Chris Loang, tloang2, tloang2@uic.ed

Sergio Covarrubias, scova2, scova2@uic.edu

Jay Patel, jpate245, jpate245@uic.edu

Voice Control Lights Blinking Cycle

Abstract of your project

We want to display leds in different styling with mainly using voice control and to also have a button to perform this task manually. This will then inform the user of the current styles being displayed and will suggest other alternative styles. The main goal for this project is to have it perform using our voices as effectively as possible.

Overall Description of Project Idea

We will have an Android emulator on Windows or Mac to connect to bluetooth with a HC - 05 bluetooth sensor module to enable voice recognition. We will have a Serial port from Arduino, and perform serial readStrings to get the string from the serial port, and that will be our input for the project. Our input will be an input to command LEDs functionality, we will have NeoPixels and use our NeoPixels LEDs to display disco, wave functionality, also changing the colors. To display disco functionality, we will have it blink multiple in random, and for a wave functionality, we will flash the first led, follow with a second one, and along the strip of the neopixel leds. At the same time, we will have an LCD display that displays current information about the NeoPixels functionality that is being used, and for the second row of the LCD, we will show suggestions for the alternative functionality of the LEDs that you can input. We will also have a button or a switch, that you can manually change the LEDs functionalities rather than have your voices as input.

Detailed Project Ideas:

For the Bluetooth connection, we decided that we will get an Android emulator, such as Bluestack, LD player so that we can connect a bluetooth from the Android emulator to the Bluetooth Sensor which is a HC - 05 Bluetooth Serial Module. If the Android Emulator does not work out, we will use a Iphone as our prime communication device.

However the Iphone is limited in apps, and the app that is being presented right now has limited functionality as well with voice recognition. We will have serial connection from our code to perform serial read from the Bluetooth Serial, and then decipher that into string and turn that into argument for function parameter.

For the LEDs, we will use neopixel, neopixel is a stripe of LEDs, and neopixel have additional api, such that you can adjust RGB lights within each LEDs. We want to use the neopixel to perform LEDs functionalities because you can use the built-in api for it. We still write functions to display LEDs to have disco balls, and wave functions. We will have each functionalities in each separate function with each parameter as a string's argument. We will have one function for waving function, with the neopixel stripe, then we have utility functions to use, and as well we can adjust each light on each LCD. In total we will have 2-3 functions all together for the LEDs stripe.

We will have separate LCD displaying the current functionality, and the second row of the LCD will have the suggested functionality, this will connect with the LEDs functionality by having a flag, and set the flags to switch the text on the LCD to print differently based on the LEDs functionality to run next. In addition, we will have a button to manually press so that it uses an interrupt signal, to interrupt and switch to different LEDs functionality.

Final Project Design stating Expected Inputs/Outputs :

Our expected input would be a phone device, or a computer device that has an android emulator installed, both devices require voice recording functionality or any voice recognition third party software to be able to store the voice into readable string or char by our software. Aside from the above mentioned input, we also have a button that works as an input for our software. The button will be another functionality to change different functionalities of the blinking neopixel leds. Our output will be the lcd, and the neopixel leds. For the LCD, the output will be for the first row lcd, we will have the current LEDs function displayed, and for the second row lcd, the lcd will display the alternative LEDs blinking functionalities, for the second row, if the string is too long, we will be scrolling the text so that it is readable.For our project, the main output would be the neopixel leds, that is either a circular neopixel leds, or a straight stripes of leds, that will display different lightings depend on the input which is a string extracted from the voice recognition. We will have three main led blinking functions, which are wave cycle, disco wave, and circular wave.

Here is an example on how our voices will be inputted into the app, and how our voice will be output. Assume one of our group members' voices was high, when speaking into the app on the android device. His voice will transfer into the bluetooth adapter, that will input a string,used in determining the amount of bits once converted into bits. The bits are used to determine whether his voice frequency is high enough for the switch to turn on all the led lights in a cycle forming a disco light display. At the same time the voice frequency will be used to display a wave frequency that will provide evidence on how high his voice was.

Final Expected Plan for Communication:

To communicate effectively, our team members decided to use discord and texting. Texting will be the main focus when attention is needed, and discord is where we can meet and chat about the project. This is where the weekly meetings will occur.

Our plan for communication between the Arduiono and another device which is a phone will be a Serial Bluetooth module device. The Serial Bluetooth device allows us to connect bluetooth through our phone with a serial device, then performs serial readings.

Final Description of the original work being attempted by our project:

Our projects involve original work such as we have functions to display multiple leds blinking. However the leds don’t blink in a default way such as blink then off, but more so that they have a variety of functionality, such as the leds demonstrate breathing cycle, wave cycle, disco cycle or random cycle. There are two ways this functionality will be performed, one is using our voice where it’ll take our voice and translate into a string which then will be used to input into different functions to perform different functionality. The other is a button which will do things manually in case there is some kind of problem with the voice control. The button in this case is used as a backup and our main focus will be making the voice control as effective as possible.

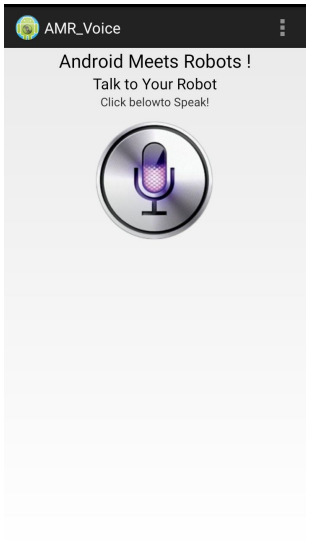
We also will display the name of these functionalities to the LCD screen on the top row, and the second row can be displayed to suggest the functionalities. We have also searched the internet looking for the exact project and guarantee that our project is original.

Discussion on how to build your project (think Lab Reports):

The project will be used for testing out the strength and weakness of a person's voice. Through string command or frequency. We will also be using a button switch to power

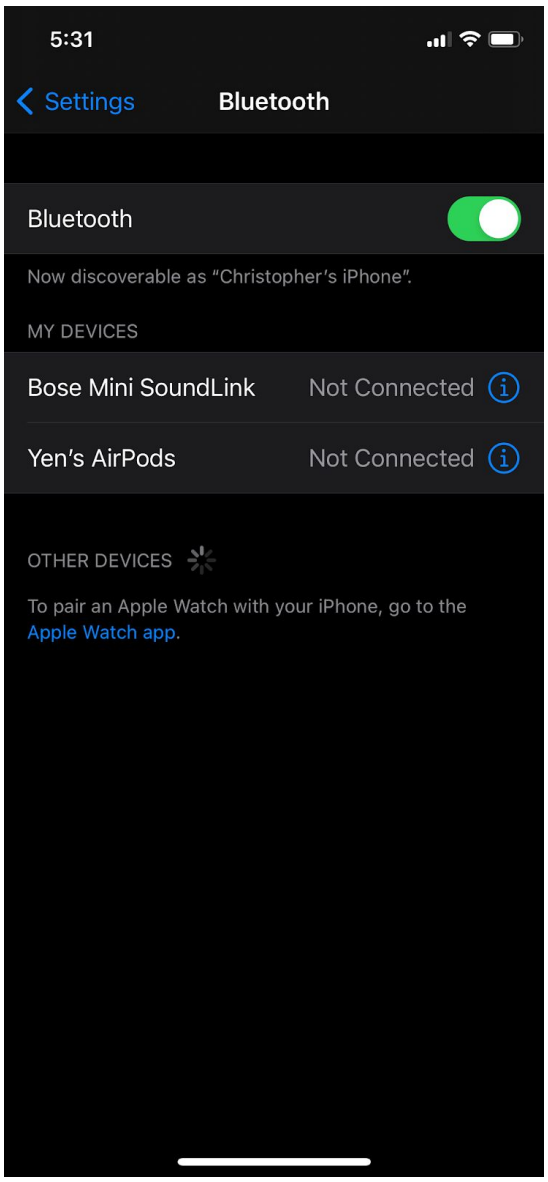
up either the RGB strip or the RGB ring. The RGB strip will be used in the wave pattern and the RGB ring will be used in a circular pattern and both the RGB strip and RGB ring will be used for the disco pattern. Let use begin on the circulation connection.

