Milestone 6

Pranav S. Rathod: pratho2-pratho2@uic.edu | Manasvi Narayanan: mnaray5-mnaray5@uic.edu

Project: Room Occupancy Counter and Temperature Alarm (ROCTA)

Abstract

With COVID-19, rooms in public areas are set to a specific capacity to reduce spreading, and therefore we must keep track of the number of people in a room at a given time. We also must keep track of temperature, to ensure it is at normal room temperature. Our project will count people coming in and out, to ensure the total number of people is less than or equal to the allotted max amount, as well as ensure the overall temperature is under a max Fahrenheit amount. If not, the room will be notified via alarm.

Description

Our project used one Arduino Uno, and read in two independent inputs, that is, the two inputs will play no role in affecting each other. These two inputs were the temperature of the room and the total number of people in the room as they are entering and/or exiting it. The temperature was read in using a DHT11 module in Celsius, which was then converted to Fahrenheit for display. The total number of people inside a room was counted by using two PIR/IR sensors, both sitting on one side of the entrance: one sensor was closest to the door, and the other sensor was at the same level as the first, but a bit further from the door. The main logic behind this counter is that If a person has entered the room, they would have passed first sensor one (or the sensor closest to the door), and then sensor two (or the sensor further from the door), thus incrementing the counter. If a person who was in the room had exited the room, then they would have passed first sensor two (or the sensor furthest from the door), and then sensor one (or the sensor closest to the

door), thus decrementing the counter.

Communication

For communication, we will be using an HC-05 Module:

The module bridged the user's smartphone and the circuit. This module was used to create a connection via Bluetooth between the user's smartphone and the arduino. This module is mainly used only when the conditions of the room are broken. When the room boundaries are broken (either there are too many people in the room or the overall temperature of the room is too high), the passive buzzer emits an alarm, notifying the room occupants as such. The user would be able to turn off the alarm via Bluetooth connection using only switch #1 on this app: Bluetooth Arduino HC-05 & HC-06 Module. This action resets both the counter and temperature variables.

Inputs

DHT11: Read in the overall temperature of the room. This module was used to monitor the temperature of the room, to ensure the overall room temperature did not exceed the standards set by the room. The module read in the temperature in Celsius, which we converted into Fahrenheit and displayed on the LCD Display.

PIR/IR Sensors: Detected motion of a person entering or exiting the room. Using two PIR (Manasvi)/IR (Pranav) sensors on the same plane, these modules were used to detect motion or obstruction, and depending on the order in which the person passed them, indicated if that person was entering or exiting the room. If the person passed sensor one (Pin A5) and then sensor two (Pin A0), it indicated they were entering the room. On the other hand, if the person passed sensor

two (Pin A0) and then sensor one (Pin A5), it indicated they were exiting the room. Depending on the order, the count variable would either increment (entering) or decrement (exiting), keeping track of the total number of occupants, to ensure that the total in the room at any given time did not exceed the standards set by the room. The overall total was displayed on the LCD Display, underneath the Temperature.

Outputs

LCD Display: Displayed room information and conditional messages. This module was used to notify room occupants on information about the room at any given time, that being the total number of people in the room on the top line, and the overall temperature of the room on the bottom lime. The message on the LCD would change based on the room conditions. If the total number of people exceeded the guidelines, the top line would display “Too Many People!”, and the bottom line would continue displaying the number of people in the room. If the overall temperature exceeded the guidelines, the top line would continue displaying the temperature, and the bottom line would display “Too High Temp!”.

Passive Buzzer: Emitted an alarm. This module was used to act as an alarm. As long as the conditions of the room were not broken (both the total number of people in the room at any given time AND the overall room temperature were less than what was set by the guidelines), the passive buzzer would not emit any noise. However, once one of the room conditions were broken, the buzzer would emit a high pitch, acting as an alarm to notify room inhabitants the room conditions were broken. The buzzer would return to emitting no noise one of two ways: 1) The room inhabitants fixed the conditions manually or 2) The user reset the conditions of the room and turned off the alarm via Bluetooth.

