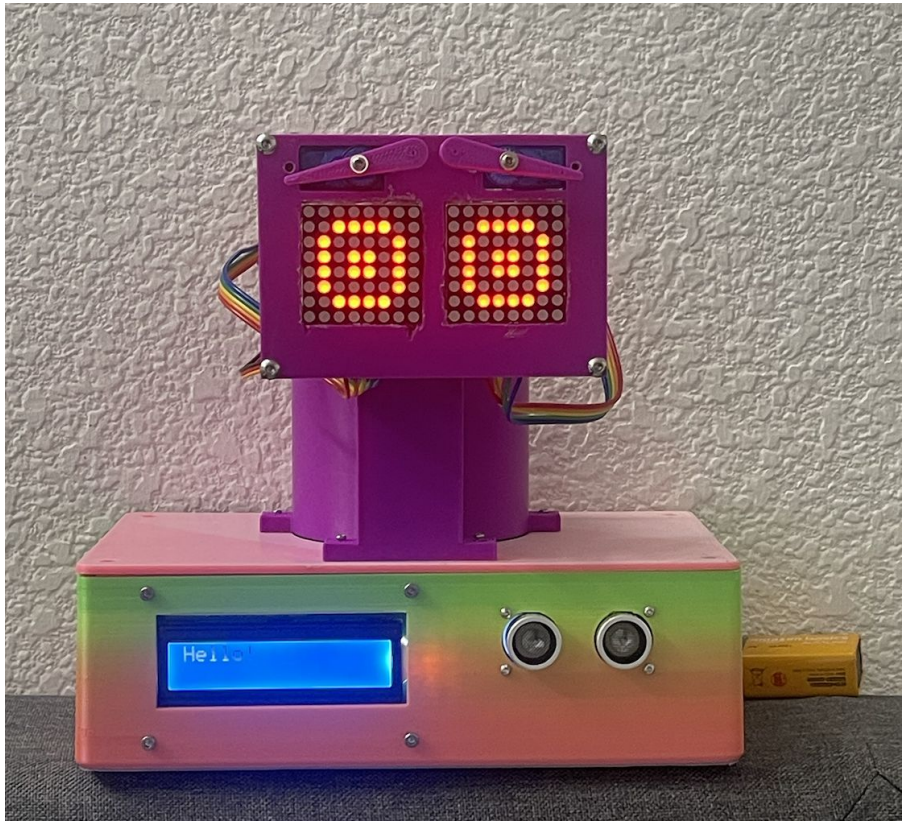
Wall-E
Emotive Robot



Introduction

WALL-E is an interactive, emotive robot designed as a personalized companion, capable of reacting to human gestures through proximity sensing and emotion simulation. The goal of this project is to create a robot that fosters human-robot interaction by providing emotional feedback using non-verbal cues such as facial expressions, head movements, and messages on an LCD screen. By using an Arduino Uno, servos, ultrasonic sensors, and a mobile app, WALL-E can respond to user proximity with different emotions.

The potential applications of WALL-E extend beyond simple companionship. Specifically, it can serve as a communication tool for individuals who face difficulties in expressing emotions, such as those with alexithymia. The integration of servo motors to mimic facial expressions, LED matrices to simulate eye movements, and an LCD screen to convey messages allows for an immersive emotional experience without requiring verbal interaction. This project contributes to the broader field of affective computing by combining robotics, sensor technology, and human-centered design.

This portfolio documents the hardware and software development process, challenges encountered, and future opportunities for enhancing WALL-E’s capabilities to better serve its intended purpose as an emotional support companion.

Human interaction with technology is rapidly evolving, with emotional engagement playing an increasingly important role in improving the human experience. Emotional robots, which are capable of detecting, processing, and responding to user behavior, have seen growing applications in healthcare, therapy, and education. The ability for machines to convey empathy or simulate emotions allows for unique, non-verbal interaction models that can serve people who have limited means of traditional communication.

The motivation for developing WALL-E stems from the idea of integrating technology into our daily lives in a way that not only provides utility but also enhances emotional well-being. WALL-E was designed to be a friendly companion that can recognize human gestures and respond with different emotional states. The inspiration for this project comes from exploring how emotionally intelligent robotics can offer supportive roles in people's lives, specifically by making human-robot interaction accessible and intuitive.

This project brings together a combination of hardware and software components, with an emphasis on emotional communication through movement, visual cues, and reactive expressions. Utilizing ultrasonic sensors, LED matrices, and servo motors, WALL-E is capable of responding to gestures with expressions that range from happiness to surprise. Additionally, a custom-built Android application adds further interaction options by enabling remote control.

