Bear Aware

Infosci 201 - Final Project - Grant Pedersen

Introduction:

Our facial expressions play a crucial role in how we express our emotions, often doing so before we are even aware of them. This phenomenon results from the complex interaction between our brain, emotions, and facial muscles (1). When we experience emotions, our brain triggers physical reactions such as changes in heart rate and muscle contractions. In response to these signals, specific facial muscles craft expressions that reflect the emotions we are feeling, often before we have fully processed these feelings (11).

This involuntary emotional display, known as "emotional leakage" or "micro-expressions," forms the foundation of a great deal of psychological and neuroscientific research. Experts in facial expressions can often identify these fleeting, involuntary movements that occur when emotions are concealed. These expressions can serve as indicators of true emotional states (1).

Our ability to recognize these expressions is crucial for empathic understanding and social interaction.

For my final project, I created BearAware. The fact that our faces can exhibit our feelings, occasionally even before we consciously register them, helped me realize the importance of early identification and response. While we are working at home, we sometimes become stressed with different tasks, and it would be important to recognize when these stressors occur.

BearAware is a 3D-printed bear designed as a fun tool that connects to your laptop via USB. It's designed to alter colors in sync with the emotions detected from the user's facial expressions, as observed on the BearAware website (bearaware.xyz). The technology configured to comprehend and differentiate various emotional states. For instance, if you display signs of anger, the bear transitions to a red hue, reflecting your

emotional state. Conversely, if sadness is detected, BearAware switches to a blue color, visually mirroring your feelings. The objective is to orchestrate an engaging, empathetic, and interactive user experience through emotion recognition technology and tangible user interfaces.

1) EKMAN, PAUL and FRIESEN, WALLACE V.. "The Repertoire of Nonverbal Behavior: Categories, Origins, Usage, and Coding" Semiotica, vol. 1, no. 1, 1969, pp. 49-98. https://doi.org/10.1515/semi.1969.1.1.49

11) Damasio, A., Grabowski, T., Bechara, A. et al. Subcortical and cortical brain activity during the feeling of self-generated emotions. Nat Neurosci 3, 1049–1056 (2000). https://doi.org/10.1038/79871

Arduino code:

This program utilizes the FastLED library to control a string of WS2812 lights. The LEDs are connected to pin 10 of the microcontroller. The code defines 60 LEDs and specifies an RGB color sequence for them. Additionally, a flag variable and an array called "leds" are initialized to keep track of the colors assigned to each LED.

In the setup() function, the serial communication is initiated and the data pin is configured as an output. The LED strip configuration is established using the FastLED library. Then, the startupLightShow() function is called, which presents a captivating light show by rapidly cycling through a diverse range of hues and intensities.

Within the loop() function, the code checks the status of the serial communication. If a command is received, the desired color is updated to reflect the indicated mood in the command (e.g., "angry," "sad," or "disgusted"). If the command is not recognized, a predetermined color is assigned as the desired color. If the current color does not match the desired color, the transitionToColor() function is invoked to smoothly transition between the two.

Following that, a for loop determines the appropriate color value for each LED. The FastLED.show() function is used to update the LEDs with the new colors, and a brief delay of 50 milliseconds is included before the loop repeats. This delay helps to slow down the LED updates, preventing a "blinking" effect.

Serial USB to JS <-> JS to serial:

The connection is written in JavaScript and demonstrates the communication between a web application and an Arduino board using serial communication and the experimental web API.

The code sets up variables and objects necessary for serial communication with the Arduino board. It defines a port variable to store the connection port, a writer object to

send data to the board, and an encoder object to convert text into a writable stream. Additionally, it initializes an object called emotionCounts to keep track of the count of different emotions detected, and a frameCount variable to limit the number of frames processed.

The connectSerial() function establishes the serial connection with the Arduino board. When the "connect-button" element is clicked, the connectSerial() function is called. It requests the serial port, opens the connection with a specified baud rate, and sets up the necessary readable and writable streams for communication.