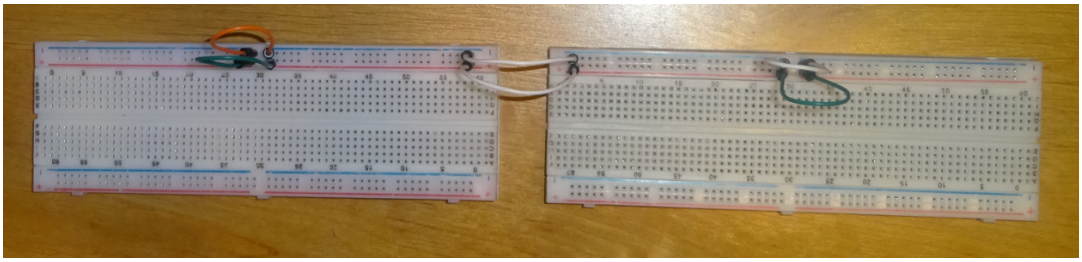
Originality

Our project idea is original because it is relevant with the time and situation we are currently in, with the pandemic and trying to remain as safe as possible. We are combining different parts to help people practice social distancing as much as possible when gathering in an enclosed space, this was to be achieved only if we limited the number of people that can be inside a room at a given point of time, and the room being under a certain overall temperature just to ensure that the room inhabitants are safe. If any of the guidelines are not followed, the alarm sounds until the conditions are ensured, and is a sign that people must vacate the room. The alarm can be turned off by the user if they choose to disregard the guidelines. Even though the idea behind having a person counter is familiar, to our knowledge, no project combines our ideas, that can be used to help people socialize but practice social distancing at the same time, and I think that is what makes our project truly unique.

Steps to build

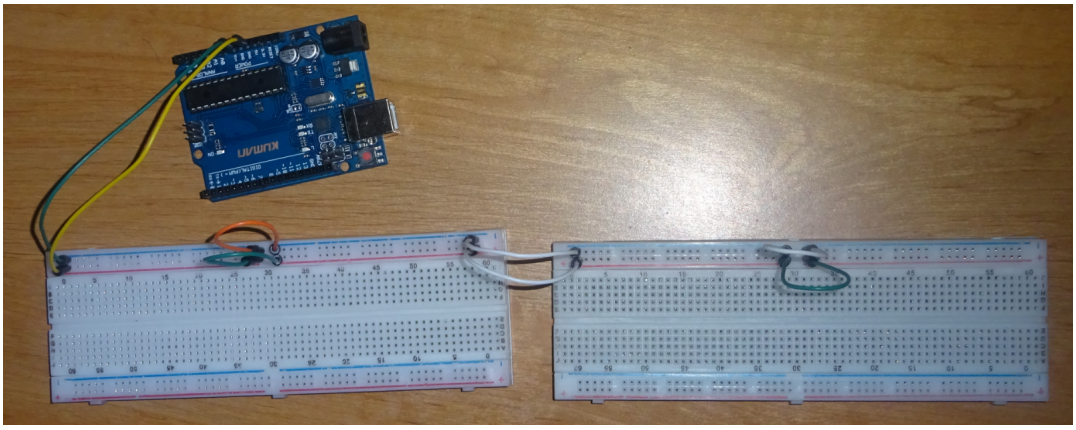
Note: The steps to build this project will utilize PIR Sensors. Depending on the sensors you are using, your wiring may vary a bit, so please keep that in mind as you build.

1) Connect Breadboards
    a) In this build, we will be utilizing two breadboards, to increase the distance between the two PIR Sensors. If you do not wish to do this, you may skip this step.
    b) Place two breadboards size by side. Connect the Power+ and Ground- lines within each breadboard (as the breadboard has a break in the middle), and then connect the +/- lines between the breadboards using, all being connected with jumper wires.

i) 

2) Connect ground and 5v wires

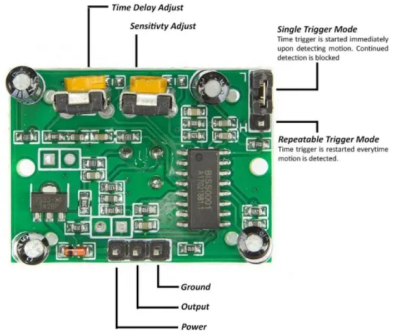
    a) Using two jumper wires, push in one wire into the Ground- line, and push the other wire into the Power+ line. Connect the Ground wire to the GND Pin, and connect the Power wire to the 5V Pin on the Arduino.