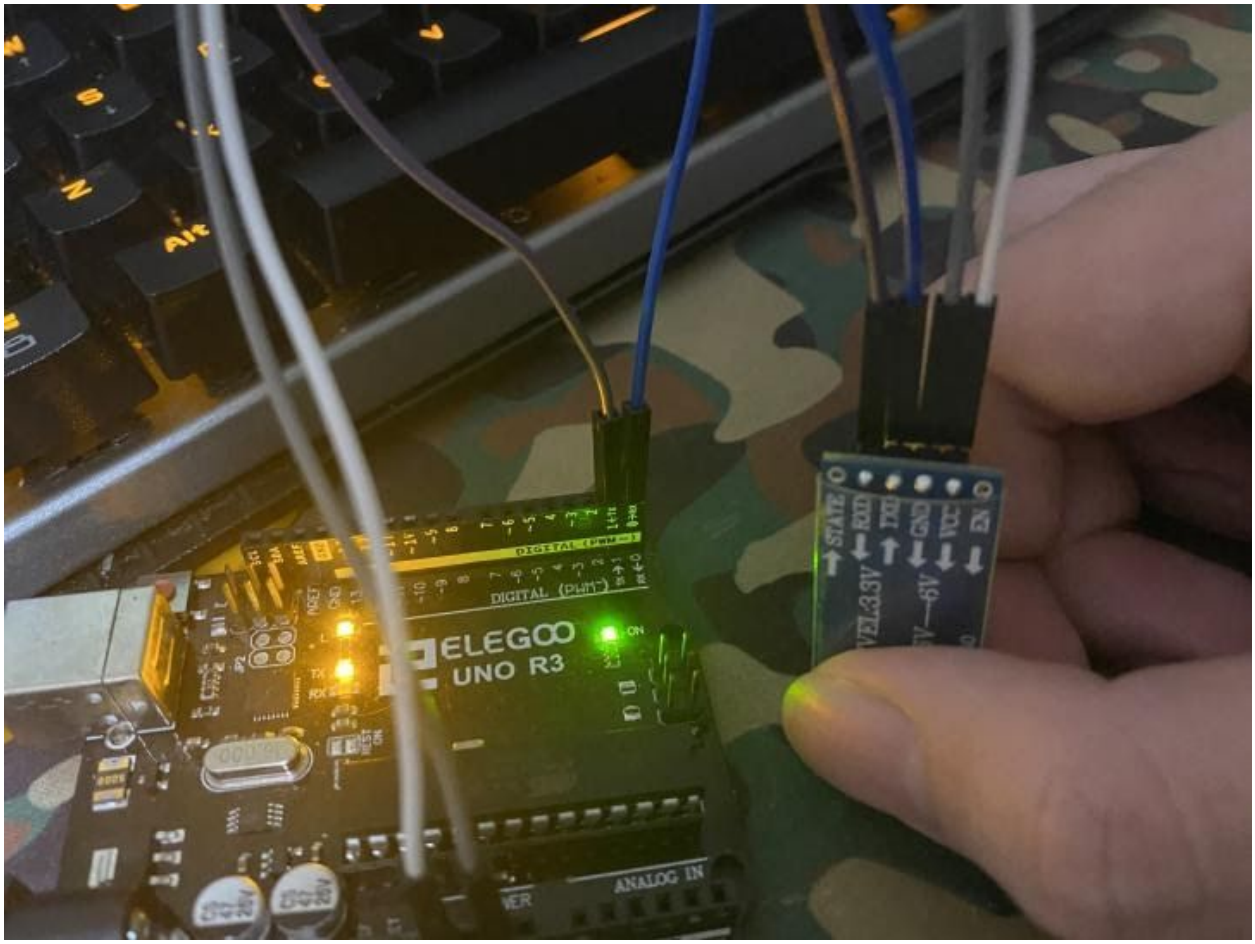
Step 1: Downloading the voice frequency app using the Arduino IDE.

