

ECE 2804 Integrated Design Project

Smart Home Project Final Solution

Project Manager:

Minh T. Nguyen, mnguyen0226@vt.edu

Instructor:

Shuxiang (William) Yu

Institution:

Virginia Polytechnic Institute and State University

Location: Blacksburg

Date:

June 2nd, 2020

Table of Contents:

I/ Project Description Break Down

II/ High Level Design

III/ Weather Station + AC Unit

IV/ Automated Night Light

V/ Fire Detection System

VI/ Intruder Detection System

VII/ Indoor Light Bluetooth

VIII/ MIT App Inventor

IX/ Final Assembly

I/ Project Description Break-Down Analysis:

1/ You should use a 16x2 LCD display with the Arduino Uno => For display **WEATHER STATION** and/or INTRUDER.

2/ You should develop an audio amplifier so that your system can deliver audio messages of critical situations => For **FIRE DETECTION**.

3/ You should control appliances (an LED will mimic an appliance) from your android application. => For **INDOOR LIGHT**/STOVE/WASHING MACHINE (+app)

4/ You should develop your own weather station that can at least read temperature and report it to both LCD Display and the Android application => **WEATHER STATION** (+app)

5/ You should develop an automated night light that will illuminate when it gets dark. => **AUTOMATED NIGHT LIGHT SYSTEM**.

6/ You should develop an intruder alert system, where the intruder alert will be broadcast to the audio system and to the phone application as well. => **INTRUDER SYSTEM**. (+app).

7/ You should be able to turn on one of your appliances when the temperature exceeds some threshold level. Assume your appliance, B is an AC and you will be using an LED to mimic the AC on your actual board. => **AUTO AC SYSTEM**

8/ You will develop an Android Phone application using MIT App Inventor, which is an open-source Android phone application development platform. You should control the entire system through a button touch. => **WEATHER STATION, INTRUDER SYSTEM, INDOOR LIGHT**.

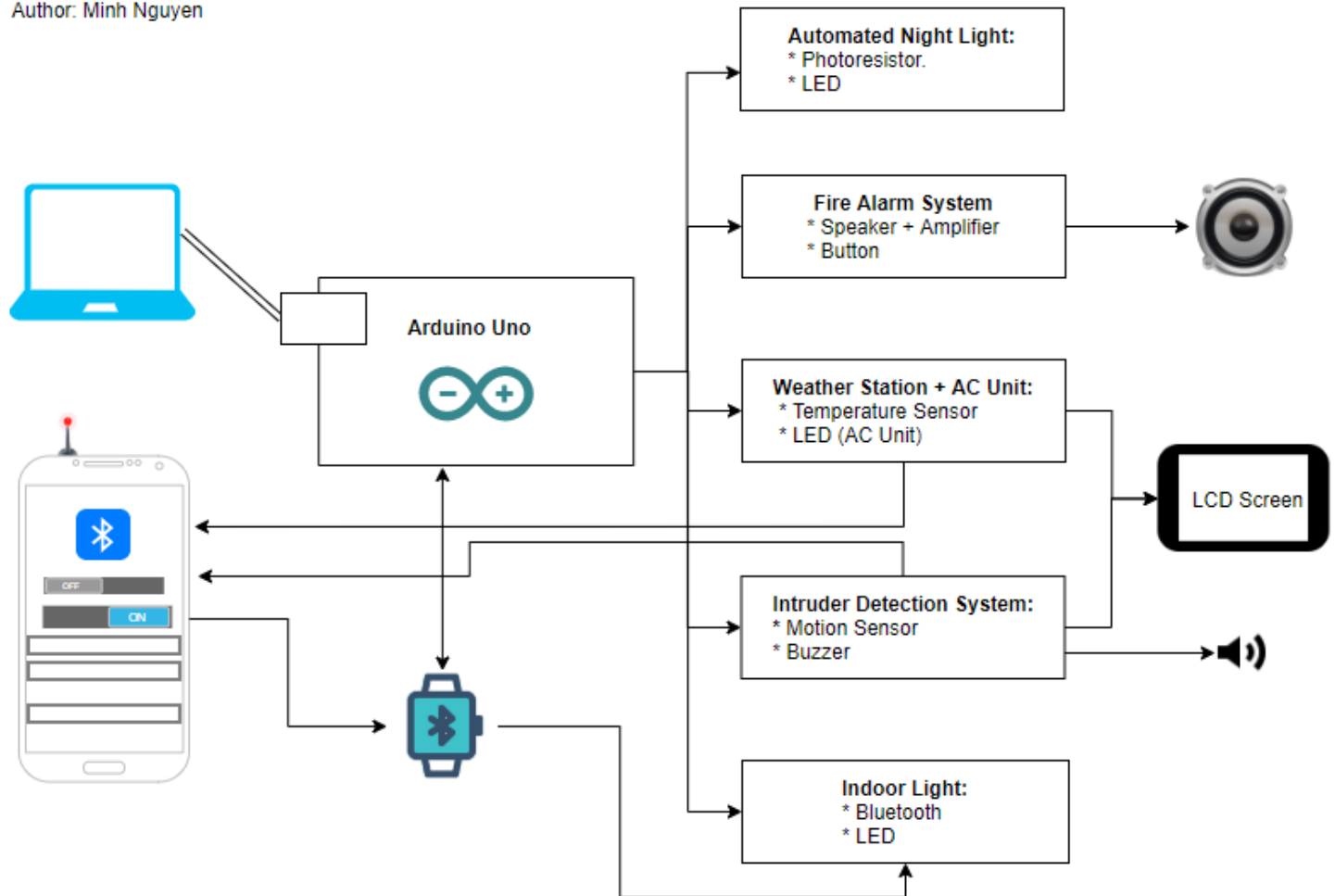
9/ You will be using a Bluetooth module to connect the Android Phone with the microcontroller to establish a wireless communication link between them. => **INDOOR LIGHT** (+app)

