# EGR 311 Final Project - Sound Reacting LED Light Strip

***Created by:*** *John Araujo* , *Kevin Cheng* , *and* *Marcus Zinke*

**Objectives and Learning Outcomes:**

- Introduce IR sensor and Sound Detector
- Build a sound-reacting LED light strip
- Learn how to build a functional circuit with various components
- Learn about Arduino code and how to use various pins

**Estimated Completion Time:**

- 1.5 hours

**Knowledge checkpoints:**

1. Analog/ digital interface of Arduino
2. Using sensors with Arduino
3. Programming logic: if/else statements and for loops
4. HSV filter of color processing (optional)

**Equipment and Materials:**

- 1 Arduino Uno
- 1 1-meter LED light strip
- 1 Light strip holder
- SparkFun Sound Detector
- Alligator clips
- 1 Breadboard
- Jumper wires
- 3D printed LED Pole

**Introduction:**

In this lab, you will build a circuit that changes the color of an LED light strip based on the amplitude of the sound in the room. The logic behind this is the conversion between analog and digital values. The other thing is the digital output that governs the on and off of the LEDs on the light strip. Lastly, the IR sensor also decides the performance by sending a digital signal to tell whether the entire system will be activated.

**Lab procedure:**

*Tips for working on this project:*

- Make sure the sound sensors, the IR sensor, and the LED strip are connected to the assigned pins declared in the code.
- Keep everything as organized as possible; stick to color-coding.
- Refer to the pictures in the google drive folder for additional clarity.

(The wiring of this lab can be confusing, so recommended wiring colors and practices are included in the following instructions. These recommendations match the figures. However, the color of the wires does not impact the functionality of the circuit.)

*Building the Circuit*

1. *Connect your Arduino as shown in **Figure 1** below.*
    1.1. Connect the 5V pin to the red side of the rail and the ground (GND) pin to the blue side of the rail.

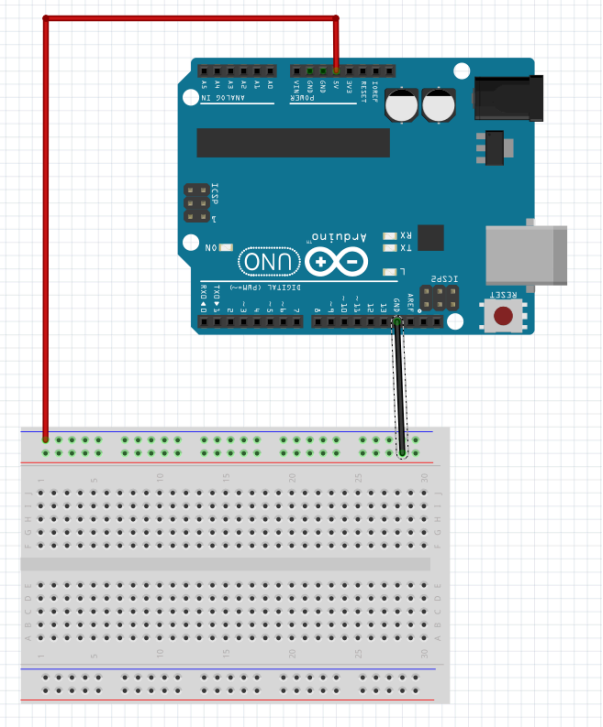

**Figure 1: Arduino power and ground connection diagram**

2. *Next, connect the LED strip terminals to power and to Pin 6 as shown in **Figure 2**.*
    2.1. Connect the red wire (Power) from the LED strip to the red 5V rail of the breadboard that is connected to the Arduino's 5V pin.
    2.2. Connect the separated black wire (Ground) from the LED strip to the GND rail of the breadboard that is connected to the Arduino's 5V pin.
    2.3. Connect the white wire (Data) from the LED strip to pin 6 of the Arduino.
    2.4. You will likely need to use the alligator clips to connect these wires. An example can be found in Figure 3 below. The other end of the alligator clips should be attached to the wires that connect to the 5V rail, GND rail, and pin 6 of the Arduino.
        2.4.1. The additional black wire can be ignored.
        2.4.2. Only use the side of the LED strip that has four separate wires. Make sure that the white arrow on the LED strip is pointing ***away*** from the wires and towards the rest of the LED strip.

