DIY Ugly Christmas Sweater

School of Engineering and Applied Sciences
Harvard University

Cambridge MA.

Description: Do-It-Yourself Ugly Christmas Sweater, a sweater made up of sewn LEDs allowing users to create their own light design through image processing of digital images

By: Mia Wright, David Garcia, Nafanua Fitisemanu, Kade Kelsch

Class: Engineering Sciences 50
Instructor: Christopher Lombardo

May 2022

ABSTRACT

We wanted to create the ultimate piece of apparel that could be fashionable and help you stand out at any party. While clothing with LEDs is not a new concept, we wanted to take it a step further. Most products out there have a single image or a select few. We wanted to expand this concept by making a sweater with LEDs that can display any image the user desires so the sweater can be worn at any party and make the user the center of attention. We did this by using MATLAB for image processing. Through MATLAB, we took an image the user uploads then resized and pixelated it. We then took each pixel's color values, rounded them to a predetermined color palette, and sent this information to an Arduino. The Arduino then sent this information to our programmable LEDs on the sweater and displayed the image.

INTRODUCTION

As stated in the Abstract, the overall goal of the project was to use the engineering skills we developed in ES50 to create a cool sweater that could display any image. Our goal was to make this a simple process so that anyone could pick up the sweater, put it on and upload any image their heart desired. Additionally, we wanted to make it an easy process so that no knowledge is required to use the sweater. We chose this project because we thought it would be fun to implement the idea and see the final result. For most electrical engineers, the first thing they ever do in the lab is light up an LED. We wanted to take this classic project and expand it because LEDs are satisfying to use. Another prospect that attracted us to this project was the ability to improve our engineering skills. While LEDs are quite simple to use, we knew this project would help us learn about image processing, Arduino, and MATLAB. Overall, we wanted a final project that met three criteria:

1. a fun project to implement and show off
2. teach us something beyond the course so we could improve our engineering skills
3. demonstrate our engineering and problem-solving ability

DESIGN

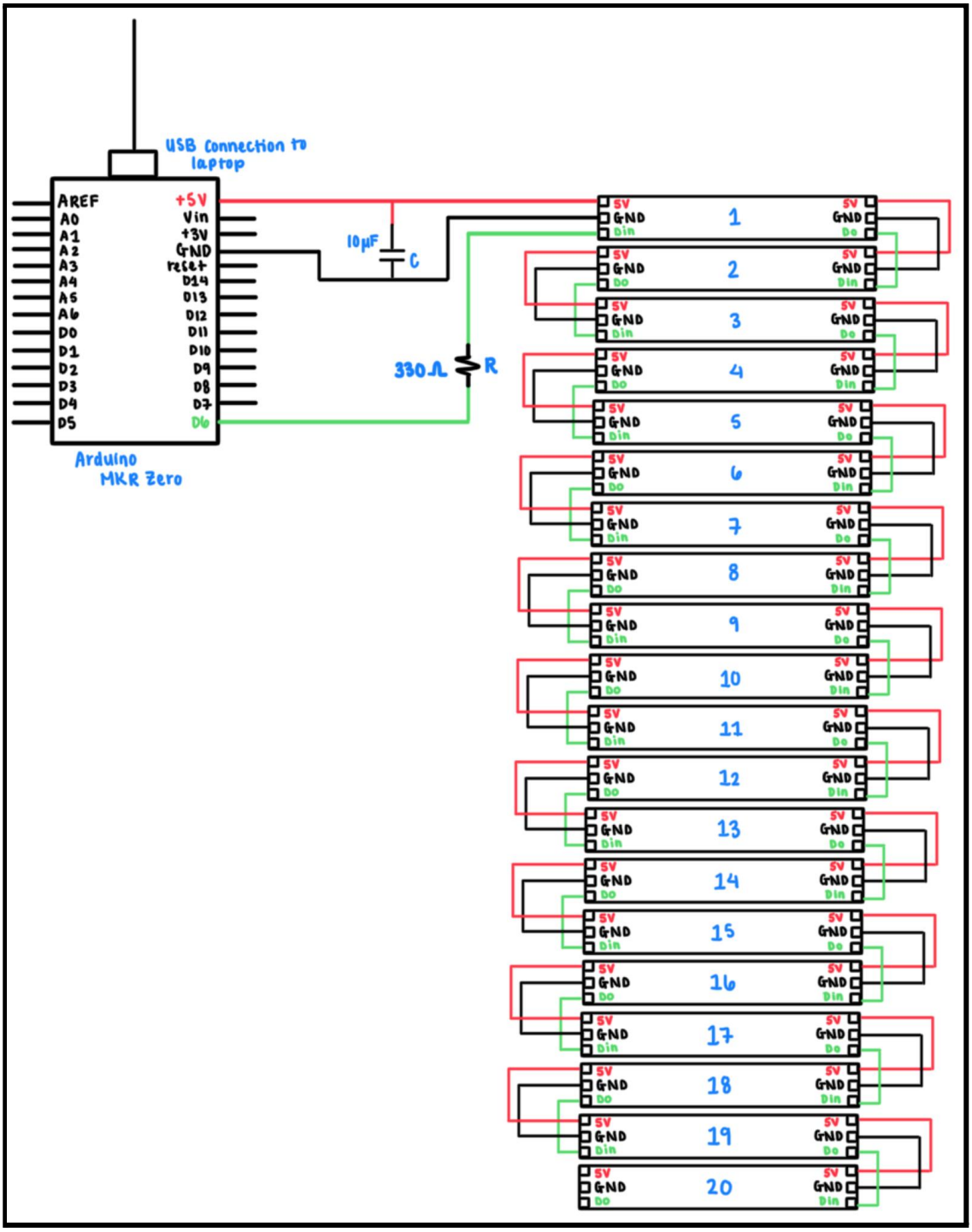


Figure 1: Overall schematic of the circuit

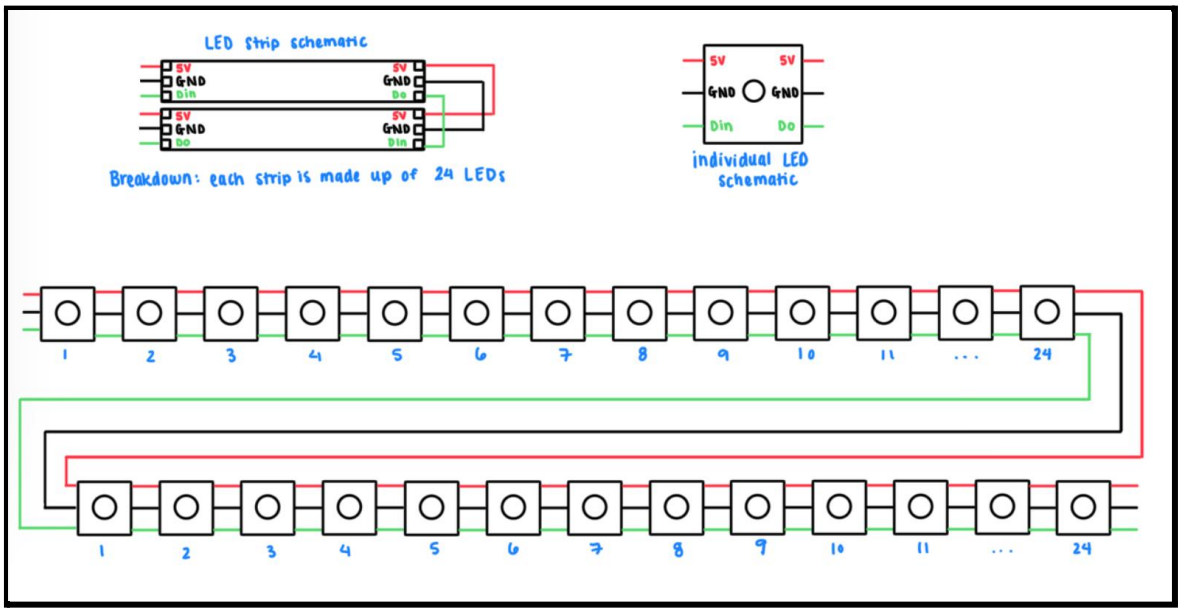


Figure 2: Schematic of a row of LED strips pictured in Figure 1

Circuit Schematic

Overall our circuit consists of 20 LED strips. Each strip has 24 LEDs which has our circuit at a total of 480 LEDs.

As seen in Figure 1, there is a USB connection between the Arduino and a laptop. This connection serves two important purposes

1. It allows the laptop to **power** the Arduino.

2. It allows the laptop to **transfer data** to the Arduino.

As seen in Figure 1, from the Arduino there is a connection between the 5-volt pin of the Arduino (labeled “+5V”) to the first LED's 5-volt pin (labeled “5V”). This allows the Arduino to send 5 volts of power to the LED which is then transferred from LED to LED, powering all 480 LEDs. The 5 volts coming from the Arduino is essentially coming from the connected laptop which means that the laptop is powering the circuit. We needed to use the 5 volt pin in comparison to the 3 volt pin because the LED strips purchased need a minimum of 5 volts to function.

The ground pin of the Arduino (labeled “GND”) is connected to the ground pin of the first LED (also labeled “GND”). Both the GND and 5V wires run through a 10 µF capacitor (labeled “C”). This capacitor allows the signal to be smoothed out to send a constant 5 volts to the LED. There is no resistor because its

insertion would cause the capacitor to act as a filter.

The Arduino Digital 6 pin (labeled “D6”) sends a voltage to the data in (labeled “Din”) of the first LED. This goes through a 330 ohm resistor (labeled “R”). This resistor limits the amount of current going through the LEDs.