By creating an emotionally responsive robot, this project aims to show how technological advancements can go beyond the functional and instead create bonds that are both comforting and beneficial, particularly for those with barriers in expressing or understanding emotions. This document will take you through the process of building WALL-E, the decisions made during the development, the challenges faced, and how this robot represents a step forward in the field of affective robotics.

Important Skills Learned

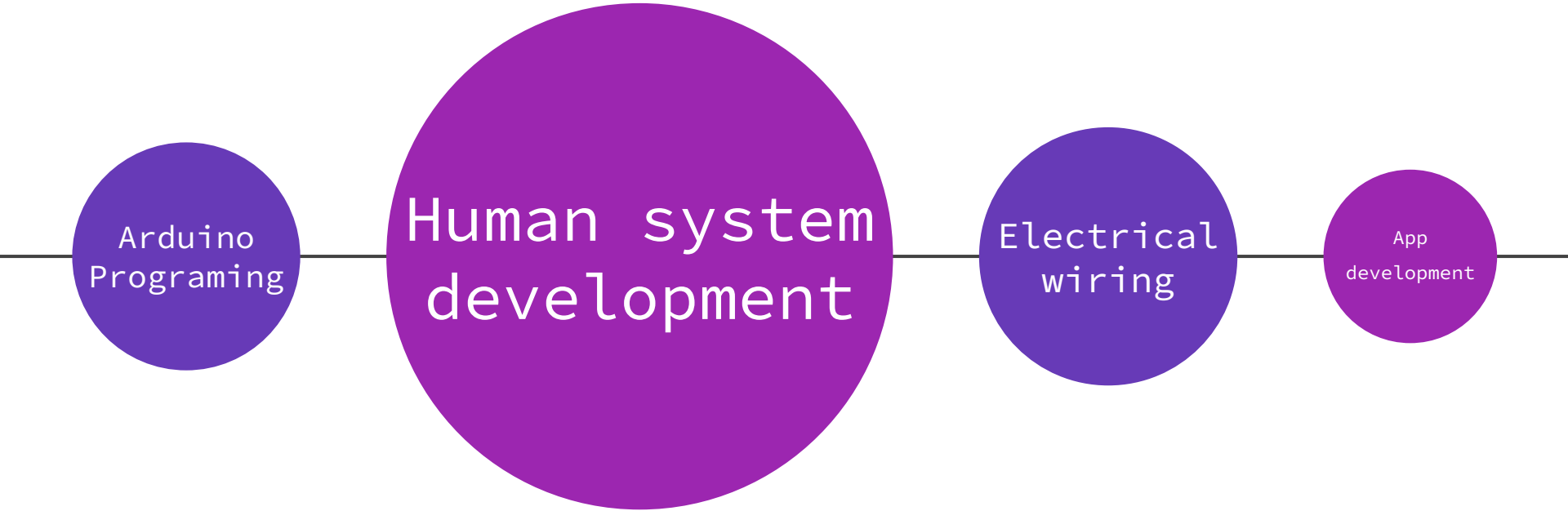
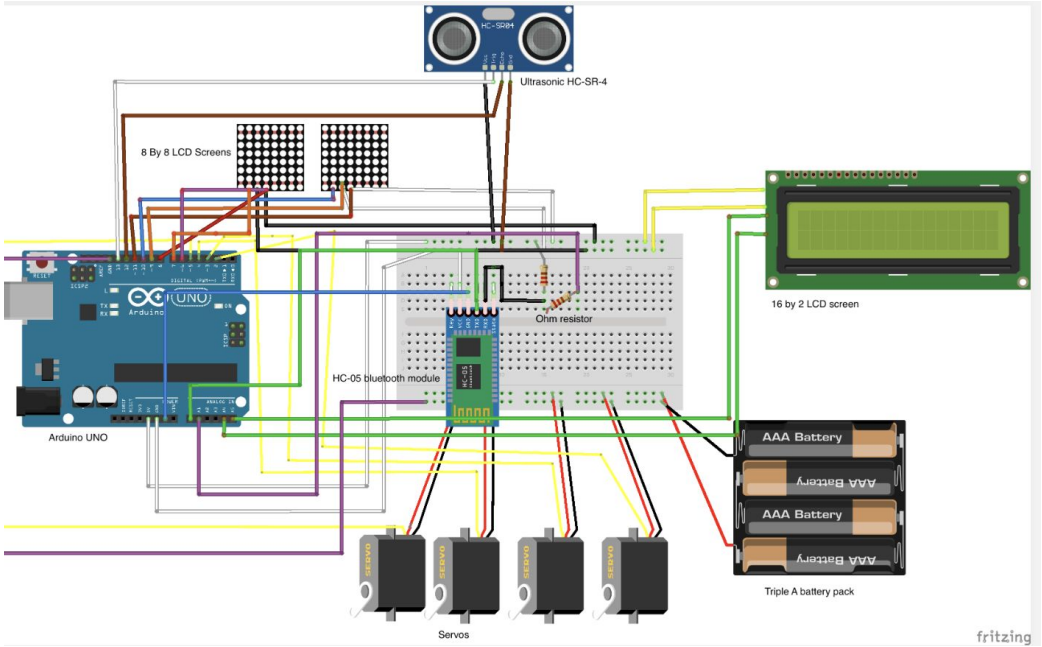
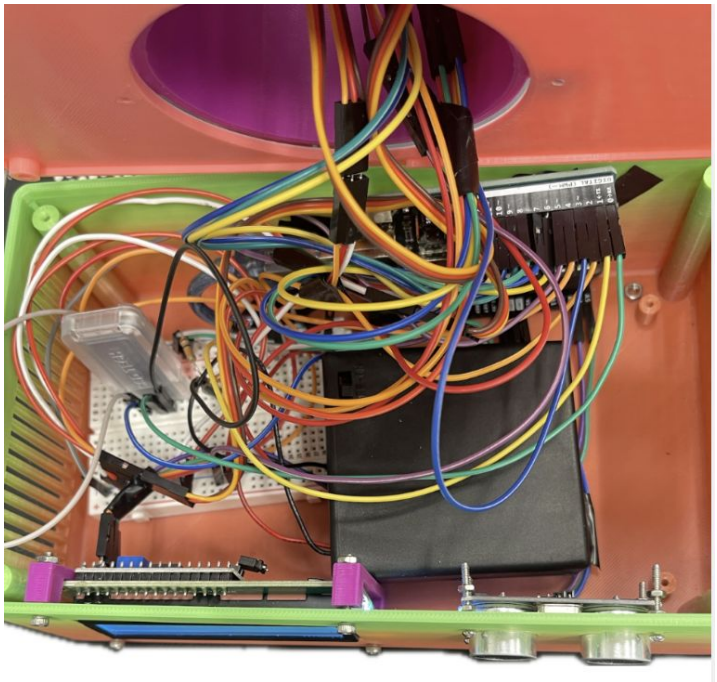
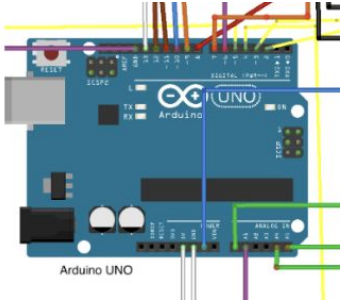


