Alyan Gillett

Tinkercad: [https://www.tinkercad.com/users/16G4AS9OnaZ?category=circuits&sort=likes&view\_mode=](https://www.tinkercad.com/users/16G4AS9OnaZ?category=circuits&sort=likes&view_mode=default) [default](https://www.tinkercad.com/users/16G4AS9OnaZ?category=circuits&sort=likes&view_mode=default)

**Github:**

<https://github.com/AlyanGillett>

**Youtube:**

[https://www.youtube.com/@AlyanENGR/featured](https://www.youtube.com/%40AlyanENGR/featured)

Embedded System Projects:

# RGB LED Fade

**Overview:**

Using an RGB LED to create a cycle of 3 different colors: red, green, and blue.

This will be done by starting off in a red state, fading to green, fading to blue, and then ending back in red.

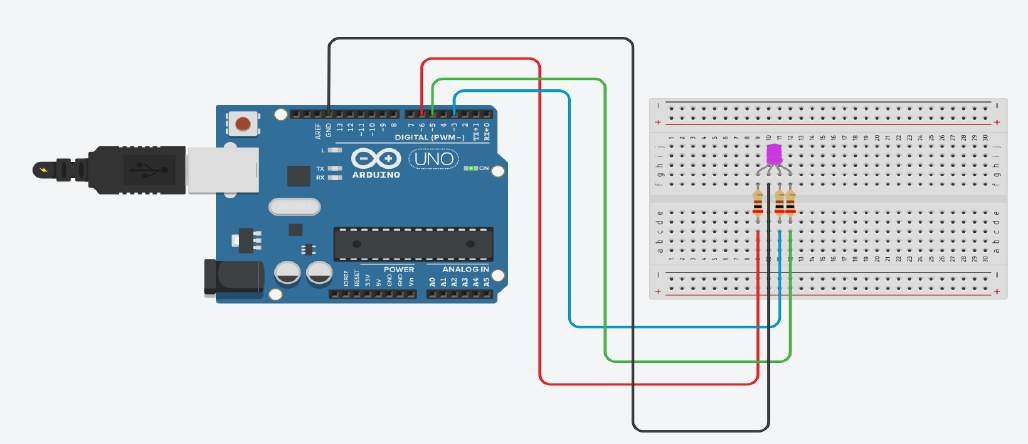
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x RGB LED**
  + **(3) x 220Ω resistor**
  + **(4) x M-M wires**

# Pictures:

## TinkerCAD design:

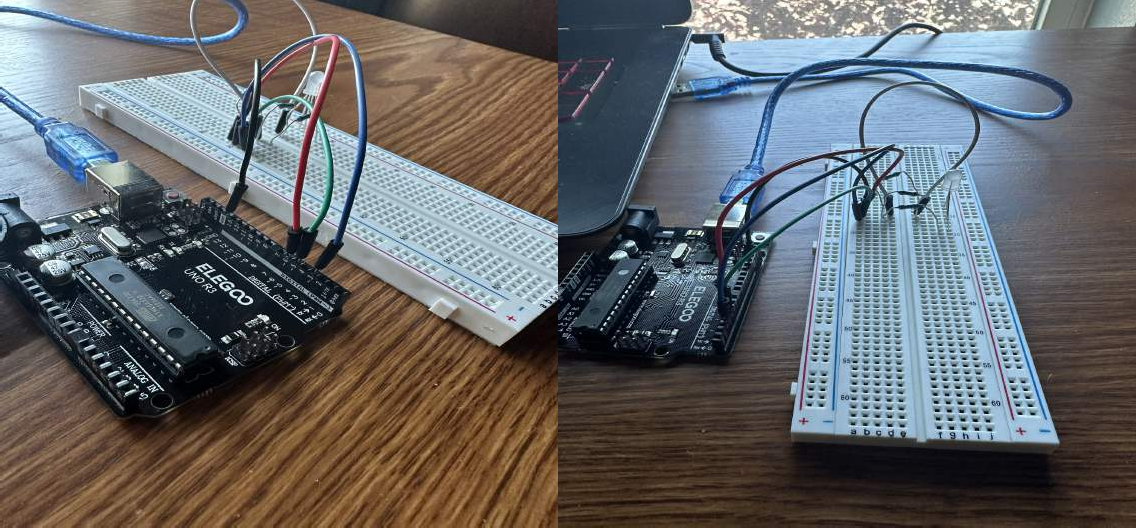
<https://www.tinkercad.com/things/ijmaU0IQkbo>



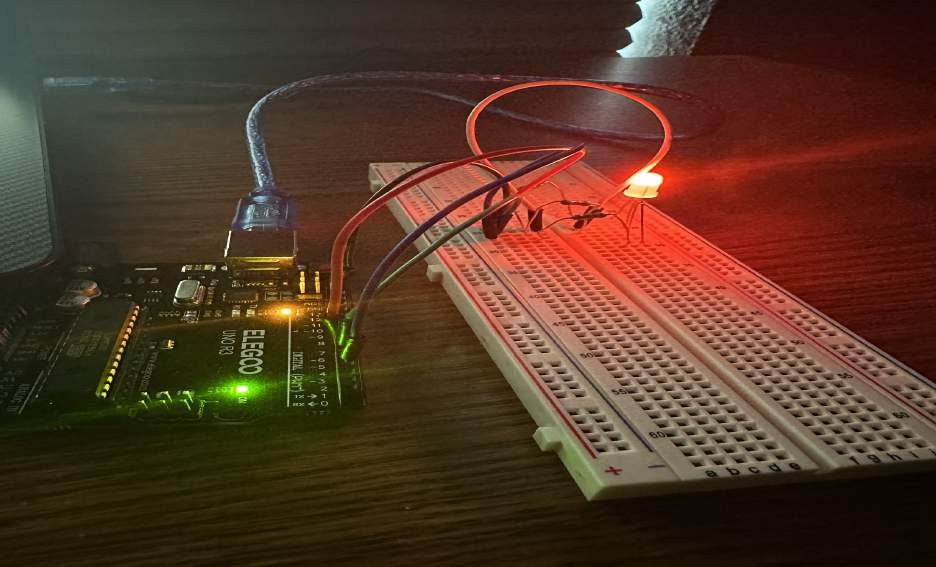
## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/RGB\_LED\_FA](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/RGB_LED_FADE.ino) [DE.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/RGB_LED_FADE.ino)

## Circuit off:



**Circuit on:**



**Live Video:**

[**https://youtu.be/l13l2Q9zUKo**](https://youtu.be/l13l2Q9zUKo)

**Things learned:**

 The initial state of the LED is set to display the color red by setting the RED pin to HIGH and the GREEN and BLUE pins to LOW using digitalWrite().

 The fading animation is achieved by gradually changing the values of the color channels (redValue, greenValue, blueValue) using the analogWrite() function. Each for loop iterates from 0 to 255, incrementing or decrementing the respective color channel values by 1 in each iteration.

 The delay() function is used to introduce a small delay (delayTime) between each change in color, creating a smooth fading effect.

# LED Push Button Display

**Overview:**

Using a push button with digital inputs in order to turn an LED on and off.

Pressing the button turns it on, and pressing the other button turns it off.