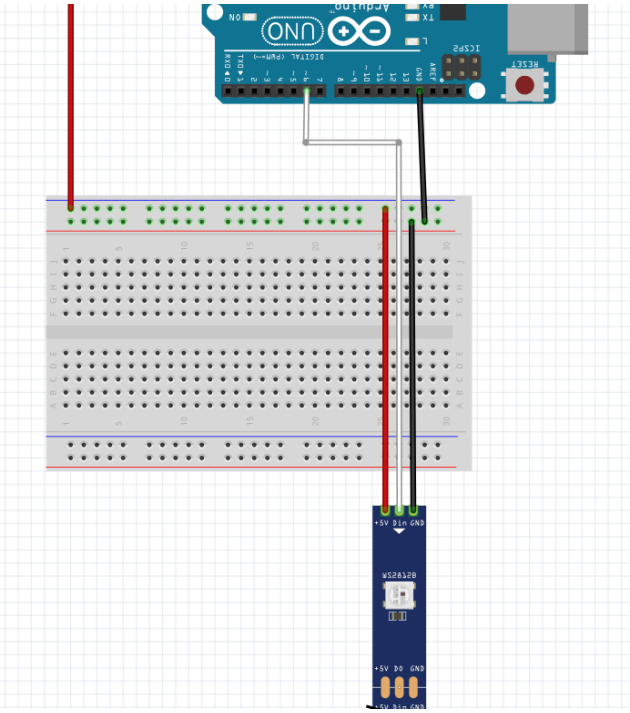

**Figure 2: LED Strip connection diagram**

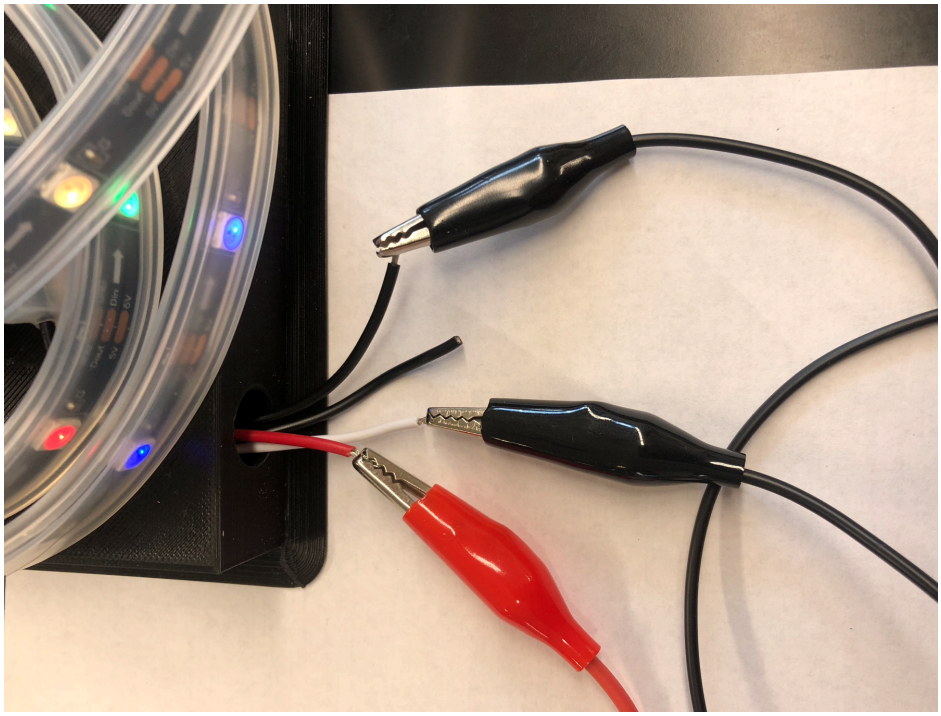

**Figure 3: Hookup for LED Strip**

3. *Connect the terminals of the IR sensor to the Arduino and breadboard as shown in* ***Figure 4*** *below.*
    3.1. Connect the **out** pin of the IR sensor to **Pin 7** of the Arduino. The out pin is the pin furthest to the right on the IR sensor in the figure below (green wire).
    3.2. Connect the ground (GND) pin of the IR sensor to the ground (blue) rail on the breadboard that is connected to the GND pin of the Arduino (black wire).
    3.3. Connect the Vcc pin of the IR sensor to the positive (red) rail on the breadboard that is connected to the 5V pin of the Arduino (white wire).