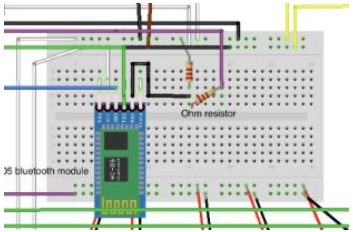
Figure 1- Schematics





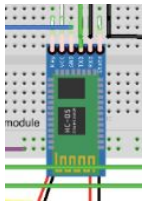


Microcontroller: Acts as the main control board, responsible for processing inputs from the various sensors and generating outputs to control the servos, LCD screen, and LED matrix.

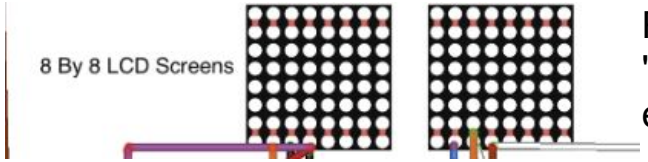


Prototyping Board: Used for creating temporary circuits without soldering.

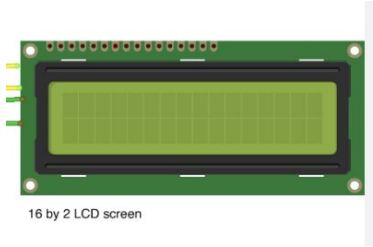
Servos: Control WALL-E's physical movements, such as head tilts and eyebrow movements, giving WALL-E dynamic responses based on different stimuli.



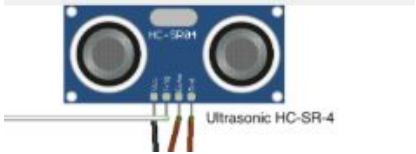
Connects WALL-E to a mobile device, allowing remote control through a custom-built Android app



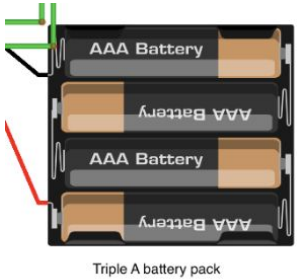
Represents different emotional states through changes in WALL-E's "eyes" by displaying different LED patterns, adding a layer of expressiveness.