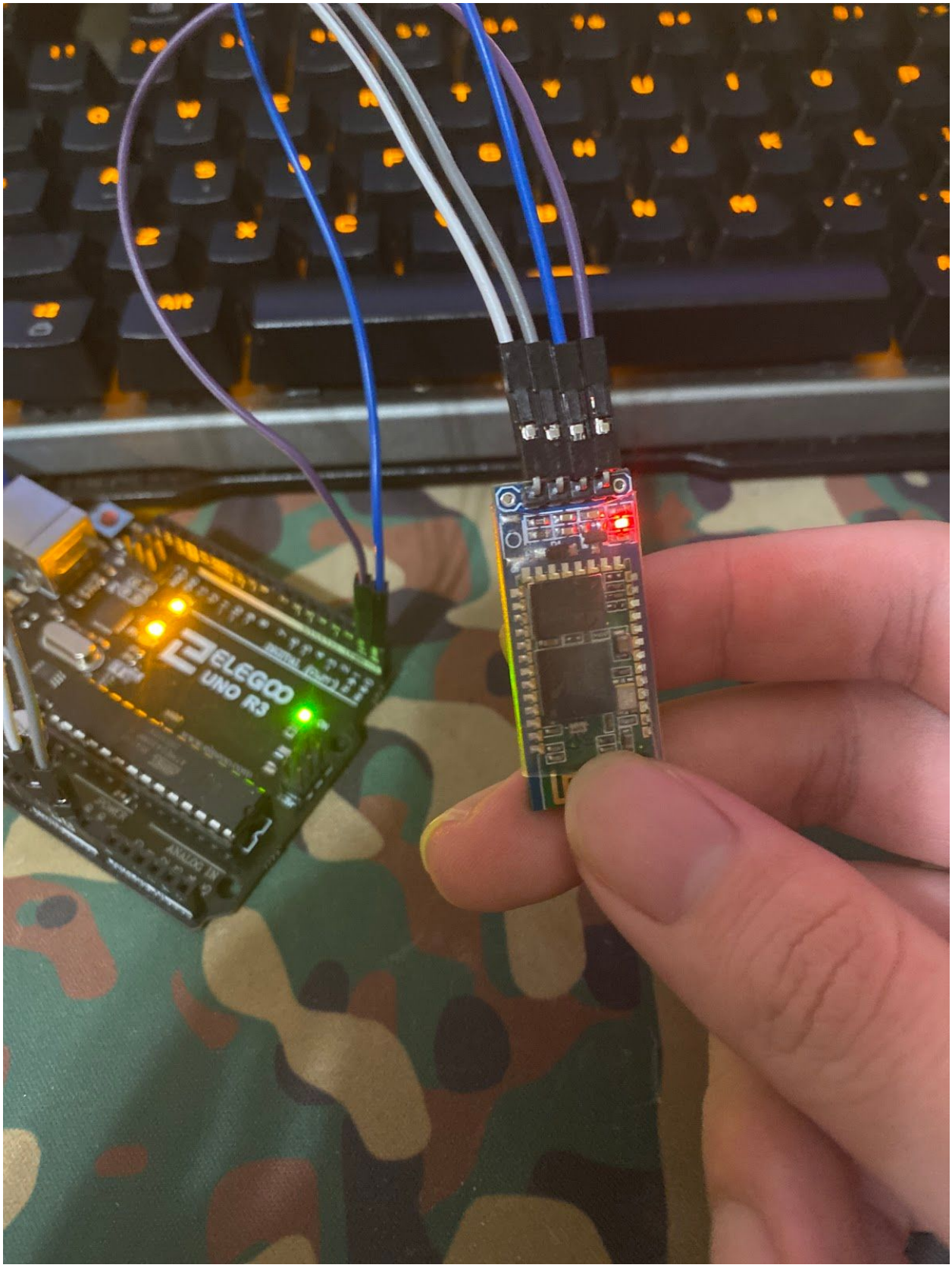


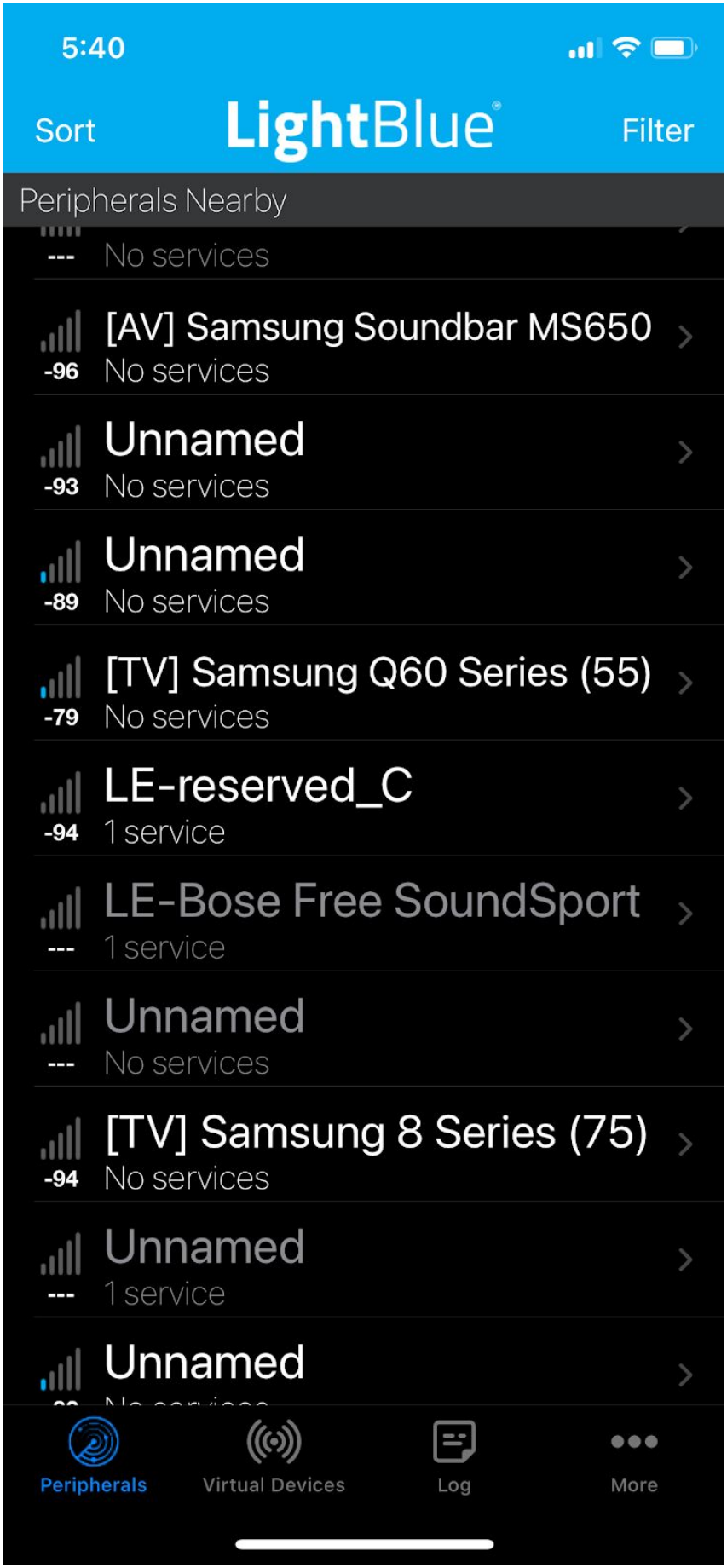
- By going to the arduino.cc website you can download the Arduino IDE software to your computer. Just make sure you select the correct software that works on your computer.
- Once you complete the Arduino IDE download, you must download our “voice frequency” program that can be uploaded to your Arduino UNO.
- Plug in the Arduino UNO into your computer using the given USB connector from ELEGOO UNO Project Super Starter Kit.
- Head into the “Tool” tab and select the appropriate Arduino UNO r3 and select the appropriate USB com, that will be either 3 com or 4 com.
- Then install the program by using the arrow button and wait until the download is complete.

- IWith an Android device you must download “Android Meets Robots!” in the google application store. The application is needed so that we can communicate with the bluetooth adaptor.
- Chris Updates: This voice frequency app is not available in App’s store, and even though I connected the electrical wires correctly, the Iphone would not detect the HC-06 Serial Bluetooth Module. I included pictures below.

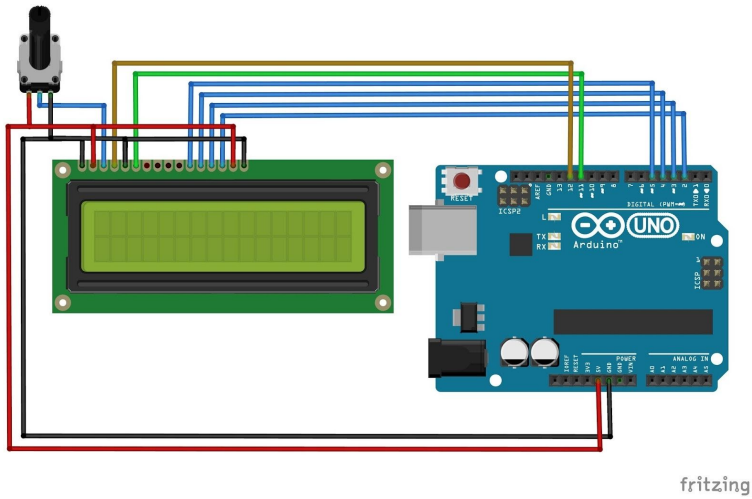

5:31
Settings
Bluetooth
Bluetooth
Now discoverable as "Christopher's iPhone".
MY DEVICES
Bose Mini SoundLink
Not Connected
Yen's AirPods
Not Connected
OTHER DEVICES
To pair an Apple Watch with your iPhone, go to the
Apple Watch app.


ELEGOO
UNO R3
DIGITAL (PWM~)
ANALOG IN
STATE
RXD
TXD
GND
VCC
EN


ELEGOO
UNO R3


5:40
Sort
LightBlue
Filter
Peripherals Nearby
--- No services
[AV] Samsung Soundbar MS650
-96 No services
Unnamed
-93 No services
Unnamed
-89 No services
[TV] Samsung Q60 Series (55)
-79 No services
LE-reserved_C
-94 1 service
LE-Bose Free SoundSport
--- 1 service
Unnamed
--- No services
[TV] Samsung 8 Series (75)
-94 No services
Unnamed
--- 1 service
Unnamed
Peripherals
Virtual Devices
Log
More

Step 2: Connecting the LCD 16*2 display, this should already be sodder.



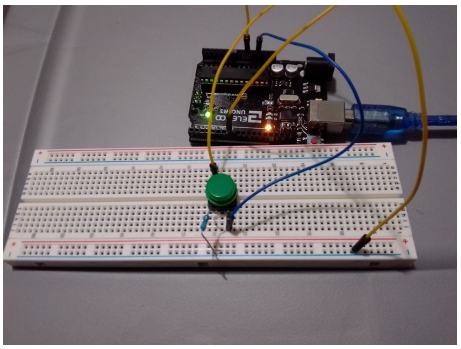
This image is an example of how to properly install the lcd 16 x 2 display(Admin, p.3). The LCD 16*2 should be with a Potentiometer and a 220 ohms resistor, for LCD 16*2.