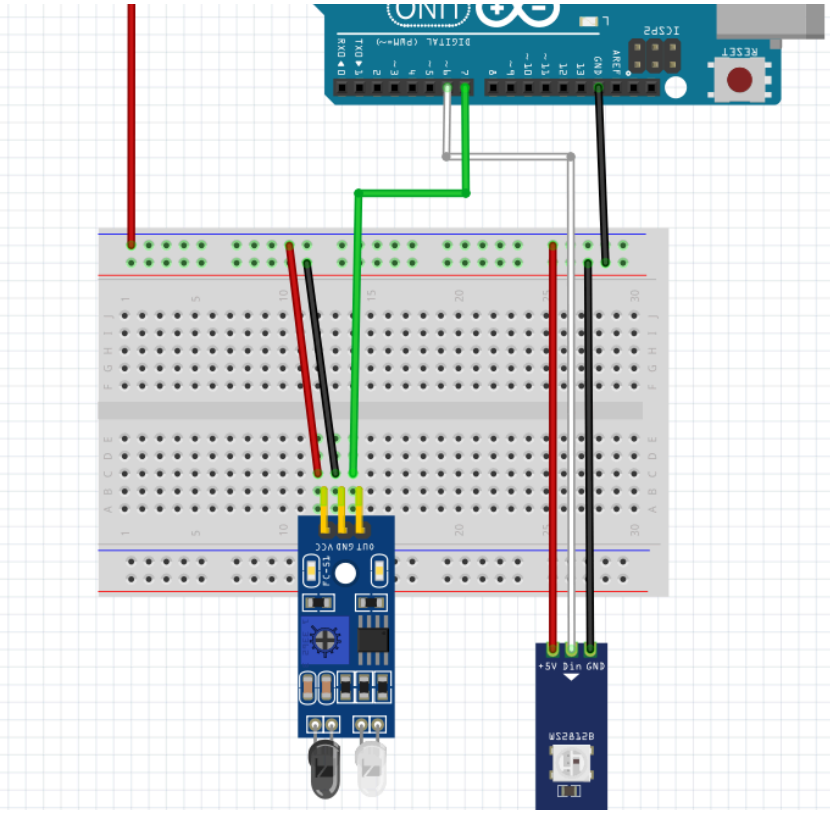


**Figure 4 Connect the IR sensor to the breadboard**

4. *Connect the sound sensor's Envelope terminal to the Arduino and power the sensor as shown in **Figure 5** below.*
    - 4.1. Connect the **Envelope pin** of the sound sensor to the **A0 pin** of the Arduino using a blue wire, as shown in the figure below.
    - 4.2. Connect the ground (GND) pin of the sound sensor to the ground (blue) rail on the breadboard that is connected to the GND pin of the Arduino (black wire).
    - 4.3. Connect the Vcc pin of the sound sensor to the positive (red) rail on the breadboard that is connected to the 5V pin of the Arduino (white wire).
    - 4.4. Leave the Audio and Gate pins **unconnected**, they are not needed for this project.