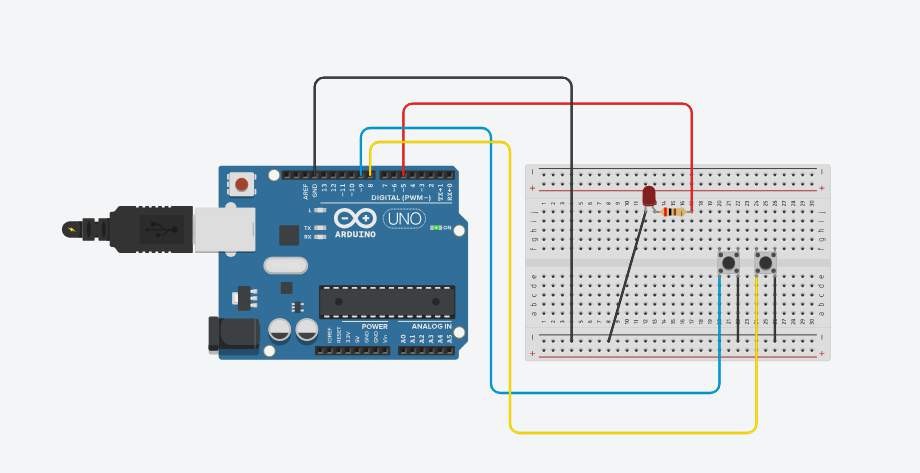
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x 5mm red LED**
  + **(1) x 220Ω resistor**
  + **(2) x push switches**
  + **(7) x M-M wires**

# Pictures:

## TinkerCAD design:

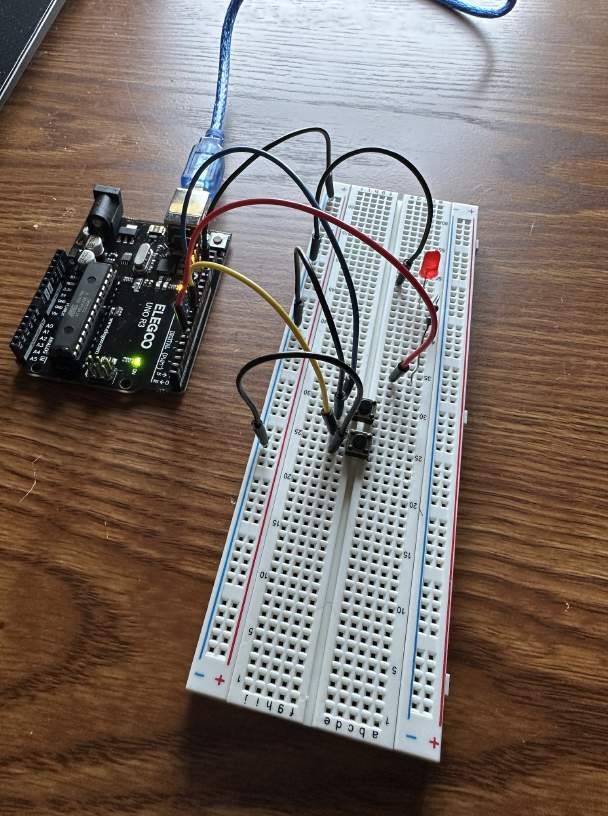
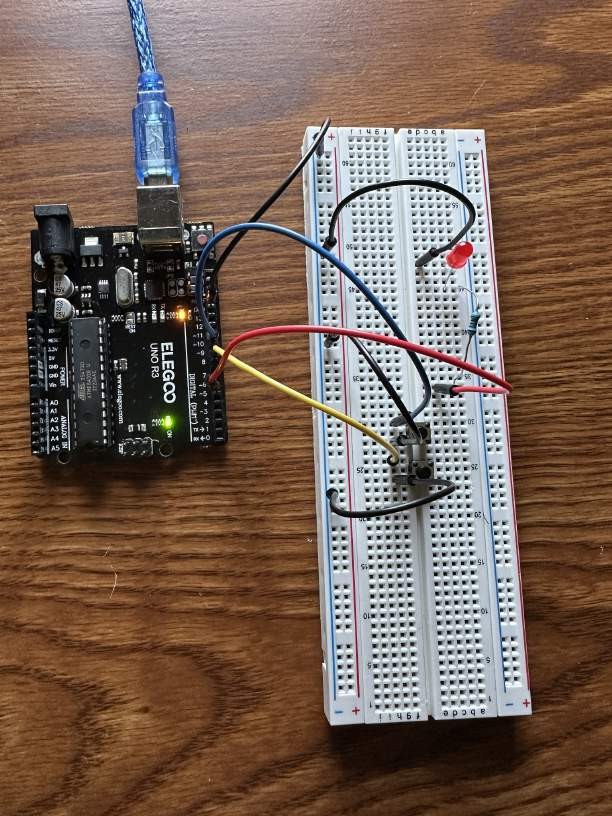
<https://www.tinkercad.com/things/5AIdDuVMtOo>



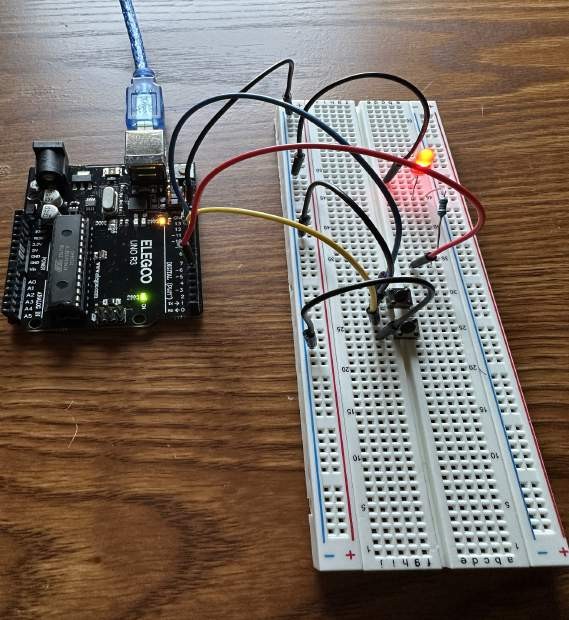
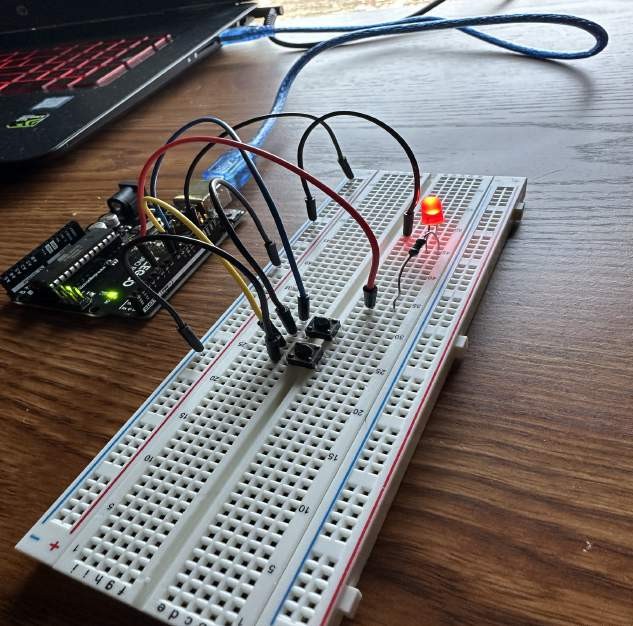
## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/LED\_PUSH\_](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/LED_PUSH_BUTTON_DISPLAY.ino) [BUTTON\_DISPLAY.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/LED_PUSH_BUTTON_DISPLAY.ino)

## Circuit off:



**Circuit on:**



**Live Video:**

[**https://youtu.be/l13l2Q9zUKo**](https://youtube.com/shorts/Z3f1ub3_-A4)

**Things learned:**

 The buttons are set as INPUT\_PULLUP, which means they are configured with internal pull-up resistors. This means that when a push button is not pressed, the corresponding input pin (buttonApin or buttonBpin) is pulled HIGH(ON state). This is the default state when the button is not being pressed.

