Wireless Sensor Networks
Project Report

CATNAP

GROUP 4
Atish Majumdar
Prasanna Natarajan
Vedant Chakravarthy

DOMAIN

Our project lies in the domain of smart homes, as it monitors disturbances in light and sound while one is sleeping and provides relevant information. When sound increases, the system's volume of white noise also increases. This works best with bluetooth earphones.

OBJECTIVE

Noise, defined as unwanted sounds, could affect people both psychologically and physiologically, with several reported negative effects.[1] Sound-absorbing treatment is a relatively effective noise reduction strategy, whereas sound masking appears to be the most effective technique for improving sleep. This is what we wish to address.The main objective of this project is to create a device which will aid in sleeping better by taking measures to curb noise disturbances using white noise (creates a masking effect, blocking out those sudden changes that frustrate light sleepers, or people trying to fall asleep) as well as monitor any changes in light intensity which might affect sleep. Our focus is on giving the user graphs (statistics) which would allow them to better choose a place to do the above mentioned. We use an analog sound sensor because we do not want to infringe privacy, hence no recordings take place, only the decibel level is noted (high or low amplitude).

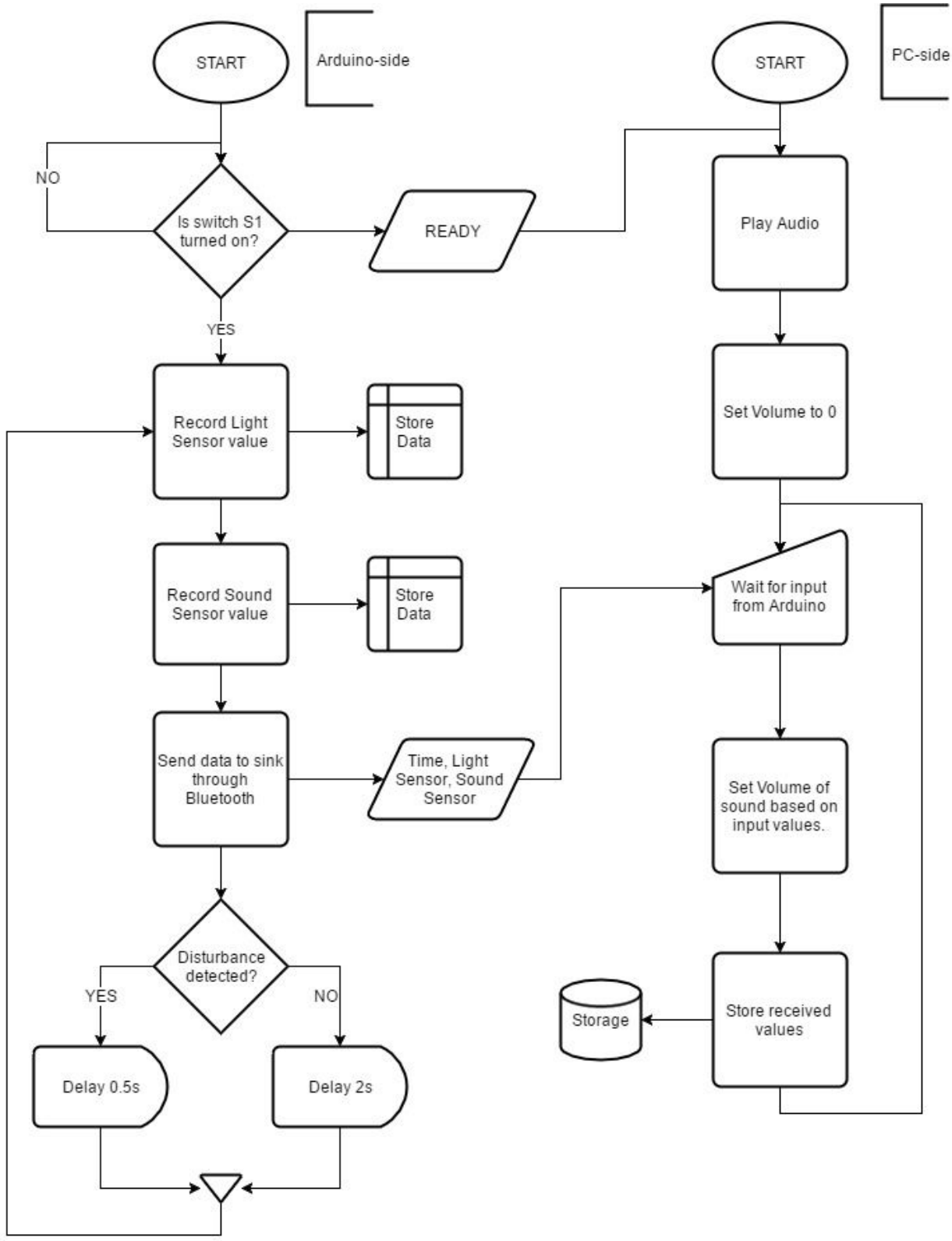
Comparison of the effectiveness of noise reduction strategies.