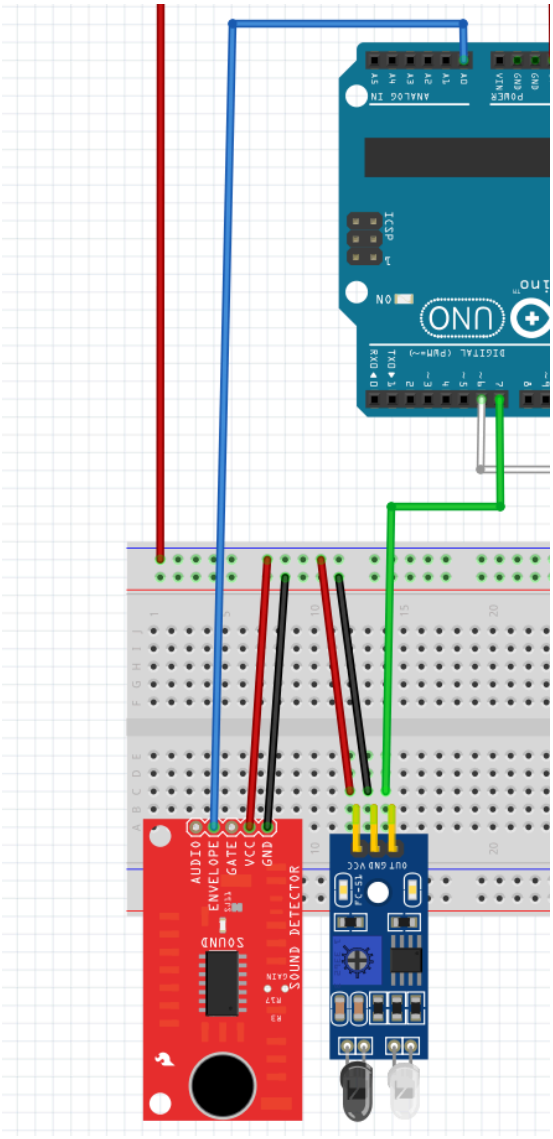


**Figure 5 Connect the sound sensor to the breadboard**

*Congrats! You have finished wiring the circuit. Your final circuit should resemble* ***Figure 6*** *below. (The speaker is not part of the circuit, it represents the functionality of the sound sensor and the suggested proximity of a speaker when using the device.)*

