REPORT

Music Controlled LEDs

Team Members:

1. Somil Dhiman
2. Akriti Jain
3. Dev Srivastava
4. Ravi Prakash
5. Ayan Shree

Mentors:

1. Anjali Rana
2. Avichal Gupta

Objective:

We planned to make a 3-Dimensional 4x4x4 Music Controlled LED Cube. Our objective was to programme the circuit for various intensity values. The circuit would detect these values from a sound/music source through a sensor and light different LEDs governing cool patterns.

Components Required:

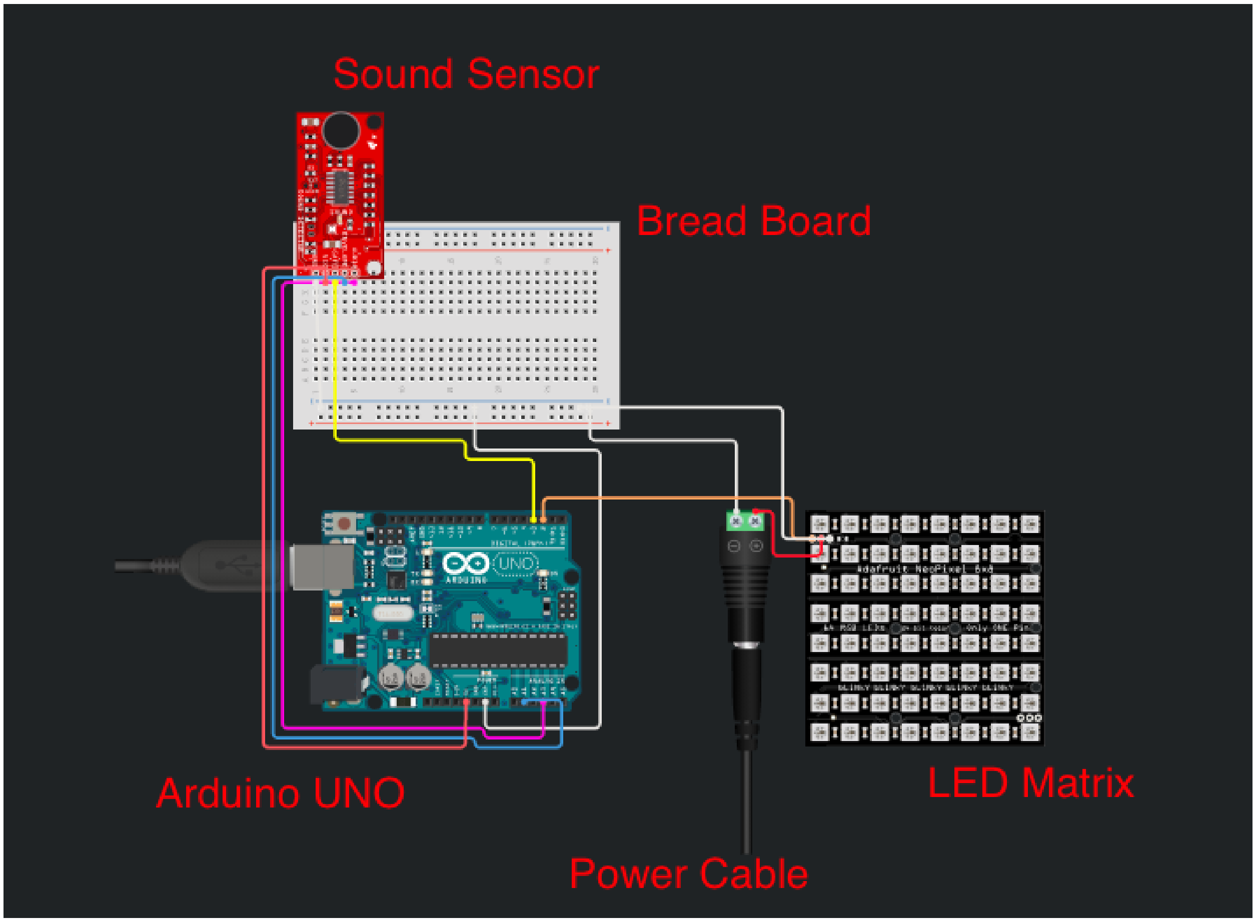
- **Arduino MEGA** -Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

- **Connecting wires** -Connecting wires allows an electrical current to travel from one point on a circuit to another because electricity needs a medium through which it can move.

