Therapy Version 1:**

Music Therapy was a single camera, that could take a picture when a button was pressed. The User Interface was created through pythons tkinter library. The main focus of version one was to have a GUI created that could run the video feed from the webcam the user interface had two buttons created using Tkinter. The first button was snap picture. Snap picture would capture the current frame and save the picture as a jpeg. The video source would keep going after the picture was taken. The second button was the check mood button. The purpose of this button is to look at the last taken picture and analyze the picture for the current emotion. This feature was not added in version one of the product.

**Music Therapy Version 2:**

Version two of Music Therapy had the same overall design as version one. Version two was a hotfix to version ones’ “Blue Screen”. Music Therapy version 1 had an issue when pictures were taken, it would print out the image with a blue tint to it. The buttons still have the same effect. Version 2 had also added the feature for faces to be recognized in images taken.

**Music Therapy Version 3:**

Version 3 added functionality to the mood check button. Clicking the mood check button would cause the mood “Happy” and a happy quote to appear in the little feedback box below the video source. The quote and the mood were hard coded into this version just to show that the mood check button finally had functionality.

**Music Therapy Version 4:**

Music Therapy version 4 added in the emotion detection for pictures. The emotion detection for Music Therapy version four was created through the method moodCheck. Moodcheck takes the picture and compares the picture to many other faces in a picture dataset and predicts the mood based off the analyzes. Now that the product can analyze the emotion of the face in the picture, the feedback and mood are no longer the hard coded. The feedback and mood printed in the feedback box below the video source will now print a mood and feedback that matches the predicted emotion analyses. The mood check button was now edited to only appear after the snap picture button has been pressed. This stopped the program from using past pictures for emotion analysis. The program was given the ability to switch camera courses. The camera source can be switched in the programs setting by switching the video source from a zero to a one. allowed the user to not be stuck using their webcam camera and now can use a camera they have plugged into their USB port.

**Music Therapy Version 4.1:**

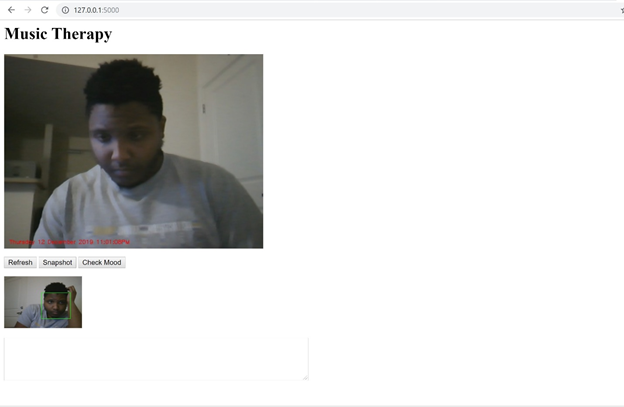
Music Therapy version 4.1 has the same exact functions as music therapy version four but the emotion methods are changed. Version 4.1 uses the method emotion finder. Emotion finder is created through the use of a CNN. The CNN take in a large dataset which is pushed to a json file. Each image is compared to the json file which the code uses to predict the emotion of the picture that is taken.

**Music Therapy Version 5:**

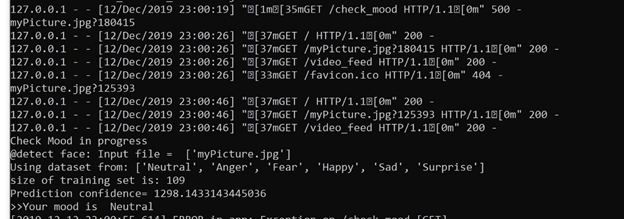
Music Therapy version 5 incorporates the strengths of both emotion recognition methods. One method was able to detect happy and neutral moods better than the other methods. The other method was able to detect angry and sad faces better than the other methods. The methods were combined to take the feedback from a specific method based on the emotion each had detected.

**Music Therapy W1:**

Music Therapy W1 is the first interaction of the project moving from a tkinter project to a flask project to become a web application. The Tkinter GUI does not work in the flask set up. Tkinter was dropped for html set up. The project now has three buttons. The extra button is the refresh. Refresh will change the picture shown in the web application to the last picture the user has taken. Refresh has to be hit to get the proper feedback. The web applications can only take the picture and show the picture taken on the web application. When the mood check button is pressed, the actually mood, feedback and YouTube link will not appear on the web application. The mood will still appear in the terminal.