Website (bearaware.xyz):

This website utilizes the face-api.js library's machine learning algorithms to achieve real-time face and emotion recognition in the browser. The underlying technology is based on TensorFlow.js, a JavaScript library for machine learning (III).

The HTML file sets up a structure that incorporates a video element for real-time video feed from the user's camera, a canvas overlay for displaying face detection results, and a text div for showing the dominant emotion. CSS is employed to layout the elements, positioning the video feed in the center of the page with the emotion text and logo arranged around it. A background image and a wooden texture frame are added to enhance the visual appeal of the video.

The JavaScript file contains the code for face and emotion recognition. It begins by loading pre-trained models from the face-api.js library (IV). These models include TinyFaceDetector for face detection, FaceLandmark68Net for facial landmark recognition (which determines the face's position and orientation), FaceRecognitionNet for face recognition, and FaceExpressionNet for emotion recognition. These models are trained using a machine learning technique called convolutional neural networks, allowing them to recognize distinct features of faces and emotions.

Upon page loading, the script acquires access to the user's camera and initiates the video feed. It continuously captures frames from this feed at a regular interval. For each frame, it employs the face-api.js library to detect all faces in the frame, identify their facial landmarks, and recognize their expressions. This process entails transforming the image data into a format that the models can comprehend, feeding the data through the models, and then converting the output back into JavaScript objects that can be easily manipulated.

The user is then displayed a button to connect to the Arduino board using the serial port. Upon clicking the button, a drop down is displayed where the user can click

which port it should send the data to.

The detected emotions for each face are represented as probabilities for different emotions. The script then identifies the emotion with the highest probability, which is deemed the dominant emotion. This emotion is displayed alongside the video feed. The detected faces and their expressions are also drawn onto the canvas overlay, enabling users to see the results in real time.

The CSS styling in the HTML file arranges the elements on the page. The video element and the canvas overlay are positioned absolutely within a relative container to maintain alignment. The emotion text is styled with a large, bold Crimson Pro font and situated beneath the video feed. A transparent, sticky navbar with the logo is affixed to the top of the page.

iii)"TensorFlow." TensorFlow, 2023, www.tensorflow.org/.

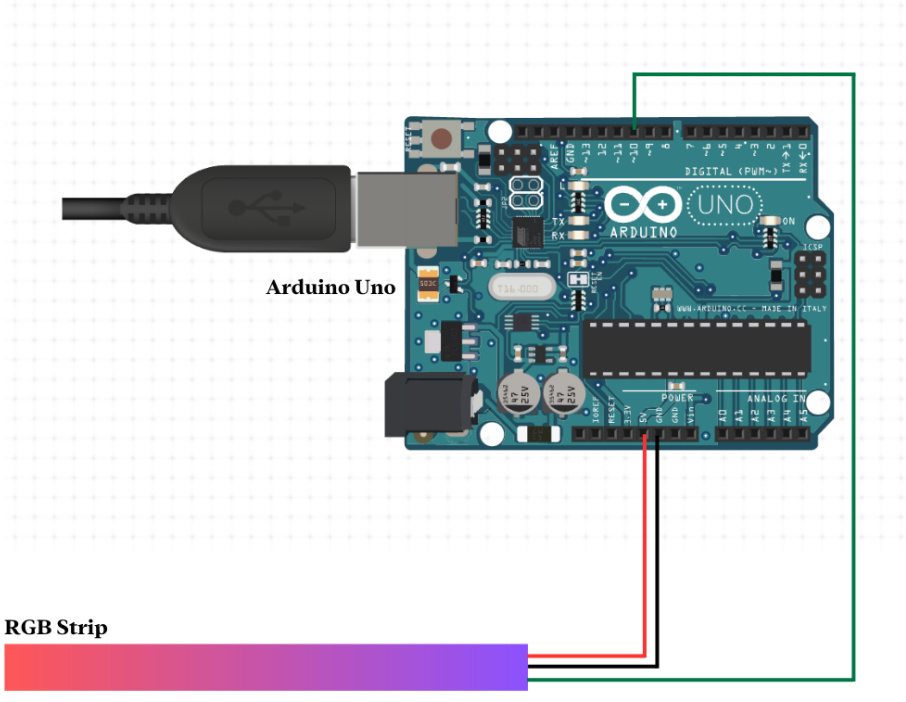
iv) "Face-Api.js." Github.io, 2023, justadudewhohacks.github.io/face-api.js/docs/index.html.