- **LEDs**(Light-emitting diode)- A semiconductor diode that emits light when conducting current and is used in electronic displays.In this project we would be using 64 LEDs.

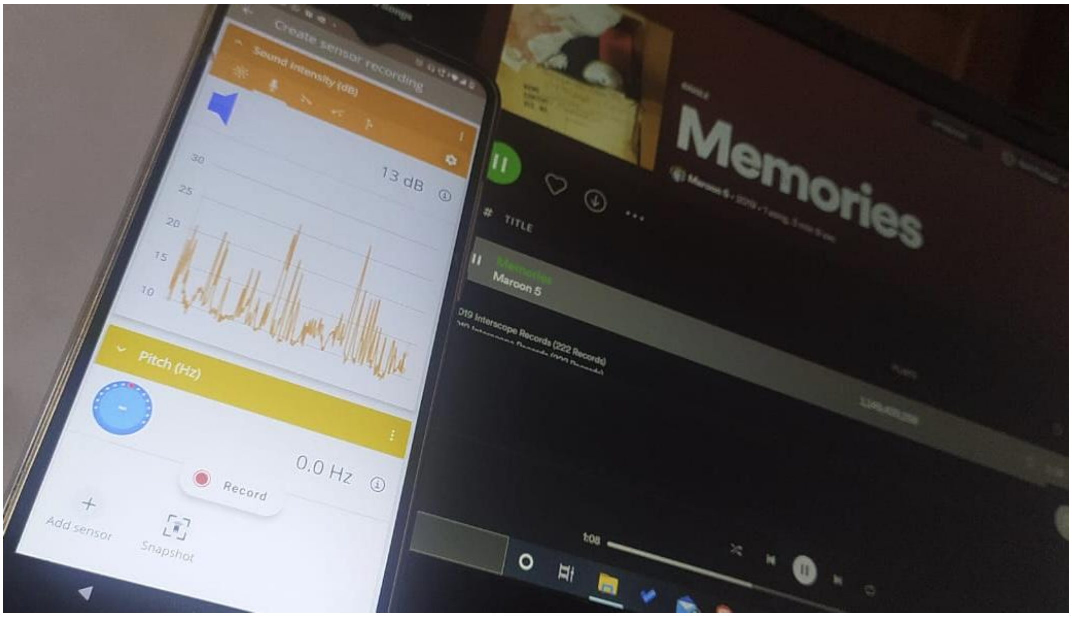
- **Sound Sensor**-A sound sensor is defined as a module that detects sound waves through its intensity and converts it to electrical signals. We were not able to simulate a sound sensor, but would be using a KY-037 sound sensor, if we are given an opportunity to present our model live.

- **Resistor**- Resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses.In this project we would be using 4 resistors of 150 kilo ohm as resistance each.


Sound Sensor
Bread Board
Arduino UNO
Power Cable
LED Matrix

Road Map:

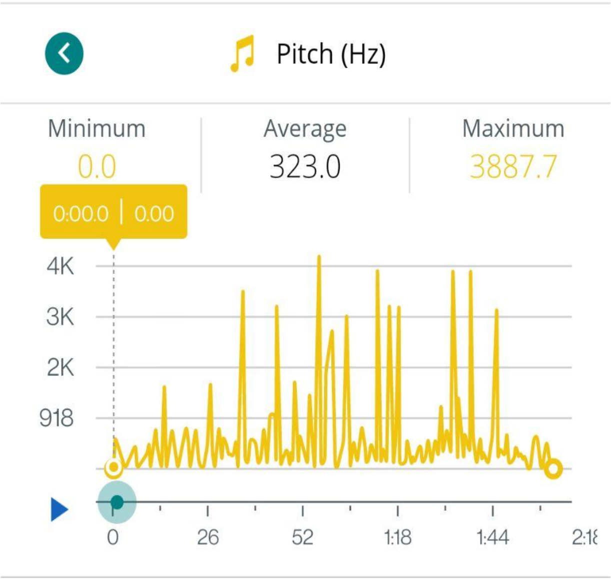
- **Collection of data sets:**
    - Due to the lack of physical sound sensors, we collected raw data using the Arduino Science Journal app. This involved recording 25 different songs and measuring their pitch and intensity.