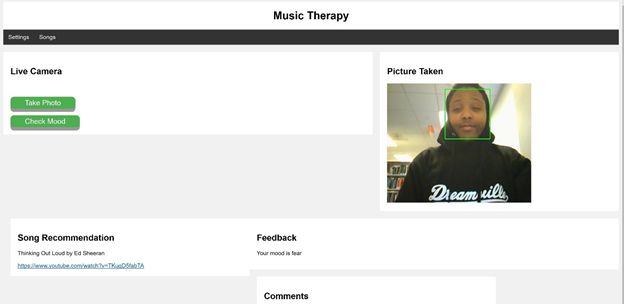
**Ex.** This is the first interaction of the web application. It contains 3 buttons, the video source, and the last picture taken below the video source. There is a feedback box but, at the moment nothing will appear in the box



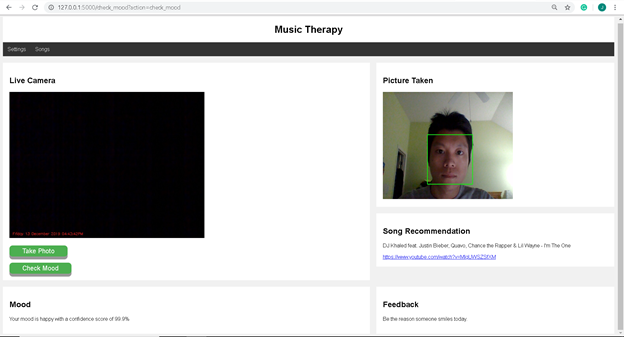
**Ex.** The mood still appears in the terminal

**Music Therapy w2:**

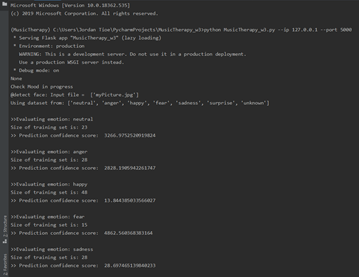
W2 is the second major iteration of the web application. The program is no longer using the refresh button, the picture taken will refresh when the take photo button is taken. The web app shows a live video feed on the left and the picture taken on the right. The look has been changed around a lot from the first iteration. The song recommendation, mood feedback, and quotes all have their own feedback boxes instead of sharing the same box. Feedback will be printed to the feedback boxes after a picture is taken and the check mood button has been pressed.



**Music Therapy w3:**

****

The layout has been moved around. The mood has now been moved to be below the video source, take photo button, and check mood button. The Picture taken, song recommendation and feedback have all been lined up on the right side of the application. All changes were made to make the interface look clean and in order.



The web application now gives a prediction confidence score for each emotion. In the mood feedback box, you will not just be given the mood but also confidence score percentage so the user will know just how confident the system is in the given prediction.

The program will now move pictures taken into the dataset based on the mood given to further increase the accuracy for future pictures to be taken.

**User Help Guide**

* Product Name & Model number:

**What the Hack: Music Therapy v4.3.2**

* **Intended use:**
  + **DISCLAIMER:**

WE ARE NOT LICENSED THERAPISTS AND CAN NOT GIVE A FORMAL MEDICAL DIAGNOSIS. THE FEEDBACK AND RESULTS GIVEN ON THIS WEBSITE ARE FOR INFORMATIONAL PURPOSES ONLY. THE PROVIDED CONTENT IS IN NO WAY MEANT TO SUBSTITUTE ADVICE FROM A LICENCED OR CERTIFIED PROFESSIONAL. IF YOU ARE EXPERIENCING EXTREME DISTRESS, OR HAVE A MEDICAL EMERGENCY, CALL 911 OR CONTACT A MEDICAL PROFESSIONAL IMMEDIATELY. WHAT THE HACK IS NOT RESPONSIBLE FOR ANY DIRECT OR INDIRECT ACTIONS TAKEN FROM THE ADVICE GIVEN IN THE WHAT THE HACK: MUSIC THERAPY WEB APPLICATION. NEVER DISREGARD THE ADVICE OF A MEDICAL PROFESSIONAL BASED OFF OF WHAT CAN BE SEEN WITH THIS WEB APPLICATION. THE UTILIZATION OF THIS APPLICATION IS AT THE RISK OF THE USER, AND BY USING THIS PRODUCT, YOU ARE AGREEING TO THESE TERMS. IF YOU DO NOT AGREE WITH THE FEEDBACK PROVIDED BY WHAT THE HACK: MUSIC THERAPY, CEASE USE OF THE APPLICATION.