	Intervention			
Outcome	Earplugs	Behavioural modification	Sound masking	Acoustic absorption
Average noise level reduction	NA	2.7dBA	NA	3.6 dBA
Average sleep improvement	25.3%	16.1%	42.7%	NA

NA, not applicable.

[1]

FLOWCHART


START
Arduino-side
NO
Is switch S1 turned on?
READY
YES
Record Light Sensor value
Store Data
Record Sound Sensor value
Store Data
Send data to sink through Bluetooth
Time, Light Sensor, Sound Sensor
Disturbance detected?
YES
NO
Delay 0.5s
Delay 2s
START
PC-side
Play Audio
Set Volume to 0
Wait for input from Arduino
Set Volume of sound based on input values.
Store received values
Storage

DIFFERENT AREAS OF APPLICATION

1. *Sleep:* When sleeping, bluetooth earphones/ speakers will be used to play white noise, which would aid in sleeping [1].Per day statistics for the user, to see how sleep patterns are. (Light and sound readings)

2. *Office:* Research[3] shows that sound in workspaces damage employee's attention spans, stress levels, short-term memory, productivity, creative thinking, and satisfaction.

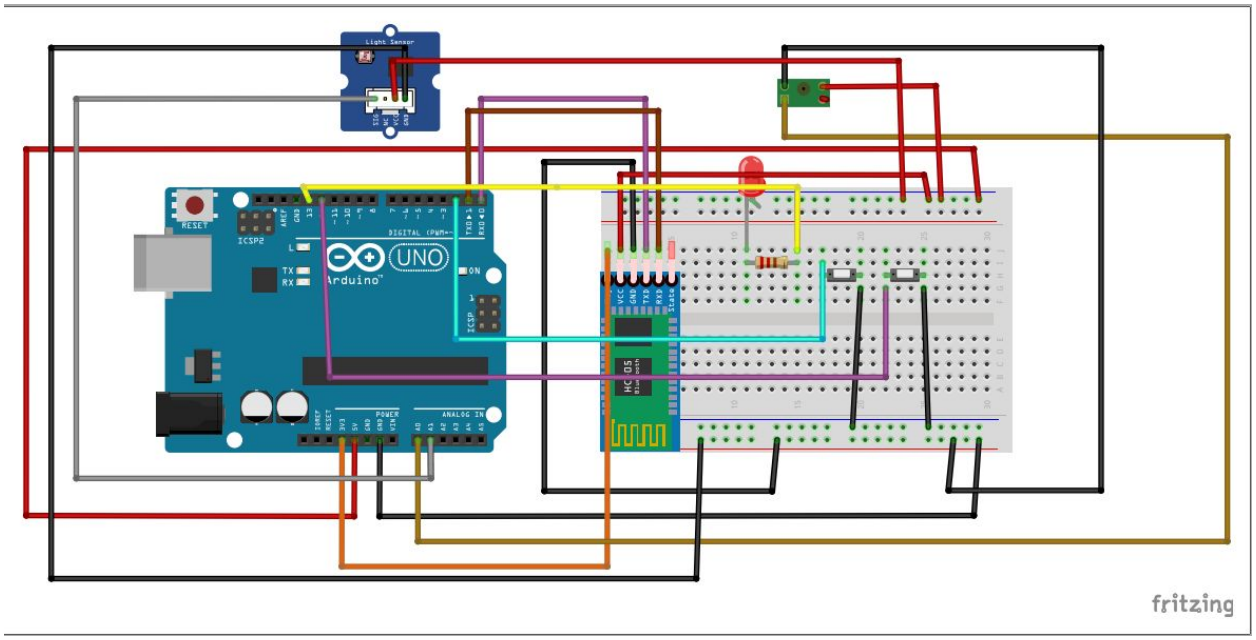
3. Employee desks, optionally plug in earphones to have mediated audio output of choice (white noise/ beats)The sensor could also be placed near the water fountains, where it can have white noise playing near based on chatter volume, to minimize gossip time. The light sensor can be used to track optimal working conditions, such that the amount of light is just right, least amount of glare and strain on the eyes of the employees.