- Using a breadboard connect an LCD 16*2 display and the potentiometer. In this step no jumper wires are needed.
- Next use two jumper wires and connect one to the Analog 5v and the second to anyone of the grounds found on the Arduino UNO r3. One end of the jumper wires will be connected to the Arduino UNO r3 and the other end of the 5v jumper wire will connect to the positive side of the breadboard and the ground connector will connect from one end of the Arduino UNO r3 to the ground of the breadboard.
- The first thing we will connect is the potentiometer.
    - Face the one pin position close to the 5v line.
    - On the left line pin place a jumper wire from one end and the other end to ground.
    - On the right pin place a jumper wire pin from one end to the 5v on the other end.
    - Between the two pins place a jumper wire pin and on the other end of the pin place it near the RS pin associated with the lcd display 16*2. This part goes from left to right so the 4th pin on the far left.

- The second thing is to connect the lcd display properly to the Arduino UNO r3.
    - First, if you have not done so place, the LCD display on the breadboard. You might find it easier to install if you install it at an angle.

- Once properly installed follow the fourth step in potentiometer connection and connect jumper wire pin to close to RS pin. .
- The GND , RW(read/write), and K(cathode) pins from left to right go to ground. Use jumper wires to connect each pin to ground.
- The VCC and A(anode), from left to right, will be connected to 5v and the A(anode) will be using a 220 ohms resistor. The 220 ohms resistor will need to be connected to 5v first and then a jumper wire will be used between the 220 ohms resistor and the pin of the LCD display.
- For other pins connections please look below and see which jumper wire will connect to what part on the Arduino UNO r3:
  - LCD RS pin to digital pin 12
  - LCD Enable pin to digital pin 11
  - LCD D4 pin to digital pin 5
  - LCD D5 pin to digital pin 4
  - LCD D6 pin to digital pin 3
  - LCD D7 pin to digital pin 2

Step 3: Connecting the switch button on to the breadboard:

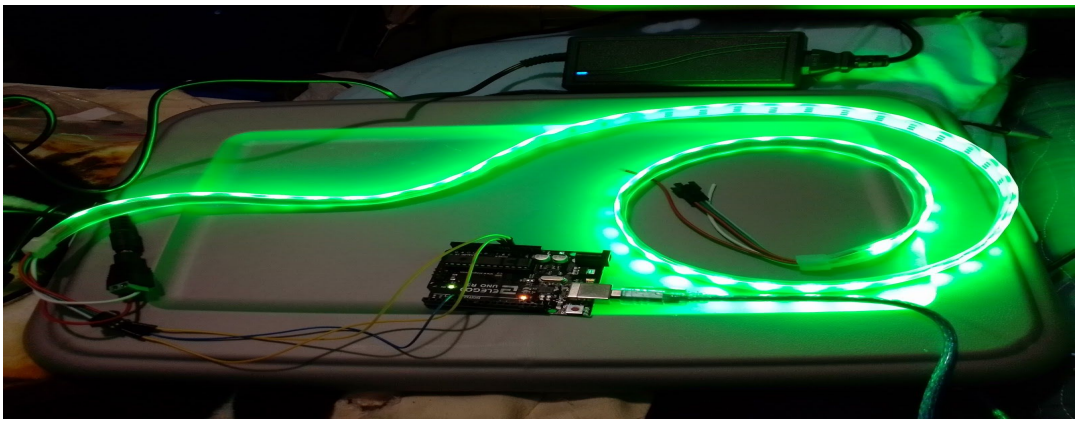


- You may use the same breadboard for the connection of the switch button. Place the switch button properly on to the breadboard. Words of advice place it into the center where there is a gap and it will be much easier to install.
- Use a 220 ohm resistor and connect one end to ground on the breadboard and the other end of the resistor to the pin on the right of the switch. Be advised whichever pin you use must be on the same side as the left pin that you want to use for the button switch. Use the Diagram of how the connection may look as an example. The switch button is on the right side of the lcd 16 x 2 display on the breadboard.
- Use a jumper wire to connect one pin end to the 5v on the breadboard and the second end to the left pin of the switch. The left pin should be the left pin closest to the 220 resistor.
- The last thing is to connect the switch button to the Arduino UNO r 3 digital pin 7 using a jumper wire. Place one end of the jumper wire to the Arduino UNO r3 digital pin 7 and the other end to the opposite right pin that does not contain the 220 ohm resistor.

- If you would like to see the connection properly please look at the photo on the left. The photo displays the proper pins and connections needed when connecting the button switch to the breadboard. Also, it's ok if the button switch does not look like the one in the image. The image of a button switch can be found in the ELEGOO UNO Project Super Starter Kit once you purchase the kit.

For the next steps you will only need the breadboard for ground and 5v connection other than that the other three steps do not need a breadboard.

Step 4: RGB strip connection:



- First the connecting the center pieces of the RGB strip.
  - On the RGB strip connect the red pin to the red jumper cable and connect the other end of the jumper cable to the 5v Analog for the Arduino. .
  - On the RGB strip connect the white pin to the black jumper cable and connect the other end of the jumper cable to the negative/ ground side of the .
  - On the RGB strip connect the green pin to any color jumper cable and connect the other end of the jumper cable to the Arduino digital 6 pin .
- Second connect the Alcohol 5V 8A 40W Power Supply and the converter transformer 5.5X2.5mm plug. You can do this by:
  1. First take the red wire, not the red wire that is connected to the green center wire, and either use a pair of scissors or wire trimmers to remove part of the protected shell.
  2. This will expose the wire and once it is exposed connect the exposed wires properly into the transformer converter, place it where it states +,

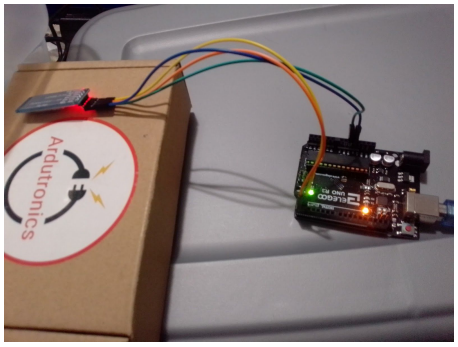
and make sure to screw it in tightly. Make sure it is properly in place and that none of the exposed wire is showing.

3. Do step 1 and 2 for the white wire, that will be ground.
4. Attach the power adapter to the transformer converter and plug in the wire to the wall. If you do not see smoke then it was done correctly.
5. Once completed the test you may unplug the power adapter from the wall.

WARNINGS:

1. Beware if you purchase an adapter with 10A then you do not need the 2A that is associated with the Arduino.
2. If you see smoke coming from the transistor it means the wires were not placed in correctly. Please unplug the external adapter from the wall before trying to place the red and white wires properly in the transistor.

Step 5: Bluetooth adaptor connection:



- Using the Female to Male Pupont Wire (10pcs) we will connect four pins of the bluetooth to four color wires next to each other. By doing so it will be easier for us to see which color pins are connected to the bluetooth adaptor. Just follow which part of the bluetooth adaptor goes where.
    - On bluetooth adapter: VCC connects to Arduino UNO r3 : 3.3v.
    - On bluetooth adaptor: Ground connects to breadboard: Ground.
    - On bluetooth adaptor: RX connects to Arduino UNO r3 : digital pin 1.
    - On bluetooth adaptor: TX connects to Arduino UNO r3: digital pin 0.

Discussion on how to your project is to be used (think User Guide) :
Hint: How does each component work? Meaning what is their role in our project?
Before we begin if you have not done so, please follow each step to connect each hardware component to its proper location on the breadboard and arduino. Once everything is properly connected follow these steps to guide you on what our project is doing.

Decide if you're going to press the switch button or will you be using the application on your Android device to communicate with the bluetooth adapter.