    - Next, we analysed all those data sets and tried to figure out the values that we would use as the test cases for the incoming new data in the circuit.

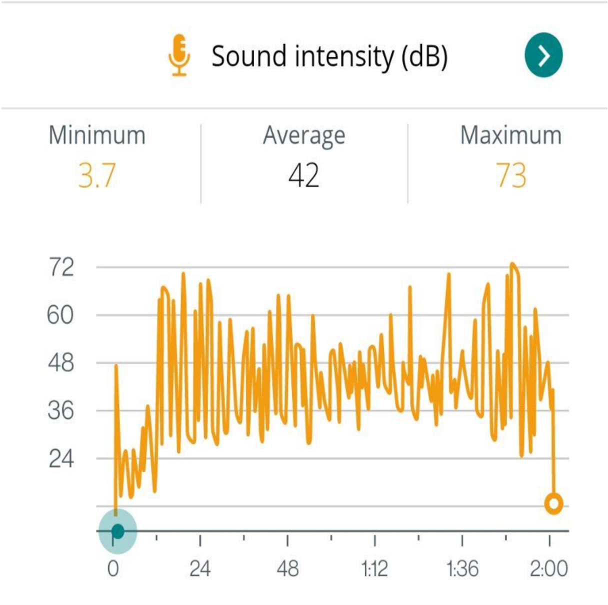
    - For this, we calculated mean, median and standard deviation in excel.

[Click here](#) to access our data sets



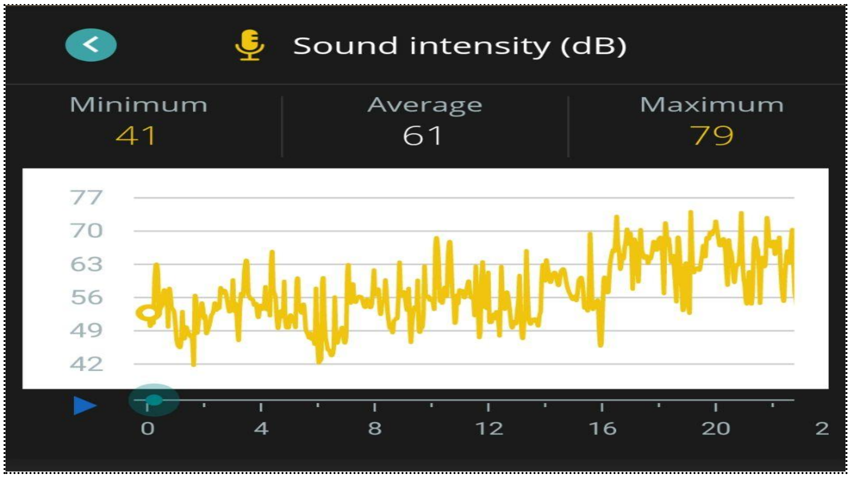
Pitch (Hz)

Minimum	Average	Maximum
0.0	323.0	3887.7



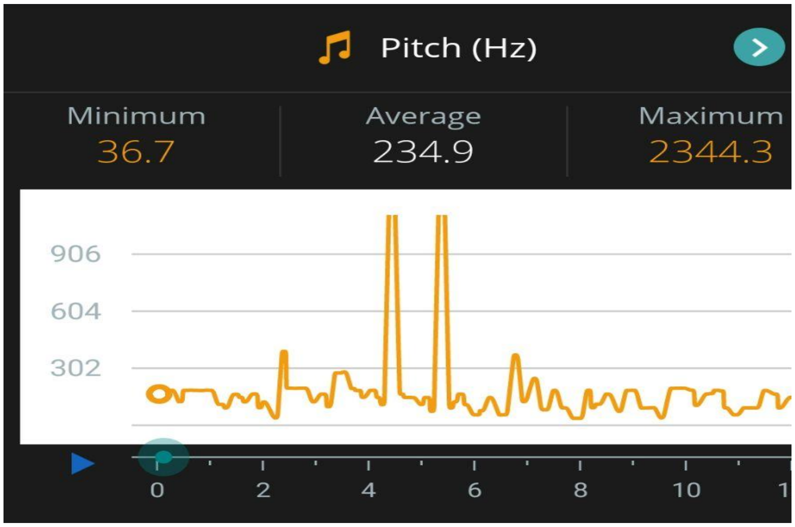
Sound intensity (dB)