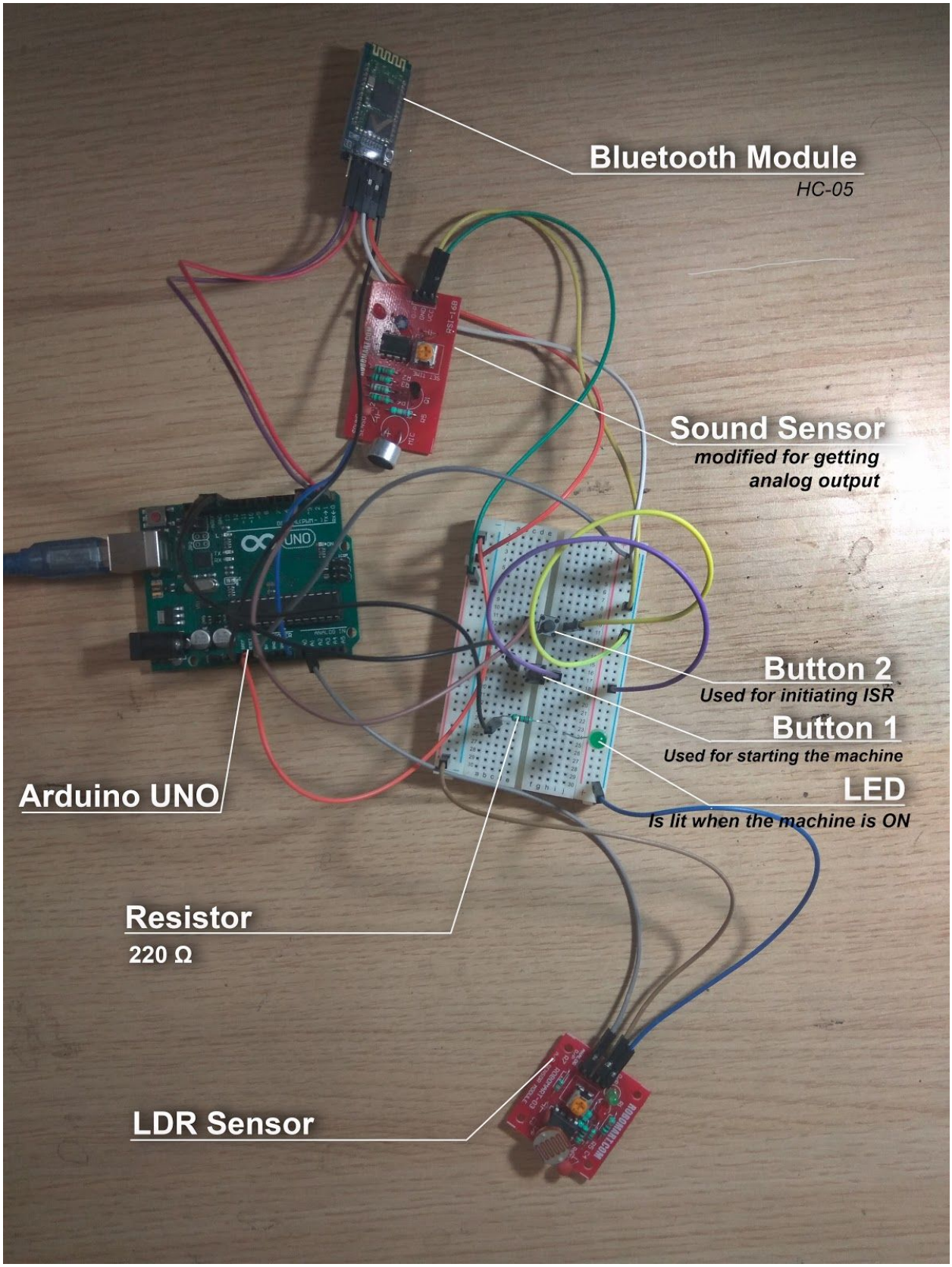
4. *Study:* The device could be placed near the study table, and optionally, earphones can be used to keep distractions at bay. The light sensor would aid in finding an optimally lit area in the house to study, to enable maximum efficiency during studying.