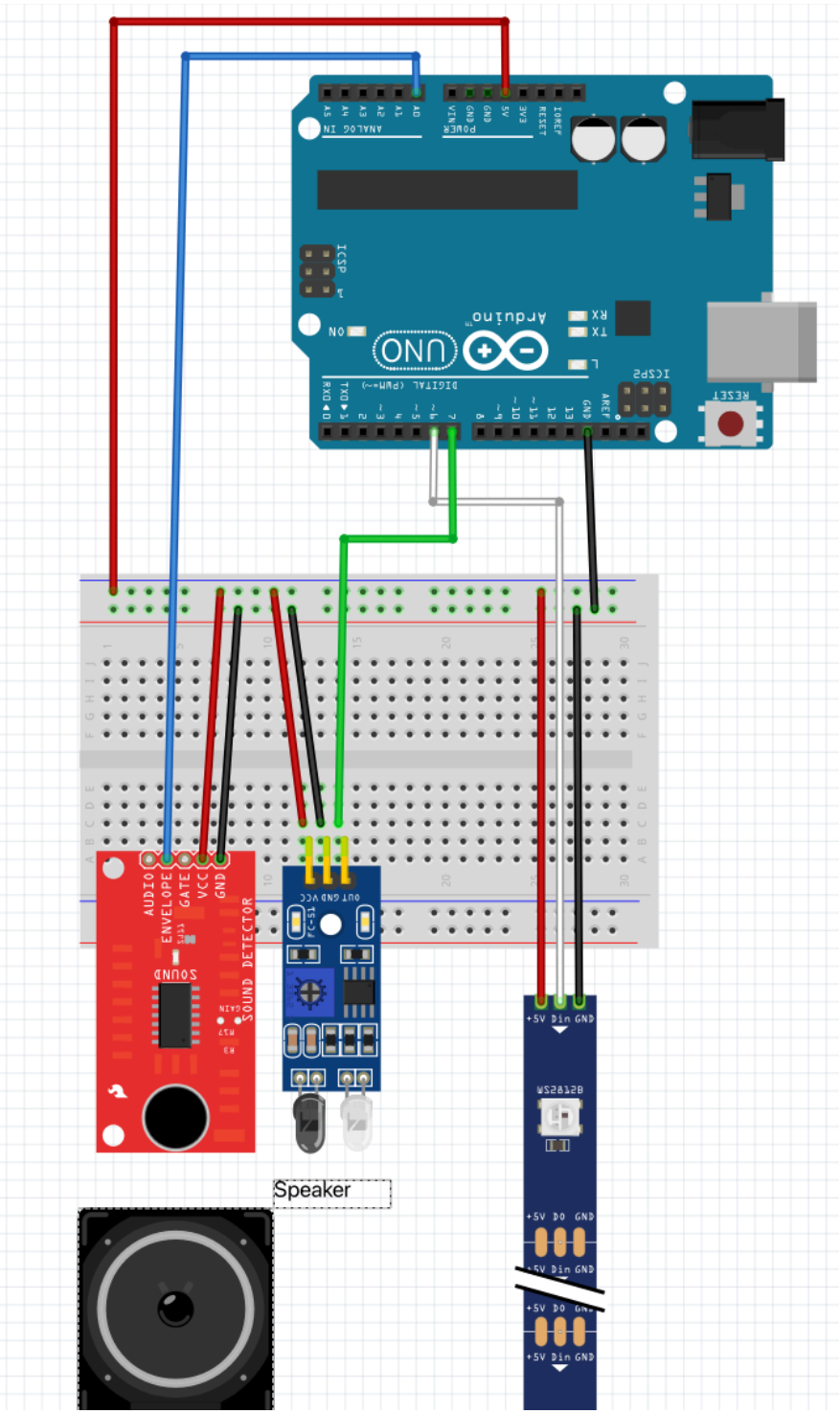


**Figure 6: Final schematic of the circuit**

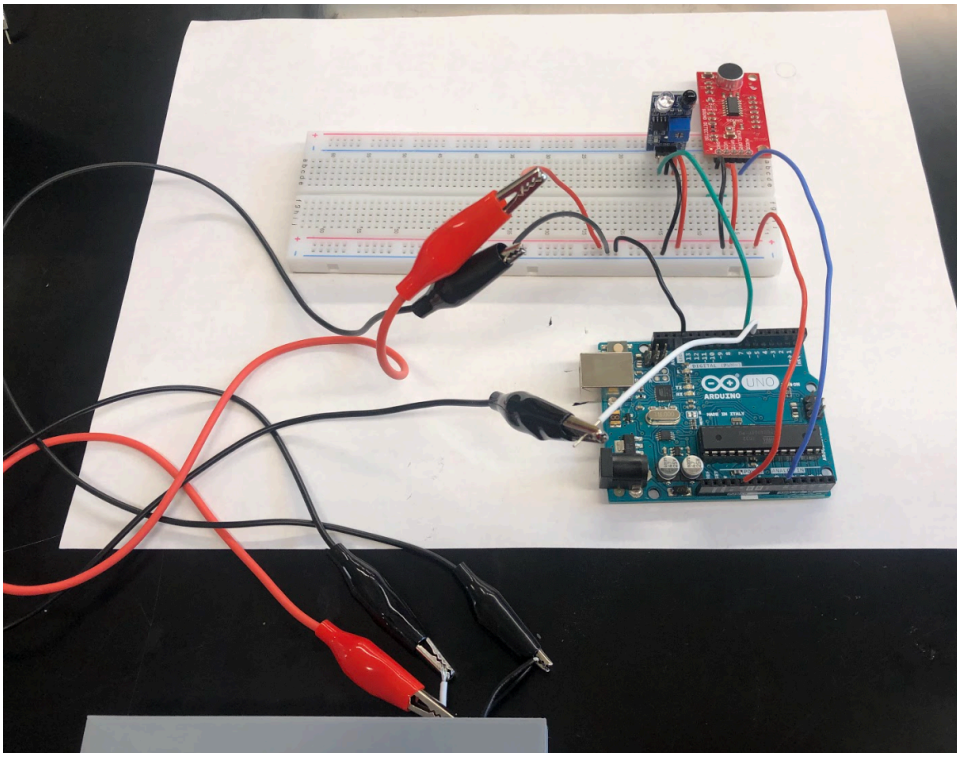

**Figure 7: Image of final circuit**

*The next step is to download and read over the Arduino code for the circuit.*

5. *Code implementation*

The code can be found in the following Google Drive folder which is named "Final_product.ino". It is also included in the Appendix A of this document.

On the Arduino program, go to your files and locate the .ino code to upload it on the IDE (File, Open, Downloads, prototype.ino). To use the correct port, go to Tools, Port, and select "Arduino Uno." Next, you must install the FastLED library (Tools, Manage Libraries, Search "FastLED", install the latest version). FastLED is a fast, efficient, easy-to-use Arduino library for programming addressable LED strips and pixels as shown in **Figure 8.**