Option 1 switch button:

If you decide to use the switch button, then once you press the switch button, the button_count variable will raise by one. Now depending on the value of button_count, either the RGB strip, WaveCycle function call, the RGB ring, CircularCycle function call, or both will light up, DiscoCycle function call. For instance, if the value for button_count = 3, then the DiscoCycle function will be called and both the RGB strip and the RGB ring will light up.

button_count results:

- button_count = 1 the WaveCycle function call that RGB strip will turn on.
- button_count = 2 the CicularCycle function call that RGB ring will turn on.
- button_count = 3 the DiscoCycle function call that turns on the RGB strip and RGB ring.
- button_count = 4 the RandomCycle function call that will allow a random function call between WaveCycle function through DiscoCycle function.

Option 2 Android app to bluetooth adapter:

If you decide to use the Android device please make sure to download the application known as “BT Voice Control for Arduino” to communicate with the bluetooth adapter .The application is used by capturing our voice and sending our voice as a string to the bluetooth adaptor. So, for the most part, the pitch of our voice will depend on how high the frequency was transferred to the bluetooth adaptor.

Step 1: Once the frequency transferred from Android smartphone application into the bluetooth adapter the frequency will be read in and stored into store_voice_frequency. The variable store_voice_frequency, stores an integer that was converted from the string/pitch of our voice.

Step 2: The store_voice_frequency, as a string command, will be used in our function ReceivingVocieFromPhone().

ReceivingVocieFromPhone(Store_voice_frequency) results:

- Using “turn on disco”, will call the function DiscoCycle() and turn on the RGB 60 Led lights strip
- Using “turn on circular” will call the function CicularCycle() and turn the last RGB 30 Led lights on
- Using “turn on wave” will call the function WaveCycle and turn on the first RGB 30 LED lights on.

The LCD display 16*2.

Once again if please make sure to connect the LCD display 16*2 on the breadboard with the proper Arduino connections. Once you have completed the proper connections, then you are ready to use the LCD display 16*2. The LCD 16*2 displays what is needed to print out for the user to know what is going on in the program. It will print out the name of our program and print out what will be turned on depending on if you use a switch button or the Android application, from string frequency to an integer that will determine what must be turned on. For instance, one of our print statements to the LCD 16*2 display is . lcd.print(" RandomCyle turns on "). As you can see there the way to print something out in the LCD 16*2 display is by using the lcd.print() and placing your string inside the two circular brackets.

RGB 60 led lights strip and RGB 93 led lights ring :

Once again please make sure that both the RGB 60 led lights strip and RGB 93 led lights ring are properly connected to the proper Arduino Uno r3 devices. If they are both RGB 60 led lights stip and RGB 93 led lights ring connect properly let me explain what they do . RGB led lights have the capability of altering coloration by using, (R = red = 0 through 255, G = green = 0 through 255, B = blue = 0 through 255). Depending what number you choose will determine the colorations brightness. Zero means unused and anything greater than zero means some type of brightness. Now since we are aware of what RGB led lights do to receive color, we can speak about how they are being used in the code.

In our code the RGB 60 led lights strip use :

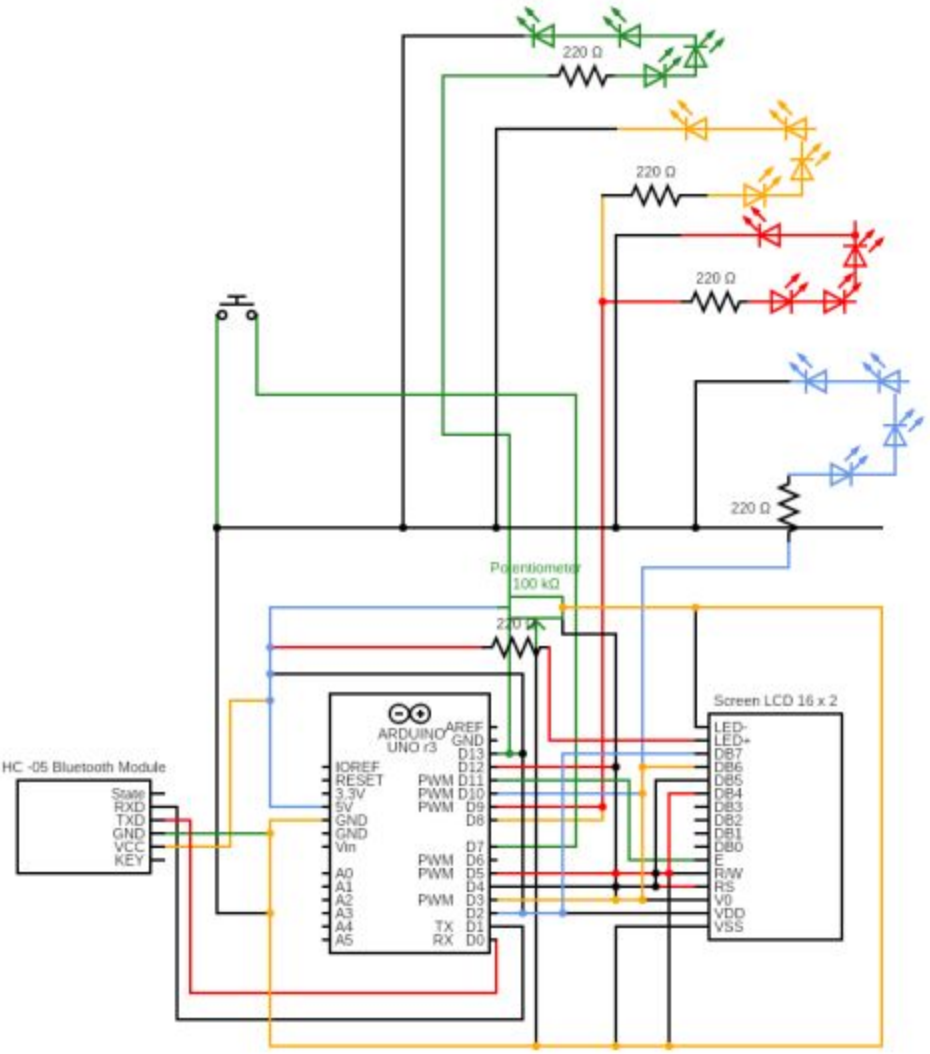
- RGB 60 led strip as a description .
    - #define wave_pin 6
    - #define wave_count 60
    - #define color__combination GRB
    - #define chip WS2812B
    - #define Bright 200
    - #define Frames_per_second 60
- This sets up the coloration for wave pattern, circular pattern and disco pattern.
    - FastLED.addLeds<NEOPIXEL, wave_pin>(leds, wave_count);

Arduino UNO r3

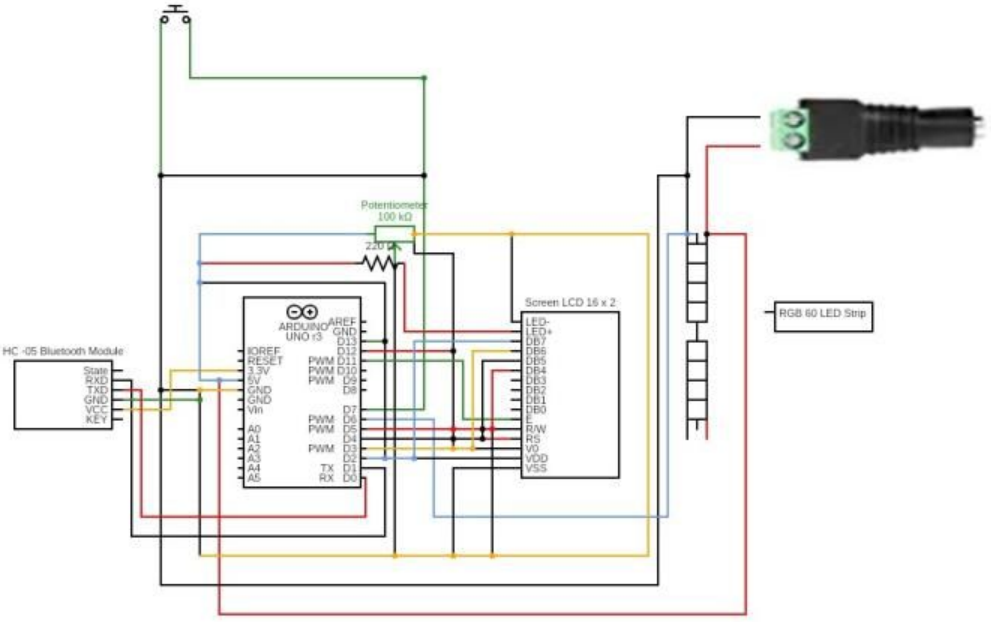
Before we can use anything from the list above we would have to make sure to connect the Arduino Uno r3 by using the usb cable and connecting it to your computer, to power up the Arduino Uno r3. The Arduino Uno r3 is used to connect every device that we will need to power up. If you would like to know which device is connected to the Arduino Uno r3 and with what pin please return to the set