Provides a text-based representation of WALL-E's emotional state, adding a layer of non-verbal communication to the interaction.

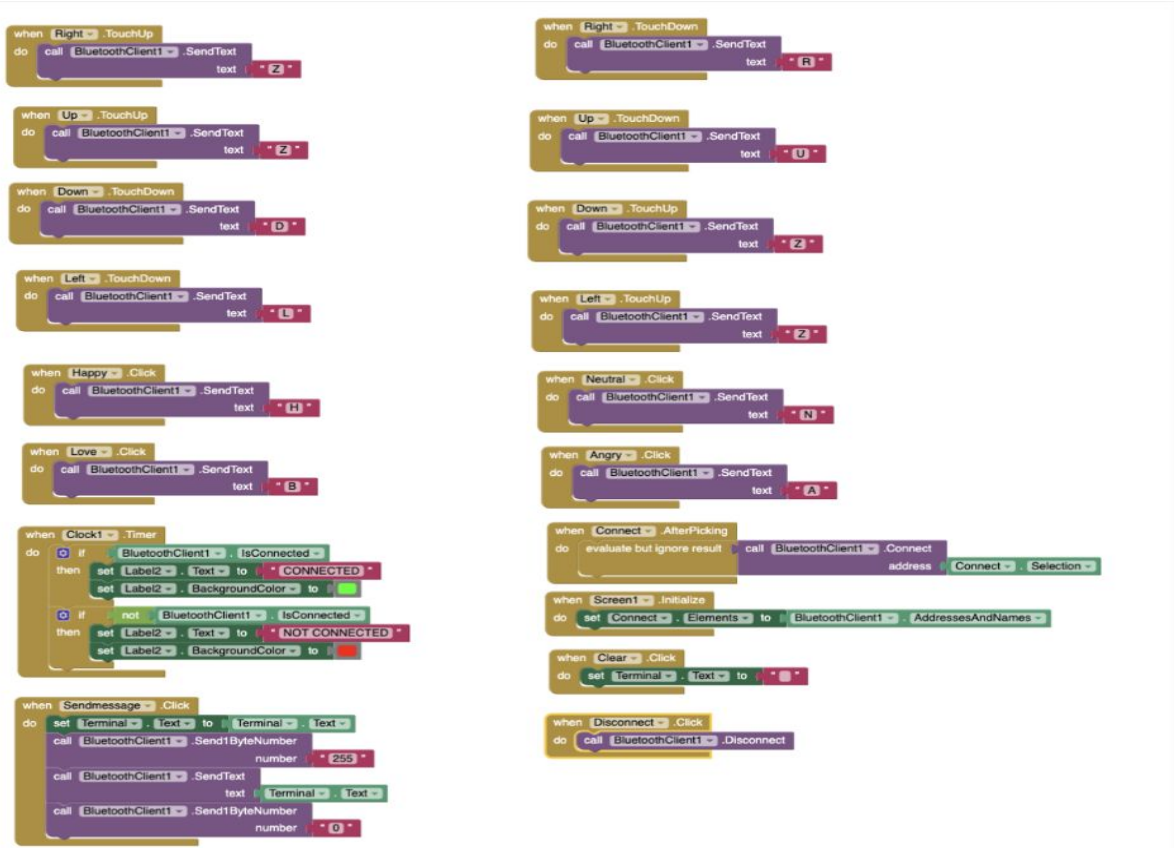


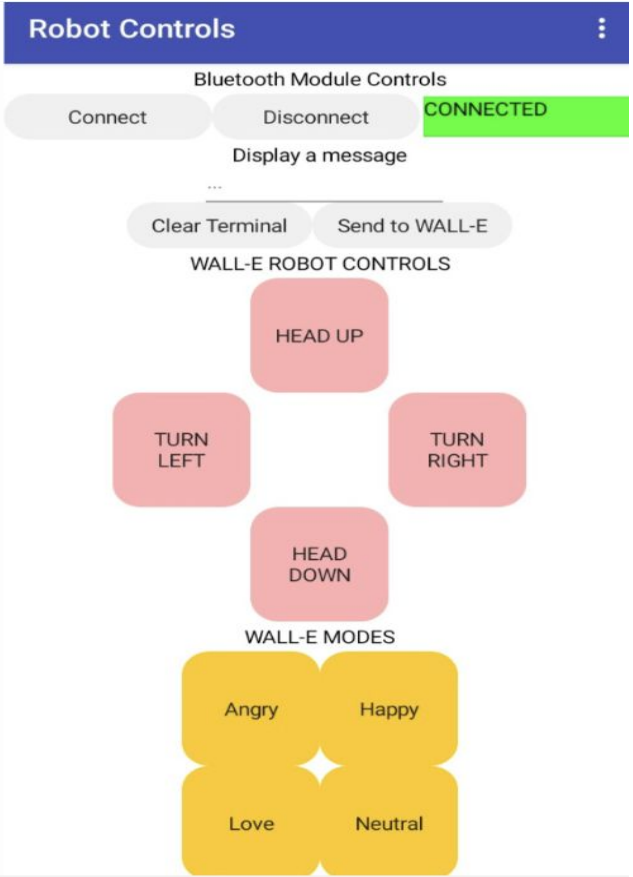
Ultrasonic Sensor (HC-SR04): Detects the user's proximity, enabling WALL-E to change its emotional expression depending on how close or far a user is.