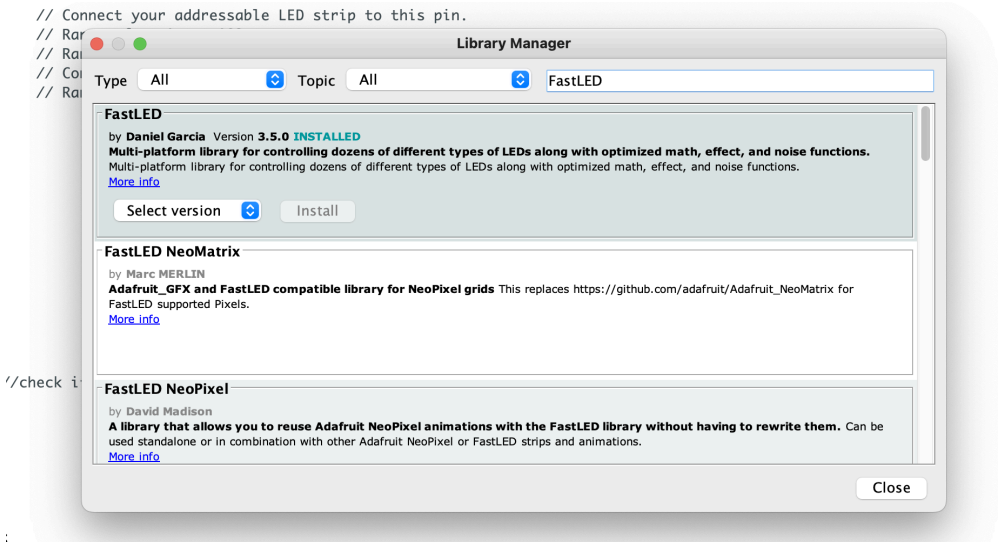


**Figure 8: Demonstration for downloading the library for FastLED**

After plugging in your Arduino, upload the code. (See Appendix A)

You should see the small LEDs on the Arduino, sound sensor, and IR sensor light up. The LED light strip should also light up.

In the code, you will see various types of read and write functions. The first one is analogRead for the sound sensor, which is connected to the output pin to the A0 pin on the Arduino shown in **Figure 9**. This function serves to return the voltage across the sound sensor between 0-5V in a scale of 10 bits. The analog input is a 10-bit analog-to-digital converter. The "10-bit" portion lets you know what your resolution is. The equation is 2^n where "n" is the number of slots you have. 2^10 = 1024. Basically you have to fit the analog signal into a discreet digital value. Think about it like the pixel resolution on a digital camera. Arduino reads from 0 to 5V so you spread the 1024 "slots" over 5V giving you 5V / 1024 = .0048V. So you can read a voltage change of about 5mV. ADCs can vary greatly between microcontrollers. The ADC on the Arduino is a 10-bit ADC meaning it can detect 1,024 (2^10) discrete analog levels.

The other interesting topic in the code is the HSV filtering of different colors. To allow changes in the color, another color filter apart from the traditional red, green, and blue filter (known as RGB), the hue, saturation, and value filter is used as shown in **Figure 10**. When at different LEDs on the strip, the hue value is always added by a constant pre-defined by the program. By changing the hue value in the code, we are changing the color while keeping its saturation and value the same. Since one color will have multiple hue values, the LEDs will eventually have repetitive colors after displaying a couple of different colors. You can see this when you are playing around with the device!

```
analogVal = analogRead(ENVELOPE_PIN);
```

**Figure 9: Section of code that defines the pin mode for the sound and IR sensor input type**

```
for(int i = 0; i <= NUMLEDS; i++){
  //the individual LEDs are showing colors with the same saturation and value in the HSV filter, except for hue, which determines the color
  leds[i] = CHSV(hue+= HUE_CHANGE, SATURATION, val);
```

**Figure 10: Section of code that uses the HSV filter function for the individual LED by changing the hue value only**

6. *Wire the LED strip up on the LED holder*

   You are provided with an LED holder in your bag with the other equipment. You can see that there is an opening at the bottom of the holder. That is where you should put the wires into. Remember that two different ends are available on the LED strip. Put the end with four wires (not the end with the black clip) into the opening. You can see a slot in the cylindrical shape. That is where you can put the other end so that the LED strip can stay on the holder without slipping off. The final result of putting the LED strip on the holder is shown in **Figure 11.**