i) 

3) Connect the PIR sensors

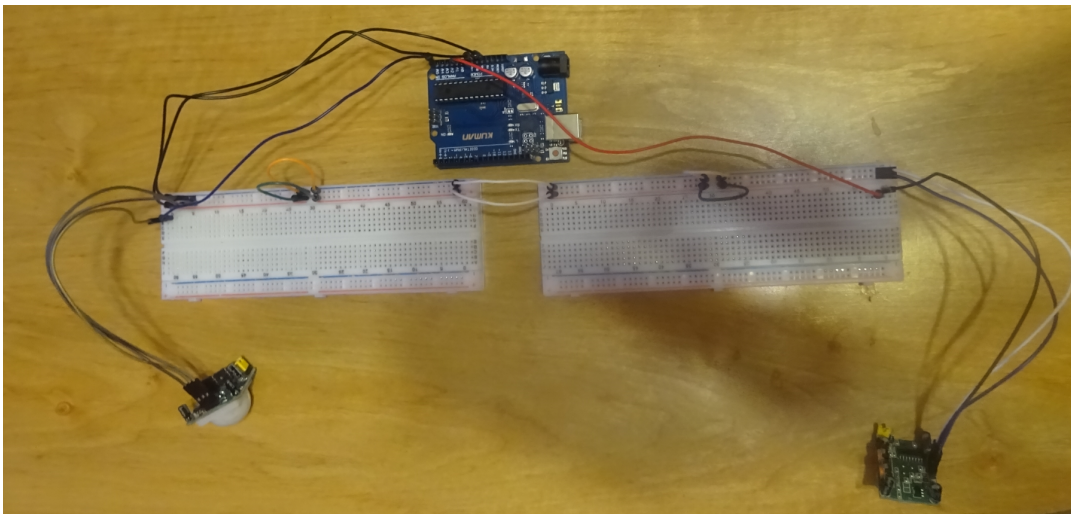
    a) We are going to be using PIR Sensors in this build. Keep in mind that if you are using different types of sensors, this step may vary.

    b) On the PIR Sensor, there are three pins as described below

i) 

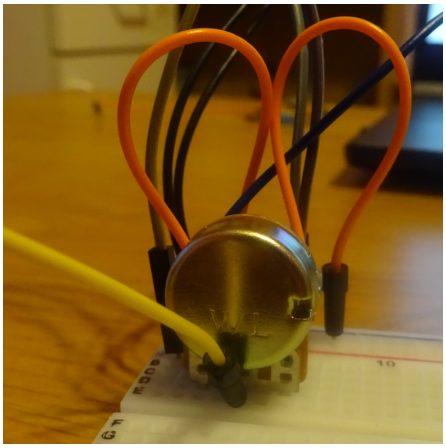
c) Using three Dupont wires, push all wire onto their respective pins. Connect the Ground wire to the top blue Ground- line on the Breadboard, and connect the Power wire to the top red Power+ line on the Breadboard. Then connect the middle Output wire to the analog pins on the Arduino. For the sensor connected to the Leftmost side of the Left Breadboard, push the Output wire into the Breadboard and connect a second Jumper wire from the Breadboard to the A5 Pin. For the sensor connected to the Rightmost side of the Right Breadboard, push the Output wire into the Breadboard and connect a second Jumper wire from the Breadboard to the A0 Pin. Note that the Pin orientation can always be changed according to the direction the sensors are facing the entrance, that is we can always plug in the sensor 1 to Pin A0 and sensor two to Pin A5, if need be.

i)



4) Connect the LCD

a) We will be wiring the LCD into the Left Breadboard with Jumper Wires. To wire in the LCD, first push in the potentiometer into the Leftmost side of the Breadboard. Going from Left to Right on the three pins, wire the left pin from behind to the blue Ground- line, connect a wire to the middle pin from the front (this will be connected to the LCD), and finally connect the right pin from behind to the red Power+ line on the Breadboard.