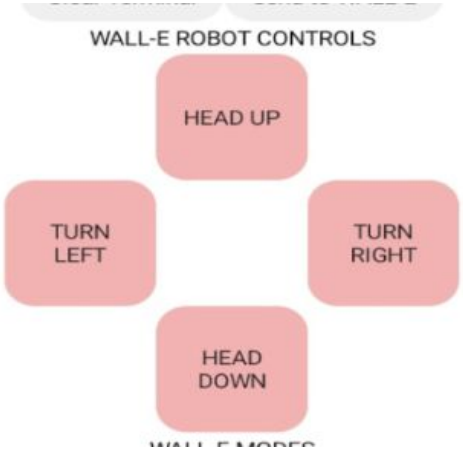
Gives power to the servos and components connected to the breadboard. The arduino is powered with a separate battery pack

Figure 2- App development- Code & interface

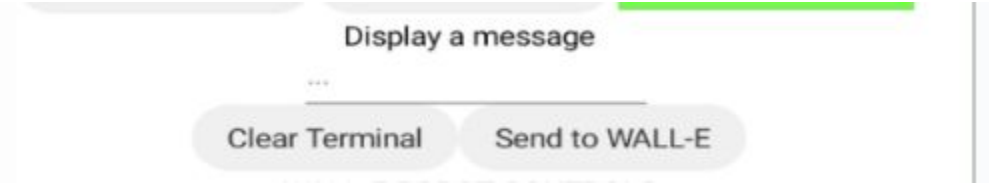




The Android application for remote control was built using MIT App Inventor.



These controls move the robot through the bluetooth sensors. Each button is stored as a state



This feature allows for a display of any message on the LCD screen.

These features make it easy for nonverbal speakers to communicate. Additionally, it helps individuals struggling with emotions to learn how to communicate different emotions by linking the buttons with different feelings.

Figure 3- C++ code for arduino

```
//pre-existing libraries
#include <LiquidCrystal_I2C.h>
#include <LedControl.h>
#include <Servo.h>
#include <NeoSWSerial.h>
// Pins for LED matrix connected to Arduino
int DIN = 11;
int CS = 9;
int CLK = 10;
int DIN_2 = 6;
int CS_2 = 8;
int CLK_2 = 7;

// Create instances for LCD and LED matrix
LiquidCrystal_I2C lcd(0x27, 16, 2);
LedControl ledmatrix = LedControl(DIN, CLK, CS);
LedControl ledmatrix_2 = LedControl(DIN_2, CLK_2, CS_2);

//create instances for bluetooth pins
NeoSWSerial bluetooth(A1, A0);

// Create instances for the Servos
Servo pan;
Servo tilt;
Servo leftBrow;
Servo rightBrow;

// Pins for the ultrasonic sensor
const int trigPin = 13;
const int echoPin = 12;

// Variable to measure distance
float duration, distance;

// bluetooth state variables
unsigned char state = 'N';
char recVbuffer[32];
unsigned recVindex = 0;
```

```
// Bitmap patterns for emotions
byte neutral_bmp[8] = {B00000000, B00111100, B01000010, B01011010, B01011010, B01000010, B00111100, B00000000};
byte happyL_bmp[8] = {B00000000, B00011100, B00100100, B01011100, B01011100, B00100100, B00011100, B00000000};
byte happyR_bmp[8] = {B00000000, B00111000, B00100100, B00111010, B00111010, B00100100, B00111000, B00000000};
byte loveL_bmp[8] = {B00011000, B00100100, B01000010, B10000001, B10000001, B10011001, B01100110, B00000000};
byte loveR_bmp[8] = {B00011000, B00100100, B01000010, B10000001, B10000001, B10011001, B01100110, B00000000};
byte sad_bmp[8] = {B10000001, B01000010, B00100100, B00011000, B00011000, B00100100, B01000010, B10000001};
byte surprisedL_bmp[8] = {B00000000, B01111110, B01000010, B01000010, B01000010, B01000010, B01111110, B00000000};
byte surprisedR_bmp[8] = {B00000000, B01111110, B01000010, B01000010, B01000010, B01000010, B01111110, B00000000};

// Function to display pattern on LED matrix
void displayPattern(byte patternL[], byte patternR[]) {
  for (int i = 0; i < 8; i++) {
    ledmatrix.setColumn(0, i, patternL[i]);
    ledmatrix_2.setColumn(0, i, patternR[i]);
  }
}

// Function to write emotion on LCD
void displayEmotion(String emotion) {
  lcd.clear();
  lcd.print(emotion);
}

// Function to move servos
void moveServos(int left, int right, int panIn, int tiltIn) {
  leftBrow.write(left);
  rightBrow.write(right);
  pan.write(panIn);
  tilt.write(tiltIn);
}

// Function to measure distance using ultrasonic sensor
float measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  return (duration * 0.0343) / 2;
}
```