As seen in Figure 1, there are 20 LED strips. Figure 2 shows a more in depth display of how the last LED on a strip connects to the first LED of the next strip. As seen in Figure 2, the last LED’s data out (labeled “Do”) connects to the next strip’s first LED’s data in (labeled “Din”). The same is true for both strips' 5-volt (labeled “5V”) and ground (labeled “GND”) connections. These connections allow the 5 volts of power, data from the Arduino, and common ground to run through all LED strips.

It is important to note, as seen in Figure 1, that the last LED's 5-volt (labeled “5V”), data out (labeled “Do”), and ground (labeled “GND”) do not have connections due to no longer needing to connect, power, or transfer data to more LEDs.

***Calculations of capacitor and resistor were found through help of References: “ES50 Lab 8” and “Guide to Using LED Strips with Arduino.”

Code

Prototyping through one strip of neopixels:

In order to get a sense of how to control the LEDs from the Arduino, we watched References: “Guide to Using LED Strips with Arduino.” This video helped to run through tests and provided libraries for our project such as the “FastLED” library to see the capabilities of the LEDs and get a start on what Arduino code may look like. This video helped us to understand how to light up the LEDs from the Arduino but not necessarily how to display an image essentially using the LEDs as its pixels.

Finding an extension to convert Matlab to Arduino:

Since we only knew how to do image processing through Matlab code, we wanted to find an extension of Matlab that could convert Matlab code to Arduino code. We found a potential extension, called “Arduino Explorer” through References: “Matlab to Arduino Extension.”

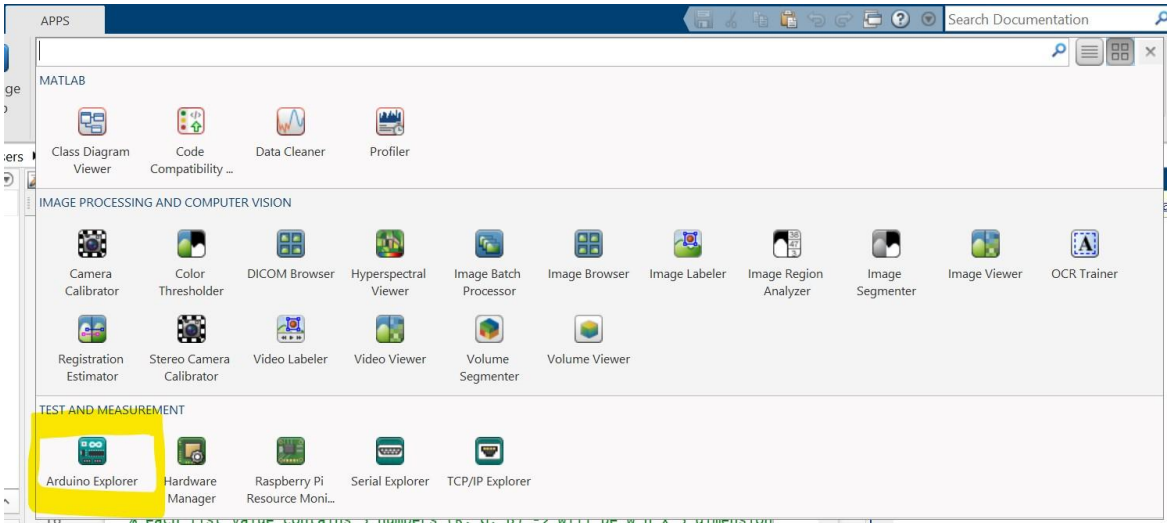


Figure 3: screenshot of installed package “Arduino Explorer” in attempt to convert Matlab code to Arduino

Unfortunately this extension did not work for our project as we were having trouble finding the right arduino port of our computer. In order to display images on our sweater through Arduino code, we had to create txt files of each image which included a list of 3 variables, RGB values ranging from 0 to 255, which represented the color of each LED through Matlab and then copy and paste that list into our Arduino code. We also had to install the “FastLED” Arduino library in order to control all the LEDs from our arduino.

***examples of txt files of RGB values can be found in Appendix: “txt example”

Colors:

As we tested the images on our sweater, we saw a variety of colors which made it difficult to identify what the image was. In order to combat this issue, we used color compression which allowed all similar colors to round to one uniform color, making it much easier to identify what the image was.

***code relating to color compression can be found in Appendix: “Color Compression”

Physical building of the sweater

Issues with soldering:

Since we were still new at soldering, we initially had difficulty making a solid connection between the wires and tabs that were at the end of the LED strips. Our initial technique was to put the wire on top of the tab and try to apply

solder to both at the same time but the connections would break. We then looked up soldering techniques on the internet and found that wetting the wire and the edge tab separately with solder and then applying solder again with the wire and tab together was the best technique. Afterwards we still had risk of losing those connections if the sweater were to be moved around a lot so with the help from one of the tfs, Kian, we applied hot glue to the connections which helped secure it.

***image of glued edge tabs can be found under Appendix: “Glued Edge Tab”

Appearance:

After we had our sweater up and running with the LEDs and arduino, we wanted the sweater to look more appealing so that potential users would want to wear it more often. Therefore, we crocheted black yarn in between each column of LEDs which created a cleaner finish and really emphasized the color of each LED. In the end, the images displayed on the sweater looked cleaner to the eye.

In the end, in order for the sweater to function, it still had to keep a connection with a power source, in this case the arduino and laptop. We wanted to create a holding for the arduino so that our user didn’t need to hold it while wearing the sweater so we crotched a side pocket that had an opening at the top and a smaller opening at the bottom so that the arduino wouldn’t fall out but an opening so that the wire that connected the arduino to the laptop could run through.

***link to time lapse of crocheting sweater can be found under Appendix: “Crochet Sweater”

***image of final sweater can be found under Appendix: “Final Sweater”

PARTS LIST

Label with included link	Company	Quantity	Cost
LED strips	Amazon	2	$63.98
Sweater	Amazon	1	$40.43
Black Yarn	Already Owned	1	$1.49
Arduino MKR Zero	Lab	1	$25.20
Solderable Breadboard	Lab	1	$2.95
USB 2.0 Cable	Lab	1	$3.28
10 microfarad capacitor	Lab	1	$0.77
330 ohm resistor	Lab	1	$0.15
Total:			**$138.25**

PROJECT IMPLEMENTATION

DIY Ugly Christmas Sweater

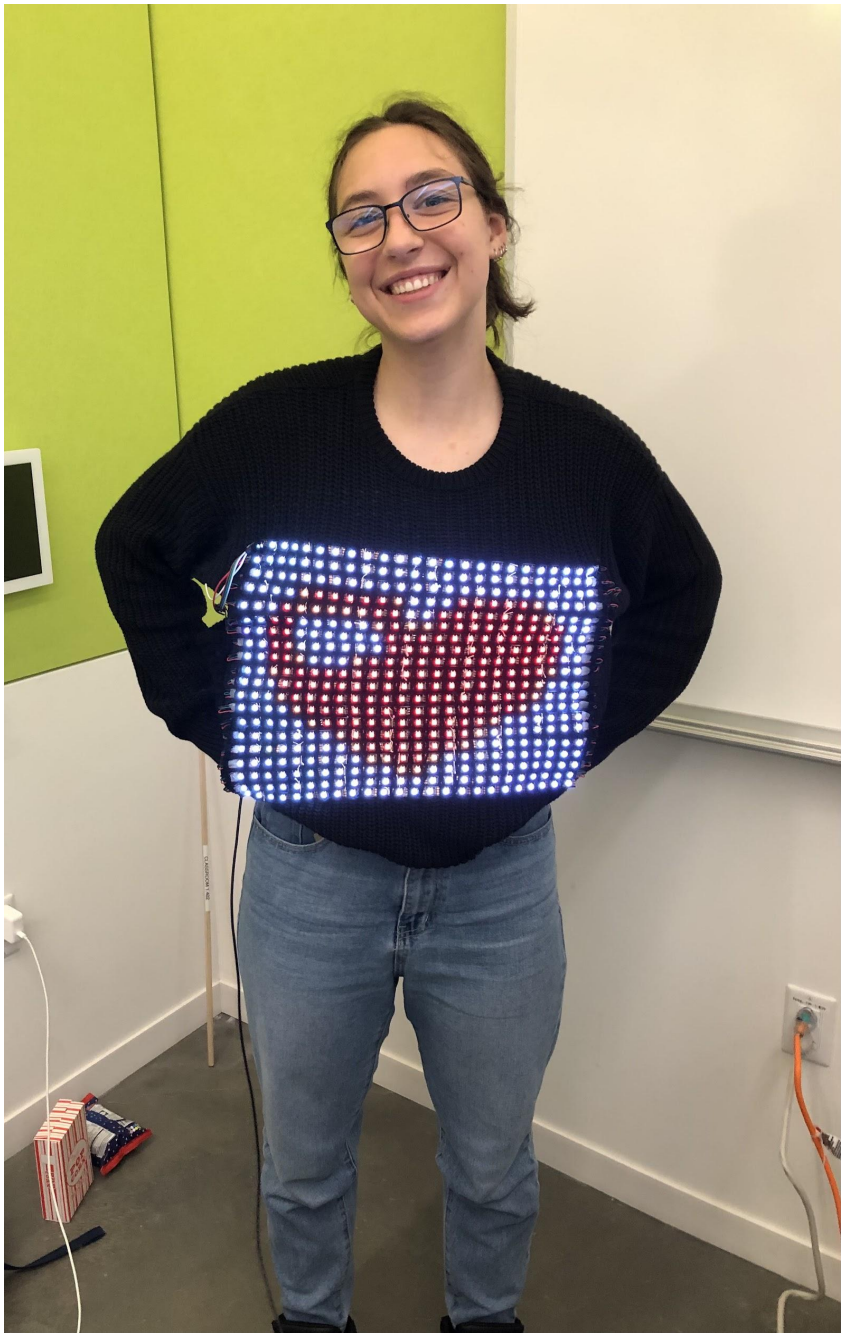
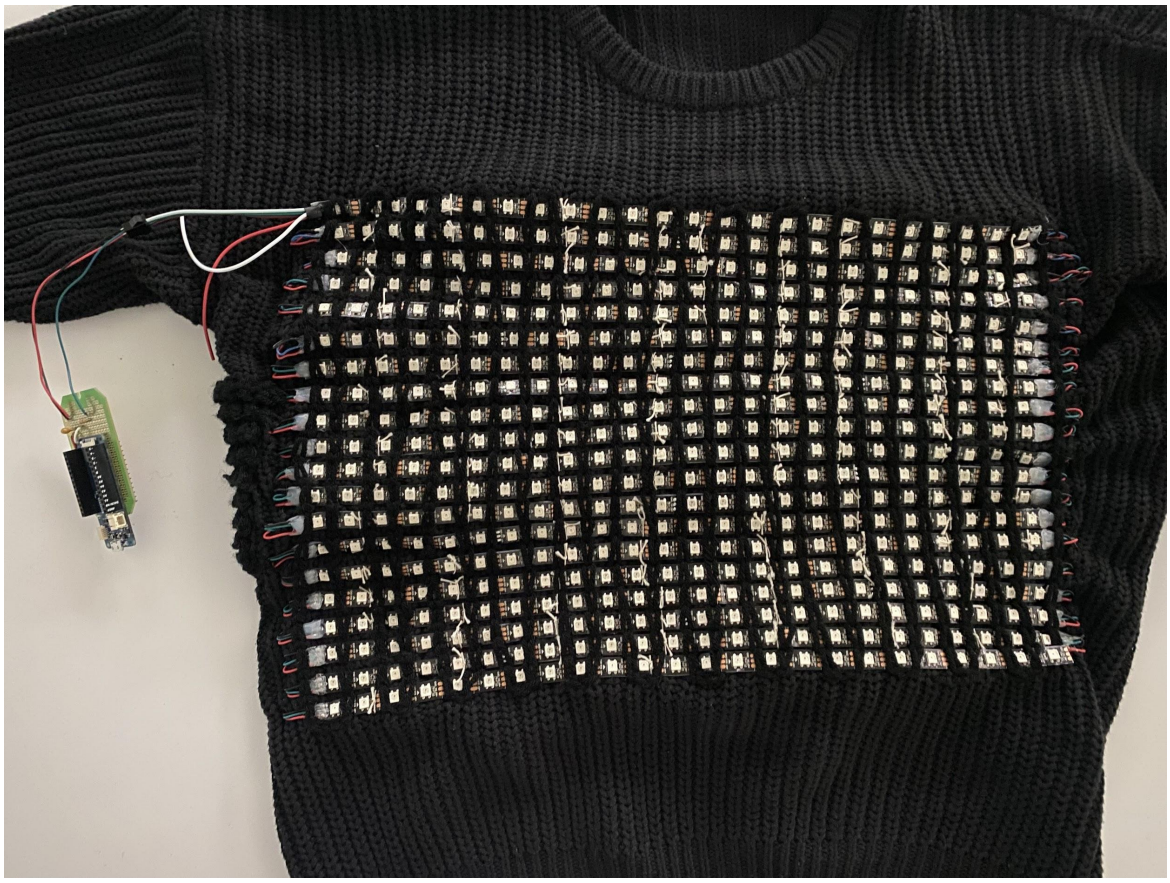