Minimum	Average	Maximum
3.7	42	73



Sound intensity (dB)

Minimum	Average	Maximum
41	61	79



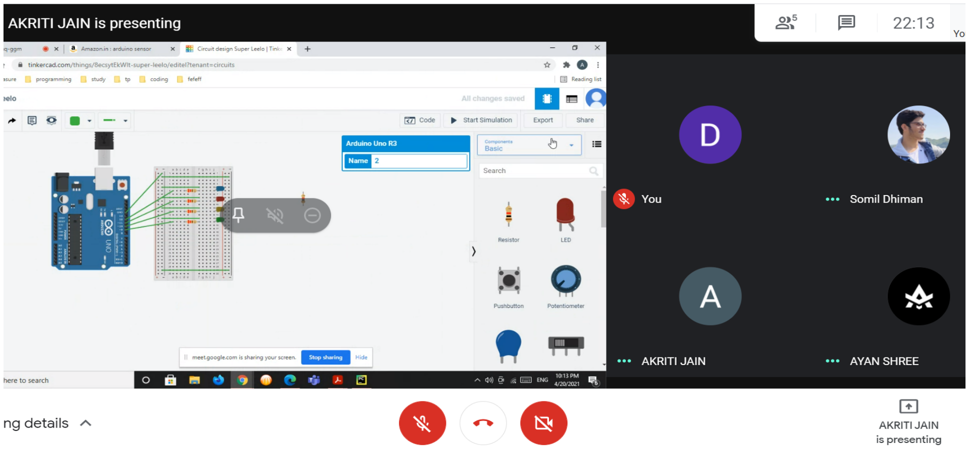
Pitch (Hz)

Minimum	Average	Maximum
36.7	234.9	2344.3

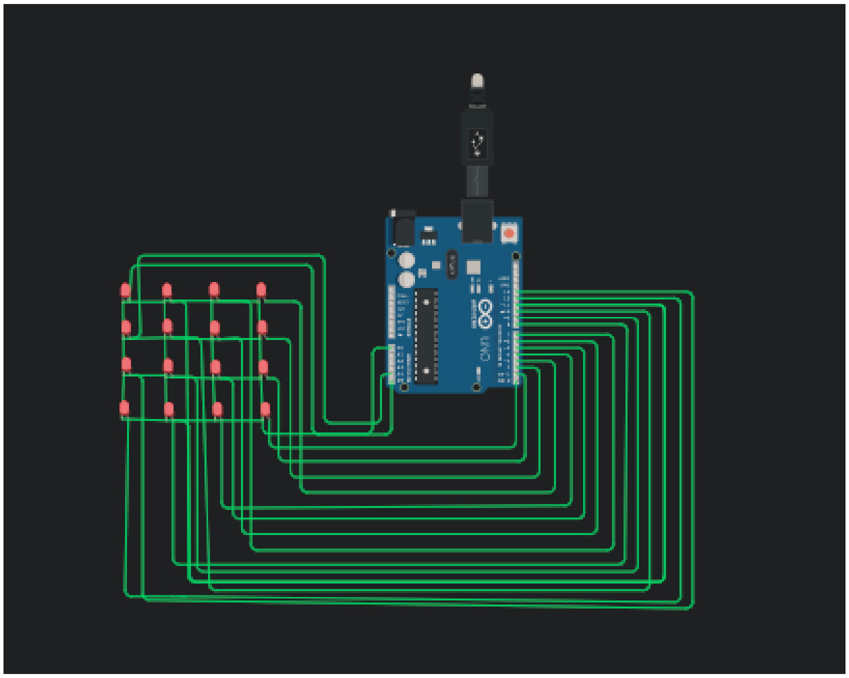
- ## Circuit Simulation:
  - For designing and simulating the circuit, we used the Tinkercad Circuit Simulator.