PROJECT COST

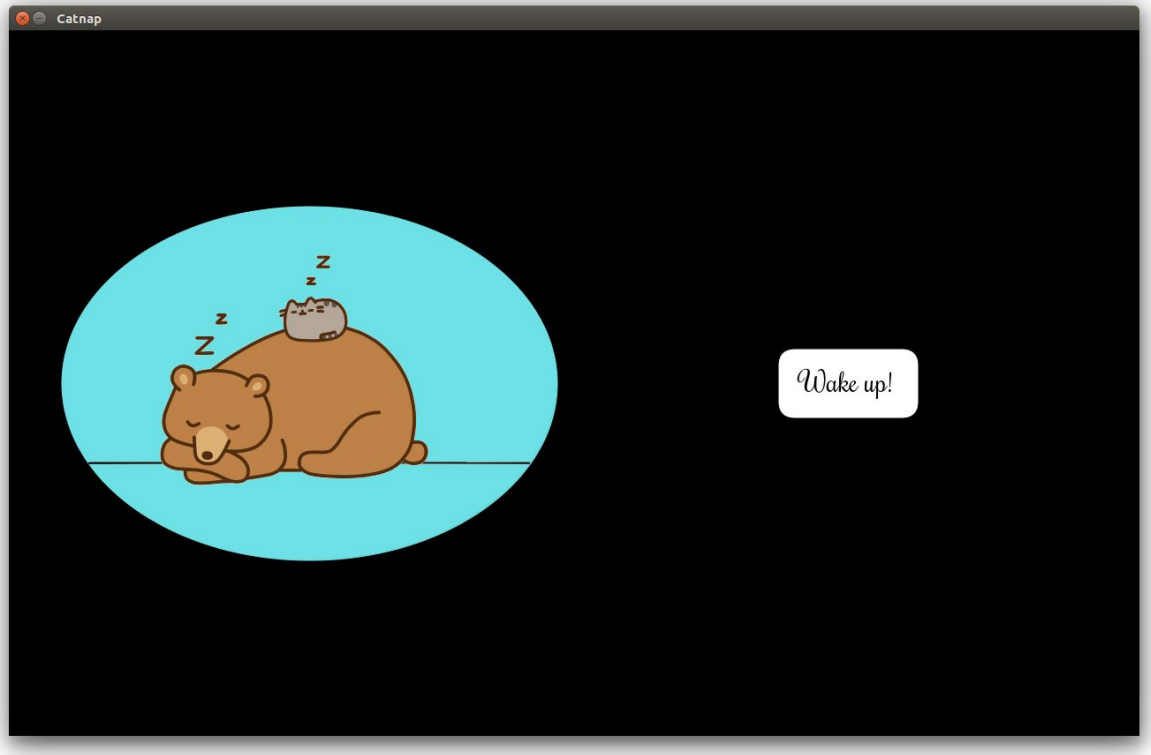
Sensor/ Component	Description	Quantity	Price (₹)
Arduino UNO microcontroller	Main board to control all operations	1	450
Sound Sensor	Sensing of sound in the environment	1	95
LDR Sensor	Sensing light in the environment	1	140
LEDs	Used to check proper functioning of other sensors and modules	Pack	70
220 Ω Resistors	To regulate current flow in the circuit	Pack	13
Jumper Wires	Connections	Pack	150
9 Volt Battery	To wirelessly power Arduino board	1	30
HC-05 Wireless Bluetooth RF Transceiver Module Serial/TTL/RS232 for Arduino	Bluetooth Module to wirelessly communicate with the Arduino	1	350
Extra	Shipping Costs	-	300
Total Cost	**Overall Project Cost**	**-**	**1598**

SNAPSHOT OF CIRCUIT
BREADBOARD VIEW




Bluetooth Module
HC-05
Sound Sensor
modified for getting
analog output
Button 2
Used for initiating ISR
Button 1
Used for starting the machine
LED
Is lit when the machine is ON
Arduino UNO
Resistor
220 Ω
LDR Sensor

UI SCREENSHOTS



GRAPH:

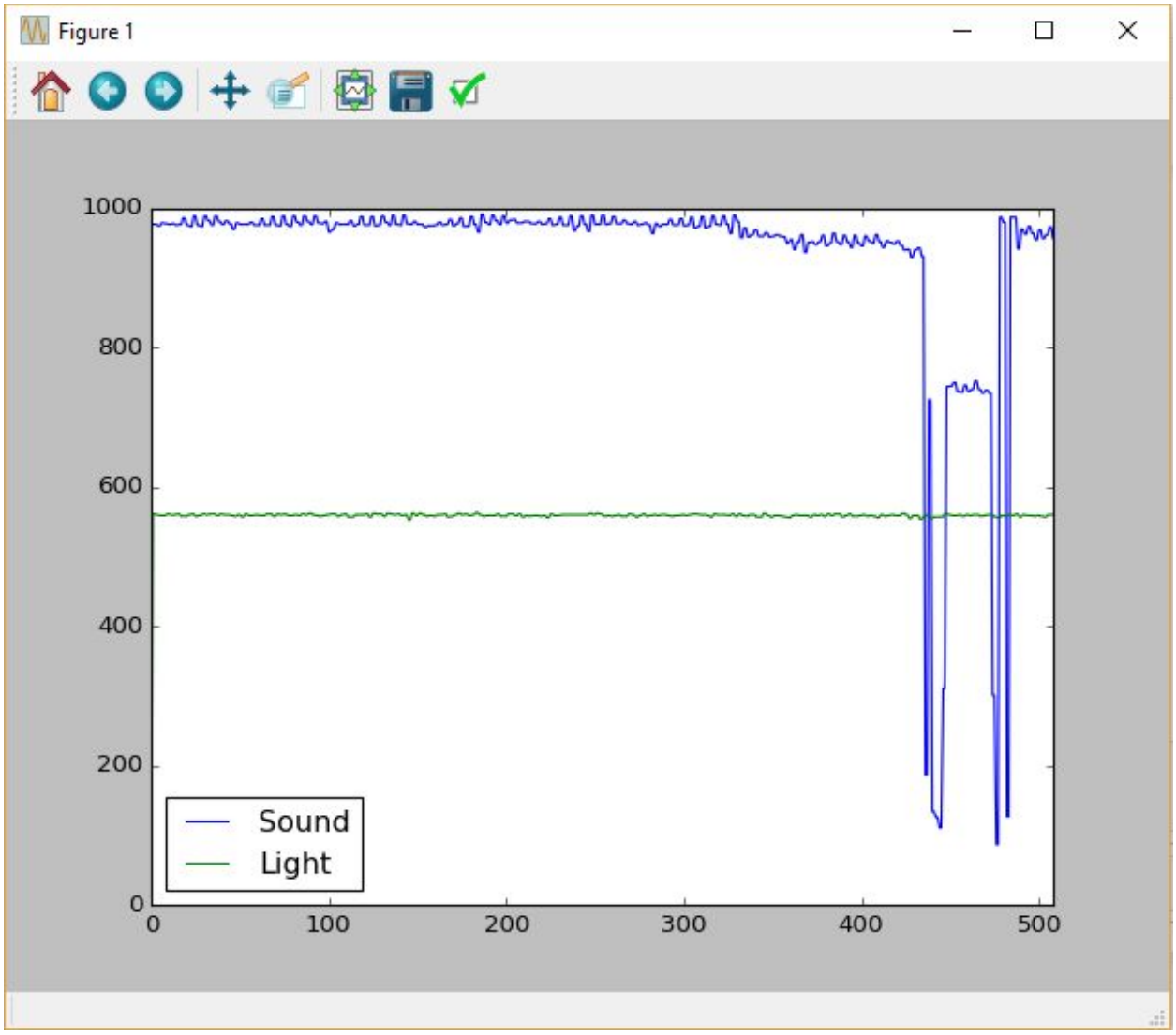


Figure 1
1000
800
600
400
200
0
0
100
200
300
400
500
Sound
Light

ARDUINO CODE:

```
/*
 This is our submission of the source code of the project for Wireless Sensor Networks
(CSD337), which consists of implementation of Catnap, a sleep assistant.
 The code has been written in Arduino IDE and Processing.

 AUTHORS
 Atish Majumdar      : 1410110081
 Prasanna Natarajan  : 1410110298
 Vedant Chakravarthy : 1410110489
*/

// defining pins
#define ledPin 13
#define soundPin A0
#define lightPin A1
#define buttonPin 12
#define bufferSize 10
#define button2Pin 2

// global variables
int state = 0;
int sound = 0;
int light = 0;
int first = 0;
int threshold = 2;
int flag = 0;
int delayMultiplier = 1;

volatile int soundMedian;
volatile int temp=0;
void setup() {

  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin,INPUT);digitalWrite(buttonPin,HIGH); // initialising button pin
  pinMode(button2Pin,INPUT);digitalWrite(button2Pin,HIGH); // initialising button pin
  // initialize the serial communication:
  Serial.begin(9600);
  state = digitalRead(buttonPin); // Reads the data from the serial port

  while(state == HIGH){
    state = digitalRead(buttonPin); // Reads the data from the serial port
    digitalWrite(ledPin, LOW); // Turn LED OFF
    Serial.println("OFF"); // Sending OFF signal to processing
    delay(10);
  }



  Serial.println("sensor Calibration... Takes upto 10s...");
```

```
  int i=0,sum=0;
  while(i<bufferSize){
    sum+= analogRead(soundPin); //reading sound analog pin
    delay(1000);                // delay to account for single ADC
    i++;
  }
  soundMedian = sum/bufferSize;
  attachInterrupt(0,calibrateISR,FALLING);  //interrupt to make the sensors calibrate
}


void loop() {

  soundMedian = 590;
  digitalWrite(ledPin, HIGH);   //indicates that the machine in powered ON
  sound = analogRead(soundPin); //reading sound analog pin
  Serial.print("L");
  Serial.println(sound);Serial.flush();//sending sound value to get stored
  delay(1000);   // delay to account for single ADC
  light = analogRead(lightPin); //reading light analog pin
  Serial.print("S");
  Serial.println(light);Serial.flush();        //sending light value to get stored
  delay(1000);
  delayMultiplier = 1;
  int diff = abs(soundMedian - sound);
  if(diff>threshold){ //if deviation from mid, then there is an increase in external noise
    flag++;//delay(1000);
    delayMultiplier= 1;
  }
  else if(flag>10){ // if diff is less than threshold and volume from initial settings has
been increased
    flag=0;//delay(1000);
    delayMultiplier=1;
  }

}

//ISR to calibrate sensors
void calibrateISR(){
  int i=0,sum=0;
  while(i<bufferSize){
    sum += analogRead(soundPin); //reading sound analog pin
    delayMicroseconds(1000000);  // delay to account for single ADC
    i++;
  }
  soundMedian = sum/bufferSize;
  delayMicroseconds(10000000);
  temp =1;
}
```