i) 

b) Next we will wire in the LCD. Push the LCD into the Left Breadboard. Going from Left to Right, connect the following pins like so with Jumper Wires:

i) VSS - Connect the VSS to the blue Ground- line

ii) VDD - Connect the VDD to the red Power+ line

iii) V0 - Connect the V0 to the middle wire from the Potentiometer

iv) RS - Connect the RS to Pin 12 on the Arduino

v) RW - Connect the RW to the blue Ground- line

vi) E - Connect the E to Pin 11 on the Arduino

vii) D4 - Connect the D4 to Pin 5 on the Arduino

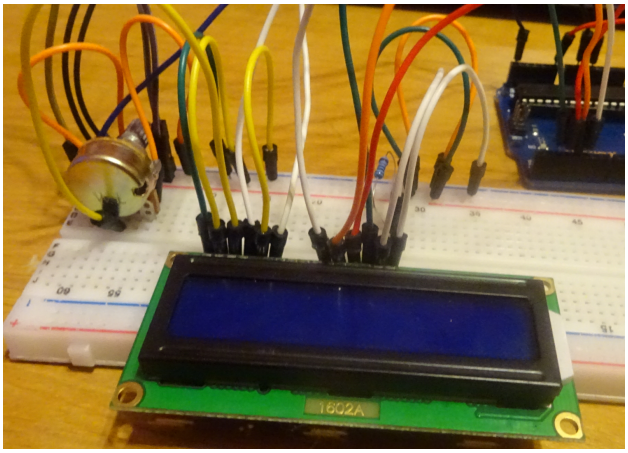
viii) D5 - Connect the D5 to Pin 4 on the Arduino

ix) D6 - Connect the D6 to Pin 3 on the Arduino

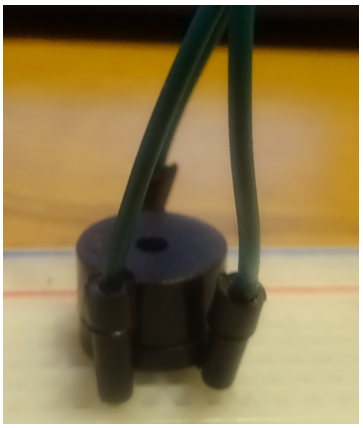
x) D7 - Connect the D7 to Pin 2 on the Arduino

xi) A - Connect the A to the red Power+ line with a 220K Ohm resistor

xii) K - Connect the K to the blue Ground- line

(1) 

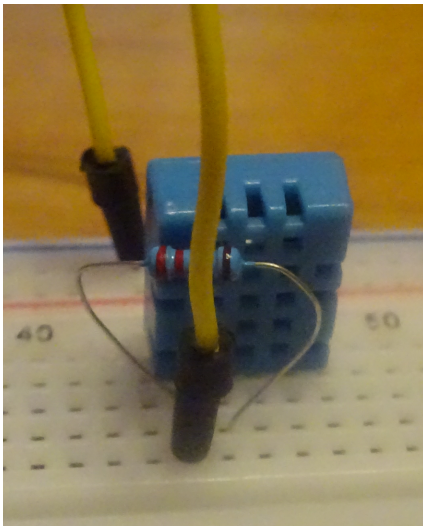
5) Connect the buzzer
    a) We will be wiring the buzzer to the Right Breadboard, so make note of the positive and negative sides of the buzzer
        i) Connect the positive side of the buzzer to a Pin 9 on the Arduino
        ii) Connect the negative side of the buzzer to the blue Ground- line

(1) 

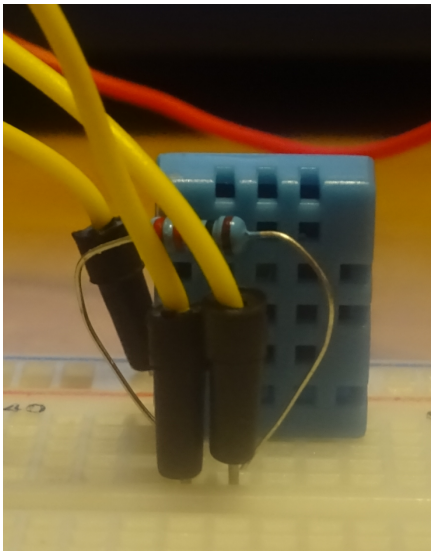
6) Connect the DHT11
    a) The DHT11 has 4 pins. We will wire using Jumper wires, working from left to right where the front of the DHT device is facing us. Start by pushing in the DHT11 into the Right Breadboard.