Steps Taken:

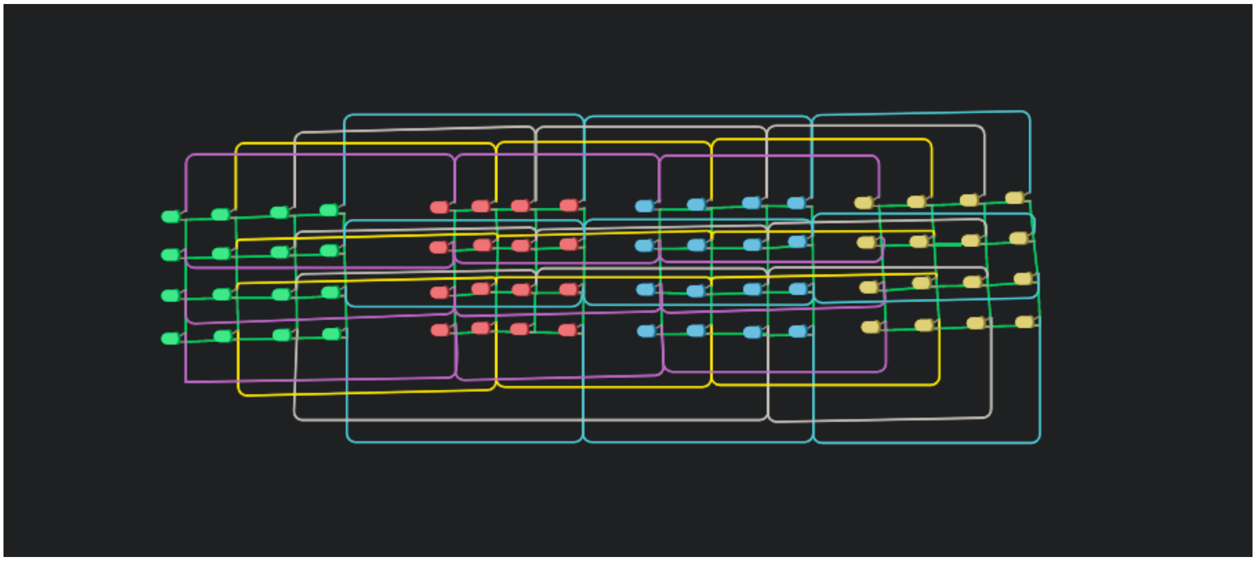
- At first we made a 4x4 layer of LEDs and connected all their cathodes together so that they have a common cathode and the LEDs can be controlled individually.

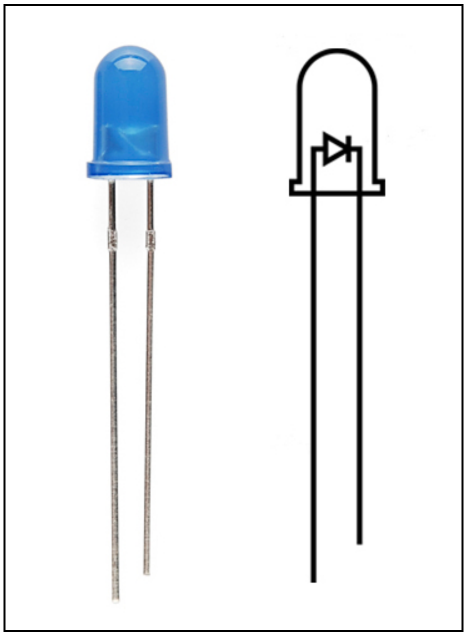


- Next, we connected the common cathode and the anodes to the digital and analog pins of the arduino board.
- This gave us control over all the LEDs in a single layer.

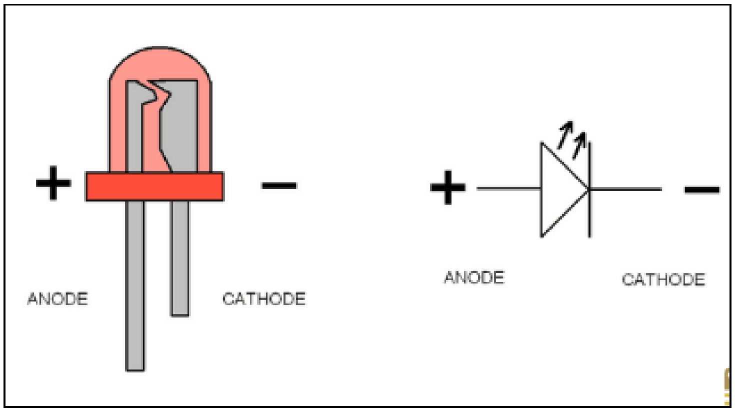


- Following this, we extended our circuit to four successive layers, in order to make a cube.
- We connected individual stacks/columns of LEDs from anode to anode.