 If button A (pin 9) is pressed (digitalRead(buttonApin) == LOW), the LED is turned on by setting pin 5 (ledPin) to HIGH (digitalWrite(ledPin, HIGH)).

 If button B (pin 8) is pressed (digitalRead(buttonBpin) == LOW), the LED is turned off by setting pin 5 (ledPin) to LOW (digitalWrite(ledPin, LOW)).

# 8 LED Light Show Display Overview:

Use eight large LEDS with arduino by using a chip called a 74HC595 Serial to parallel Converter. This process is so we don’t have to give up 8 output pins on the UNO.

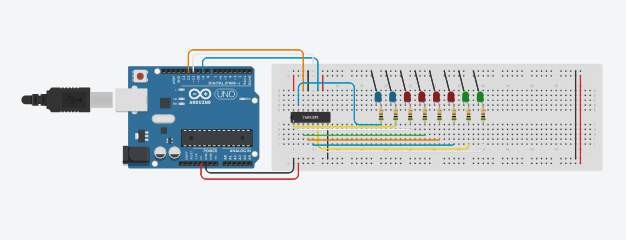
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(8) x 5mm LEDs**
  + **(8) x 220Ω resistor**
  + **(1) x 74hc595 IC**
  + **(14) x M-M wires**

# Pictures:

## TinkerCAD design:

<https://www.tinkercad.com/things/e5217bLDfZf>

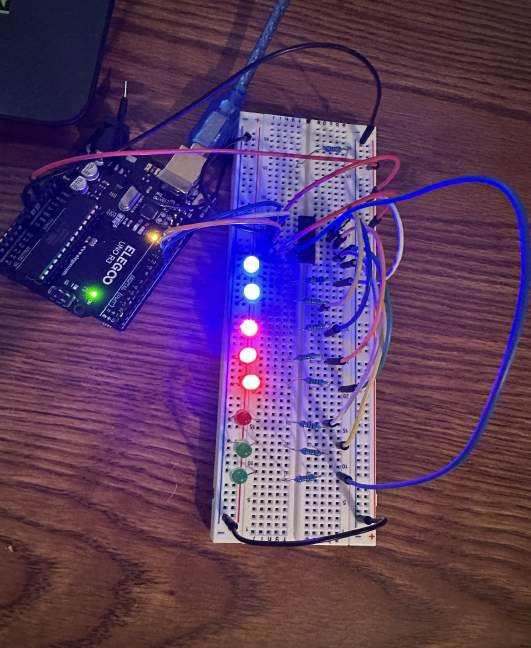


## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/EIGHT\_LED\_](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/EIGHT_LED_LIGHTSHOW_DISPLAY.ino) [LIGHTSHOW\_DISPLAY.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/EIGHT_LED_LIGHTSHOW_DISPLAY.ino)

## Circuit off:

**Circuit on:**



**Live Video:**

[**https://youtu.be/l13l2Q9zUKo**](https://youtube.com/shorts/C7xxOmLmXl0?feature=share)

**Things learned:**

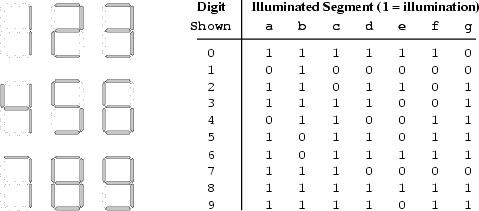
 learned how to shift data into the register and use it to control multiple LEDs efficiently.

 bitSet() and bitClear() are used to turn on and off individual LEDs in the leds variable. This provides an introduction to bitwise operations and manipulating binary values.

 Configuring GPIO pins of the microcontroller as outputs and set their states using pinMode() and digitalWrite()

# Seven Segment Display Cathode Overview:

Use a seven-segment display cathode in order to illustrate the count 0-9.



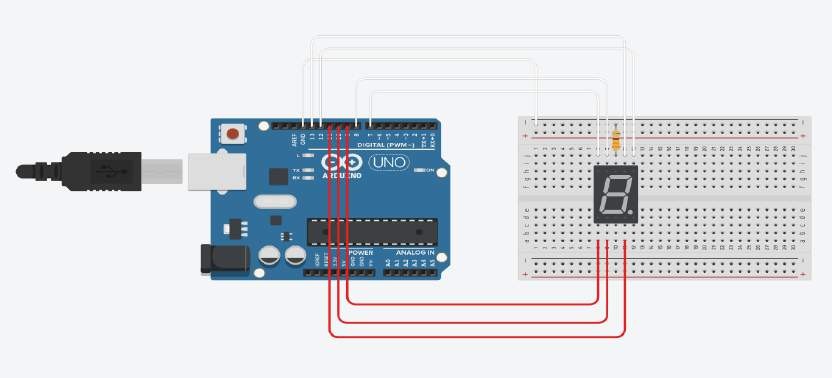
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x Seven Segment Display**
  + **(1) x 330Ω resistor(1)**
  + **(7) x M-M wires**

# Pictures:

## TinkerCAD design:

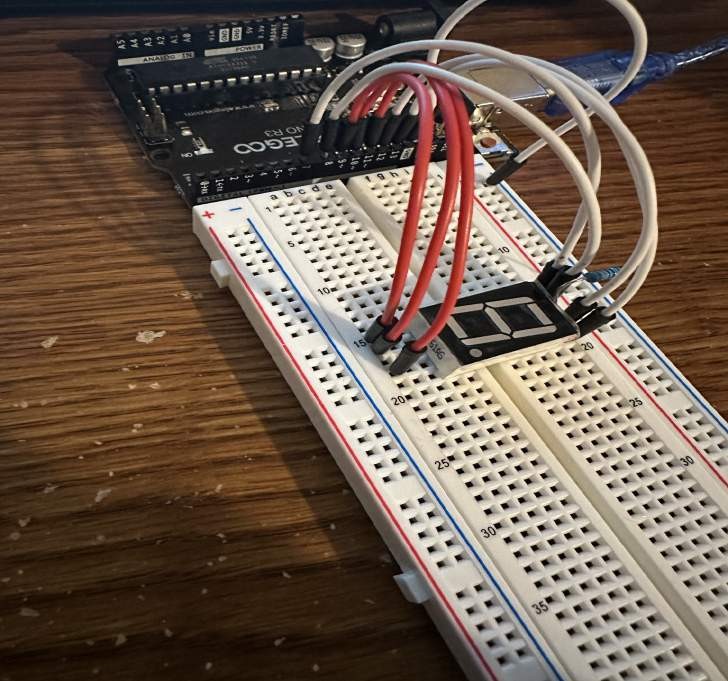
<https://www.tinkercad.com/things/l342quXkDsv>