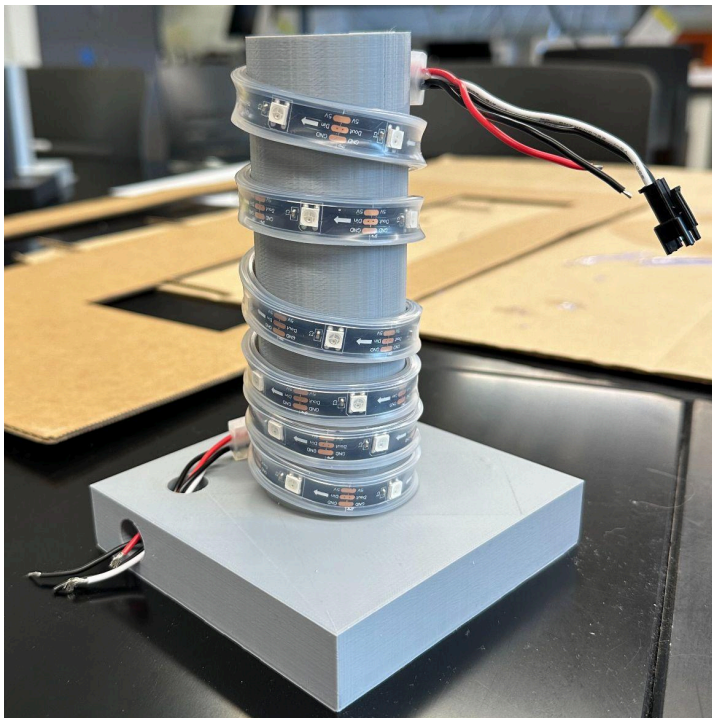

**Figure 11: demonstration of the wiring of the LED strip on the 3D-printed holder**

**Testing the prototype:**

By now, you should have completed building the circuit and uploading the code. To test the subcomponents, do the following steps.

1. Check the IR sensor by putting your finger on top of the black cap. Both the LEDs on the IR sensor as shown in **Figure 12** board should light up when you touch it and it should turn off when your finger is off.
    a. If this is not the case, refer back to the wiring procedure for the IR sensor.
2. To check if the sound sensor is working, make loud sounds near the sensor. You should see the red LED light on the sensor board turn on as it detects sound as shown in **Figure 13**
    a. If this is not the case, refer back to the wiring process for the sound sensor.
3. If the LED strip is not turning on, you should either:
    a. Refer back to the wiring procedure
    b. Make sure the power wire is connected properly to the location of the 5V voltage provided by the Arduino Uno
    c. Look at the Arduino program and make sure you're using the right port.

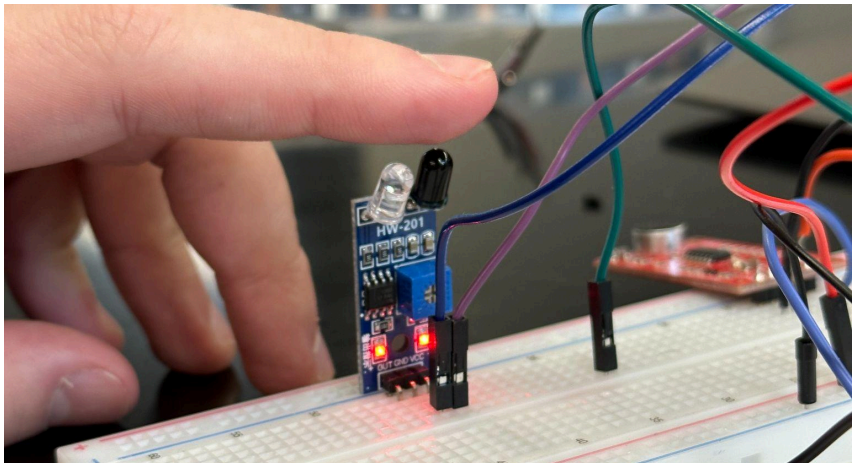


**Figure 12: Desired LED display when the finger is present above the IR sensor**

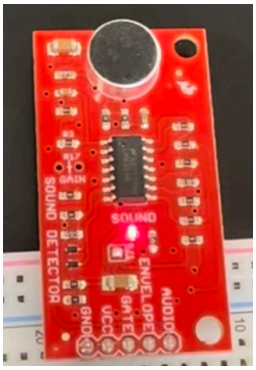


**Figure 13: Red LED on the sound sensor board blinking when sound is detected**

If everything is working correctly, congratulations! You have built a working model for a sound-sensitive LED strip that can blink with different colors! You may also change the delay time in the code at the very end to make it blink as fast as you wish.