up and see how all of our devices are connected to the Arduino Uno r3. For instance, many of our devices, like the bluetooth and LCD 16*2 displays need the Analog pin 5v to power up. Therefore, it is important to have a working Arduino Uno r3.

Project Hardware diagram previously:

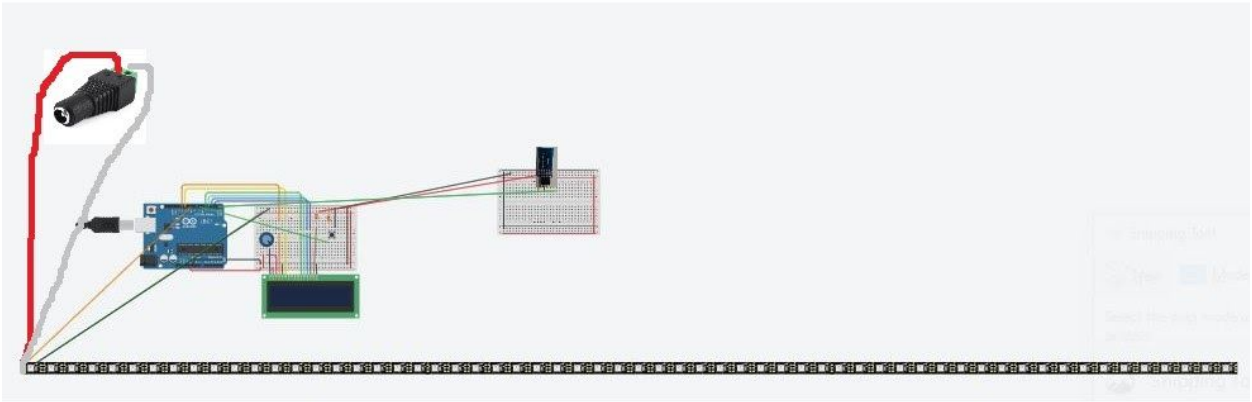

220 Ω
220 Ω
220 Ω
220 Ω
Potentiometer
100 kΩ
220 Ω
HC -05 Bluetooth Module
State
RXD
TXD
GND
VCC
KEY
ARDUINO
UNO r3
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5
AREF
GND
D13
D12
PWM D11
PWM D10
PWM D9
D8
D7
PWM D6
PWM D5
D4
PWM D3
D2
TX D1
RX D0
Screen LCD 16 x 2
LED-
LED+
DB7
DB6
DB5
DB4
DB3
DB2
DB1
DB0
E
R/W
RS
V0
VDD
VSS

New Project Hardware diagram :



Comparing the previous sketch from the new sketch:.
The major differences are, we removed all the led lights from the wave and added a RGB led lights strip with a digital connection associated with 6. Another change is with the RGB LED strip we connected a converter transistor in the event that there is not enough power that is generated by the Arduino Uno r3.

Diagram of how the connection may look:



The diagram above shows the proper connection for every item we will be using in our project.

- Starting with the left we have a 16*2 lcd display with a potential meteor
- Next on the bottom we have our 60 RGB led light strip with a red and white wire that will connect to a Converter Transformer.5X2.5mm Plug.
- The top right has one item on the breadboard. The item is the bluetooth adaptor.
- On the first breadboard, on the left, has a switch, next to the 16*2 lcd display..
- One thing we want to mention is, on this diagram we have a Converter Transformer 5.5X2.5mm Plug plugged into a power supply that will be attached to a 5v 8amps adaptor.

Discussion on how your project is to be used (think User Guide):

The project can be used in two ways to display the different functionality of the lights. One way is to use a button, so everytime the button is pressed it will go in this order: wave, circular, disco, and random. The other way is to use voice recognition which is the main goal for this project. To do that first you'd have to connect to the bluetooth and use a 3rd party software that is only available to the Android devices. Unfortunately for the Iphone, the connection is harder such that a connection between the serial bluetooth and Iphone application might not work. Therefore, for Apple's applications, the application is more limited such that there's an option for voice recognition, but a typing of inputs, where the user needs to type in the functionality manually. Whereas, for the Android Users, they can use the voice recognition to turn different lighting functions like Disco lights, waves, and a bunch of other options to choose from. We also have an LCD, which will say which functionality is being used in real time.