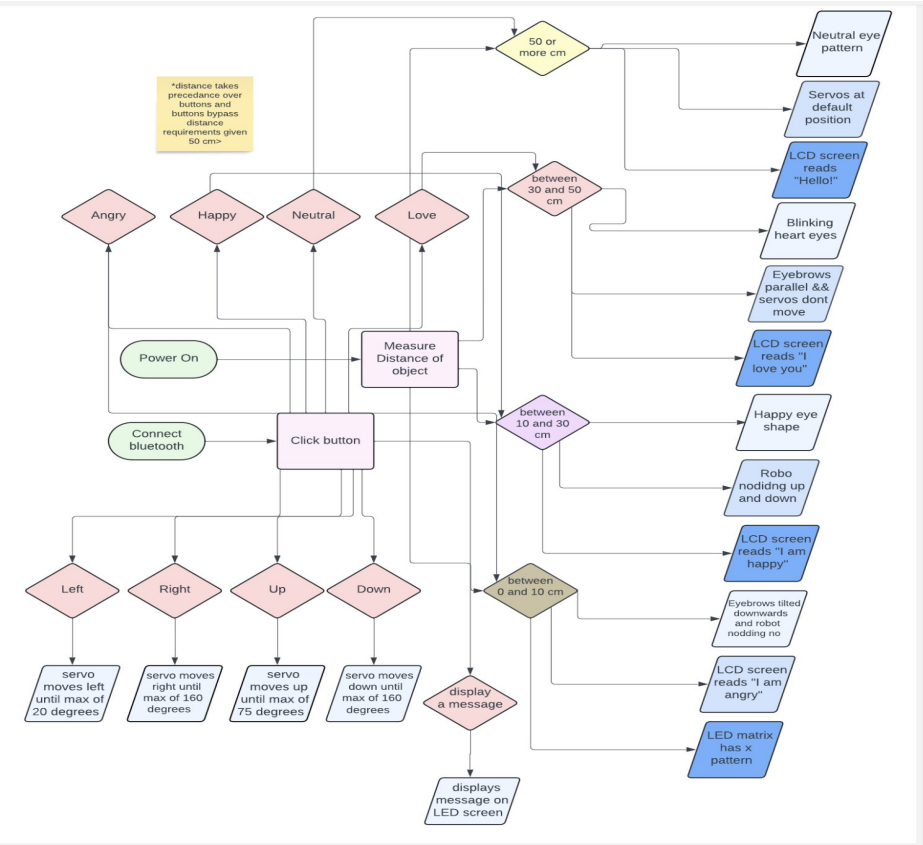
```
//variables for led lights to make code quicker
unsigned long lastBlinkTime = 0;
bool isDisplayOn = true;

void setup() {
  // Debugging
  Serial.begin(9600);
  bluetooth.begin(9600);

  // Set up LCD
  lcd.init();
  lcd.backlight();

  // Set up LED matrices
  ledmatrix.shutdown(0, false);
  ledmatrix.setIntensity(0, 1);
  ledmatrix.clearDisplay(0);

  ledmatrix_2.shutdown(0, false);
  ledmatrix_2.setIntensity(0, 1);
  ledmatrix_2.clearDisplay(0);

  // Initialize columns with letters
  displayPattern(neutral_bmp, neutral_bmp);  // In
  displayEmotion("Neutral");  // Initial neutral e

  // Define input/outputs
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  // Attach servos
  pan.attach(5);
  tilt.attach(4);
  leftBrow.attach(2);
  rightBrow.attach(3);

  // Set initial positions
  moveServos(60, 90, 130, 90);
}
```

```
void loop() {
  //logic to display text on LCD screen
  if (state == 255) {
    if (bluetooth.available() > 0) {
      char received = bluetooth.read();
      if (received == 0) {
        state = 0;
        recVbuffer[recVindex] = 0;
        displayEmotion(recVbuffer);
        Serial.println(recVbuffer);
        delay(3000);
      }
      if (recVindex < 33) {
        recVbuffer[recVindex] = received;
        recVindex += 1;
        if (recVindex == 16) {
          recVindex += 1;
        }
      }
    }
    return;
  }

  distance = measureDistance();

  //initialize bluetooth
  if (bluetooth.available() > 0) {
    state = bluetooth.read();
    Serial.println((unsigned)state);
  }

  if (state == 255) {
    recVindex = 0;
    return;
  }
```

```
  // Priority handling: Motion detection takes precedence
  if (distance <= 10 && distance > 0) {
    moveServos(60, 120, 130, 90);
    displayPattern(sad_bmp, sad_bmp);
    displayEmotion("Please back up!");
    for (int i = 90; i > 45; i--) {
      delay(10);
      tilt.write(i);
    }
    for (int j = 45; j < 135; j++) {
      delay(10);
      tilt.write(j);
    }
  } else if (distance <= 30 && distance > 10) {
    tilt.write(90);
    displayPattern(happyL_bmp, happyR_bmp);
    displayEmotion("I am so happy");
    leftBrow.write(90);
    rightBrow.write(90);
    for (int i = 150; i > 110; i--) {
      delay(10);
      pan.write(i);
    }
    for (int j = 110; j < 150; j++) {
      delay(10);
      pan.write(j);
    }
  } else if (distance <= 50 && distance > 30) {
    displayEmotion("I love you");