b) 

    i) Pin1: Connect Pin1 to Pin2 with a 10 k Ohm resistor, and also connect a wire from Pin1 to the blue negative ground line

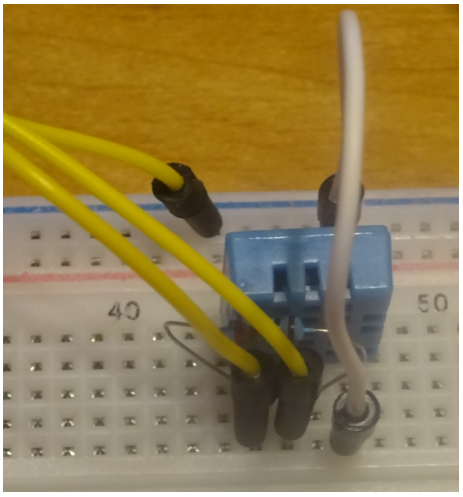
(1) 

ii) Pin2: In addition to the resistor, connect a wire from Pin2 to Pin 13 on the Arduino

(1) 

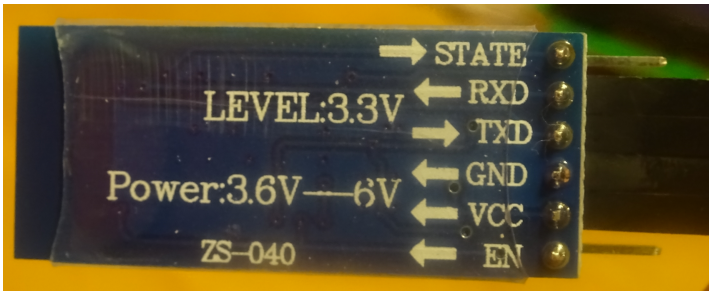
iii) Pin3: Skip pin3, it will not be used

iv) Pin4: Connect a wire from Pin4 to the top red Power+ line

(1) 

7) Connect the HC-05 Module

a) For our communication potion, we need to connect a Bluetooth Module. There are different types, but in this build we will be using the HC-05 Module, of which we will be using 4 pins: RXD, TXD, GND, and VCC

i) 

b) Using 4 Dupont wires, connect the 4 pins as so:

i) RXD - Connect to the 1/TX Pin on the Arduino

ii) TXD - Connect to the 0/RX Pin on the Arduino

iii) GND - Connect to the Ground- line on the Breadboard

iv) VCC - Connect the Power+ line on the Breadboard

c) When the Arduino unit is plugged into a power supply, the HC-05 should turn on and light up red, indicating it is connected.

How is our project used

The Idea behind making this device is to practice discipline in maintaining the guidelines of social distancing especially after experiencing how things have been during this pandemic, where public places, like stores or closed gatherings, require the presence of people in order to function. This device can help monitor the number of people in that enclosed space at any given time along with the overall temperature of the room, just to ensure that people are in a safe environment.

The Guidelines are mostly set the same for everybody, having the uniform number of people in a room no matter it's size. That is why we have set a maximum number of people in the room as well as the maximum temperature in the code and not accepting it from the user. The purpose behind this is that if the user is given the choice of setting the maximum capacity, they may get tempted to set an arbitrary limit that may never be reached and the purpose of the device is lost.

After following the steps to build the that are mentioned above and having the code uploaded in the arduino, all the user needs to do is place the device at the entrance in such a way that the person entering, passes both the sensors, and depending on the which side/direction that the sensors are facing, the orientation of entering and exiting may vary. It is assumed that entrance of the enclosed space is also the way the person leaves for the counter to keep track of them entering and exiting the room. In case that the room conditions are broken the user, who is monitoring the room has the option to turn off the alarm if need be which will reset the capacity to zero and start counting again. This can be done by connecting the smartphone to the device via bluetooth and using the app, "*Bluetooth Arduino HC-05 & HC-06 Module*" and turning on Switch 1. It is always advised to let the alarm turn off automatically once the room conditions are restored, but if need be this option gives the user to override the guidelines.