Code Sketches:

```
/*
Voice Control Lights Blinking Cycle.
Created by:
Chris Loang, tloang2,
Jay Patel, jpate245.
Sergio Covarrubias, scova2,


Link to a demo of our project:


Description of our project.
We will have an Android emulator on Windows or Mac to connect to bluetooth with a HC
- 05 bluetooth sensor module
to enable voice recognition. We will have a Serial port from Arduino, and perform serial
readStrings to get the string from the serial port,
and that will be our input for the project. Our input will be an input to command LEDs
functionality, we will have NeoPixels and use our NeoPixels LEDs to display disco,
wave functionality, also changing the colors. To display disco functionality, we will have
it blink multiple in random, and for a wave functionality,
we will flash the first led, follow with a second one, and along the strip of the neopixel
leds. At the same time, we will have an LCD display that displays current
information about the NeoPixels functionality that is being used, and for the second row
of the LCD, we will show suggestions for the alternative functionality of the
LEDs that you can input. We will also have a button or a switch, that you can manually
change the LEDs functionalities rather than have your voices as input.

 The circuit:
 * LCD RS pin to Arduino digital pin 12
 * LCD Enable pin to Arduino digital pin 11
 * LCD D4 pin to Arduino digital pin 5
 * LCD D5 pin to Arduino digital pin 4
 * LCD D6 pin to Arduino digital pin 3
 * LCD D7 pin to Arduino digital pin 2
 * LCD R/W pin to Arduino Analog pin GND
 * 10K resistor
 * ends to +5V and ground
 * wiper to LCD VO pin (pin 3)
```

```
 * Bluetooth RXD pin to  Arduino digital pin RX/D0
 * Bluetooth TXD pin to  Arduino digital pin TX/D1
 * Bluetooth VCC pin to Arduino Analog  pin 5V
 * Bluetooth GND pin to Arduino Analog pin GND
 * Potentiometer 100K
 * 220 resistor:

 *RGB 60 LED light ground pin to Arduino Analog GND
 *RGB 60 LED light power pin to Arduino Analog 5v
 *RGB 60 LED light pin connector to Arduino digital pin 6
 *RGB 60 LED external(RED) power cable to power adaptor 5v port
 *RGB 60 LED external(WHITE) GND cable to power adaptor GND port

 *Switch button pin to digital pin 7
 *Switch button pin ground to Arduino Analog GND
 *Switch button 220 resistor to Arduino Analog 5
*/

// include the library code:
#include <LiquidCrystal.h>
#include <SoftwareSerial.h>
#include <FastLED.h>

//RGB 60 led strip.
#define wave_pin 6
#define wave_count 60
#define color__combination GRB
#define chip WS2812B
#define Bright 200
#define Frames_per_second 60

CRGB leds[wave_count];// used to turn on a random lights

// initialize any needed LCD interface pin, bluetooth module pin, button switch, and LED.
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

//call each pin to the bluetooth adaptor
```

```
 String store_voice_string = "";

//used for button switch strings that will be combine and be able to print out one string in the lcd.print instead of multiple printouts:
String bl1 = "";

//used for the bluetooth adaptor strings that will be combine and be able to print out one string in the lcd.print instead of multiple printouts:
String ba1 = "";
//ba2 and ba3 will be used for both switch button and bluetooth adapter.
String ba2 = "";
String ba3 = "";

//assist in shifting words on the screen to the left
int toleft = 0;
int aidleft = 16;
//call a variable for the button switch
const int button_switch = 7;
int button_count = 0;
long random_number = 0;

//this will help change the led lights to different color pallete
CRGBPalette16 gPal;

//setup has changed to fit a neopixel led light.
void setup(){
  // set up the LCD's number of columns and rows and input:
  lcd.begin(16, 2);

 // setting up the random colors for Disco pattern
  delay(3000); // sanity delay
  FastLED.addLeds<chip, wave_pin, color__combination>(leds,
wave_count).setCorrection( TypicalLEDStrip );
  FastLED.setBrightness( Bright );
  gPal = HeatColors_p;

  //Button Switch:
  pinMode(button_switch, INPUT);

  //setting up the coloration for wave pattern and circular pattern
```

```
  FastLED.addLeds<NEOPIXEL, wave_pin>(leds, wave_count);

  // All outputs will be set up.

  Serial.begin(9600);

  randomSeed(analogRead(0));
  // Print a message that introduces the group(optional).
  lcd.print("Welcome to Voice Control Lights Blinking Cycle.");
}

//loop is not done yet please continue working on it.
void loop() {

 int readButton = digitalRead(button_switch);

//seeing if the bluetooth is connected and if it worked use the store_voice_string in void
voice_ledfrequency(int voice_recoded){}.
if(Serial.available ()){
  while (Serial.available()){
    delay(10);
    char c = Serial.read();
    if (c == '#'){break;}
    store_voice_string += c; // shorthand for voice = voice + c
  }
  lcd.setCursor(0,1);
  store_voice_string.toLowerCase();//makes the entire string have lower case character.
  ReceivingVocieFromPhone(store_voice_string);
  lcd.print(move_left(comblue()));
    }
else if (readButton == HIGH){
      readButton = LOW;
      button_count++;
      lcd.setCursor(0,1);
      blinkWave();

      lcd.print(move_left(comswitch()));
    }

  delay(1000);//delay can be modified
```

```
}

//-------------------------------------------------------------------------------------------

//Helper Functions:
//if the switch button selects this function then
//let the button_count be randomly selected to choose one of the three patterns.
void RandomCycle(){
 ba2 = "Random selects ";
  button_count = random(3);// will select a number between 0 through 2.

  // figuring out which one to turn on.
  if (button_count == 1){
    ba3 = "Wave pattern.";
    WaveCycle();
    }
  else if (button_count == 2){
    ba3 = "Circular pattern.";
    CircularWave();
   }
  else{
    ba3 = "Disco pattern.";
    DiscoWave();
  }
 button_count = 0;
}

//this function is called when the switch button has been pressed.
void blinkWave(){
//depending on the button_count determines which of the four wave patterns, circular
pattern, disco pattern or random selection pattern will be activated.

 //Depending on the button_count, determines which one will be turned on.
   if (button_count == 1){
   bl1 = " turn on Wave pattern.";
   WaveCycle();
    }
  else if (button_count == 2){
    bl1 = " turn on Circular pattern.";
    CircularWave();
```

```
   }
  else if (button_count == 3){
    bl1 = " turn on Disco pattern.";
    DiscoWave();
  }
  else{
    bl1 = " give random a chance to select which one will turn on.";
    RandomCycle();
  }
}


// Depending on our command issued and the frequency of our voice either one of the
//four WaveCycle(), CiccularCycle(), DiscoCycle() or RandomCycle(), will be called.
void ReceivingVocieFromPhone(String voice_recoded){
  //this will turn the wave pattern on.
  if (store_voice_string == "*turn on wave"){
    WaveCycle();

            ba1 = " Wave pattern!";
  }
  //this will turn the circular pattern on.
    else if(store_voice_string == "*turn on circular"){
    CircularWave();

    ba1 = " Circular pattern!";
  }
  //this will activate the disco pattern.
    else if(store_voice_string == "turn on disco"){
    DiscoWave();
    ba1 = " Disco pattern!";
  }
  //this will activate a random pattern.
  else if(store_voice_string == "turn on random"){

    ba1 = " a random pattern! ";
    RandomCycle();
  }
    ba1 = " not a command try again.";
}
```

```
//this function will create a string that will be used in printing out a message on what was selected after our voice frequency command turns on one of the three selections.
String comblue(){
  String combine = "";

  //test to see if ba1 is the string for random selection.
  if (ba1 == " a random pattern! "){
    combine = "Voice frequency detected and will power up " + ba1 + ba2 + ba3;
    }
   else{
    combine = "Voice frequency detected and will power up " + ba1;
    }
    return combine;
  }

 //this function will create a string that will be used in printing out a message on what was selected after the button switch was pressed.
String comswitch(){
  String combine = "";

  //test to see if ba1 is the string for random selection.
  if (bl1 == " a random pattern! "){
    combine = "Switch button has the power to " + bl1 + ba2 + ba3;
    }
   else{
    combine = "Switch button has the power to " + bl1;
    }
    return combine;
  }


//helps display messages by scrolling to the left.
String move_left(String personal){

  String shift = personal + "";//used to store in the string and create an ending.

  //shifting the characters starting from zero all the way to the end.
  toleft++;
  aidleft++;
```

```
  int strlength = shift.length();
  if (toleft > strlength){
    toleft = 0;
    aidleft = 16;
    }
  return (shift.substring(toleft, aidleft));
  }
//-------------------------------------------------------------------------------------------------------------

//Strip activation functions:

//Splitting the RGB 60 LED lights into two groups.examples found
// In the FastLed library made by Daniel Garcia.
//The Mirroring method has been used in both the WaveCycle2() and CircularWave2().

//This function will test the first 30 neopixel lights in a wave pattern.
void WaveCycle(){

  int j = 153;//Alters Green
  int p = 255;//Alters Red
  int q = 51;// Alter Blue
  int i = 0; //Random color selection


  //This while loop will help in altering the color per each led light on the strip.
  while(i < 254){
      p = p + i;
      j = j - i;
      q = q + i;

      //testing each variable to make sure they are not out of bounce.If they are we will
rest them to default.
      if (i > 254){
        i = 0;
        }
      if (j < 0 ){
        j = 255;
        }
     if (q > 254){
        q = 0;
```

```
      }
    if (p > 254){
       p = 0;
       }

    //this for loop is forward.
    for(int t = 0; t < 30; t++) {
     // setting up our current led light
      i++;
      p = p + i;
      j = j - i;
      q = q + i;

      leds[t] = CRGB((p),(j),(q));
      FastLED.show();
      // clear our current dot before we move on
      leds[t] = CRGB::Black;
      delay(1000);
    }

    //this for loop is to go backward.
      for(int t = 29; t > -1; t = t - 1) {
      // setting our current led light.
      i++;
      p = p + i;
      j = j - i;
      q = q + i;

      leds[t] = CRGB((p),(j),(q));
      FastLED.show();
      // clear our current dot before we move on
      leds[t] = CRGB::Black;
      delay(1000);
    }

  }
}

  //This function will assist to test the  last 30 neopixel lights
  //in a circular pattern.
```

```
void CircularWave(){
int j = 153;//Alters Green
int p = 255;//Alters Red
int q = 51;// Alter Blue
int i = 0; //Random color selection


//This while loop will help in altering the color per each led light on the strip.
while(i < 254){
    p = p + i;
    j = j - i;
    q = q + i;

    //testing each variable to make sure they are not out of bounce.If they are we will rest them to default.
    if (i > 254){
      i = 0;
      }
    if (j < 0 ){
      j = 255;
      }
   if (q > 254){
      q = 0;
     }
  if (p > 254){
      p = 0;
      }

  //this for loop is forward.
  for(int t = 30; t < 60; t++) {
   // setting up our current led light
    i++;
    p = p + i;
    j = j - i;
    q = q + i;
    leds[t] = CRGB((p),(j),(q));
    FastLED.show();
    delay(100);
  }
  //this for loop is forward.
```

```
  for(int t = 30; t < 60; t++) {
   // setting up our current led light
    leds[t] = CRGB::Black;
    FastLED.show();

  }

  //this for loop is to go backward.
    for(int t = 59; t > 29; t = t - 1) {
    // setting our current led light.
    i++;
    p = p + i;
    j = j - i;
    q = q + i;

    leds[t] = CRGB((p),(j),(q));
    FastLED.show();
    delay(100);
  }
  //this for loop is backwards.
  for(int t = 59; t > 29; t = t-1) {
   // setting up our current led light
    leds[t] = CRGB::Black;
    FastLED.show();

  }
 }
 }

 // for DisoWave()we will be using the same method as the other two with an added
while loop instead.
 void DiscoWave(){

  int i = 0; // the changer of lights
  int p = 153; // holds the Green positions.
  int j = 255; // holds the Red postions.
  int q = 51; // holds the Blue positions.


  //this while loop will change the color per iteration.
```

```
    while (i < 255){

      p = p + i;
      j = j - i;
      q = q + i;
      //testing each variable to make sure they are not out of bounce.If they are we will
rest them to default.
      if (i > 254){
        i = 0;
        }
      if (j < 0 ){
        j = 255;
        }
     if (q > 254){
        q = 0;
       }
    if (p > 254){
        p = 0;
        }

    //using the for loop to light up and turn off all 60 led lights.
      for(int t = 0; t < 60; t++) {
      // set each light to the same pattern
        leds[t] = CRGB(p,j,q);
      }
      FastLED.show();
      //this for loop will turn all the lights off.
       for(int t = 0; t < 60; t++) {
      // set each light to the same pattern
        leds[t] = CRGB::Black;
      }
      i++;
      delay(1000);//delay for a second
    }
  }
```