Figure 5: picture taken of a model (Kade Kelsch) rocking the sweater with a heart-image display

Have you been invited to an Ugly Christmas Sweater party and have nothing to wear? Well this product is perfect for you!

With the DI-Wirez DIY Ugly Christmas Sweater, you will automatically become the party at every event you attend.

Step 1: How It Works



The DIY Ugly Christmas Sweater consists of 20 strips of LEDs. Each strip contains 24 LEDs, adding up to a total of 480 LEDs. The LEDs are connected to an Arduino which is then connected to a computer as its power source. With help from Matlab, you will use image processing to get any digital image to a 20 (height) x 24 (width) grid (same grid of the sweater's LEDs). From there you will create txt files of those RGB values for each grid box which will then transfer to the Arduino and into the LEDs' displayed color which will then display the image on the sweater.

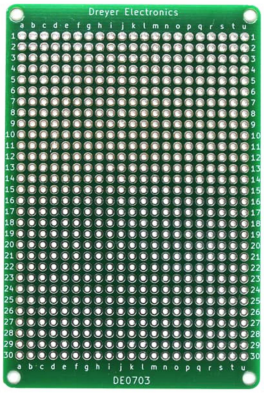
Step 2: Skills Required

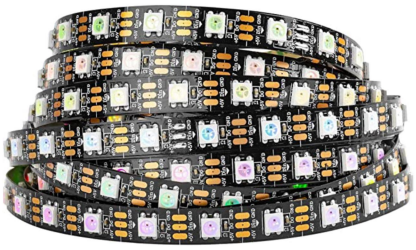
- Basic soldering
- (not required but helpful) Knowledge with Matlab and Arduino
- (optional) basic crocheting

Step 3: Tools

- Soldering machine
- Glue gun and glue sticks
- Wire (preferably 3 colors to represent 5V, GND, and D in/out)
- Wire cutter/strip
- Protective glasses
- White string (different than black yarn)
- Needle
- Scissors
- (optional) Crochet tools

Step 4: Materials



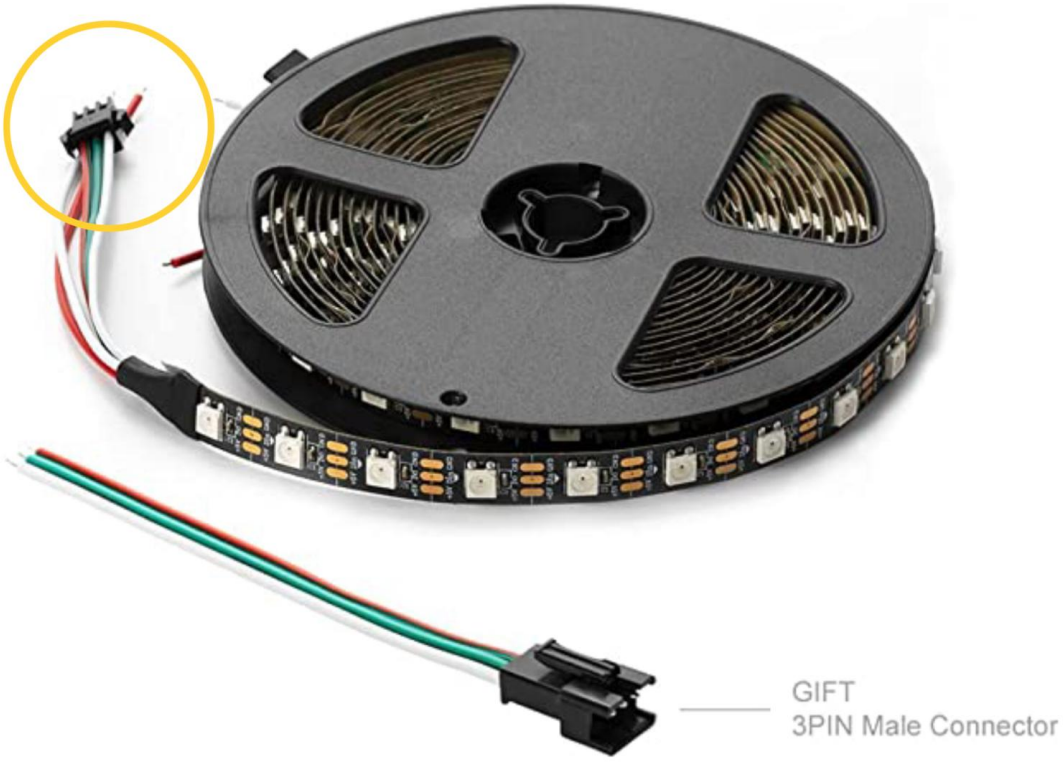



- (2) ~$63.98 LED strips
- (1) ~$40.43 Sweater
- (1) ~$1.49 Black Yarn
- (1) ~$25.20 Arduino MKR Zero
- (1) ~$2.95 Solderable Breadboard
- (1) ~$3.28 USB 2.0 Cable
- (1) ~$0.77 10 microfarad capacitor
- (1) ~$0.15 330 ohm resistor