As we know, LED is a semiconductor device that emits light when current flows through it. It is a special type of p-n junction diode which operates in the forward biased region.



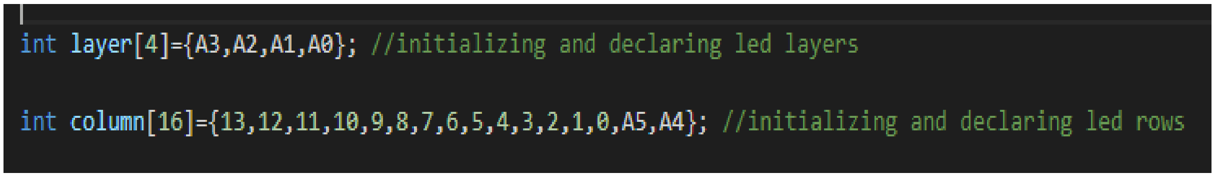
For the LED to glow, the potential of anode should be more than the cathode. Since the 4 layers are governed by the cathode and the 16 columns are denoted by anode, so as evident to turn the LED on, we keep the layer LOW and the column HIGH to complete the circuit and ensure current flow.

- **Working on the Code:**
  - We plan to write the code in such a way that the intensity input governs the 3-dimensional glowing of the circuit.

- We also plan to explore various patterns with various intensity inputs to make the circuit more visually appealing.

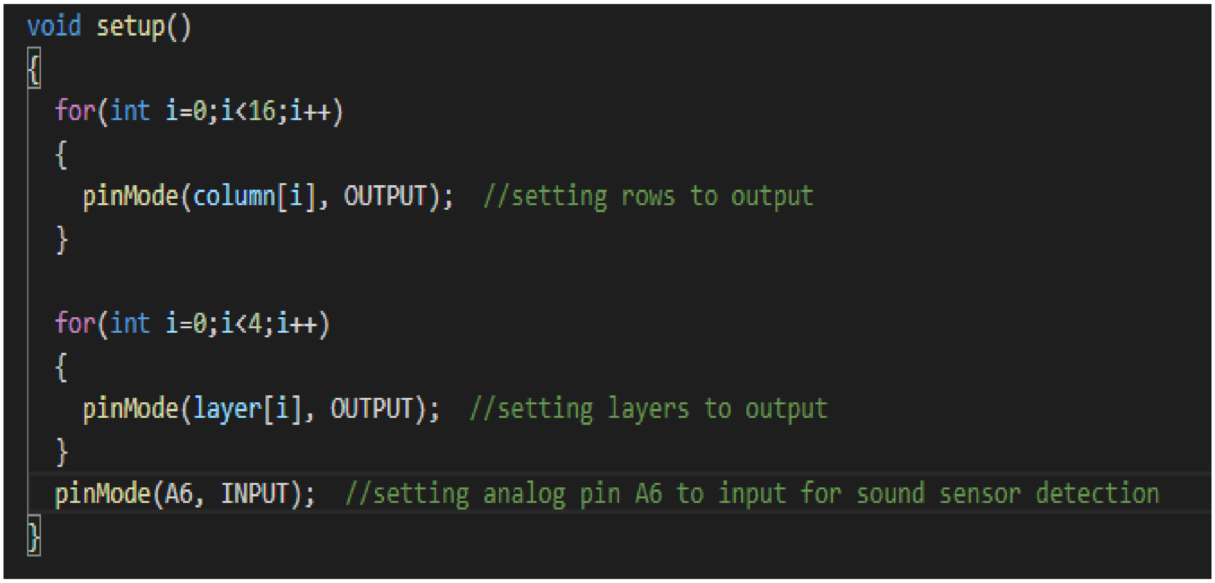
- The code will be written in Arduino IDE.