List of Materials Expected to be Needed:

To receive most of these items purchase an ELEGOO UNO Project Super Starter Kit with Tutorial and UNO R3 Compatible with an Arduino IDE that contains a solder LCD display.

1. One Arduino Uno R3.
2. 16 LED lights 4 different colors.
3. Hook up wires.
4. An LCD display
5. 3 or more breadboards
6. 2 or more switches
7. 2 or more 10 k resistors, used for the switches.
8. WS2812B 5050 60 RGB LED strip.
9. WESIRI 93 LEDs 6 Rings WS2812B 5050 RGB LED Ring Lamp Light Individually Addressable Full Dream Color DC5V with Mini Controller.
10. Bluetooth adaptor
11. 2 Female to Male Pupont Wire (10pcs). We will be using only 4 to 5 wires.
12. USB power cable.
13. 10k ohm potentiometer
14. The app “BT Voice Control for Arduino”
15. An Android device
16. Aclorol 5V 8A 40W Power Supply AC/DC Power Adapter AC 100~240V to DC 5volt 8amps Converter Transformer 5.5X2.5mm Plug
17. 2 220 resistors .
18. A set of jumper cables for power and ground. You may use any color if not red and black jumper cables are preferred.
19. A tiny screwdriver for the screws on the convertor transformer.
20. A wire trimmer or scissors

Timeline:

Note: Each part of the code will be complete separately, depending on the date provi.l`m ded.

- Part 1 of the project and code connect and test the lcd properly Due on Sunday 10/25/2020

- Part 2 of the project and code is to connect the bluetooth adaptor on a second breadboard and make sure our voices work, using the app the

frequency on our android devices and make sure the LCD screen is able to read our voice frequency.Due on Sunday 11/15/2020.

- Design Presentation on Monday 11/23/2020

- Part 3 of the project and code connect the LED lights to the third breadboard and connect the switch button properly. Test to see if the switch works correctly. Due Sunday 11/29/2020.

- Final Report and Final Design Document Friday 12/04/2020

- Team Work Assessment - done individually, critique of group performance, submitted via web form before Monday of week 16, 12/07/2020 at 11:59pm.

Our group would like to mention a couple of things that we used primarily in our code.

Labs

- The Libraries used in Lab 2-Input and Output
    - Used to understand how to use the switch function
- The Libraries used in Lab 3-Scrolling output
    - For usage of LCD 16 x 2 display and how to create shift to the left function. We used the processes of creating shifts to the left to display our strings as each word shifted to the left.
- The Libraries used in Lab 7 - Serial Communication
    - Used to understand how TX and RX, from both the Bluetooth adaptor and Arduino Uno Device are connected.
- The Libraries used in Lab 9 - Graphing Sensor data on a PC
    - Used to combine multiple strings to one large string.

Usable source

- The main usable source that our group found helpful was FASTLED created by Daniel Garcia. His Libraries were very useful on how to use our RGB 60 LED light strip. The example we used to inspire use, in creating WaveCycle(), CircularCycle() and DiscoCycle(), was Mirroring.

References:

1. Admin, Arduino. “LCD Arduino Tutorial - How to Connect LCD with Arduino.” *Mechatrofice*, 12 Oct. 2020, mechatrofice.com/arduino/lcd-matrix-interface.

2. Arduino.cc. 2019. *Arduino - Softwareserial*. [online] Available at: <https://www.arduino.cc/en/Reference/SoftwareSerial> [Accessed 5 October 2020].

3. Arduino.cc. 2018. *Button*. [online] Available at: <https://www.arduino.cc/en/Tutorial/BuiltInExamples/But ton> [Accessed 5 October 2020].

4. Arduino.cc. 2018. *Scrolldisplayleft() And Scrolldisplayright() Methods*. [online] Available at: <https://www.arduino.cc/en/Tutorial/LibraryExamples/LiquidCrystalScroll> [Accessed 5 October 2020].

5. .Arduino Project Hub. 2020. *Arduino Frequency Counter With 16×2 LCD Display*. [online] Available at: <https://create.arduino.cc/projecthub/jasirtp/arduino-frequency-counter-with-16x2-lcd-display-c99779?ref=similar&ref_id=61668&offset=5> [Accessed 5 October 2020].

6. Burgess, Phillip. “Adafruit NeoPixel Überguide.” *Adafruit Learning System*, 30 Aug. 2013, learn.adafruit.com/adafruit-neopixel-uberguide/arduino-library-use.

7. Garcia, Daniel. “FastLED/FastLED.” Edited by Mark Kriegsman, *Mirroring Sample* , GitHub, 4 Sept. 2016, github.com/FastLED/FastLED/wiki/Multiple-Controller-Examples.

8. Ladyada.net. 2020. *Arduino Tutorial - Lesson 5*. [online] Available at: <http://www.ladyada.net/learn/arduino/lesson5.html> [Accessed 5 October 2020].

9. Sabaa, Y., 2015. *Bluetooth Control Led With Lcd Led Status Display Real Time.*. [online] Arduino Project Hub. Available at:

<https://create.arduino.cc/projecthub/YoussefSabaa/bluetooth-control-led-with-lcd-led-status-display-real-time-58f9ca?ref=similar&ref_id=57135&offset=5> [Accessed 5 October 2020].

10. Team, The Arduino. “String Case Change Functions.” *Arduino*, 11 Aug. 2015, www.arduino.cc/en/Tutorial/BuiltInExamples/StringCaseChanges.