PROCESSING CODE

```
/*
 This is our submission of the source code of the project for Wireless Sensor Networks
(CSD337), which consists of implementation of Catnap, a sleep assistant.
 The code has been written in Arduino IDE and Processing.

 AUTHORS
 Atish Majumdar      : 1410110081
 Prasanna Natarajan  : 1410110298
 Vedant Chakravarthy : 1410110489
*/

import processing.serial.*;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.text.SimpleDateFormat;
import java.util.Date;
//import processing.sound.*;

int numFrames = 40;  // The number of frames for GIF1 in the animation
int numFramesLine = 25;  // The number of frames for GIF2 in the animation
int numFramesAwake = 20; // The number of frames for GIF3 in the animation
int currentFrame = 0;
int currentFrameLine=0;
int currentFrameAwake=0;
PImage[] images = new PImage[numFrames];  // Creating GIF1
PImage[] lineI = new PImage[numFramesLine];  // Creating GIF2
PImage[] awake = new PImage[numFramesAwake]; // Creating GIF3
PFont font;
//WhiteNoise wn;
SoundFile sf;
BufferedWriter bw = null;
FileWriter fw = null;
/* First button */
int rectXBase=-1280;
int rectX=870, rectY=400-40;            // Position of "Wake UP" button
int rectSizeX = 160,rectSizeY = 80;     // Dimensions of "Wake UP" button
color rectColor;                        // Color of button
color rectHighlight;                    // Hover color
boolean rectOver = false;
/* First button */

/* Second button */
int rectX2=560, rectY2=350;            // Position of "Sleep" button
int rectSizeX2 = 160,rectSizeY2 = 80;  // Dimensions
color rectColor2;                      // Color of button
color rectHighlight2;                  // Hover color
```

```
boolean rectOver2 = false;
/* Second button */

/* Third button */
int rectX3=560, rectY3=175;            // Position of "Show Statistics" button
int rectSizeX3 = 160,rectSizeY3 = 80;  // Dimensions of "Show Statistics" button
color rectColor3;                      // Color of button
color rectHighlight3;                  // Hover Color
boolean rectOver3 = false;
/* Third button */

boolean initV=true;

Serial myPort;  // Create object from Serial class

String input;  //to store received value from arduino
int sound;     //to store sound value from input
int light;     //to store light value from input
int volUp;     //to store volume up value from input
int volDown;   //to store volume down value from input

File file;

void setup(){

  size(1280, 800);                      // set the window size to be 1280*800
  //wn = new WhiteNoise(this);
  sf= new SoundFile(this,"song.mp3");    // Reference to audio file to be played containing
White Noise
  frameRate(60);                         // Frame rate = 60
  surface.setTitle("Catnap");            // Page Title
  file = new File("data.csv");  //Statistics storage file path
    // if file doesnt exists, then create it
    if(file.exists()){
         file.delete();
    }
    //if (!file.exists()) {
      try {
        file.createNewFile();
        fw = new FileWriter(file.getAbsoluteFile(), true);
        bw = new BufferedWriter(fw);
        PrintWriter pw = new PrintWriter(bw);

        pw.println("Time,Sound,Light");   // Heading for .csv file to store data
        pw.flush();
        System.out.println("Done heading");
        pw.close();
      } catch (IOException e) {
        e.printStackTrace();
      }
```

```
  font = createFont("Rochester-Regular.ttf", 32);  //import font
  textFont(font);                                   //set font to text

  rectColor = color(255);
  rectHighlight = color(240);
  rectColor2 = color(255);
  rectHighlight2 = color(240);
  rectColor3 = color(255);
  rectHighlight3 = color(240);

  // Preparing GIF files

  images[0]  = loadImage("output/1.gif");
  for(int x=1;x<=9;x++)
    images[x]  = images[x-1];

  images[10]  = loadImage("output/2.gif");
  for(int x=11;x<=19;x++)
    images[x]  = images[x-1];

  images[20]  = loadImage("output/3.gif");