Total cost should be around $138.25

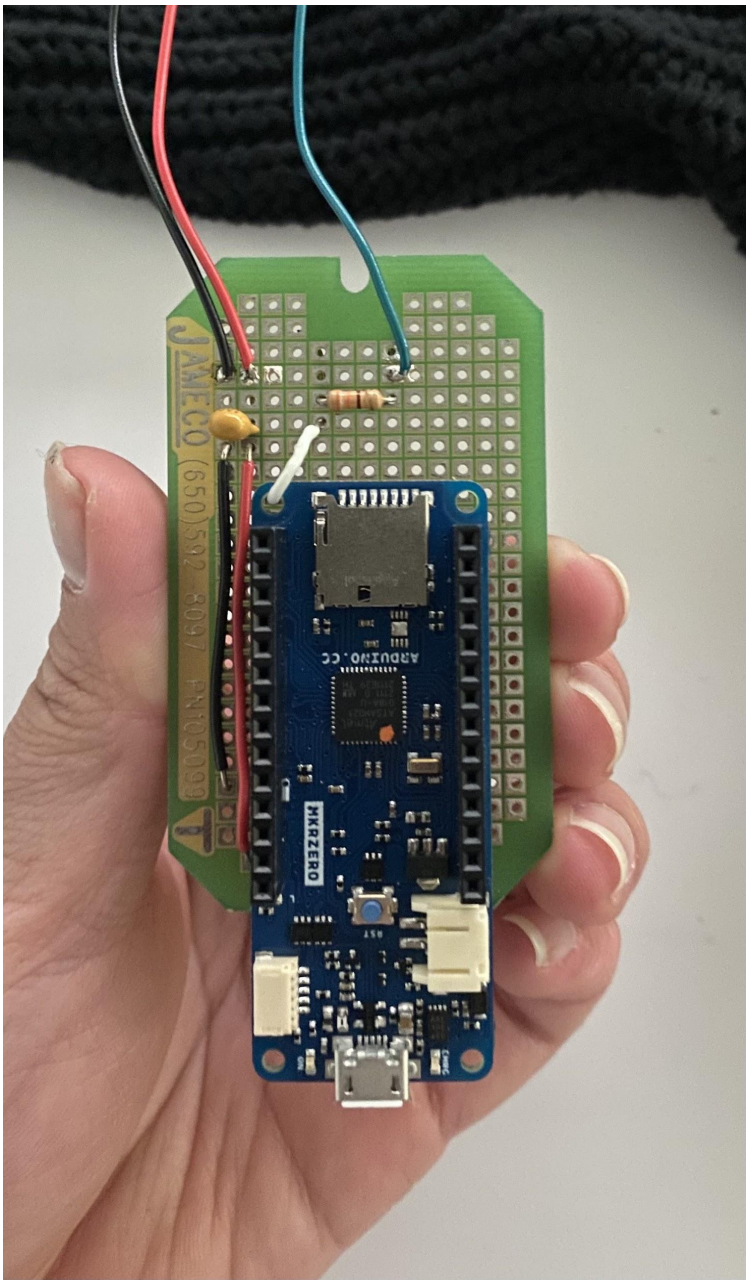
Step 5: Cutting LED Strips

1. **Start off by laying out all of the LED strips**
2. **Use scissors to cut 20 strips of 24 LEDs**
    a. Each strip should have 24 LEDs
    b. Make sure to keep the 3PIN Female Connector to one of these strips (shown below with yellow circle)
3. **Lay them out on sweater**



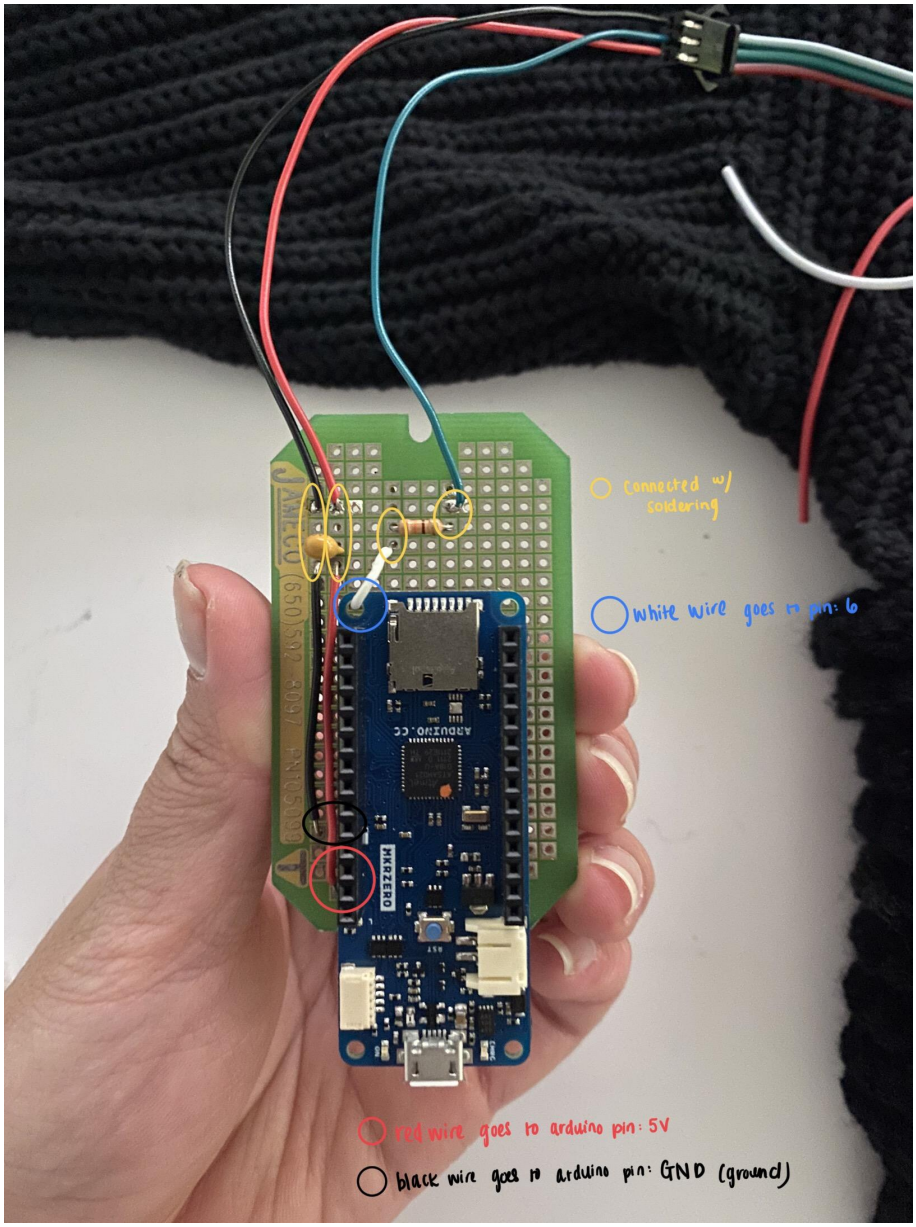
Step 6: Arduino Build

1. **Take out your Arduino and attach it to the solderable breadboard by simply putting the spikes of the Arduino into the holes of the breadboard**
    a. We recommend placing the Arduino in the middle of your breadboard and aligning the last pins of the Arduino with the last row of breadboard holes as shown below



Step 7: Soldering within Arduino

Refer to image for guidance



1. **Find the Arduino that is connected to the solderable breadboard as shown in Step 6**

2. **Wire, Capacitor, and Resistor connections within Arduino**

Keep in mind that the wires with the Arduino are never connected directly to the pins of the Arduino. The wires are instead stripped and then pulled through the solderable breadboard then solder is applied to both the wire and another wire that is coming from Arduino pins.

3. **Focus on the connection made by the red circle**
    a. This connection is of the Arduino's pin: 5V and the red wire
    b. Measure how long the red wire needs to be in order to connect to the Arduino's pin: 5V and to the top of the Arduino
        i. Before cutting that length, make sure to add some slack so that there's enough to strip the wire which will be pulled through the solderable breadboard
    c. Strip the red wire then pull the stripped wire through the holes of the solderable breadboard.
        i. One end through the breadboard hole to the side of the Arduino's pin: 5V
        ii. The other end through the breadboard hole that is a row above the Arduino
        iii. We recommend placing the ends of the red wire in the same column
    d. Apply solder to the end of the red wire that is on the side of pin: 5V and to another stripped wire that is pulled through pin: 5V.

4. **Focus on the connection made by the black circle**
    a. This connection is of the Arduino's pin: GND and the black wire
    b. Measure how long the black wire needs to be in order to connect to the Arduino's pin: GND and to the top of the Arduino, in the same row as the red wire.
        i. Make sure to also add slack so that there's enough to strip the wire which will be pulled through the solderable breadboard
    c. Strip the black wire then pull the stripped wire through the holes of the solderable breadboard
        i. One end through the breadboard hole to the side of the Arduino's pin: GND
        ii. The other end through the breadboard hole that is a row above the Arduino
        iii. We recommend placing the ends of the black wire in the same column

d. Apply solder to the end of the black wire that is on the side of pin: GND and to another stripped wire that is pulled through pin: GND

5. **Focus on the placement of the capacitor**
    a. Place the capacitor in the row above the ends of the red and black wire
        i. Place one end of the capacitor in the same column as the red wire then apply solder to them together
        ii. Place the other end in the same column as the black wire then apply solder to them together
        iii. Note: Don't solder the red and black wires together. There should be two different solderings.

6. **Focus on the placement of the resistor**
    a. There is leeway on where you can insert the resistor
    b. Make sure to put it at least 4 rows above the edge of the Arduino
    c. We recommend placing it in the center of the solderable breadboard
    d. Insert the ends of the resistor in two different holes but make sure they are in the same row

7. **Focus on the connection made by the blue circle**
    a. This connection is of the Arduino's pin: D6 and the white wire
        i. The white wire is actually be pulled through the hole of the Arduino's corner, not the hole right beside pin: D6
    b. Measure how long the white wire needs to be in order to go through the Arduino's corner and the same column of the left leg of the resistor but in the row below
        i. Make sure to also add slack so that there's enough to strip the wire which will be pulled through the solderable breadboard
    c. Strip the white wire then pull the stripped wire through the holes of the solderable breadboard
        i. One end through the breadboard hole that aligns with the hole of the Arduino's corner
        ii. The other end through the breadboard hole that is the same column as the left leg of the resistor but a row below
    d. Apply solder to the end of the white wire that is not by the resistor and to another stripped wire that is pulled through pin: D6