v) "Node.js." Node.js, 2023, nodejs.org/en.

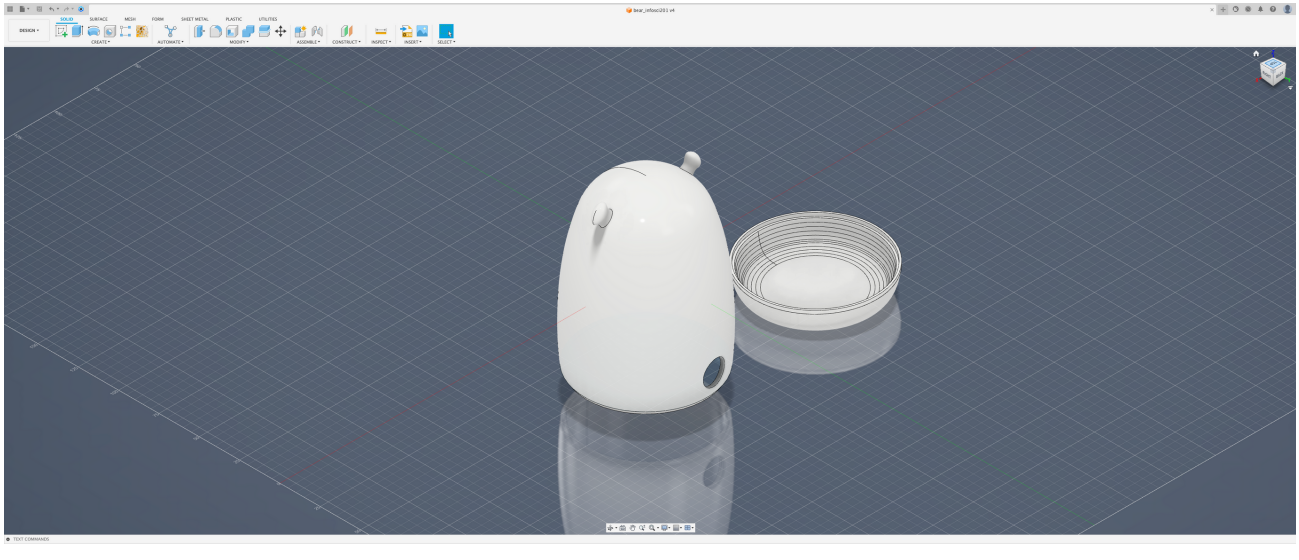
Emotion detection (using JS):