    moveServos(90, 90, 130, 90);
    displayPattern(loveL_bmp, loveR_bmp);
    delay(500);  // On for 500ms
    ledmatrix.clearDisplay(0);  // Turn off left matrix
    ledmatrix_2.clearDisplay(0);  // Turn off right matrix
    delay(500);  // Off for 500ms
  } else {
```

```
} else {
  // Switch cases to handle buttons pressing in app
  switch (state) {
    case 'A':
      displayPattern(sad_bmp, sad_bmp);
      displayEmotion("I am Angry!");
      moveServos(60, 120, 130, 90);
      for (int i = 90; i > 45; i--) {
        delay(10);
        tilt.write(i);
      }
      for (int j = 45; j < 135; j++) {
        delay(10);
        tilt.write(j);
      }
      break;

    case 'H':
      displayPattern(happyL_bmp, happyR_bmp);
      displayEmotion("I am Happy!");
      moveServos(120, 60, 130, 90);
      leftBrow.write(90);
      rightBrow.write(90);
      for (int i = 150; i > 110; i--) {
        delay(10);
        pan.write(i);
      }
      for (int j = 110; j < 150; j++) {
        delay(10);
        pan.write(j);
      }
      break;

    case 'N':
      displayPattern(neutral_bmp, neutral_bmp);
      displayEmotion("Hello!");
      moveServos(120, 60, 130, 90);
      break;

    case 'B':
      displayEmotion("Love!");
      moveServos(120, 60, 130, 90);
      if (millis() - lastBlinkTime >= 500) {
        lastBlinkTime = millis();
        if (isDisplayOn) {
          ledmatrix.clearDisplay(0);
          ledmatrix_2.clearDisplay(0);
        } else {
          displayPattern(loveL_bmp, loveR_bmp);
        }
        isDisplayOn = !isDisplayOn;
      }
      break;

    case 'L':
      displayEmotion("Looking Left");
      if (tilt.read() > 20) {
        tilt.write(tilt.read() - 3);
        delay(1);
      }
      break;

    case 'R':
      displayEmotion("Looking Right");
      if (tilt.read() < 160) {
        tilt.write(tilt.read() + 3);
        delay(1);
      }
      break;

    case 'U':
      displayEmotion("Looking Up");
      if (pan.read() > 75) {
        pan.write(pan.read() - 3);
        delay(1);
      }
      break;

    case 'D':
      displayEmotion("Looking Down");
      if (pan.read() < 160) {
        pan.write(pan.read() + 3);
        delay(1);
      }
      break;

    case 'Z':
      // Do nothing, keep the current position
      break;

    default:
      displayPattern(neutral_bmp, neutral_bmp);
      displayEmotion("Hello!");
      moveServos(120, 60, 130, 90);  // Eyebrows up, slight head til
      break;
  }
 }
}
```