Timeline

Timeline of Development given in a week-by-week format of completed items:

Week 10: We Tested our PIR sensors and to check if they were triggered.

Week 11: We finished our Slide Deck for the Expo and Worked on the DHT11 Sensor.

Week 12: We worked on the counters section of the code as well as added the buzzer, that would make a sound in case the counter goes over maximum capacity, or beyond a certain temperature.

Week 13: We added the HC-05 bluetooth module that will make use of an app, "Bluetooth Arduino HC-05 & HC-06 Module" That would turn off the alarm when the room conditions are broken, via the "Switch 1" Toggle in the app.

Week 14: We had our project finalised and made the slide deck and the YouTube video for our design presentation for the Expo, as well as started working on the Final Design Document.

Week 15: We evaluated 2 Groups's demonstration of their project. We also finished working on the Final Design Document and submitted it on gradescope.

What Worked and What Didn't Work

Worked:

- Using the HC-05 to create a connection between the phone and the arduino via Bluetooth
- The Passive Buzzer makes a sound when the guidelines are broken (Too many people, too high temperature)
- IR and PIR sensors picked up obstruction/ movement respectively when someone passed them
- DHT-11 sensor reads in the room temperature in celsius we convert it to be displayed in Fahrenheit

Didn't Work:

- Pranav - Hardware issues with PIR sensor, thus switched to IR sensors, since PIR sensor was not available to be procured due to a complete lockdown in the residing state of India

List of Materials

- 2 PIR or IR Motion Sensors
- 1 16X2 LCD Display
- Breadboard - We recommend using two to increase the distance between sensors
- 1 Passive Buzzer
- 1 HC-05 Bluetooth Module
- 1 DHT 11
- Jumper Wires
- Dupont Wires

List of References

ElectroPeak. PIR Motion Sensor: How to Use Pirs w/ Arduino & Raspberry Pi. 8 Apr. 2019, create.arduino.cc/projecthub/electropeak/pir-motion-sensor-how-to-use-pirs-w-arduino-raspberry-pi-18d7fa.