- **Explanation:**



```
int layer[4]={A3,A2,A1,A0}; //initializing and declaring led layers

int column[16]={13,12,11,10,9,8,7,6,5,4,3,2,1,0,A5,A4}; //initializing and declaring led rows
```

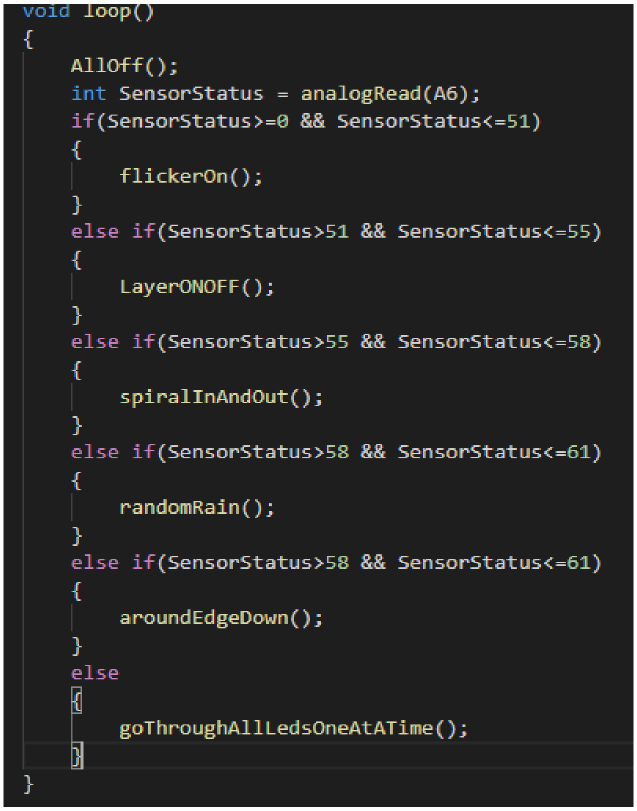
We have declared two arrays namely “layer” and “column”. It's being used to indicate which Arduino pin the LED is connected to.



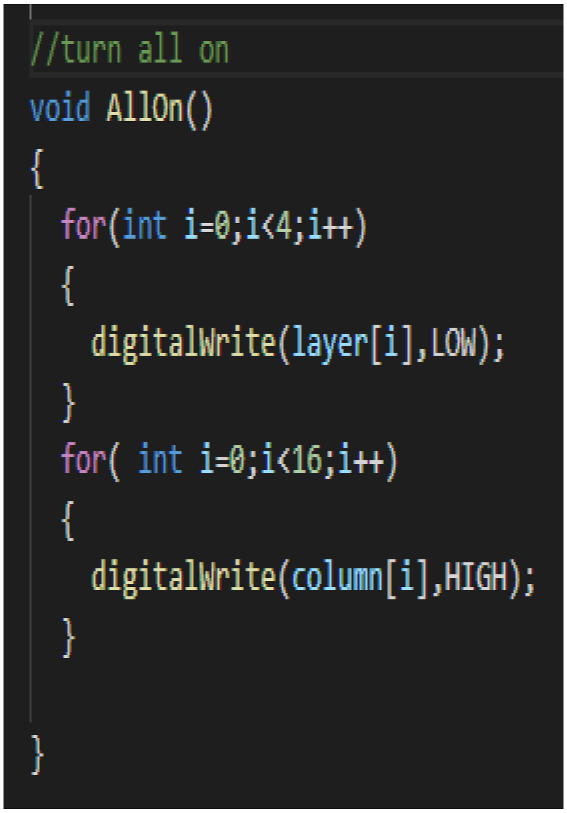
```
void setup()
{
  for(int i=0;i<16;i++)
  {
    pinMode(column[i], OUTPUT);  //setting rows to output
  }

  for(int i=0;i<4;i++)
  {
    pinMode(layer[i], OUTPUT);  //setting layers to output
  }
  pinMode(A6, INPUT);  //setting analog pin A6 to input for sound sensor detection
}
```

The setup() function is triggered only once in the entire code run and here we are using it to initialize pin modes.