Figure 4- Flowchart to visualize code



Key features

- Motion detection takes precedence
- States are used to read bluetooth commands
- Distance is used to display different emotions
- Emotions are shown through LED patterns on the eyes and the body and eyebrow servos

Figure 5-

Bill of Materials

Part	Note	Price	Link
HC-05 Bluetooth Module	Connect Bluetooth with phone allowing servos to move	$10.39	Link
Microservos (2x)	Controlling eyebrows	$7.00	Link
Servos (2x) MG995	Controlling the head and neck	$10.39	Link
Ultrasonic Sensor HC-SR04	Detect motion using sound waves	$6.40	Link
8x8 LED Matrix	Controlling eye emotions	$4.88	Link
LCD Screen	Displaying messages	$9.99	Link
Battery Pack	Power servos	$5.98	Link
Arduino UNO	Code processor	$27.60	Link
Mini Breadboard	Helps provide processing power to servos	$6.75	Link
Various M2 & M3 Screws	Assembly toolbox	$4.99	Link
Jumper Cables	Connecting all components together	$5.98	Link
3D Printed Parts	Container of WALL-E robot	Variable	N/A

Conclusion- Potential applications

Education and Social Interaction
In educational settings, WALL-E could serve as an engaging way to teach emotional intelligence to children. Using a robot to teach children how to identify emotions provides a safe and interactive method for practicing social skills without the complexities and unpredictability of human interaction. Studies have shown that robots can significantly improve the learning experience by providing feedback that children perceive as non-threatening, thereby boosting their confidence and willingness to learn.

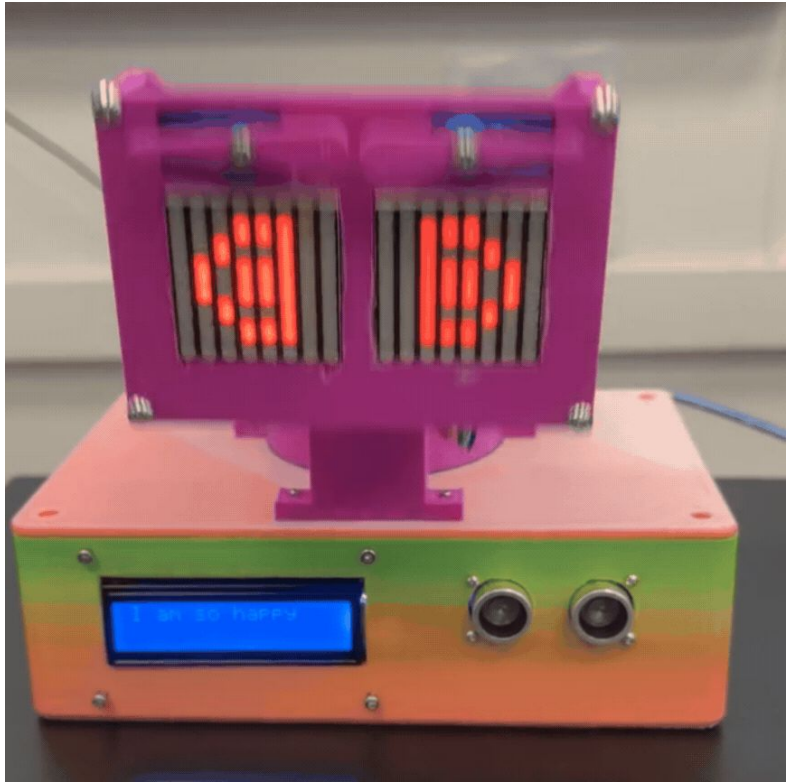
Therapy and Emotional Support
Therapists could use WALL-E to facilitate emotional awareness exercises. The robot's expressions could serve as visual aids during therapy sessions, helping patients understand how emotions are represented through physical movements. For example, WALL-E can react with sad expressions when approached too closely or display happiness when interacted with gently, thereby serving as a non-verbal medium that makes emotional recognition more intuitive. This capability has promising applications in treating conditions such as autism, where understanding social cues is often a challenge.

Conclusion- Future improvements

Machine Learning for Emotional Adaptation
WALL-E could become more sophisticated by incorporating a machine learning model that allows it to adapt its emotional responses over time. A reinforcement learning approach could be used, wherein WALL-E learns what emotional responses are appropriate based on the feedback it receives. For instance, if a particular expression consistently results in positive user reactions, WALL-E could learn to use it more frequently. Such adaptability would move WALL-E from being a pre-programmed device to an intelligent, evolving robot capable of learning from its environment.

Mobility and Autonomous Behavior
Currently, WALL-E is limited to stationary movements of its head and facial features. Adding mobility by equipping WALL-E with wheels or a crawler mechanism would enable it to follow users around, making it more akin to a robotic pet. Autonomous behavior could also be programmed—such as approaching users who appear to be sad or retreating when someone seems agitated. This mobility feature would allow WALL-E to act independently, enhancing the feeling of companionship.

Project outcomes



Through this project, I developed greater innovation and resilience as an engineer. Whether it was restarting 3D prints after unexpected failures or completely disassembling the build to address a critical issue, I persevered through these challenges to successfully create a solution aimed at social good.