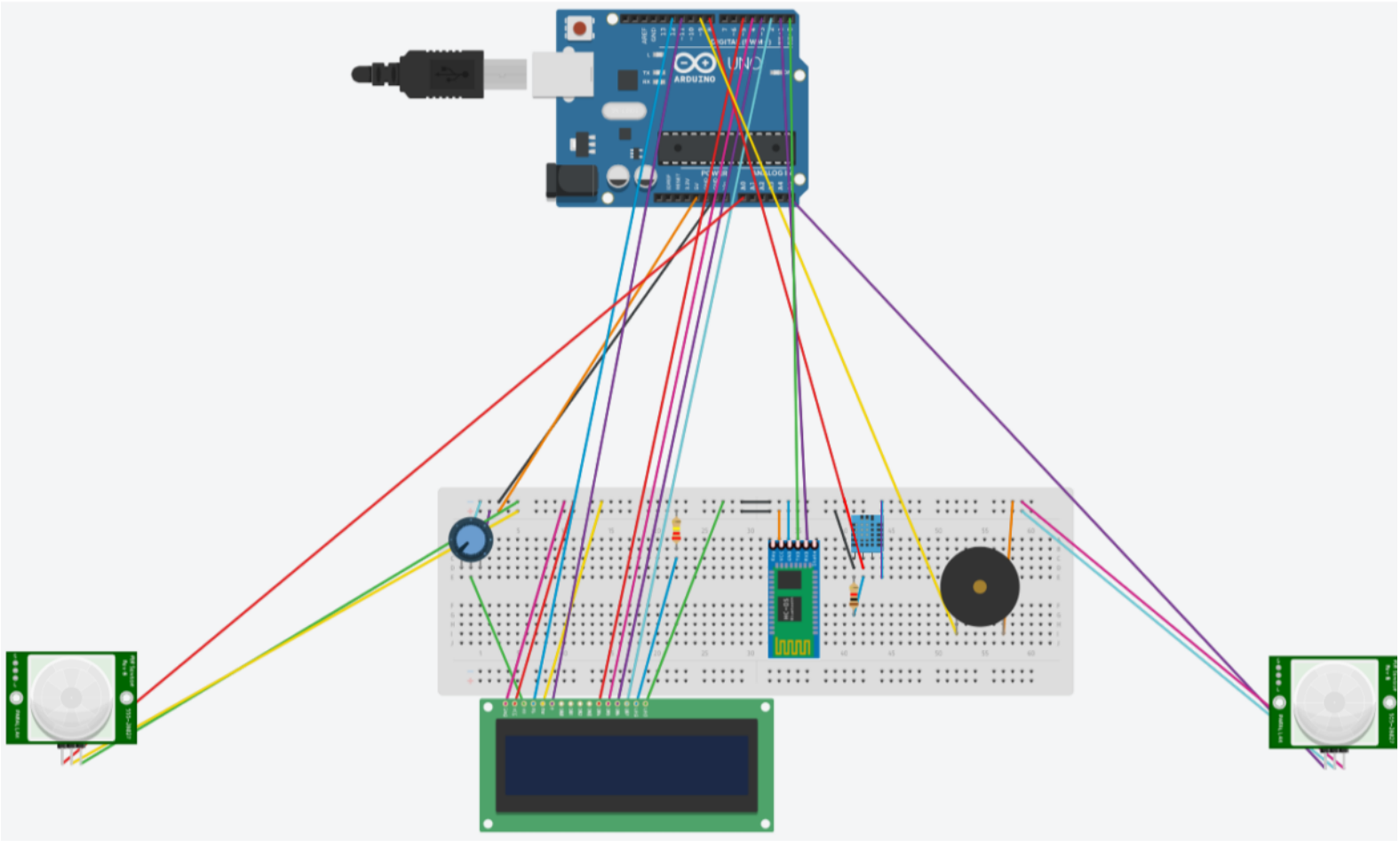
Khondaker, Mahamudul Karim. Temperature Sensor with Arduino Uno. 17 June 2020, create.arduino.cc/projecthub/m_karim02/temperature-sensor-with-arduino-uno-9806a3.

Team, The Arduino. Play a Melody Using The Tone() Function. 5 Feb. 2018, www.arduino.cc/en/Tutorial/BuiltInExamples/toneMelody.

Team, The Arduino. Scrolldisplayleft() and Scrolldisplayright() Methods. 5 Feb. 2018, www.arduino.cc/en/Tutorial/LibraryExamples/LiquidCrystalScroll.

Campbell Scott. How to Set Up the DHT11 Humidity Sensor on an Arduino. www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/