  for(int x=21;x<=29;x++)
    images[x]  = images[x-1];

  images[30]  = loadImage("output/4.gif");
  for(int x=31;x<=39;x++)
    images[x]  = images[x-1];

  for(int x=0;x<=24;x++)
    lineI[x]  = loadImage("line/"+(x+1)+".gif");

  awake[0]  = loadImage("awake/1.gif");
  for(int x=1;x<=9;x++)
    awake[x]  = awake[x-1];

  awake[10]  = loadImage("awake/2.gif");
  for(int x=11;x<=19;x++)
    awake[x]  = awake[x-1];

  // End of preparing GIF files

  myPort = new Serial(this, "COM6", 9600);        // Initialise COM port to connect to Arduino
  myPort.bufferUntil('\n');                       //buffer the input received from serial
connection till \n
}


int offset=0;
int isMove=0;
int offsetShapeX=0;
```

```
void draw(){
  background(255);
  {                                                  // UI elements of frame1
    update(mouseX, mouseY);                          // Update variables based on mouse current
position
    if (rectOver2) {
    fill(rectHighlight2);
    } else {
      fill(rectColor2);
    }
    stroke(0);
    strokeWeight(2);
    rect(rectX2, rectY2, rectSizeX2, rectSizeY2,20);// Draw "Sleep Now" button
    fill(0);
    text("Sleep now!", rectX2+ 30, rectY2+50);

    if (rectOver3) {
      fill(rectHighlight3);
    } else {
      fill(rectColor3);
    }
    stroke(0);
    strokeWeight(2);
    rect(rectX3, rectY3, rectSizeX3, rectSizeY3,20);// Draw "See Statistics" button
    fill(0);
    text("See Statistics", rectX3+ 10, rectY3+50);

    currentFrameAwake = (currentFrameAwake+1) % numFramesAwake;  // Used to cycle through
frames for GIF
    image(awake[(currentFrameAwake) % numFramesAwake], 470, 530);
  }
  {                                                              // UI elements of frame1
    fill(0);
    rect(rectXBase,0,1280,800);
    fill(110,225,230);
    stroke(110,225,230);
    currentFrame = (currentFrame+1+(currentFrame%2)) % numFrames; // Used to cycle through
frames for GIF
    if(isMove==1)                                                 // Set ground line to shake
when sound detected
      currentFrameLine = (currentFrameLine+1+(currentFrameLine%2)) % numFramesLine;  // Used
to cycle through frames for GIF
    else
      currentFrameLine=0;
    ellipse(340+rectXBase,400,560,400);
    image(lineI[(currentFrameLine) % numFramesLine], 90+rectXBase, 350);
    image(images[(currentFrame) % numFrames], 100+rectXBase, 200);
    fill(120);
```

```
    stroke(120);

    if (rectOver) {
      fill(rectHighlight);
    } else {
      fill(rectColor);
    }
    stroke(0);
    strokeWeight(2);
    rect(rectX+rectXBase, rectY, rectSizeX, rectSizeY,20);    // Draw "Wake up" button
    fill(0);
    text("Wake up!", rectX+ 26+rectXBase, rectY+50);
    rectXBase-=mult;                          // Slide from frame1 to frame2 on button click
    if((rectXBase>=0)||(rectXBase<=-1280))
      mult=0;
  }
}

int mult=0;
void update(int x, int y) {
  if ( overRect(rectX+rectXBase, rectY, rectSizeX, rectSizeY) ) {// Hover over "Wake up!"
button
    rectOver = true;
    rectOver2 = false;
    rectOver3=false;
  } else if (overRect2(rectX2, rectY2, rectSizeX2, rectSizeY2) ){// Hover over "Sleep!" button
    rectOver = false;
    rectOver2 = true;
    rectOver3=false;
  }
  else if(overRect3(rectX3, rectY3, rectSizeX3, rectSizeY3)){// Hover over "See Statistics"
button
    rectOver = false;
    rectOver2 = false;
    rectOver3= true;
  }
  else{                                                   // Hover outside all button areas
    rectOver = false;
    rectOver2 = false;
    rectOver3 = false;
  }
}
int buttonActivate = 0;        //for checking if the machine is ON or OFF
void mousePressed() {
  if (rectOver&&(rectXBase > -1280)&&buttonActivate==1) {// Clicked on "Wake Up!" button
    mult=10;
    sf.stop();
  }
  else if (rectOver2&&buttonActivate==1){ // Clicked on "Sleep!" button
    mult=-10;
```

```
   sf.loop();
   sf.amp(0.9);
  }