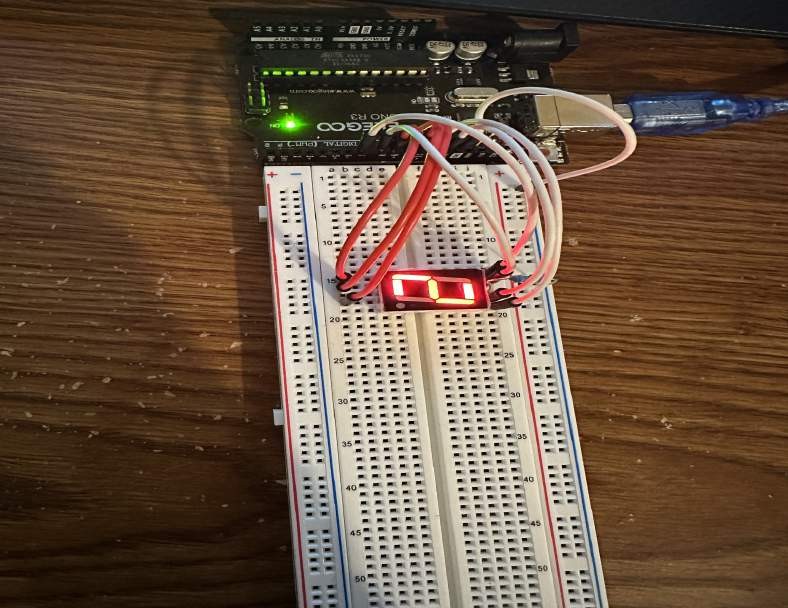


## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/SEVEN\_SEG](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/SEVEN_SEGMENT_DISPLAY_CATHODE.ino) [MENT\_DISPLAY\_CATHODE.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/SEVEN_SEGMENT_DISPLAY_CATHODE.ino)

## Circuit off:



**Circuit on:**

**Live Video:**

[**https://youtu.be/l13l2Q9zUKo**](https://youtube.com/shorts/DVrDo1Pobeo)

**Things learned:**

 The code sequentially displays the digits 0 to 9 on the seven-segment display.

 Each digit is displayed by selectively turning on/off the segments using digitalWrite() function and setting the pin values accordingly.

# LCD Lyrics Display With Song Synchronization Overview:

This Tinkercad project incorporates an LCD screen to display lyrics while playing a song simultaneously. The project utilizes an Arduino board and a LiquidCrystal library for controlling the LCD. The lyrics of the song are stored in an array, and each line is displayed on the LCD screen in a loop. A timer is also implemented to synchronize the start of the song with the lyrics on the screen.

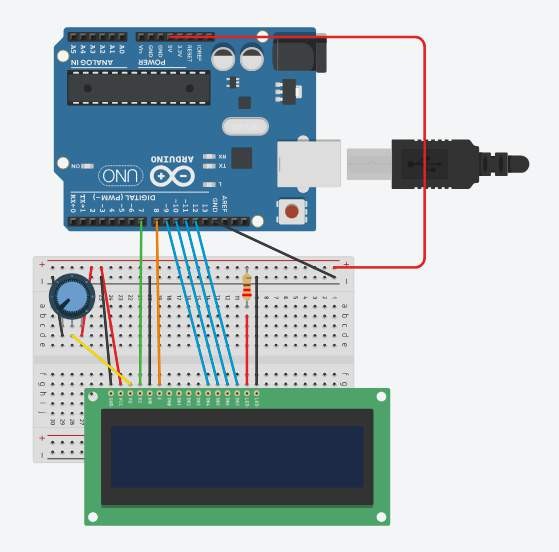
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x LCD1602**
  + **(1) x 220Ω resistor(1)**
  + **(16) x M-M wires**

# Pictures:

## TinkerCAD design:

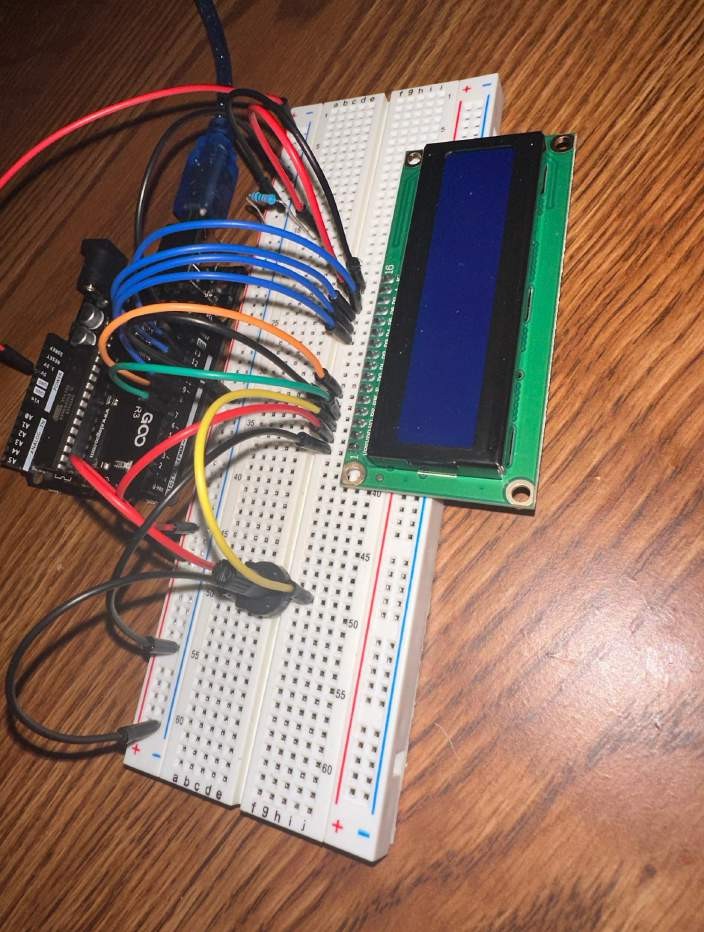
<https://www.tinkercad.com/things/gKByVYoqsOr>

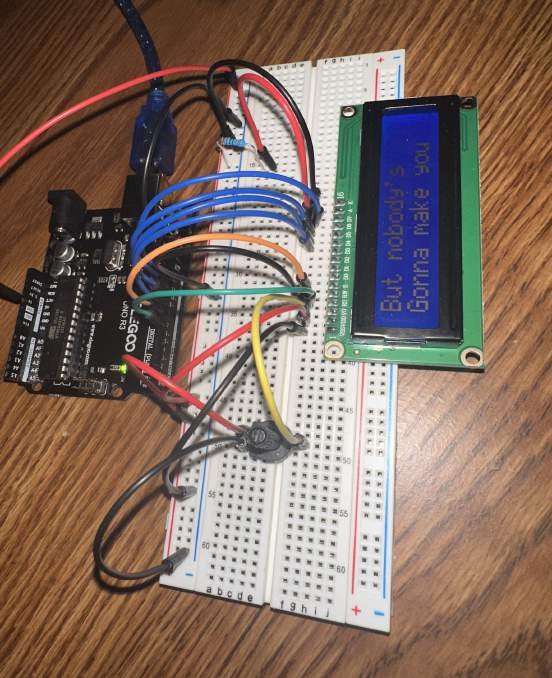


## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/LCD\_LYRICS\_](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/LCD_LYRICS_SONG_SYNCHRONIZATION.ino) [SONG\_SYNCHRONIZATION.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/LCD_LYRICS_SONG_SYNCHRONIZATION.ino)

## Circuit off:



**Circuit on:**

**Live Video:**

[**YT VIDEO**](https://youtube.com/shorts/uklZGwgLY1Y)

**Things learned:**

 Working with LCD screens and controlling them using an Arduino board.  Storing and retrieving data efficiently using arrays.

 Synchronizing the display of lyrics with the playback of a song using delay times.

 Understanding the importance of loops and iteration in controlling program flow.

 Enhancing music and visual displays through creative technological integration.