Step 8: Soldering between Arduino and LED Strip

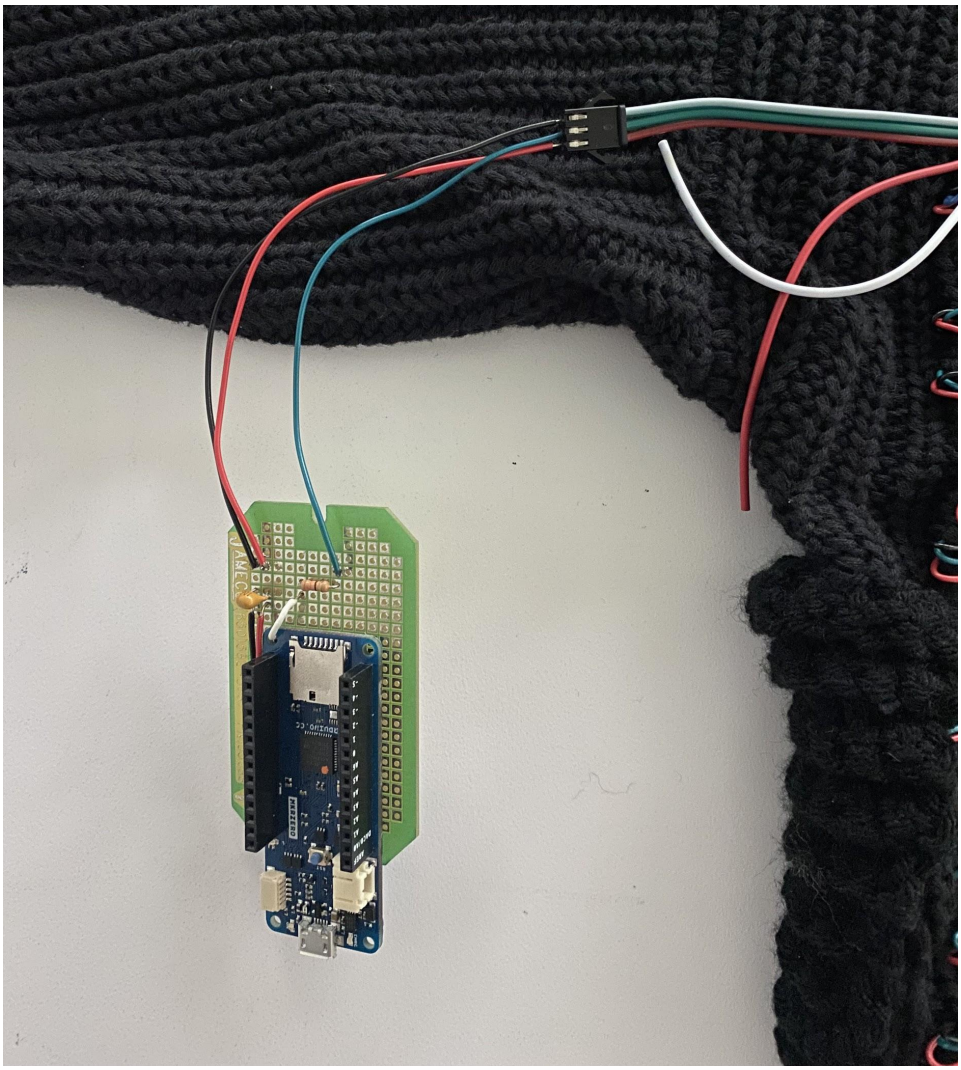
1. **Locate**
    a. Arduino that is connected to the solderable breadboard shown in Step 7
    b. LED strip that is connected to the 3PIN Male Connector shown in Step 5

    Refer to the image below for guidance

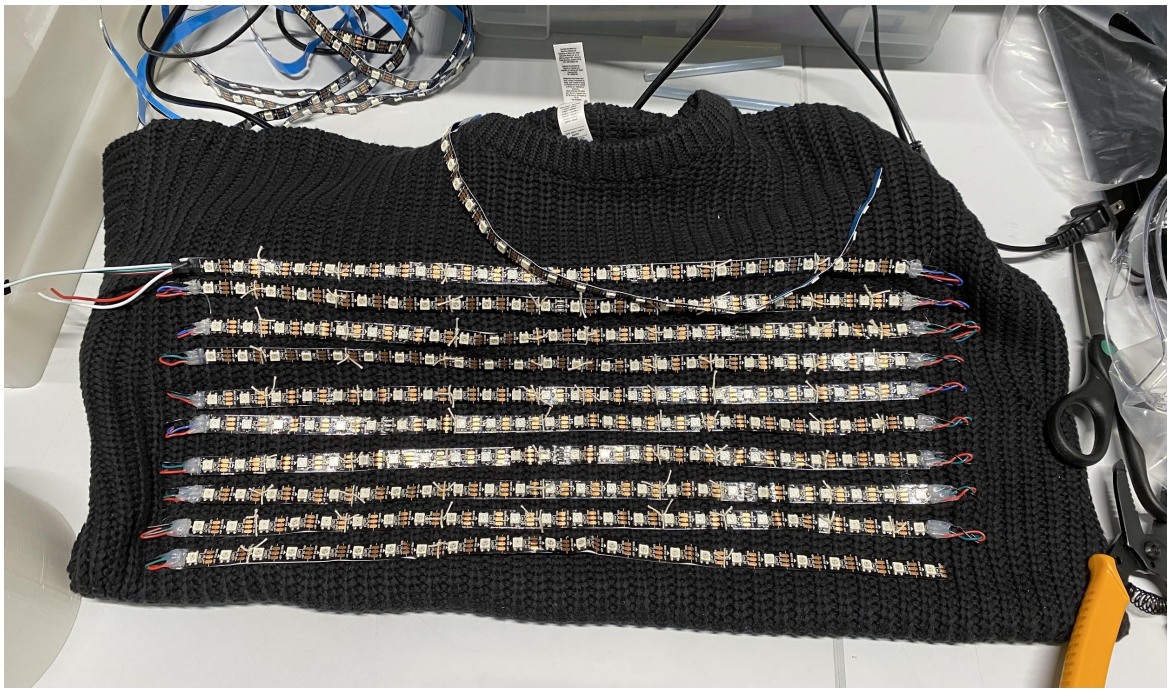
2. **Connecting red wire to breadboard and 3PIN Male Connector**
    a. Cut 7 inches of red wire and strip its ends
    b. Pull one end through the breadboard hole that is in the same column as the other red wire and one of the capacitor legs
        i. Make sure to insert it in a row above the capacitor
    c. Apply solder to this end of the red wire and also the previous connection between the capacitor and other red wire together
        i. This solder connection makes sure that there is a conductive connection between all the red wires on the breadboard, capacitor, and pin: 5V.
    d. Now connect the other end of the red wire to the red wire of the 3PIN Male Connector shown in image below

3. **Connecting black wire to breadboard and 3PIN Male Connector**
    a. Cut 7 inches of black wire and strip its ends
    b. Pull one end through the breadboard hole that is in the same column as the other black wire and one of the capacitor legs
        i. Make sure to insert it in a row above the capacitor
    c. Apply solder to this end of the black wire and also the previous connection between the capacitor and other black wire together
        i. This solder connection makes sure that there is a conductive connection between all the black wires on the breadboard, capacitor, and pin: GND.
    d. Now connect the other end of the black wire to the white wire of the 3PIN Male Connector shown in image below

4. **Connecting blue wire to breadboard and 3PIN Male Connector**
    a. Cut 7 inches of blue wire and strip its ends
    b. Pull one end through the breadboard hole that is in the same column as the other end of the resistor that the white wire is not in
        i. Make sure to insert it in a row above the resistor
    c. Apply solder to this end of the blue wire and the resistor leg that is in the same column
        i. This solder connection makes sure that there is a conductive connection between the blue wire, white wire on the Arduino, resistor, and pin:D6.
    d. Now connect the other end of the blue wire to the green/blue wire of the 3PIN Male Connector shown in image below



Step 9: Soldering between LED Strips



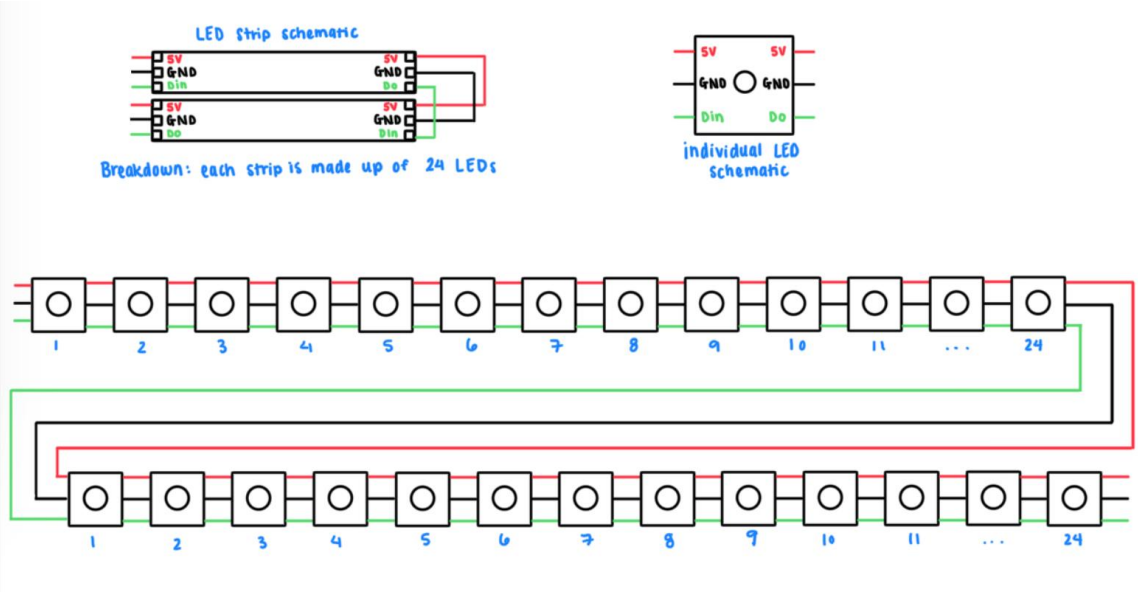
1. **Layout LED strips**
    a. Should have 20 LED strips
    b. Each strip containing 24 LEDs

2. **Place first LED strip on sweater**
    a. Should be the one that we just connected the Arduino to in Step 8

3. **Take black, green, and red wire then measure the length of one strip**
    a. Before cutting add 6 inches to the end which will provide slack for connections between different LED strips