**Possible errors and debugging steps:**

- LED strip not being lit: check if the positive and the negative leads of the LED strip are correctly connected to the breadboard: refer back to Lab Procedure: Building the Circuit section.
- Arduino code uploading error: “An error occurred while uploading the code.”: If this error message is shown on your Arduino screen, go to “Tools” in **Figure 14**, and check if the “Port” is selecting the correct port for the Arduino. The port we will be using should contain the name “Arduino” as shown in **Figure 15**.
- Arduino code uploading error: “No such file or directory”. If this error message occurs in your serial monitor as **Figure 16**, you may have forgotten to download the library for the FastLED library that is required to run the program. Refer back to Lab procedure: Code implementation.

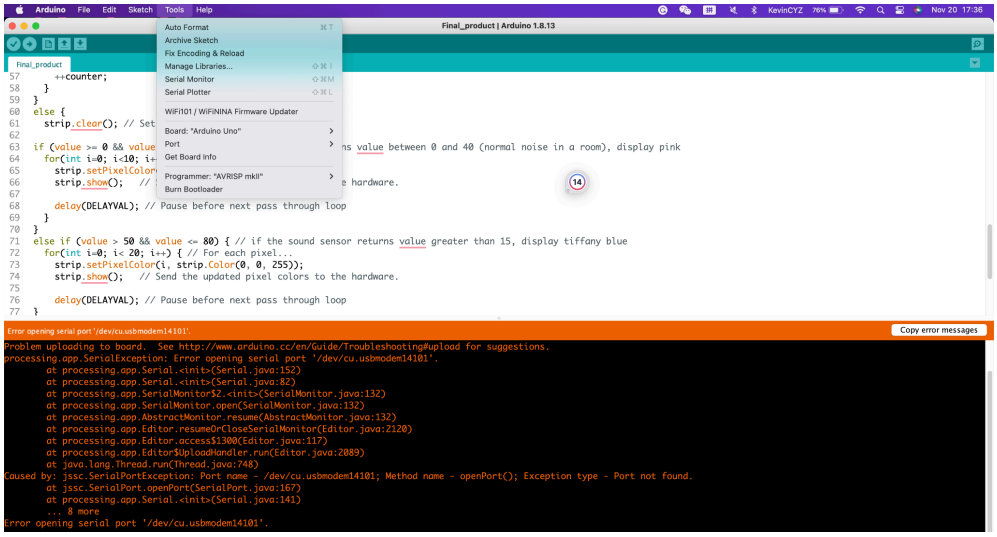


**Figure 14: Debugging process for code uploading error. You can find the tool using the toolbar on the top**

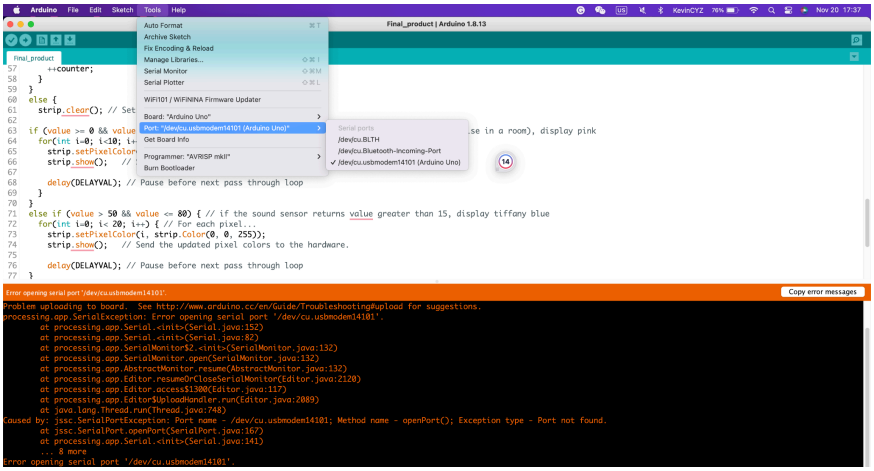


**Figure 15: Second step of debugging code uploading error**


**Figure 16: Error message of “No such file or directory”**

**References:**

https://cults3d.com/en/3d-model/home/vornoi-triangles-lamp-with-2-switchable-shades

https://www.reddit.com/r/3Dprinting/comments/s66sqy/fully_3d_printed_universal_spool_holder_no/

https://github.com/Rupakpoddar/Sound-reactive-LED-strip/blob/master/SoundReactiveLed/SoundReactiveLed.ino