# Ultrasonic Sensor With LCD Display Overview:

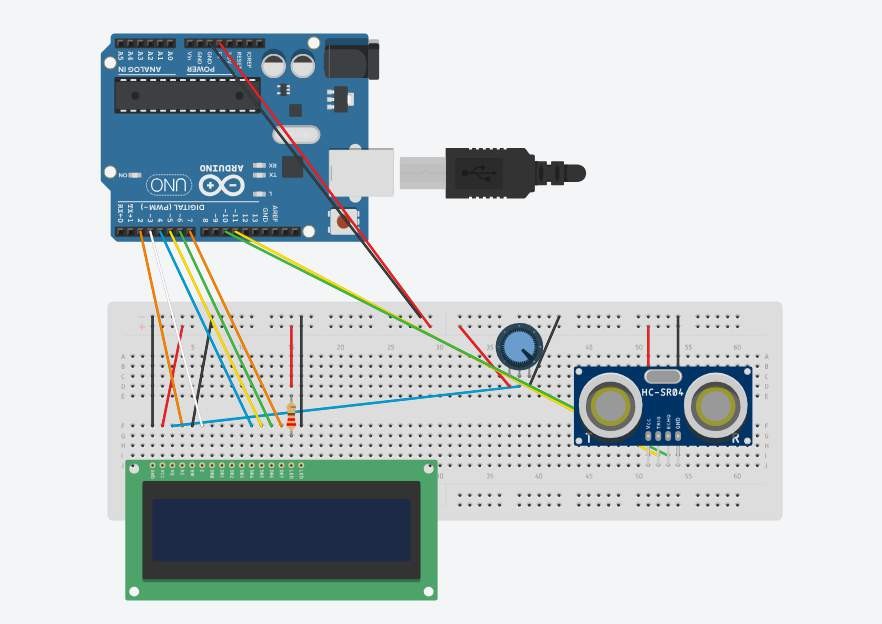
This project uses an Arduino Uno board, an ultrasonic sensor, and an LCD display to create a distance measurement device. The ultrasonic sensor emits sound waves and measures the time it takes for the sound waves to bounce back after hitting an object. By calculating the time delay, the distance to the object can be determined.

# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x LCD1602**
  + **(1) x 220Ω resistor**
  + **(20) x M-M wires**
  + **(1) x Ruler**
  + **(1) x object with a flat surface preferably**

# Pictures:

## TinkerCAD design:

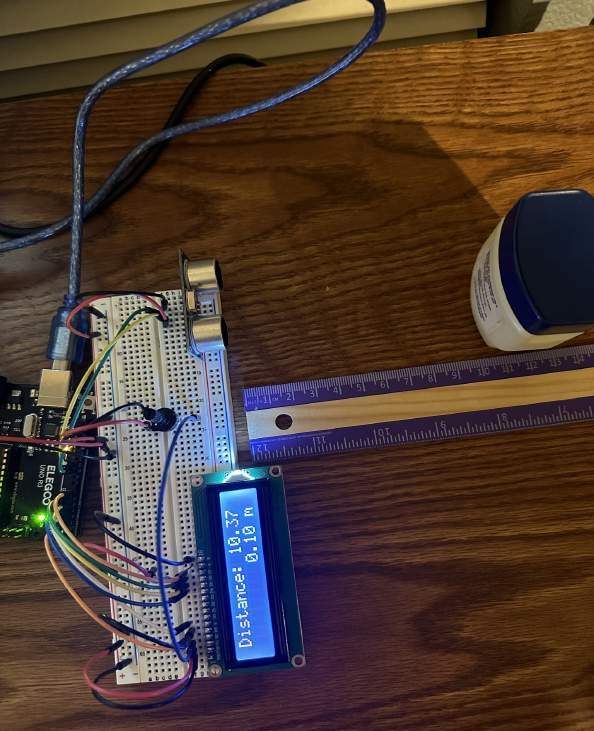
<https://www.tinkercad.com/things/gpbJOINUNQm>

## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/ULTRASONIC](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/ULTRASONIC_SENSOR_DISPLAY.ino)

[\_SENSOR\_DISPLAY.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/ULTRASONIC_SENSOR_DISPLAY.ino)

## Circuit off:

**Circuit on:**

**Live Video:**

[**YT VIDEO**](https://youtube.com/shorts/rJajmdL_77A)

**Things learned:**

 The ultrasonic sensor consists of a transmitter (trig pin) that emits sound waves and a receiver (echo pin) that detects the reflected waves.

 Using the pulseIn() function to measure the duration of a pulse.

 The calculation of the distance in centimeters involves multiplying the pulse duration(µs) by the conversion factor of 0.0343 and dividing it by 2 to account for the round-trip time of the sound wave.

# Servomotor Overview:

This Tinkercad involves creating a servo motor control system using an Arduino and a potentiometer. The potentiometer allows users to manipulate the servo motor's position, enabling control over its movement and direction.

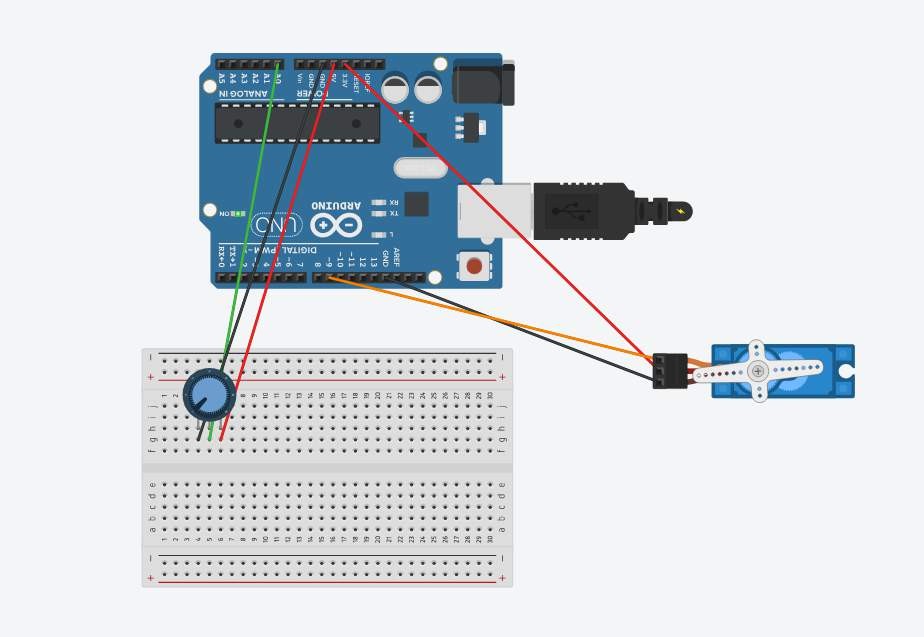
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(6) x F-M wires**
  + **(1) x potentiometer**
  + **(1) x Servomotor**

# Pictures:

## TinkerCAD design:

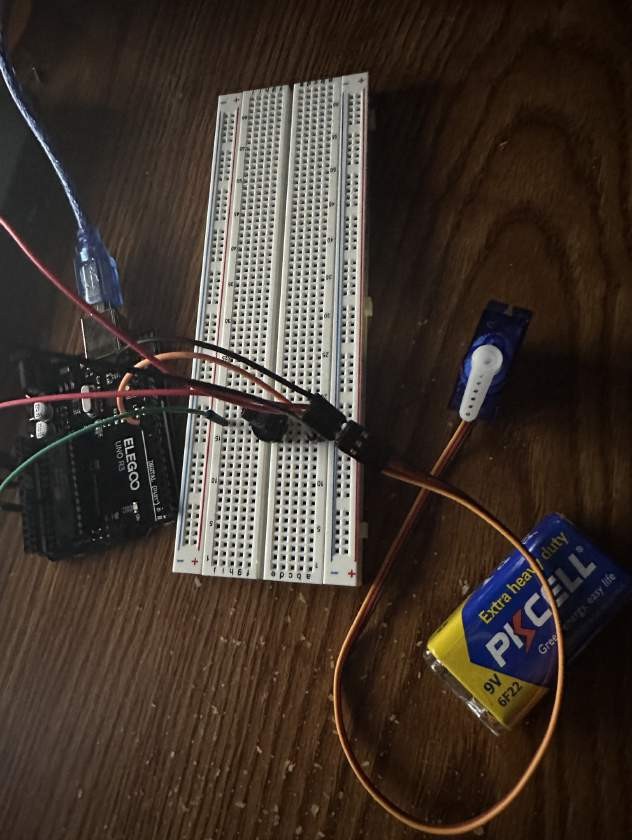
<https://www.tinkercad.com/things/ewzMGsisuG3>