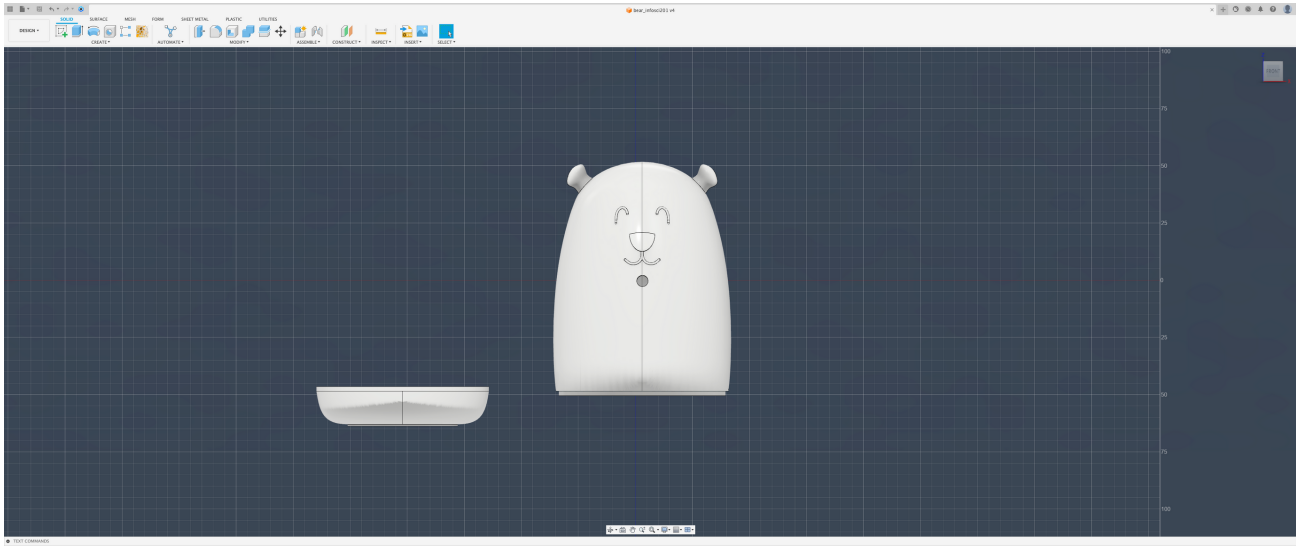
An interval function runs asynchronously every 100 milliseconds after the video starts playing. The faceapi.detectAllFaces() method locates all the faces in the current video frame and computes the facial expression probabilities for each face using the withFaceExpressions() method. This generates a detections object, which is resized to match the display size, creating the resizedDetections object. For every detection, the faceapi draw functions illustrate the detection boxes, landmarks, and expressions on the canvas. Over the span of a second, each detected emotion increments its respective counter. After each second, the most dominant emotion—determined by the highest counter—is capitalized and displayed on the webpage via the emotion div. Once displayed, all counters reset for the calculation of the next dominant emotion in the following second. This process repeats, continuously analyzing the video feed, calculating the most dominant emotion per second, and displaying that emotion on the webpage.

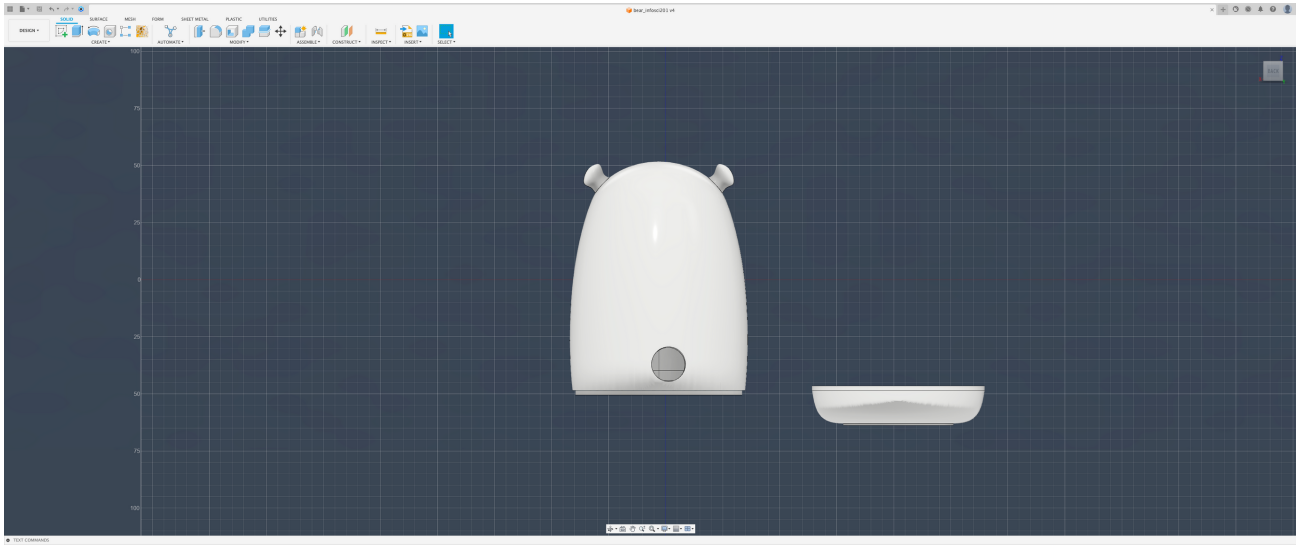
Circuit design:

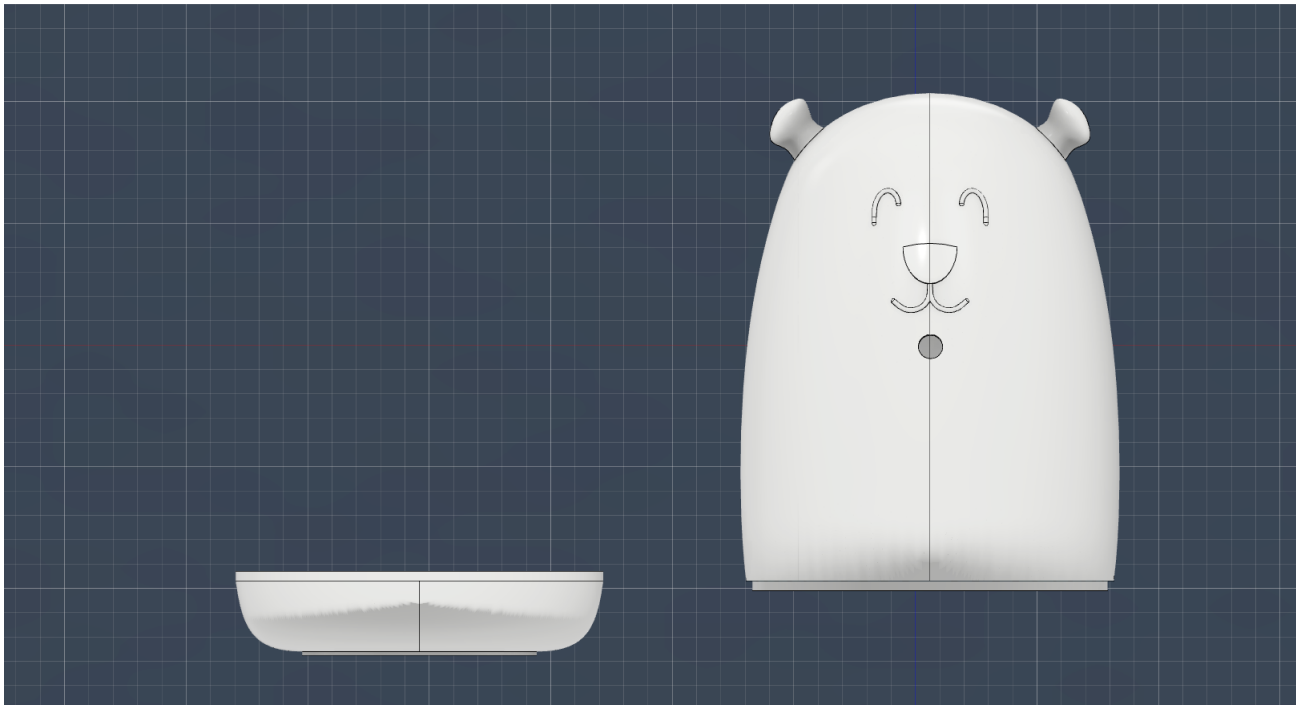


3D model:

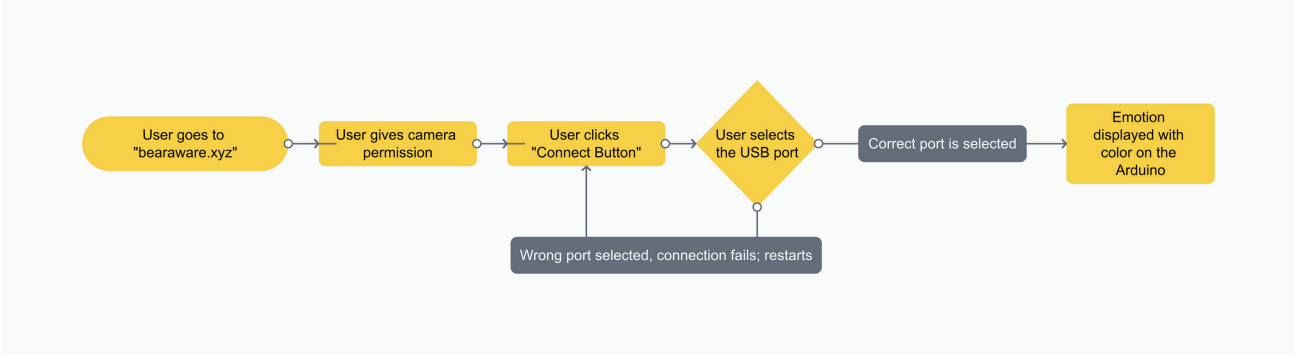








Userflow:



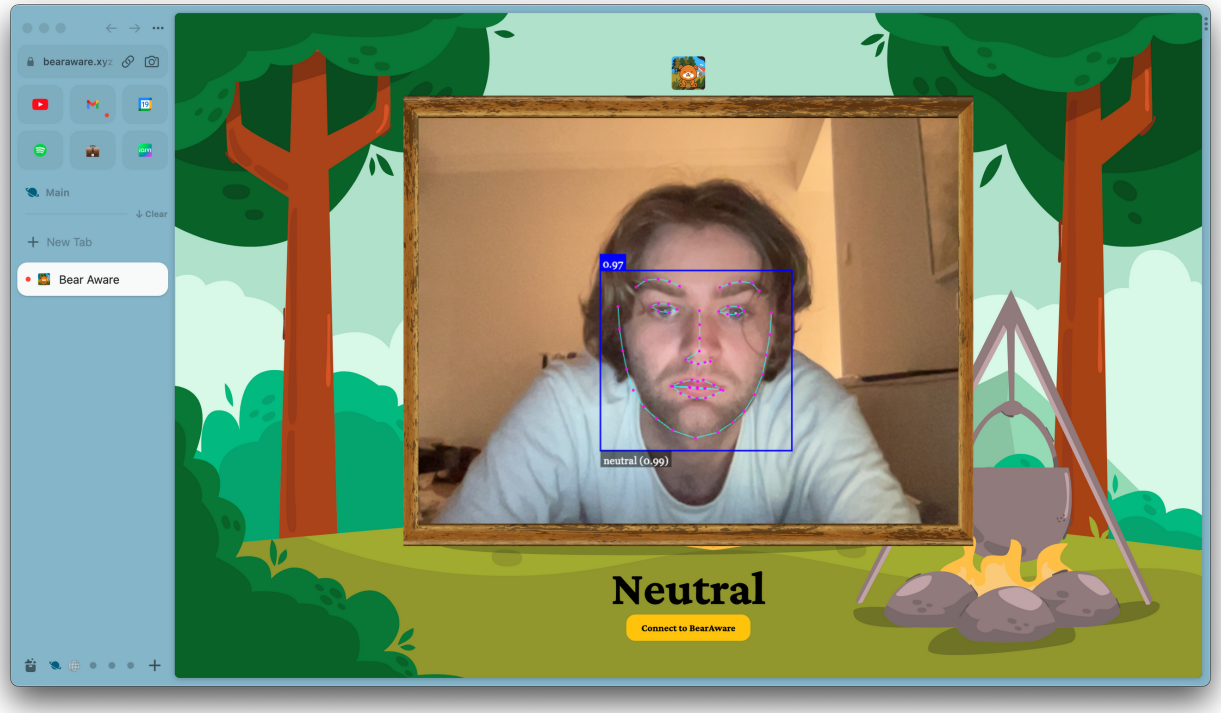
Website design elements:

background.svg

logo.svg and favicon.ico

wood.svg

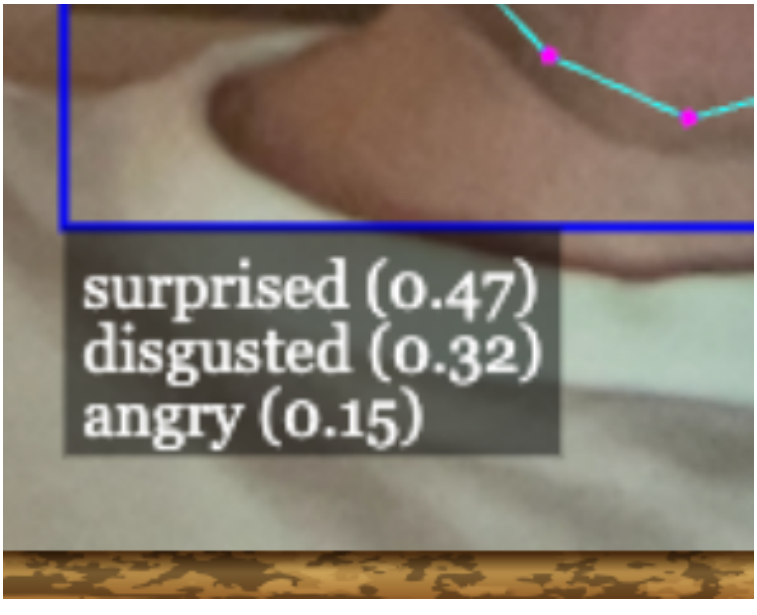
Website in action:




bearaware.xyz
Main
Clear
New Tab
Bear Aware
0.95
happy (0.95)
Happy
Connect to BearAware

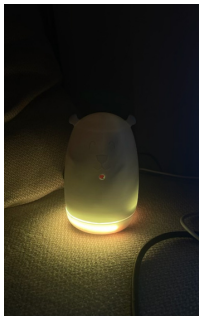
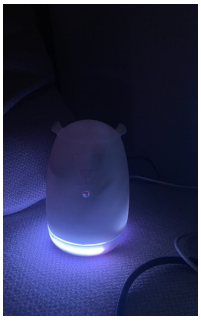
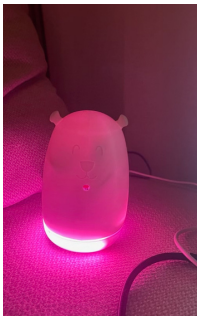
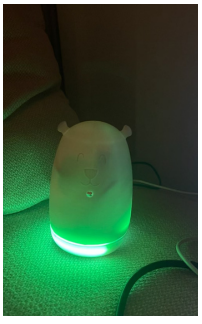

bearaware.xyz
Main
New Tab
Bear Aware
0.88
angry (0.83)
disgusted (0.16)
Angry
Connect to BearAware

Averages demonstrated:



As shown above, the program guesses the emotion by guessing what it thinks the emotion should be. Depicted above, I intentionally made a complicated face. The guess percentages are depicted to the right of the face, so it can be shown what it thinks it could be.

BearAware "Bear" showing emotions:

From left to right: Neutral/Happy → Sad → Anger → Disgust