4. **The LED strips have 3 tabs at the edge of them as shown below**
    a. Left-hand side
        i. First tab representing the 5V coming from the Arduino
        ii. Second tab representing the GND
        iii. Third tab representing the Din
    b. Right-hand side
        i. First tab representing the 5V coming from the Arduino
        ii. Second tab representing the GND

iii. Third tab representing the Dout



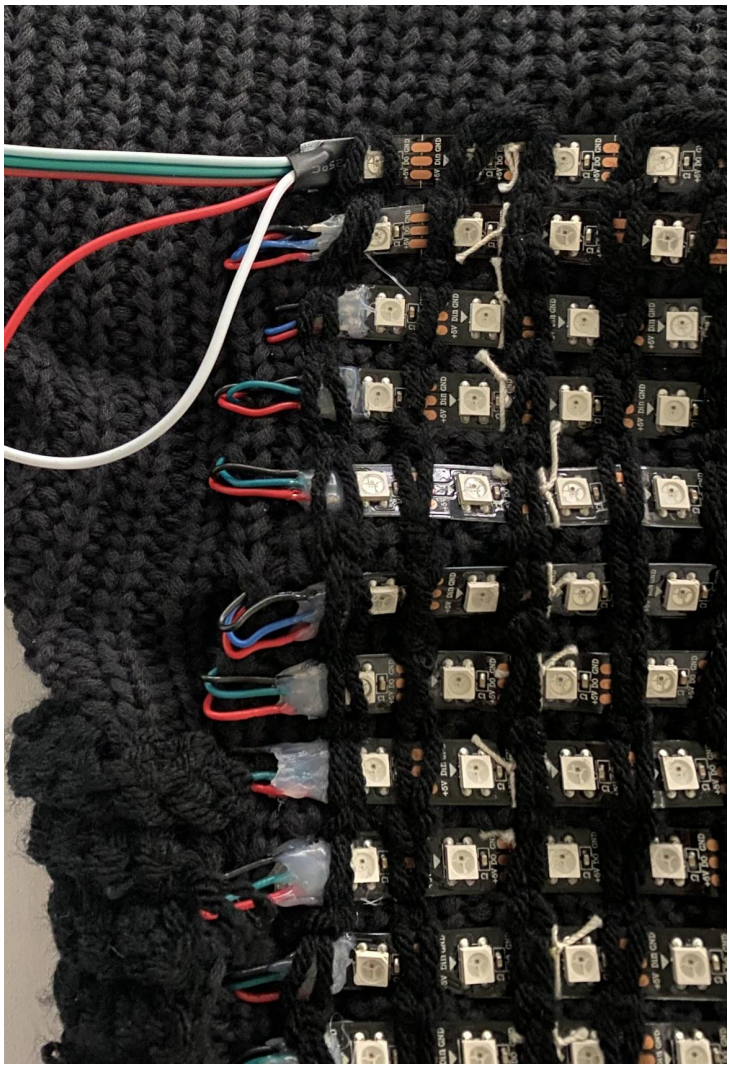
5. **Black Wire**
    a. Strip both ends
    b. Apply solder to one of the ends
    c. Then apply solder to the GND tab of the LED
        i. Should also align with the white wire coming out of the 3PIN Male Connector
    d. Then place black wire on top of GND tab
    e. Apply solder to both

6. **Green Wire**
    a. Strip both ends
    b. Apply solder to one of the ends
    c. Then apply solder to the Dout tab of the LED
        i. Should also align with the blue/green wire coming out of the 3PIN Male Connector
    d. Then place green wire on top of Dout tab
    e. Apply solder to both

7. **Red Wire**
    a. Strip both ends
    b. Apply solder to one of the ends
    c. Then apply solder to the 5V tab of the LED

        i. Should also align with the red wire coming out of the 3PIN Male Connector
    d. Then place red wire on top of 5V tab
    e. Apply solder to both

8. **Glue gun to soldering of tab as shown below**
    a. To secure these soldering connections apply hot glue to the tab as shown below



9. **Braiding of Black, Green, and Red wires**
    a. Once the glue dries, braid the long strand of black, green, and red wires
    b. Twist the edges of the wires and insert them inside the sweater

c. Let it run through the inside of the sweater and come back out to start a new row of LEDs
d. The braiding of LEDs through the inside of the sweater should look like below

10. **Sew white string at different increments of the LED strip and braided wires on the inside so that it helps lay the strip flatly on the sweater as shown above**

11. **Setting up new row of LEDs**
    a. Make sure that the new row of LEDs start with the tabs of 5V, GND, and Din

12. **Repeat steps numbers 3-11 for the rest of the LED strips**

Step 10: Code of Matlab and Arduino

1. **Open Matlab on computer**
2. **Insert code from the Appendix: “Matlab Code”**
    a. Matlab Code includes image processing as it takes digital images and converts them to a 20 (height) x 24 (width) grid
    b. It also includes the process of color compression so that colors are more uniform making the image display on the sweater more clean and easier to identify
    c. The Code then creates a list of 3 variable - RGB values ranging from 0 to 255 as a txt file.
3. **Open Arduino on computer**
4. **Insert code from the Appendix: “Arduino Code”**
    a. Arduino Code includes the process of transferring the list of RGB values to the LEDs so they know what color to display

Step 11: Test an image of your own

1. **Connect USB cord to computer and Arduino**
2. **Search any image on your browser**
3. **Save image as a jpeg to your computer that you can easily access**
    a. We recommend giving a simple title to the image
4. **Go to Matlab**
5. **Place image directly into Matlab's "Current Folder"**
6. **Find "greenlanterngreen.jpg" and replace with "title of image.jpg"**
7. **Find "greenlanterngreen.txt" and replace with "title of image.txt"**
    a. As seen below highlighted in red

```
%imgName = input("What image would you like to display on the sweater?")
image = imread('greenlanterngreen.jpg');
imgpixelated = pixelate(image, 3);
% resize the image to match the dimensions on our sweater
imgrszd = imresize(imgpixelated, [h, w]);

% Recoloring image
imgrszd_recolored = recolor(imgrszd);
%imshow(imgrszd_recolored);


% sanity check to see what the resized image does to the original
%subplot(3, 1, 1), imshow(image), title('image');
%subplot(3, 1, 2), imshow(imgpixelated), title('pixelated');
%subplot(3, 1, 3), imshow(imgrszd), title('pixelated then resized');
%subplot(4, 1, 4), imshow(X_no_dither,map);

% take the matrix that makes up the image and turn it into a list where
% each list value contains 3 numbers (R, G, B) -> will be w*h x 3 dimension
list = [];
for i = 1:h
   for j = 1:w
       list = [list imgrszd_recolored(i, j, :)];
   end
end

% Creating new list so we can transform it into a 2 dimensional array with
% rgb as the rows instead of columns
listFinal = zeros(3,480);

% Adding list to listFinal
for i = 1:size(listFinal, 2)
    listFinal(1,i) =  list(1,i,1);
    listFinal(2,i) =  list(1,i,2);
    listFinal(3,i) =  list(1,i,3);
end

% Helpful lists to test code with
listRG = [255, 0, 0; 0, 255, 0; 255, 0, 0; 0, 255, 0; 255, 0, 0; 0, 255, 0; 255, 0, 0; 0, 255, 0];
listGB = [[0, 255, 0], [0, 0, 255], [0, 255, 0], [0, 0, 255], [0, 255, 0], [0, 0, 255], [0, 255, 0
[0, 0, 255]];

% Exporting listFinal to a txt file
fid = fopen(['greenlanterngreen.txt'], 'wt'); % open a file for text writing (wt)
formatSpec = '{%d, %d, %d}, ';
```

8. **Run the Matlab code**
    a. Should create a txt file of the image with 480 groupings
    b. Each grouping having RGB values representing the color of each LED

c. Example of a txt file below

harvard.txt

{{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0,
0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255,
255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255,
255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255,
75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255},
{255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255,
255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75,
0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0},
{255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255,
0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75,
0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0},
{255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},}

9. **Open txt file**
10. **Copy the entire text**
11. **Go to Arduino**
12. **Find this region of Arduino code**

```
CRGBArray <NUM_LEDS> leds; // array of length of number of leds we have

int heart[NUM_LEDS][3] =
  {{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0},
{255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0},
{255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75,
0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}}
  ;
```

13. **Add the same format of code beneath it**
    a. Press Enter
    b. Insert code “int [image title][NUM_LEDS][3] =”
    c. Press Enter
    d. Paste entire txt file
    e. Press Enter
    f. Press the “;” sign
14. **Go to the end of Arduino code**

```
void loop() {

 //displayImageBlink(heart);
 //displayImageBlink(pacmanWhite);
 //displayImageBlink(thumbsUp);
 //displayImageBlink(no);
 //displayImageBlink(star);
 //displayImageStatic(blueCool);
 //displayImageBlink(trees);
 //displayImageBlink(harvard);
 //displayImageBlink(smiley);
 //displayImageBlink(es50);
 //displayImageBlink(menorah);
 //displayImageBlink(peaceHand);
 //displayImageStatic(elmo);
 //displayImageBlink(eggplant);
 displayImageBlink(Superman);
 //displayImageBlink(rainbow);
 //displayImageBlink(prideflag);
 displayImageBlink(batman);
 //displayImageBlink(jason);
 //displayImageBlink(greenlantern);
 displayImageBlink(flash);
 //displayImageBlink(dc);
 displayImageBlink(greenlanterngreen);
```

15. **Add same format of code beneath it**
    a. Press Enter
    b. Insert code “displayImageBlink(image title);”
    c. As shown above
16. **Run Code**

Helpful notes:

- Can display a series of maximum 4 images
    - Simply run code for the 4 images you want to display
    - Use displayImageBlink([desired image]) in arduino to have it blink

        - This will insert a pause of all LEDs off between each display
    - Use displayImageStatic([desired image]) in arduino on the single image you want to display
        - This will display a single image. If you run more than one image, it will cycle between the images with no blinking between them
    - If you run code for more than 4 images, the code will not run through the Arduino
- Instead of deleting lines of code you're not using right now, simply put (2) slash signs "//" in front of the code of the image you don't want to display
    - This allows you to keep the code you have but not physically display the image you don't want

***refer to Appendix: "DC Logo Display" for an awesome video of the DIY Ugly Christmas Sweater going through an image series of DC Superhero logos

OPTIONAL Step 12: Crochet

If you want to create an even cleaner look after creating the sweater. Follow the steps below

1. Crochet in between each column of LEDs
2. Crochet a side pocket for where the Arduino can be stored as user wears sweater
    a. The crocheted side pocket will have a wide top opening and a narrow bottom opening so that the Arduino can be secured in and with the ability to have the wire connecting to the computer pull out

***refer to Appendix: "Crochet Sweater" for the link to time lapse video of crocheting the sweater

That is all for building our DIY Ugly Christmas Sweater! Now you have a full functioning sweater that can display any image and is definitely guaranteed life of the party accessory.