* Features:
  + Webcam Access:
* When accessing the What the Hack : Music Therapy web application, the user must allow the app access to their devices integrated webcam, or a separate web camera (example: USB attached web camera).
  + Take Picture:
* Once the web application has been granted access to the web camera of the user’s choice, the user is able to have their taken by the JavaScript communicating with the Python Script within the application.
  + Check Mood:
* After the user has taken their picture using their webcam of choice, they are able to analyze their facial expressions in order to have the application give an output that determines the mood that they were in.
  + Get Feedback:
* Succeeding the evaluation of the user’s face, the user is also able to receive a lighthearted message that will coincide with the user’s determined mood.
  + Song Recommendation:
* Subsequently the the application will produce a song for the user in order to improve their mood from whatever was read by the application.
* Installation instructions
  + For Source Code:
* **Install Flask for Python**

**Resource:**<https://www.youtube.com/watch?v=Z1RJmh_OqeA>

Needed PowerShell/ Linux Commands

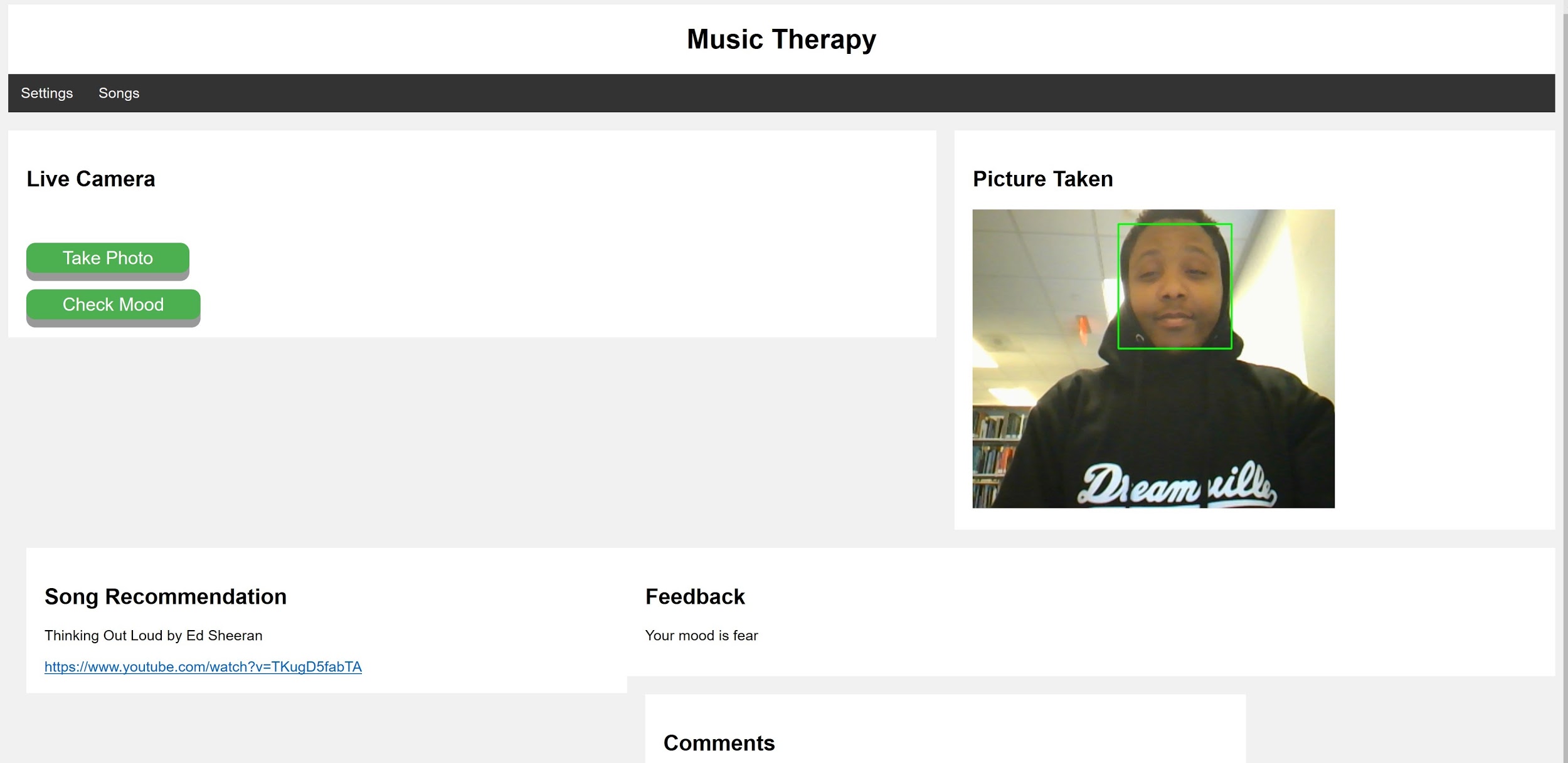
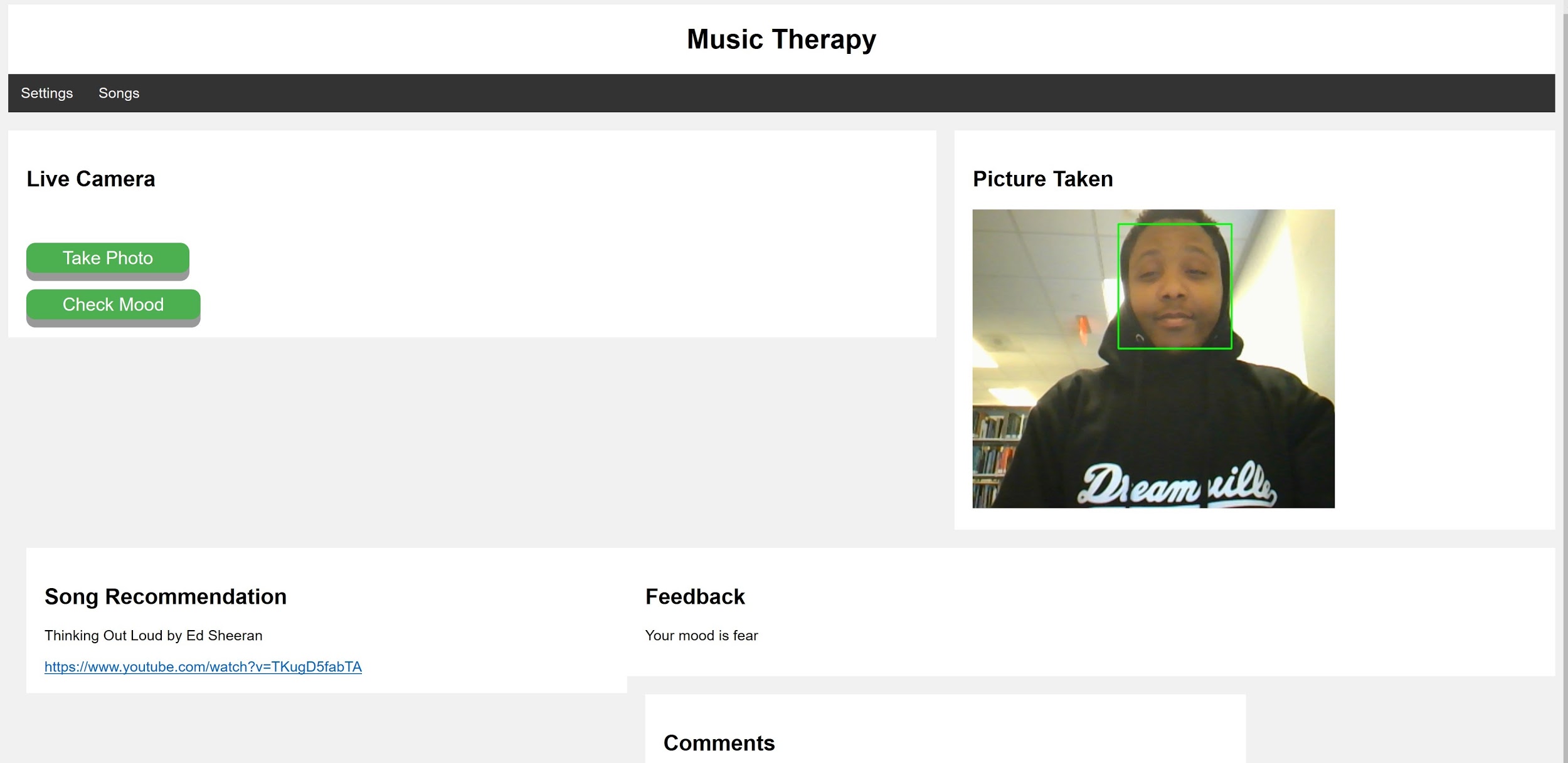
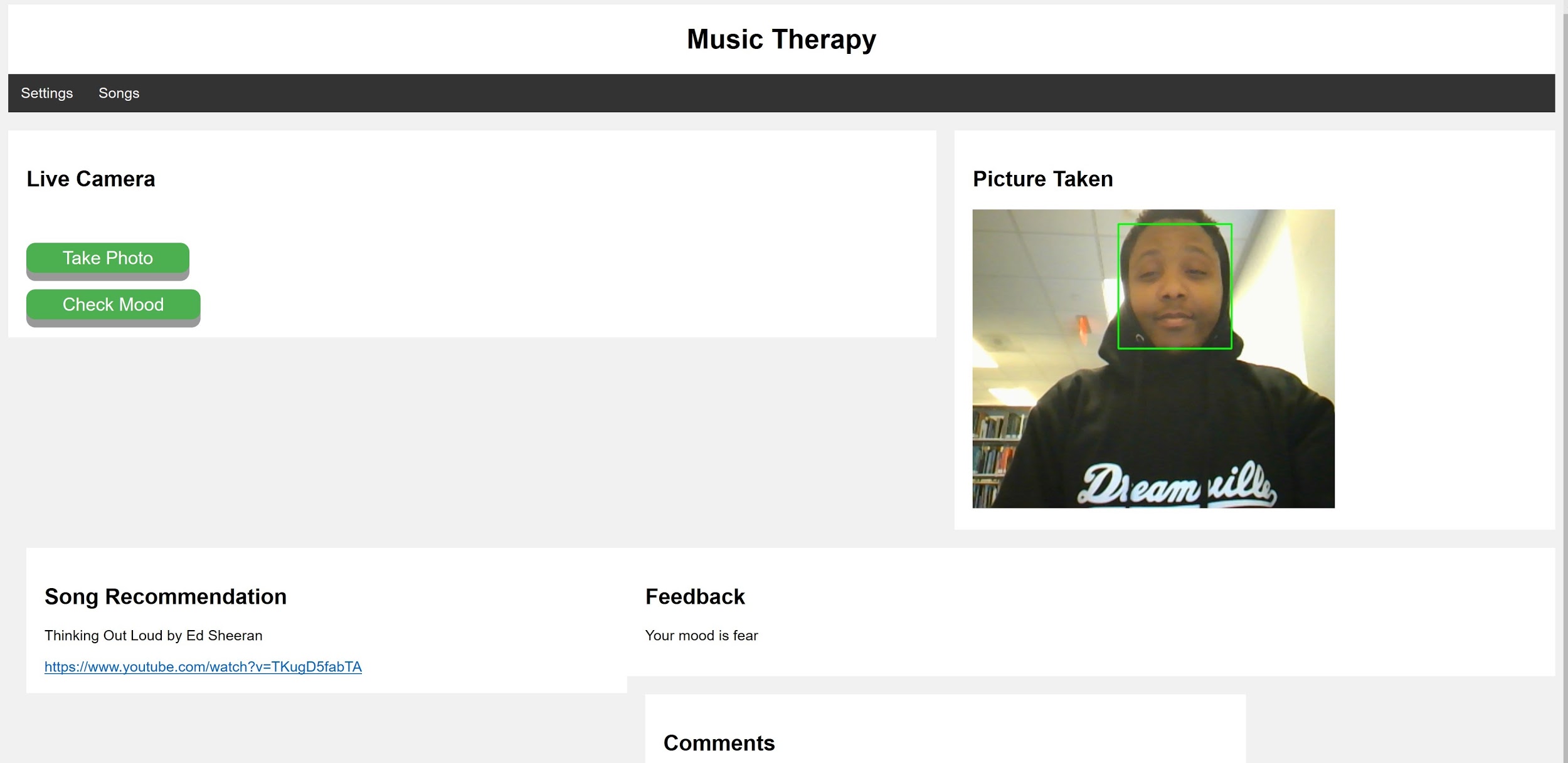
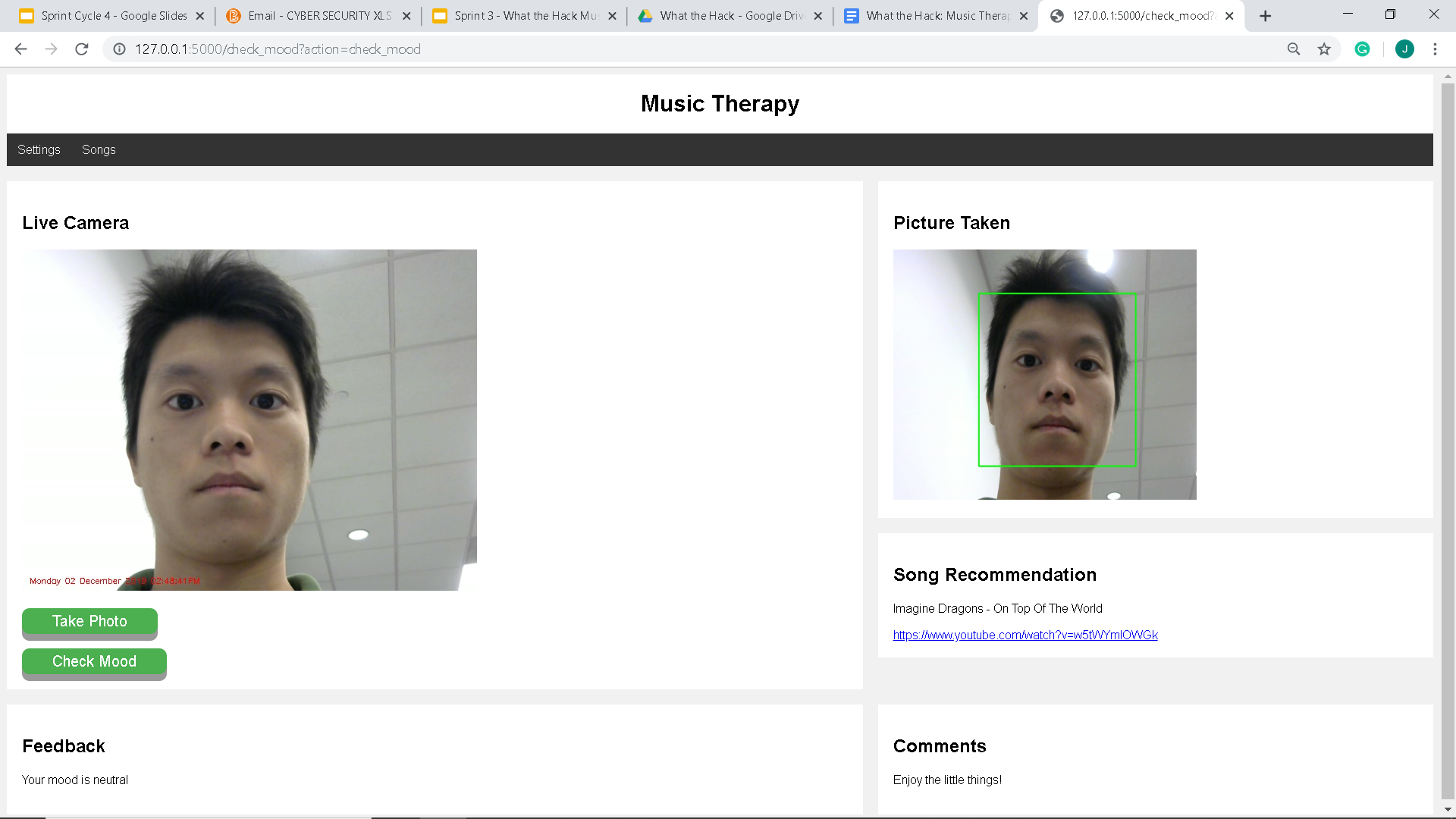
>pip3 install virtualenv