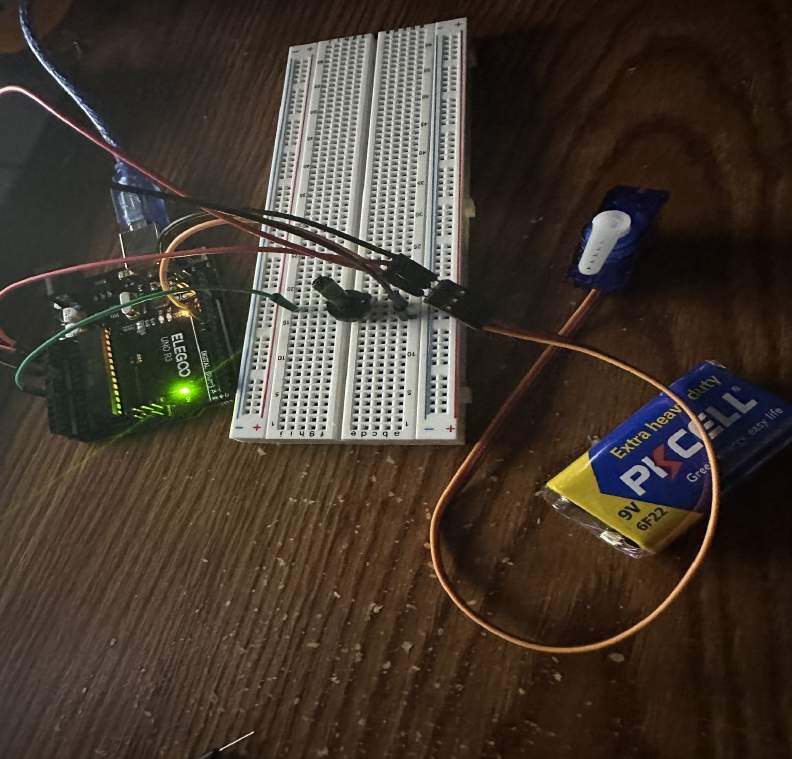


## Code: written in TinkerCAD

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/SERVO\_MOT](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/SERVO_MOTOR.ino) [OR.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/SERVO_MOTOR.ino)

## Circuit off:



**Circuit on:**

**Live Video:**

[**YT VIDEO**](https://youtube.com/shorts/fn3sweWq53M?feature=share)

**Things learned:**

 By adjusting the delay value, you can control the speed at which the servo motor moves, with smaller delay values resulting in faster movement.

 The Servo library in Arduino provides convenient functions for controlling servo motors, allowing precise positioning based on input values

 The potentiometer's reading is obtained using analogRead(potPin), which returns a value between 0 and 1023. This value is then mapped to a servo angle between 0 and 180 degrees using the map() function

# Temperature And Humidity Display

This Tinkercad involves creating a temperature and humidity monitoring system using an Arduino, a DHT11 sensor, and a LiquidCrystal display. The system continuously retrieves temperature and humidity data from the DHT11 sensor, converting the temperature reading into Fahrenheit and then presenting both temperature and humidity values on the LCD screen.

Moreover, it provides a proper display with real-time updates of the temperature and humidity, with a 2-second pause between readings which can be seen on the Serial Monitor data stream.

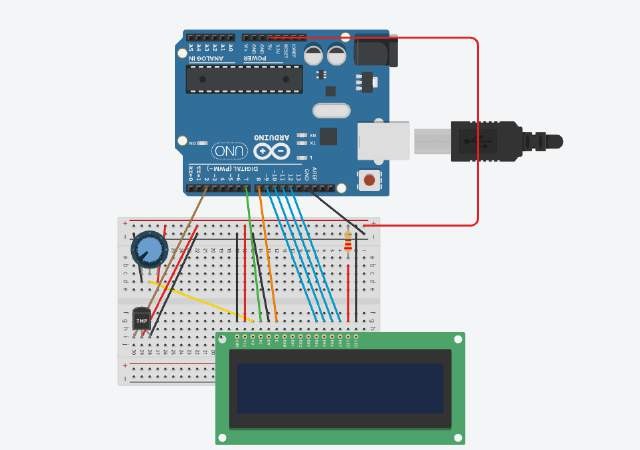
# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x LCD1602**
  + **(1) x 220Ω resistor**
  + **(3) x M-M wires**
  + **(16) x F-M wires**
  + **(1) x potentiometer**
  + **(1) x DHT11**

# Pictures:

## TinkerCAD design:

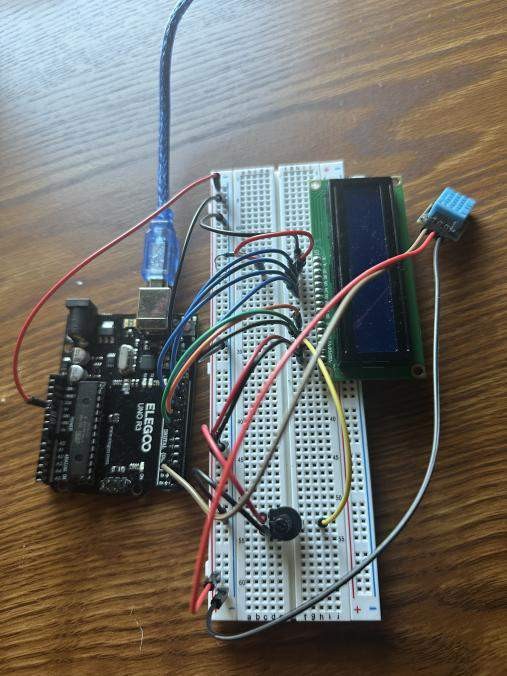
<https://www.tinkercad.com/things/5PaZ3cR6REk>



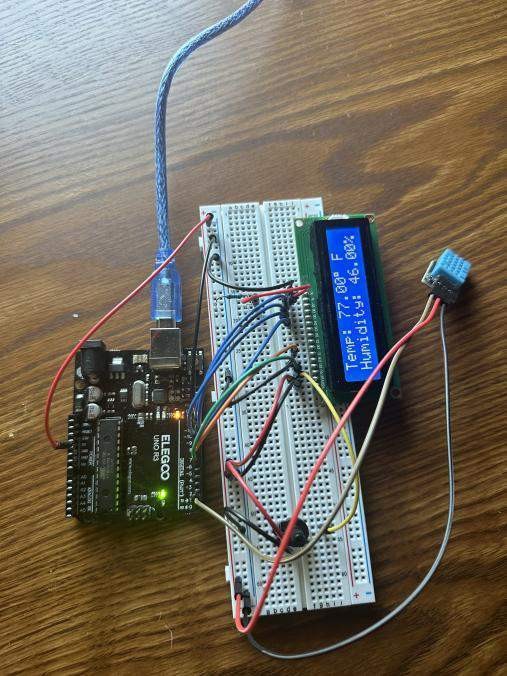
## Code:

[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/TEMPERATU](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/TEMPERATURE_HUMIDITY_DISPLAY.ino) [RE\_HUMIDITY\_DISPLAY.ino](https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/TEMPERATURE_HUMIDITY_DISPLAY.ino)

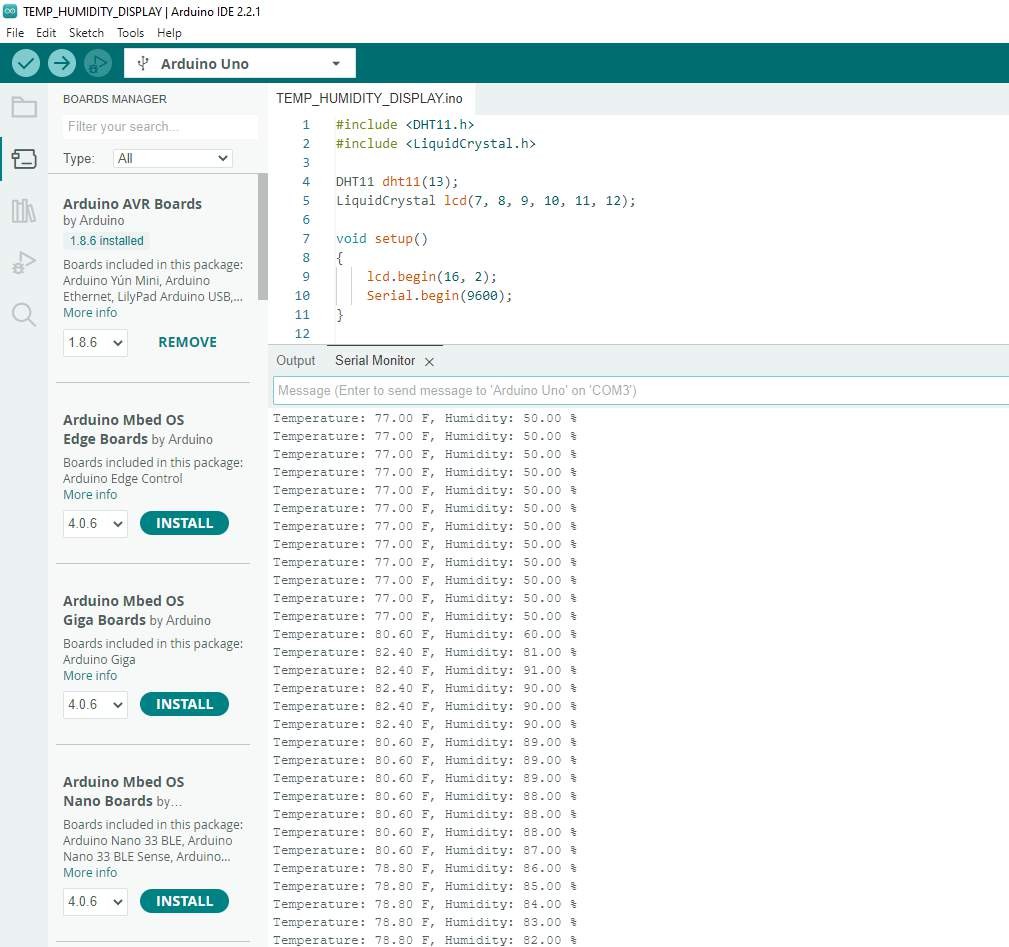
## Circuit off:



**Circuit on:**



**Serial Monitor:**



**Live Video:**

[**YT VIDEO**](https://youtube.com/shorts/HeeDZDpap-w)

**Things learned:**

 Reading sensor data(temperature and humidity) from the DHT11.

 Utilizing the Serial communication to send data from the Arduino to the computer for debugging and monitoring purposes.

 The potentiometer's reading is obtained using analogRead(potPin), which returns a value between 0 and 1023. This value is then mapped to a servo angle between 0 and 180 degrees using the map() function

# Climate System Control Display

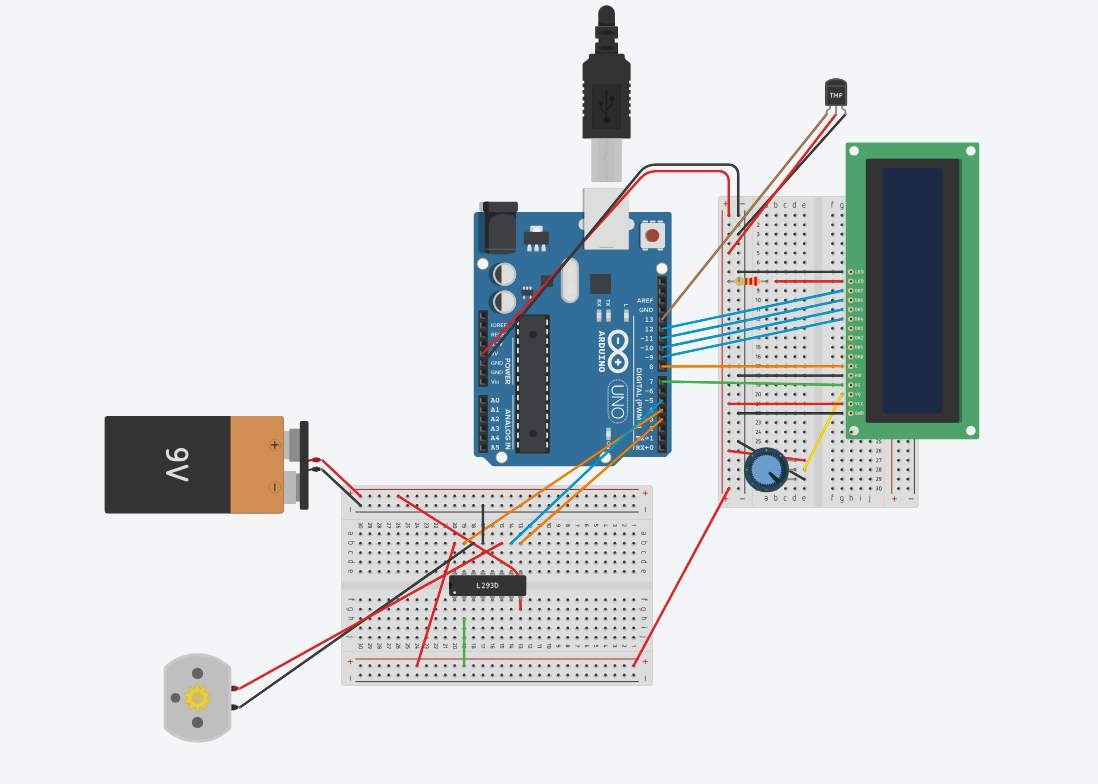
This Tinkercad involves creating a climate control system with real-time monitoring and fan automation using an Arduino board, a DHT11 sensor, and a LiquidCrystal display. The code initializes the necessary libraries and pins for the components, sets a temperature threshold, and controls a fan based on the temperature readings. Nonetheless, it continuously reads temperature and humidity data from the sensor, illustrates it on the LCD screen, and turns on the fan if the temperature exceeds the specified threshold, helping to maintain the desired temperature. Additionally, the code includes functions for controlling the fan's direction and speed.