TEAM MANAGEMENT

Mia and Kade took the lead of the code with help from David and Nafi. Specifically with Mia being in charge of the Arduino code and Kade in charge of the Matlab code. David took the lead for circuit calculations and schematic design. Nafi took the lead for the written report. Mia and Nafi took the lead for the video. Kade also soldered and crocheted the sweater. Although there were different leads for each aspect of our project, everyone helped where they could. Specifically for the soldering and crocheting this was only done by Kade due to them being the only group member that could do so.

OUTLOOK AND POSSIBLE IMPROVEMENTS

Overall the project turned out well, however, several implementations can be made to improve the overall quality of the project. One simple implementation is increasing the quality of the image display through adding more LEDs. Essentially the LEDs are acting as the pixels of an image, the more pixels, the higher the image definition. We could also replace the LED strips with individual neopixels. Although this would increase the cost of the sweater, it would make the light up part of the sweater more flexible, and could possibly decrease the production time as the neopixels can be connected with conductive thread, thus eliminating the need for soldering.

Another implementation is making the sweater portable. Currently, the sweater needs to be plugged into a computer to be powered and for the computer to communicate with the Arduino. While this isn't inherently a problem, it would be better to be able to upload an image and walk around with the sweater without having to be connected to a computer. This would be a two-step process. We would need to power the LEDs and Arduino, and we would need to be able to communicate with the Arduino wirelessly. One method of powering the Arduino would be using a 9-volt battery. To power the LED strips, we would use a voltage regulator IC to limit the voltage to 5 volts in order to power the LEDs. The 9-volt battery can also be used to power the Arduino.

***For a possible design, refer to the Appendix: “9-volt to 5-volt regulator design” to see a

schematic and PCB design (made using the open-source software Kicad) for a 9-volt to 5-volt voltage regulator.

After the power issue is solved, wireless communication is the next issue. MATLAB can communicate via Bluetooth or wifi with an Arduino if a proper Arduino is used. However, an Arduino MKRZero does not have this capability. If an Arduino without Bluetooth or wifi support is used, the same effect can be achieved by using two Arduino with radio modules. One Arduino will be a receiver Arduino. The receiver is the one connected to the sweater. The sender Arduino will be connected to the computer, and it will transmit the data to the receiver.

ACKNOWLEDGEMENTS:

We would like to acknowledge TFs, Simon for helping us out with our image processing code and Kian for soldering tips with hot glue.

DISCLAIMER

We allow the sharing of our report, codes, photos, and videos of our project.

REFERENCES:

Matlab to Arduino Extension:
https://www.mathworks.com/discovery/arduino-programming-matlab-simulink.html

ES50 Lab 8:
https://canvas.harvard.edu/files/14622122/download?download_frd=1

Guide to Using LED Strips with Arduino: https://youtu.be/5M24QUVE0iU

APPENDIX

Useful drawings, diagrams, and code mentioned throughout the report

txt example:

harvard.txt

{{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0,
0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255,
255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255,
255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255,
75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255},
{255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255,
0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255,
75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255,
255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75,
0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255,
255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0},
{255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255,
0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255,
255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75,
0}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 75, 0}, {255, 75, 0},
{255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0,
0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 75, 0}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0}, {255, 0, 0},
{255, 75, 0}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},
{255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 75, 0}, {255, 75, 0}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255}, {255,
255, 255}, {255, 255, 255}, {255, 255, 255}, {255, 255, 255},}

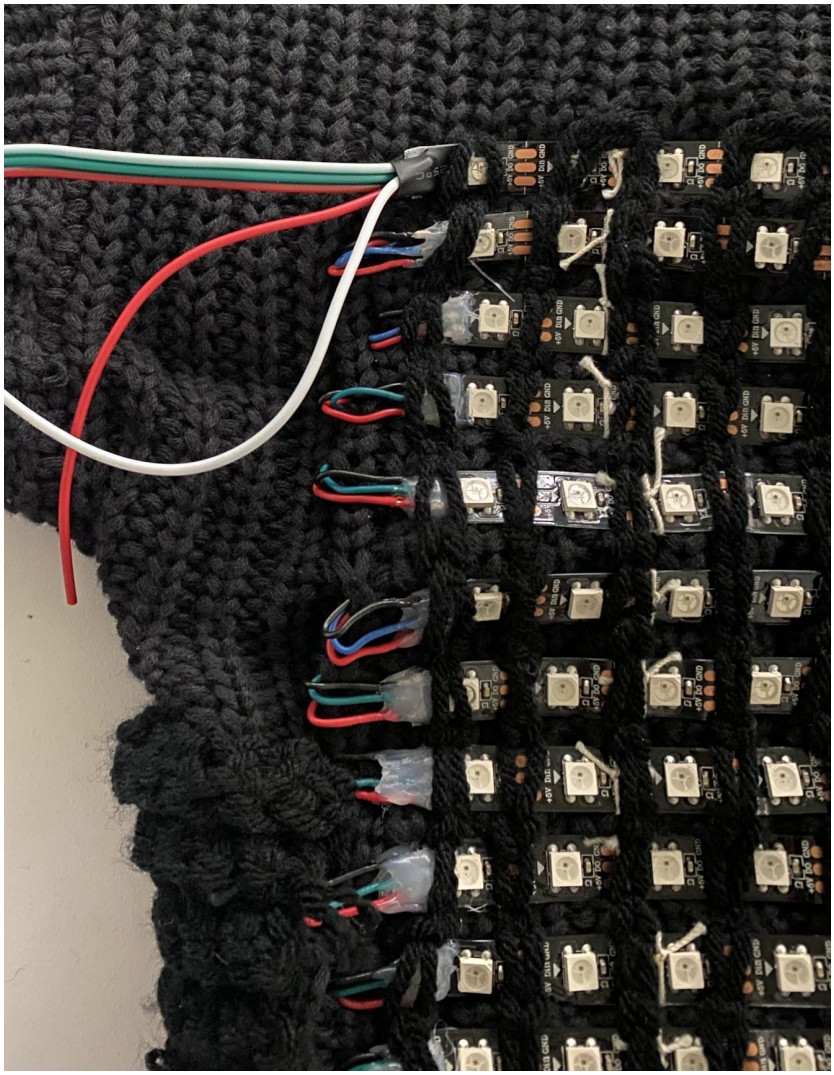
Color Compression:

% Recoloring function that changes colors to pure rgb values
function imgrecolored = recolor(image)

```
imgrecolored=double(image);
   % red, orange, yellow, green, cyan, blue, purple, pink, white
   base_colors_list = [255, 0, 0;  255, 255, 0;  0, 255, 0;  0, 255, 255;  0, 0, 255;
200, 0, 255;  255, 0, 255;  255, 255, 255];
   % Changing list into type double
   base_colors = double(base_colors_list);
   % Create an emtyp array for the difference between the colors
   differenceArray=[];

   % Function to find the difference between each color and palette
   % Cycling through each pixel in image
   for i = 1:size(imgrecolored,1)
      for k = 1:size(imgrecolored,2)
         % Cycling through each color in palette
         for j = 1:size(base_colors,1)
            differenceArray(j) = sqrt( (imgrecolored(i, k, 1)-base_colors(j,1))^2 +
(imgrecolored(i,k,2)-base_colors(j,2))^2 + (imgrecolored(i,k,3)-base_colors(j,3))^2
);
         end
         % Function to find the smallest value in differenceArray
         [smallest_val, location] = min(differenceArray);
         imgrecolored(i,k,1) = base_colors(location,1);
         imgrecolored(i,k,2) = base_colors(location,2);
         imgrecolored(i,k,3) = base_colors(location,3);
      end
   end
end
```