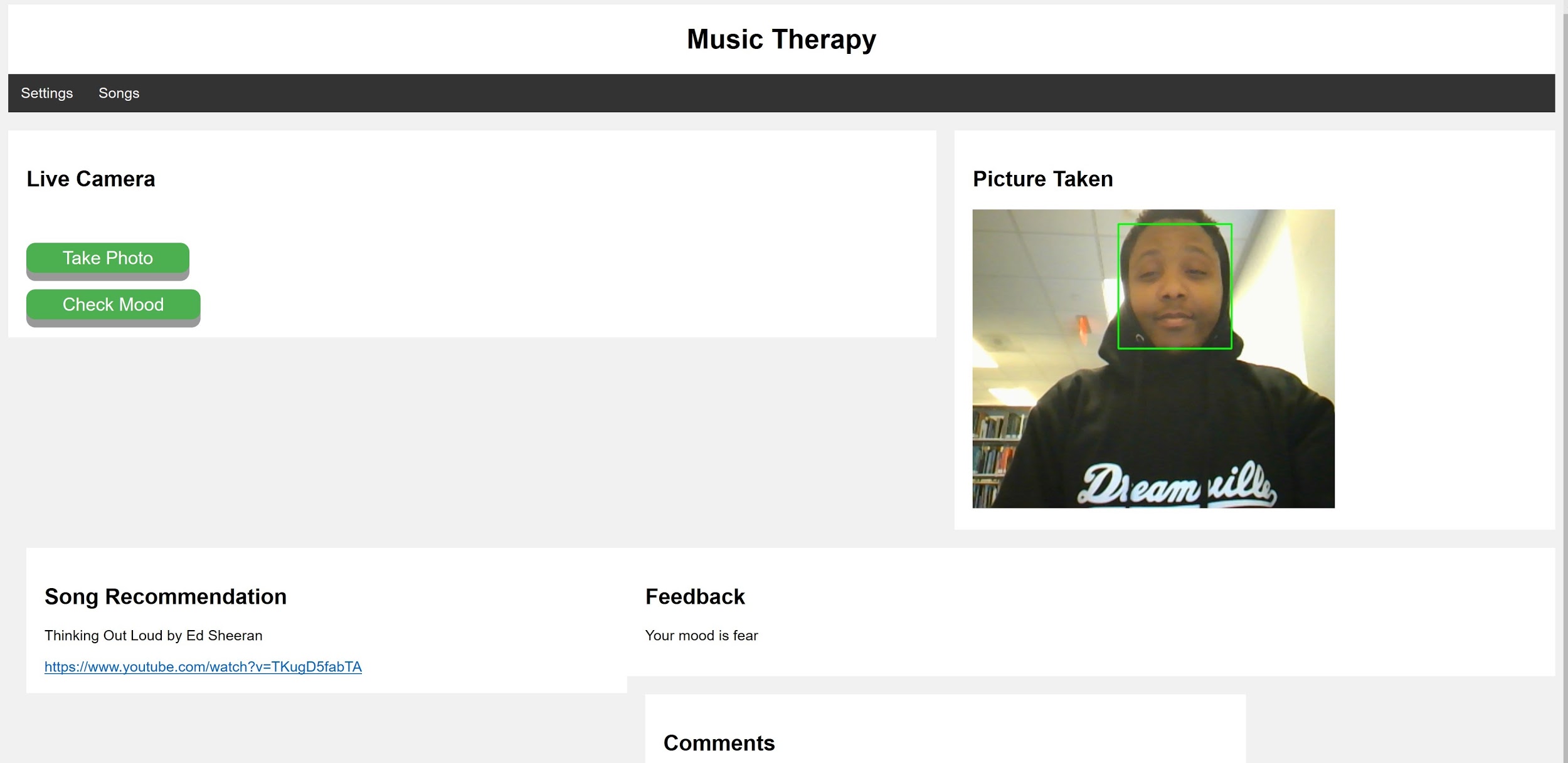
>virtualenv env

>cd env

>pip3 install flask flask-sqlalchemy

>cd ..

* **Within Python Directory**
  + put MusicTherapy\_w3.py in the root folder
  + put index.html to root/templates folder
* Additional Necessary Python Packages:
  + You need to install additional package such as
    - **Numpy**
    - **Opencv-contrib-pythom**
    - **Keras**
    - **Tensor flow**
    - **Matplotlib**
    - **imutils**
  + To start the webserver, you need to type the following command from Terminal
    - > python MusicTherapy\_w3.py --ip 127.0.0.1 --port 5000
  + URL to access application:
    - <http://127.0.0.1:5000/>
  + For Web Application:
* **Open Working Web Browser:**
  + What the Hack : Music Therapy is ready for use on any web browser including (but not limited to) the following:
    - Google Chrome
    - Internet Explorer
    - Microsoft Edge
    - Safari
    - Mozilla Firefox
    - Opera
* Description of how to use/operate the product
  + Website URL:
* Insert Link into Browser:
  + <http://127.0.0.1:5000/>
    - This link should transport you to the What the Hack : Music Therapy web application landing page.
  + Take photo (button):
* Once on the web application landing page, you will see a button to the left of the screen that appears like the one represented above. Clicking on this button will take a photo of the love feed on the website, and then display that photo on the page for viewing.
  + Check Mood (button):
* After the picture has been taken, click on the “Check Mood” button that is located directly underneath the “Take Photo” button.
  + Receive Feedback:
* Once satisfied with the photo that has been taken, and the “Check Mood” button has been activated, the mood and feedback will be presented in the “Comments” section.
  + Song Recommendation (text box):
* The song recommendation will be displayed directly underneath the user’s displayed photo and will contain a link to a YouTube video that will be used to improve the mood of the user.