# Components used:

* + **(1) x Elegoo Uno R3**
  + **(1) x Breadboard Mb-102**
  + **(1) x LCD1602**
  + **(1) x DHT11**
  + **(1) x potentiometer**
  + **(1) x 9V Battery**
  + **(1) x L293D Motor Driver**
  + **(1) x 220Ω resistor**
  + **(1) x Lighter**
  + **(3) x M-M wires**
  + **(27) x F-M wires**

# Pictures:

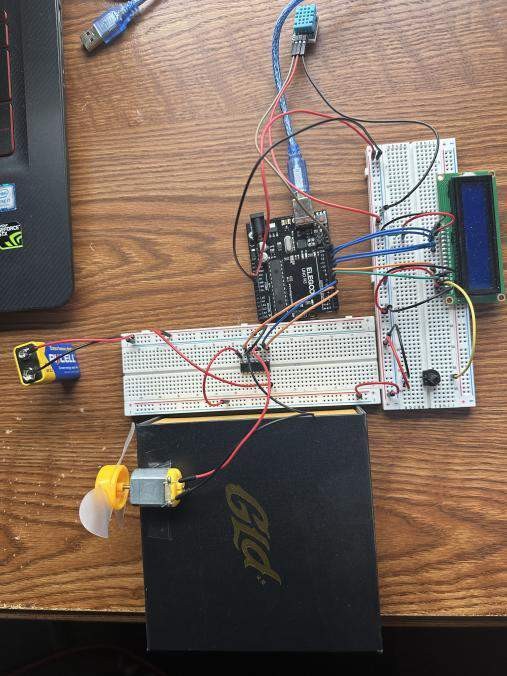
## TinkerCAD design:

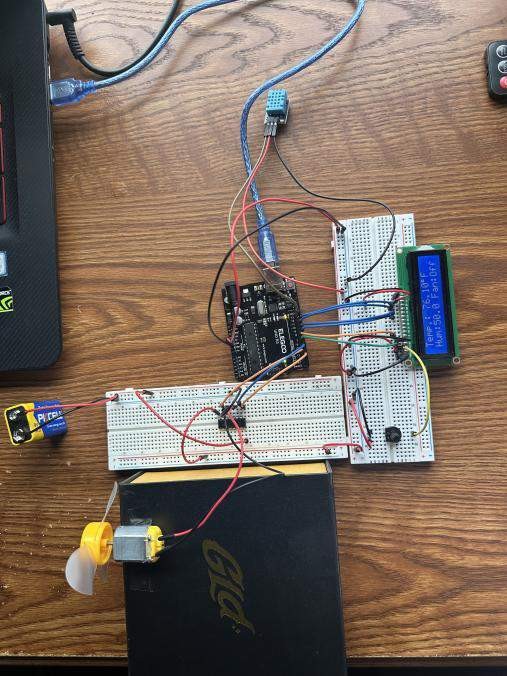
<https://www.tinkercad.com/things/i1pChzmCHvU>

## Code: written in TinkerCAD

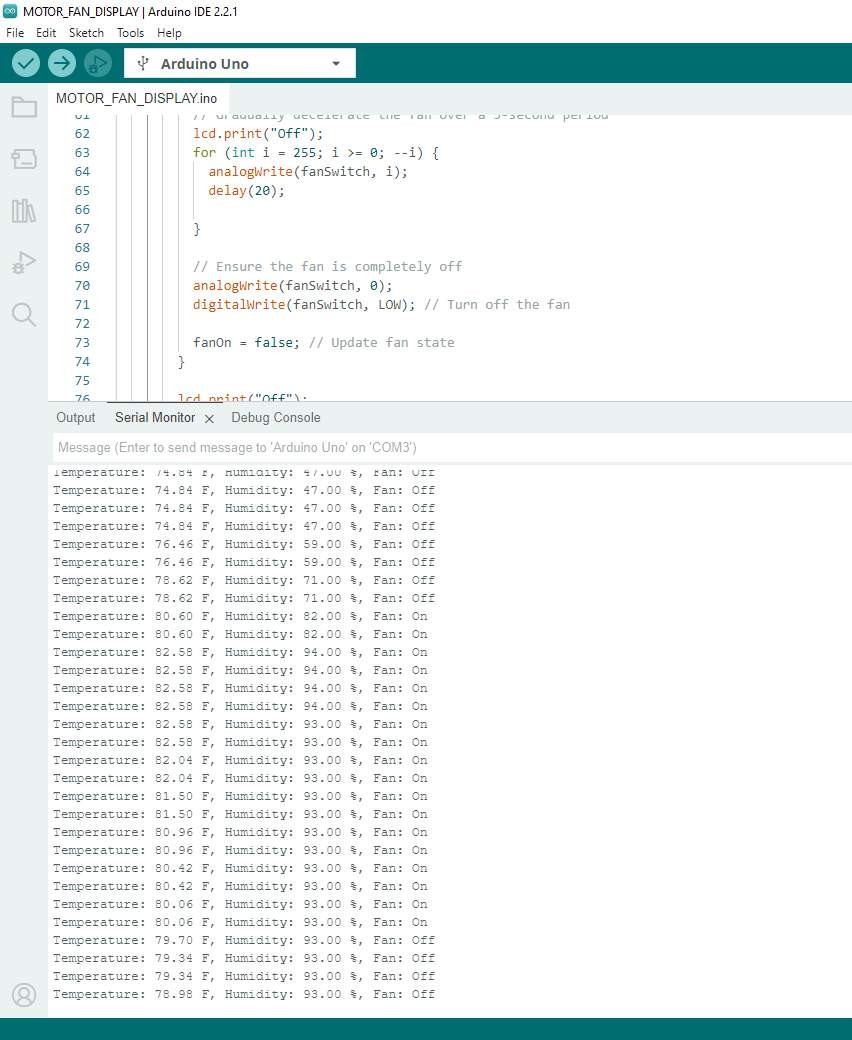
[https://github.com/AlyanGillett/Embedded-Systems-Projects/blob/main/CLIMATE\_SY](https://www.tinkercad.com/things/i1pChzmCHvU?sharecode=UJr_tbVJ9pCM51wlfbVPrKCgBOSAQSC-Ts3D9qKa7_cOg2rT5D4uz9xM184m_KhoTgpzRyYoOg) [STEM\_CONTROL\_DISPLAY.ino](https://www.tinkercad.com/things/i1pChzmCHvU?sharecode=UJr_tbVJ9pCM51wlfbVPrKCgBOSAQSC-Ts3D9qKa7_cOg2rT5D4uz9xM184m_KhoTgpzRyYoOg)

## Circuit off:



**Circuit on:**

**Serial Monitor:**



**Live Video:**

[**YT VIDEO**](https://youtube.com/shorts/h_ixe-wYg58)

**Things learned:**

 Uses a DHT11 sensor to read temperature and humidity data. It demonstrates how to obtain these readings using the readTemperature() and readHumidity() functions provided by the DHT library.

 Illustrate how to interface with a 16x2 LCD screen using the LiquidCrystal library. The code sets up the LCD and displays temperature, humidity, and fan status information on it.

 Implements a fan control system based on temperature readings. When the temperature exceeds a predefined threshold (fanThreshold), the fan is turned on using the speedControl() function. If the temperature drops below the threshold, the fan is gradually decelerated using a for loop and eventually turned off. The code efficiently manages the fan's state with the fanOn variable.

 Serial communication is used for debugging and monitoring purposes.

Temperature, humidity, and fan status are sent to the Serial Monitor, allowing real-time monitoring of the system's behavior.

 Functions created like (directionControl() and speedControl()) for controlling the fan's direction and speed, which may be useful in scenarios where fans need precise control.

 Error Handling in case the sensor fails to provide valid data, the code handles the error gracefully by displaying an error message on the LCD and the Serial Monitor.

 Timed Loop where the code operates with a delay of 1000 milliseconds (1 second) between each cycle, which determines the rate at which temperature and humidity readings are updated and displayed.