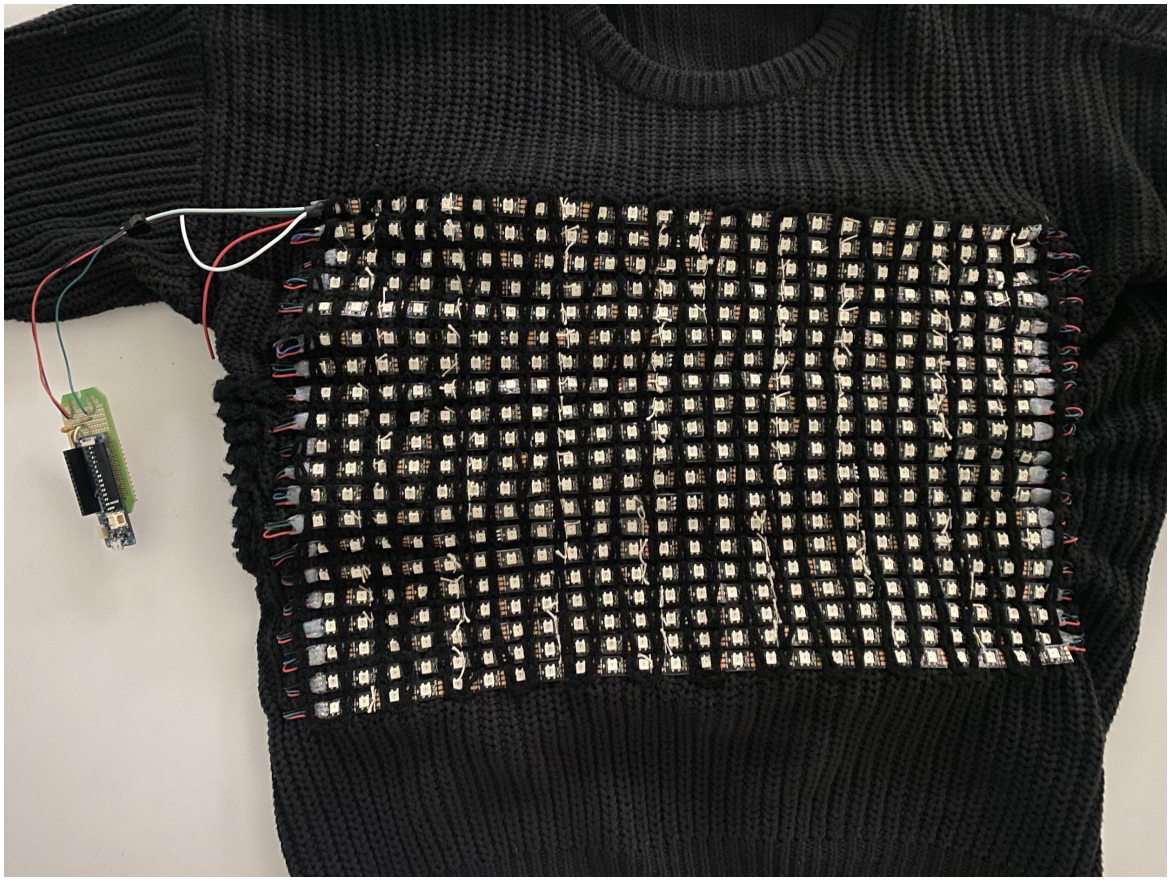
Glued Edge Tab:



Crochet Sweater:

https://drive.google.com/file/d/1f_K_zbklKdFtvDxeQYK7jwTF0iLmnr1S/view?usp=sharing

Final Sweater:



Matlab Code:

Separate pdf titled: “DI_WIRES_V4.m”

Arduino Code:

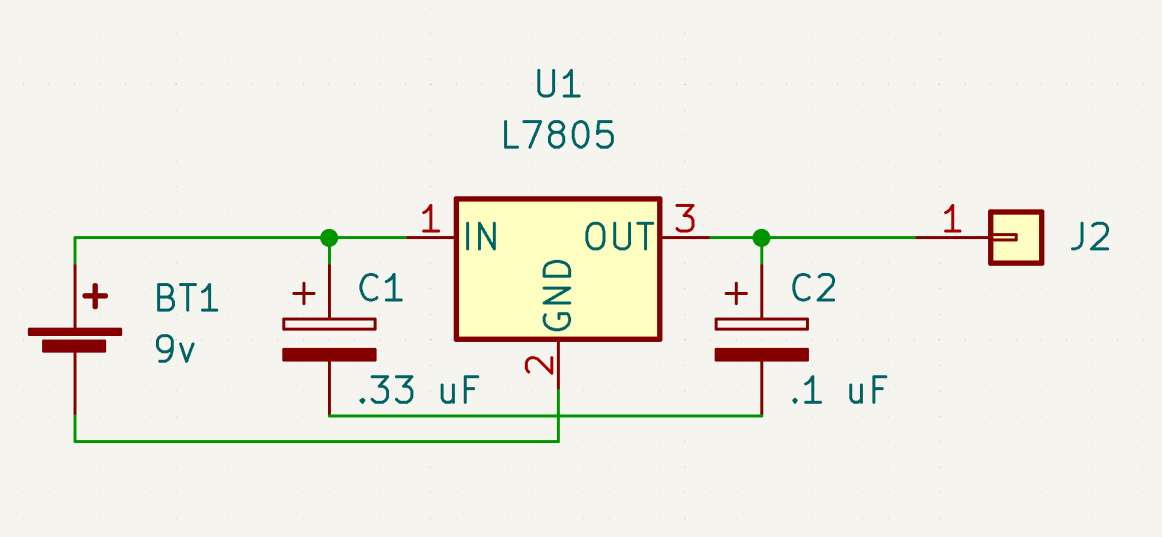
Separate pdf titled: “slideshowv2.ino”

DC Logo Display:

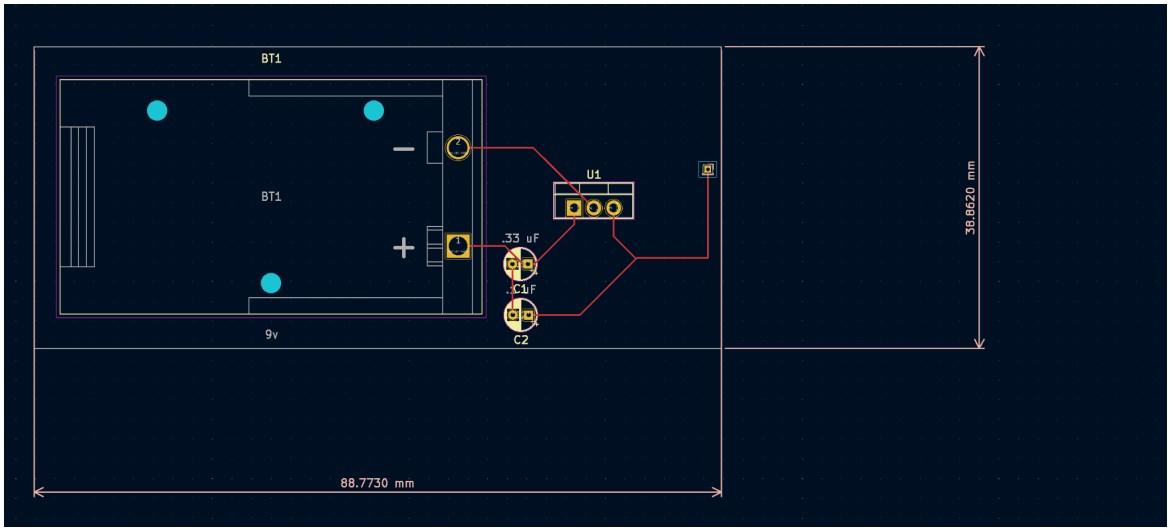
https://drive.google.com/file/d/1oMEiB8s29UDPnu95calIKA1U6Hbj8TRv/view?usp=sharing

9-volt to 5-volt regulator design:

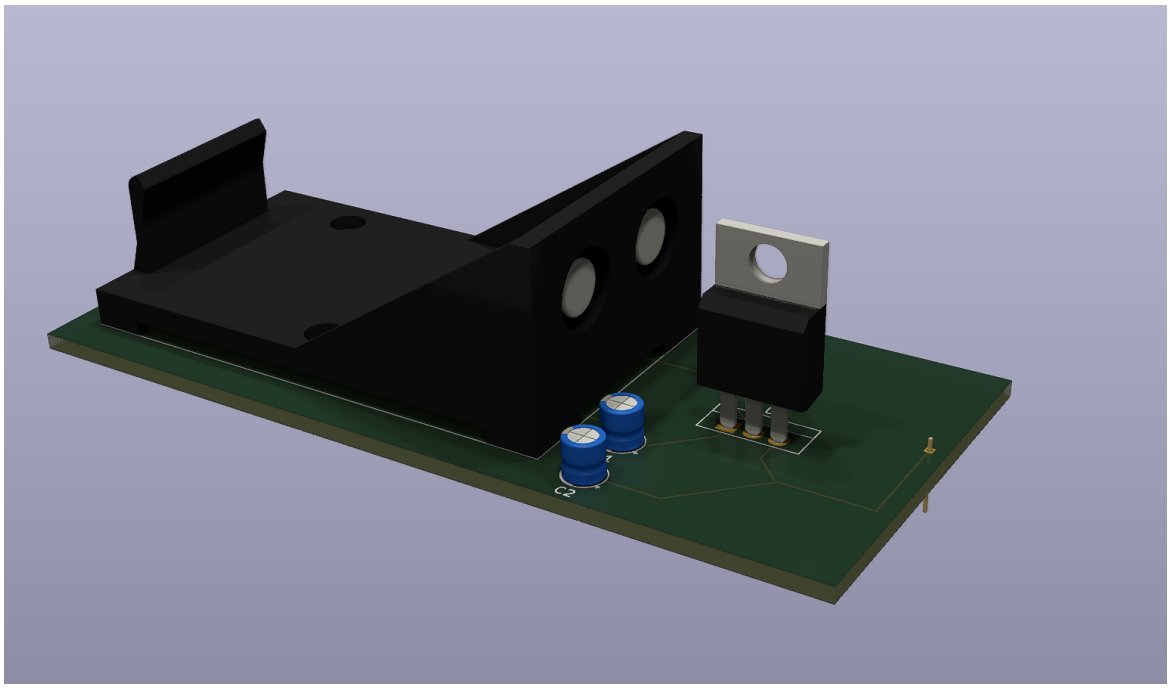
Note: The capacitor values are from the L7805CV Datasheet by STMicroelectronics. The capacitors help smooth out the signal to keep it at a constant 5 volts.



Schematic of the voltage regulator



PCB design of the voltage regulator



3D model of the pcb rendered and created by the Kicad software.