Circuit Sketch



Code Sketch

```
#include <LiquidCrystal.h>
#include <dht.h>

int sensor1 = A5; //Pin for sensor 1
int sensor2 = A1; //Pin for sensor 2
int counter = 0; //Integer for number of people in the room
int temp = 0; //Integer for the temperature
int maxPeople = 6; //Limit on people in the room (change per room capacity)
int maxTemp = 70; //Limit on temp of room (change per room capacity)
int off = 0; //Integer to indicate which condition is broken
int buzzerPin = 9;            //Pin for the buzzer/alarm
String tempString, pString; //String for LCD messages
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2; //LCD Pins
#define DHT11_PIN 8 //DHT Pin
```

```
LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //LCD setup
dht DHT; //DHT object

void setup() {
  pinMode(sensor1, INPUT); //Sensor1 is input
  pinMode(sensor2, INPUT); //Sensor2 is input
  Serial.begin(9600); //Debugging
  lcd.begin(16, 2); //Setup LCD
  lcd.clear(); //Clear LCD just in case
  delay(5000); //Delay to give time to put in the HC-05 Pins
} //end setup()

void loop() {
  int chk = DHT.read11(DHT11_PIN); //Read in the DHT
  temp = DHT.temperature*(9/5)+32; //Calculate the read in Celcius temp to Fahrenheit
  if(off == 0 || off == 2){ //If the person condition is not broken
    //Print out a string of the current room temperature on the LCD
    tempString = ("Temperature: ");
    tempString = (tempString + temp);
    tempString = (tempString + "F");
  } else if (off == 1){ //Otherwise, print out the Too Many People message
    tempString = "Too Many People!"; //Indicates the person condition is broken
  }

  lcd.setCursor(0,0);
  lcd.print(tempString); //Print this out on the top line of the LCD


  if(off == 0 || off == 1){ //If the temperature condition is not broken
    pString = ("People: ");
    pString = (pString + counter); //Print out a message on the total room occupants
  } else if(off == 2){//Otherwise, print out the Too High Temp message
    pString = "Too High Temp!";//Indicates the person condition is broken
  }

  lcd.setCursor(0,1);
  lcd.print(pString); //Print this on the bottom line of the LCD
  delay(2000);

  //If a person is entering
  if(digitalRead(sensor1) == HIGH){
    //First check for sensor1
    checkBluetooth(); //Call checkBluetooth in case they want to turn off the alarm
```

```
    Serial.println("Sensor 1 triggered"); //Debugging
    while(digitalRead(sensor2) != HIGH){ //Wait for the user to pass sensor2
        Serial.println("WAITING For Sensor 2"); //Debugging
        checkBluetooth(); //Call checkBluetooth in case they want to turn off the alarm
    }
    Serial.println("Sensor 2 triggered"); //Debugging
    counter++; //Increment counter
    checkBluetooth(); //Call checkBluetooth in case they want to turn off the alarm
  }

  //If a person is exiting
  if(digitalRead(sensor2) == HIGH){ //Check for sensor2 then
    checkBluetooth(); //Call checkBluetooth in case they want to turn off the alarm
    Serial.println("Sensor 2 triggered"); //Debugging
    while(digitalRead(sensor1) != HIGH){ //Wait for the user to pass sensor1
        Serial.println("WAITING 1"); //Debugging
        checkBluetooth(); //Call checkBluetooth in case they want to turn off the alarm
    }
    Serial.println("Sensor 1 triggered"); //Debugging
    if(counter > 0){
      counter--; //We cannot have negative people, so just to prevent that
    }
    checkBluetooth(); //Call checkBluetooth in case they want to turn off the alarm
  }

  if(counter >= maxPeople){ //Check for the person condition
    off = 1; //Set off to 1 if it is broken
  } else if (temp >= maxTemp){ //Check for the temperature condition
    off = 2; //Set off to 2 if it is broken
  }

  if(off != 0){ //If off is not 0, a condition is broken
    analogWrite(buzzerPin, 215); //Sound the alarm
    checkBluetooth(); //Call checkBluetooth
  } else { //Otherwise, no conditions are broken
    analogWrite(buzzerPin,0); //So we keep/turn off the alarm
  }
}//end loop()


void checkBluetooth(){
  if(counter >= maxPeople){ //Check to see if the person condition is broken
    off = 1; //If so, set off to 1 to indicate
```

```
  } else if (temp >= maxTemp){ //Check to see if the temp condition is broken
    off = 2; //If so, set off to 2 to indicate
  }

  if(off != 0){ //if off does not equal 0, then a room condition is broken
    analogWrite(buzzerPin, 215); //Sound the alarm
    if(Serial.available()> 0){ //Look for the Bluetooth device
      char inputByte = Serial.read(); //Read in the char
      if(inputByte == 'Z'|| inputByte =='z'){ //With our app, Z/z was used for switch 1
        analogWrite(buzzerPin, 0); //I we get Z/z, turn off the alarm
        counter = 0; //Reset counter
        if(off == 2){
            delay(5000); //Delay to give time for the temperature to go down
            lcd.setCursor(0,1);
            pString = ("People: ");
            pString = (pString + counter);
            lcd.print(pString);
        }
        temp = 0; //Rest temp
        off = 0; //Reset off
      }
//For manual fix
    } else if(off == 1){ //if off is 1, then we want to fix the people
      if(counter < maxPeople){ //check if the total is less than the max
        analogWrite(buzzerPin, 0); //if so, turn off the alarm
        off = 0; //Reset off
      }
    } else if(off == 2){ //if off is 2, then we want to fix the temp
      if(temp < maxTemp){ //check if the temp is less than the max
        analogWrite(buzzerPin, 0); //if so, turn off the alarm
        off = 0; //Reset off
      }
    }
  }
}//end checkBluetooth()
```