* Troubleshooting
  + Any issues with the What the Hack: Music Application should be reported to the following: whatthehackgsu@gmail.com

**Lessons Learned**

* Frequent communication is effective to ensure that the project is on track.
* Coming up with a design plan for each Sprint can make the project manageable.
* Breaking down work into smaller chunks to fit into a Sprint cycle is efficient.
* Try to fail early by implementing a small piece of functionality, so that the issue can be resolved without impacting the timeline.
* A better distribution of tasks among team members is important in reducing stress
* Role switching (scrum master, developer, or tester) is an effective way to build leadership and ownership of a project.
* It takes the whole agile team to deliver a successful project in a timely manner.
* It is important to focus on solving problems rather than blaming others.
* It is not always possible to fulfill all the functionalities in time or with the resources at hand, in which case prioritizing is essential
* Using UML diagrams for representation makes the software systems and use cases easier to understand
* It is important to not get personally or emotionally attached to the project
* Risks are unavoidable, therefore it is critical to identify possible sources for those risks and set up a plan to avoid or mitigate their effect on the project

**Glossary**

**Accuracy** - how accurate the mood prediction is based on percentage

**ApplicationSettings** - configuration JSON file that the user can configure the number of songs per mood, as well as the camera source to take picture

**Architectural Design** - the overall structure of the system; how the system should be organized

**CheckMood** - Python file which does the mood checking after the user clicks on the check mood button

**Convolutional Neural Networks (CNN)** - Deep Learning algorithm which allows the recognition of faces and classifies the emotions

**Emotion** - the feeling that one expresses; Ex. Happy, sad, anger, fear, surprise, neutral (see Mood)

**Feedback** - the comment displayed on the screen as a result from the given emotion

**Flask** - framework to build the web application in Python

**iMutils** - series of functions to make basic imaging processing functions more convenient with OpenCV

**JSON** - JavaScript Object Notation - open-standard file format used for ApplicationSettings and SongConfiguration

**Log File** - file that records events that occur during the runtime of the MusicTherapy code

**Mood** - the feeling that one expresses; Ex. Happy, sad, anger, fear, surprise, neutral (see Emotion)

**MusicTherapy** - the main Python program to run the code

**myPicture** - most recent picture after snapshot, saved as a JPEG format (myPicture.jpg)

**myPictureCropped** - crops the face from the picture image and saved as a JPEG file (myPictureCropped.jpeg)

**OpenCV** - Open Source Computer Vision - used to set up the camera user interface

**opencv-contrib-python** - wrapper package for OpenCV python bindings. This is primarily used for the camera

**PERT chart** - Program Evaluation and Review Technique. It is a diagram used to organize, schedule and coordinate tasks of a project

**Prediction confidence** - how confident the computer is at detecting the specific emotion. When the prediction confidence is lower, the accuracy of the emotion detection is higher (prediction confidence below 100 is ideal for the emotion result)

**Python** - the main programming language that is used for the project

**Refactoring** - improving the structure and organization of a program. This allows the code to be simpler and maintainable

**Snapshot** - the process of taking a picture

**SongConfiguration** - configuration JSON file that contains all the songs and feedbacks

**Tensorflow** - open source library useful for artificial intelligence, which uses data flow graphs to build models

**Test Case** - set of conditions where the tester determines whether a system under test satisfies the requirements

**Tkinter** - Python library that contains the camera user interface, but is not supported by the web application

**Training set** - the set of all emotions: ['neutral', 'anger', 'happy', 'fear', 'sadness', 'surprise', 'unknown']. Each of these emotions need to be trained; in other words, more facial pictures in the training set would make the set more accurate (excluding ‘unknown’)

**UML-**Unified modelling language. A general-purpose, developmental, modelling language that is intended to provide a standard way to visualize the design of a system.

**Use Case** - describes the interaction between the user and the system, such as the user and the camera

**User Interface** - the frontend of the web application

**Version control** - software tools that can help a software team manage the changes to the source code over time, such as MusicTherapy\_v1, MusicTherapy\_v2, etc.

**Video stream** - video feed for the user to pose in front of camera

**Webcam** - camera user interface for video feed