  else if(rectOver3&&buttonActivate==1){ // Clicked on "See Statistics!" button
    System.out.println("Clicked Inside");
    clickStat();
  }

}

boolean overRect(int x, int y, int width, int height)  {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}
boolean overRect2(int x, int y, int width, int height)  {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}
boolean overRect3(int x, int y, int width, int height)  {
  if (mouseX >= x && mouseX <= x+width &&
      mouseY >= y && mouseY <= y+height) {
    return true;
  } else {
    return false;
  }
}
int sum=0, ii, medianVal=590;
float soundCounter = 0.8;
void serialEvent(Serial p){
 try
 {
  input = p.readStringUntil('\n');

  if(input !=null){
    // println(input);
    // println("here");
  }

  if(input.indexOf("S") !=-1){
    buttonActivate = 1;
    input = input.replace("\n","");
```

```
    input = input.replace(" ","");
    sound = Integer.parseInt(input.substring(1,input.length()-1)); //Parse received Sound val
    if(sound!=0){
      sum+=sound;
      ii=(ii+1)%15;
      if(ii==0){
        medianVal = sum/15; // Calculate a current mean value from given data
        sum=0;
      }
      println(medianVal + "," + sound);
    }

    if(Math.abs(sound-medianVal)>3 && sound!=0){// Disturbance detected, disturbance
threshold = 3
      isMove=1;
      if(sound>medianVal)
      {
        soundCounter+=0.1;                          // Increase volume
        if(soundCounter>1)                          // Check bounds
        {
          soundCounter-=0.1;
        }
        sf.amp(soundCounter);                       // Set output volume
      }
      else
      {
        soundCounter-=0.1;                          // Decrease volume
        if(soundCounter<0.4)                        // Check bounds
        {
          soundCounter+=0.1;

        }
        sf.amp(soundCounter);                       // Set output volume
      }

    }
    else{
      isMove=0;
    }
  }

  if(input.indexOf("L")!=-1){
    buttonActivate = 1;
    input = input.replace("\n","");
    light = Integer.parseInt(input.substring(1,input.length()-1));// Parse received Light val
  }

  //call functions for controlling audio output and volume control

  if(sound!=0){
```

```
    writeIntoFile(sound,light);      // Record values if valid
    delay(1000);
  }

 }catch(Exception e){
   e.printStackTrace();
  }


}

void writeIntoFile(int sound, int light){  // Record values in a format readable as a comma
delimited .csv

    //writing into a csv file
    try {
      fw = new FileWriter(file.getAbsoluteFile(), true);
      bw = new BufferedWriter(fw);
      PrintWriter pw = new PrintWriter(bw);
      Date dNow = new Date( );
      SimpleDateFormat ft = new SimpleDateFormat ("hh:mm:ss");
      pw.println(ft.format(dNow)+","+sound+","+light);
      pw.flush();
      System.out.println("Done");
      pw.close();
    } catch (IOException e) {
      e.printStackTrace();
    }


}
void clickStat(){    // Triggered on click of "See Statistics" button to run utility to
display statistics trends
 println("inside here");
  /*try {
      Process p = Runtime.getRuntime().exec("D:\\showStat.cpython-36.pyc");
    } catch (IOException e1) {
      // TODO Auto-generated catch block
      println("inside catch");
      e1.printStackTrace();
    }*/
    launch("D:\\showStat.cpython-36.pyc"); // Run graph generation for recorded statistics
}
```

REFERENCES:

[1] Kang J. Urban Sound Environment. London: Taylor and Francis; 2006.
[2]Xie, Hui, Jian Kang, and Gary H. Mills. "Clinical review: The impact of noise on patients' sleep and the effectiveness of noise reduction strategies in intensive care units." *Critical Care* 13.2 (2009): 208.
[3]Davis, Matthew C., Desmond J. Leach, and Chris W. Clegg. "6 The Physical Environment of the Office: Contemporary and Emerging Issues." *International review of industrial and organizational psychology* 26.1 (2011): 193-237.