The pinMode() function configures the specified pin to behave either as input or output.



```
void loop()
{
    AllOff();
    int SensorStatus = analogRead(A6);
    if(SensorStatus>=0 && SensorStatus<=51)
    {
        flickerOn();
    }
    else if(SensorStatus>51 && SensorStatus<=55)
    {
        LayerONOFF();
    }
    else if(SensorStatus>55 && SensorStatus<=58)
    {
        spiralInAndOut();
    }
    else if(SensorStatus>58 && SensorStatus<=61)
    {
        randomRain();
    }
    else if(SensorStatus>58 && SensorStatus<=61)
    {
        aroundEdgeDown();
    }
    else
    {
        goThroughAllLedsOneAtATime();
    }
}
```



```
//turn all on
void AllOn()
{
  for(int i=0;i<4;i++)
  {
    digitalWrite(layer[i],LOW);
  }
  for( int i=0;i<16;i++)
  {
    digitalWrite(column[i],HIGH);
  }

}
```

Next, the loop() function is used to actively control the arduino board and as the name suggests, this function runs consecutively.

Firstly, the AllOff() function is called that turns all the LEDs off.

The function analogRead() reads the value from the specified analog pin. This means that it will map input voltages between 0 and the operating voltage(5V) into integer values between 0 and 1023. The analog output voltage changes with the intensity of sound received by the microphone of the sound sensor. We will connect this output to Arduino analog pins and process the output voltage

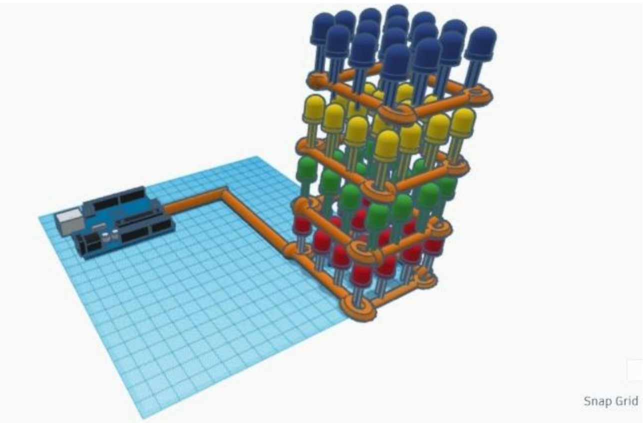
This value is then stored in the variable SensorStatus.

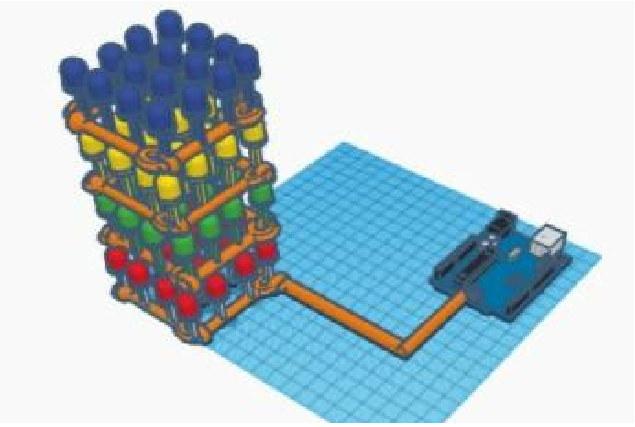
Furthermore, we have used multiple if-else statements to trigger various patterns in accordance with the intensity ranges that we had gathered manually by recording songs.

- **Challenges:**
  - First of all, executing a completely hardware related project in online mode was itself the biggest challenge in execution of this project.
  - No Sound sensor was available online in the simulators, thus we had to pre-define the input variables.
  - Initially we had a plan to include both pitch and intensity sensors in our project. But there was no physical pitch sensor available in the repository. Thus, we had to drop the plan.
  - Due to lack of existing datasets we had to manually analyse the collected data and find out the relevant ranges.

- **Future Aspects**
  - In this project, we were unable to add the sound sensor KY-037, so we aspire to add the same in our project, if available to us in hardware;.
  - We would like to include a pitch sensor too in the project.
  - We would also like to include Wi-Fi backed input.

Following is a snapshot of our circuit:





After repeated Testing and improvements, we made a smooth-running project that is both electronically efficient and visually attractive.