II/ High-Level Design

High-Level Design of the Smart Home

Date: 6/4/2020

Author: Minh Nguyen

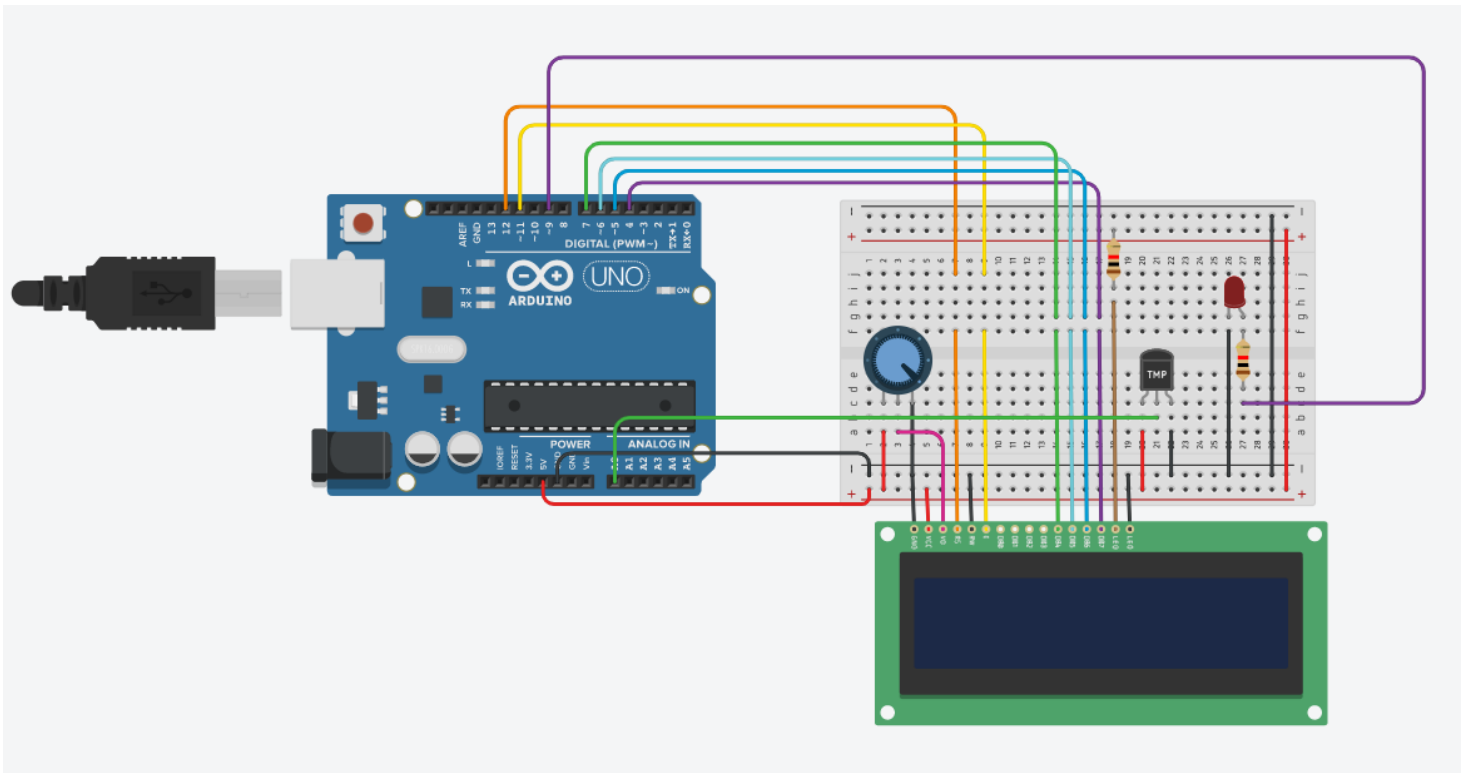


The system meets the course's requirement by containing five systems, and all of them will be controlled via Arduino Uno microcontroller with the written code from the laptop. The Automated Night Light System contains a photoresistor and a LED. If there is no light, the LED will turn on. The Fire Alarm System contains a speaker, an amplifier, and a button. When the owner presses the button then a recorded message will be processed by the amplifier which then delivers the message to the speaker. The Weather Station + AC Unit contains a temperature sensor and an LED which represents the AC Unit. The weather station will take real-time temperature, and if the temperature in F goes above 67F, then the LED (AC Unit) is on. The temperature will constantly be updated on the LCD and MIT App. The Intruder Detection System contains a motion sensor and a buzzer. If a motion is detected, then the buzzer will make buzzing noise, the LCD asks the user to call 911, and the MIT App will alert the owner of the theft's moving and stopping time. The Indoor Lights contains a Bluetooth and an LED (indoor light). The LED on/off state will be controlled via MIT app, Bluetooth, and Arduino Uno.

III/ Weather Station + AC Unit:

(This one just print out the temperature on the LCD because we have not introduce the MIT App yet, the printing on MIT App is in the last section)

a/ Arduino Assembly:

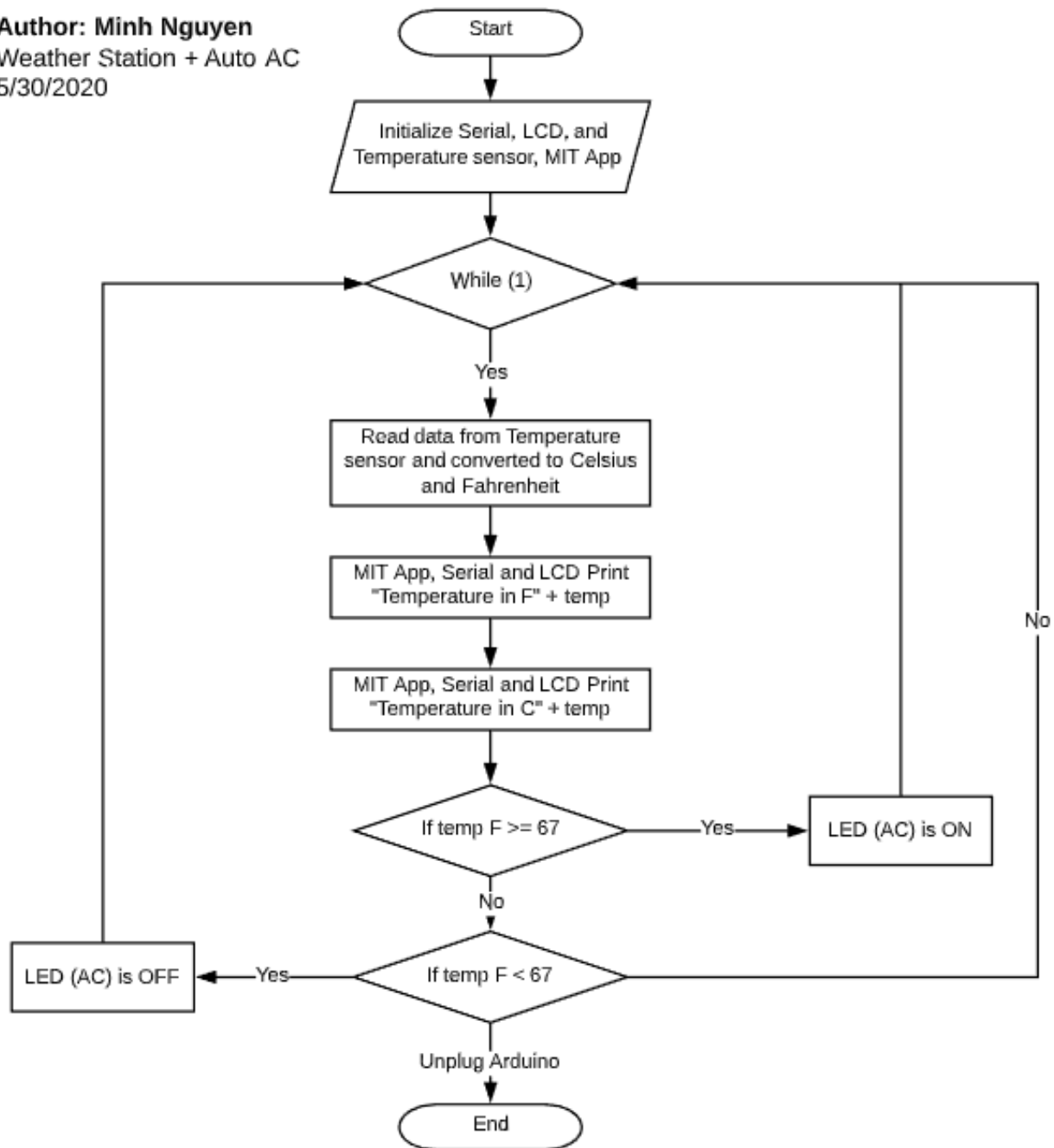


b/ Components:

- 16x2 LCD
- Temperature Sensor
- LED
- Potentiometer.
- Resistor for LCD: Not needed in reality but for simulation on Tinkercad is a must!
- Resistor Calculation:
 - + V_{cc} of Arduino pin = 5V
 - + V needed for LED = 0.7V
 - + Efficient 20mA
 - + $R = (5 - 0.7) / 0.02 = 235 \text{ Ohm} \Rightarrow$ Use 270 Ohm in the ECE 1004 Kit

c/ Flow Chart:

*Author: Minh Nguyen
 *Weather Station + Auto AC
 *5/30/2020



d/ Code:

```

/**
 * Author: Minh Nguyen
 * Date: 6/4/2020
 * Weather Station + AC Unit
 */
#include <LiquidCrystal.h>

//Initialize the library by associating ICD interface pin with the arduino pin number
const int rs = 12, en = 11, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
LiquidCrystal lcd(rs,en, d4, d5, d6, d7);

int sensorPin = A0; // Adjustment for what Analog Pin, currently A0
int tempC, tempF;
int AC = 9;

void setup() {
  Serial.begin(9600);
  pinMode(AC, OUTPUT);
  lcd.begin(16,2);
  lcd.clear();
}

void loop() {
  tempC = get_temperature(sensorPin);
  tempF = celsius_to_fahrenheit(tempC);
  lcd.setCursor(0,0);
  lcd.print("Temp: ");

  lcd.setCursor(6,0);
  lcd.print(tempF); lcd.print((char)223); lcd.print("F");

  // Print new line Serial Monitor
  String res1 = "Temperature in F is: ";
  Serial.println(res1);
  Serial.println(tempF);

```

```
lcd.setCursor(12,0);
lcd.print(tempC); lcd.print((char)223); lcd.print("C");
```

```
// Print new line Serial Monitor
String res2 = "Temperature in C is: ";
Serial.println(res2);
Serial.println(tempC);
```

```
//This is a condition to turn the AC on.
if(tempF >= 67)
{
    digitalWrite(AC, HIGH);
    Serial.println("AC is on");
}
else if(tempF < 67)
{
    digitalWrite(AC, LOW);
    Serial.println("AC is off");
}
delay(500);
}
```

```
//Helper Method 1: Read in Celsius
int get_temperature(int pin) {
    int temperature = analogRead(pin);
    float voltage = temperature * 5.0;
    voltage = voltage / 1024.0;
    return ((voltage - 0.5) * 100);
}
```

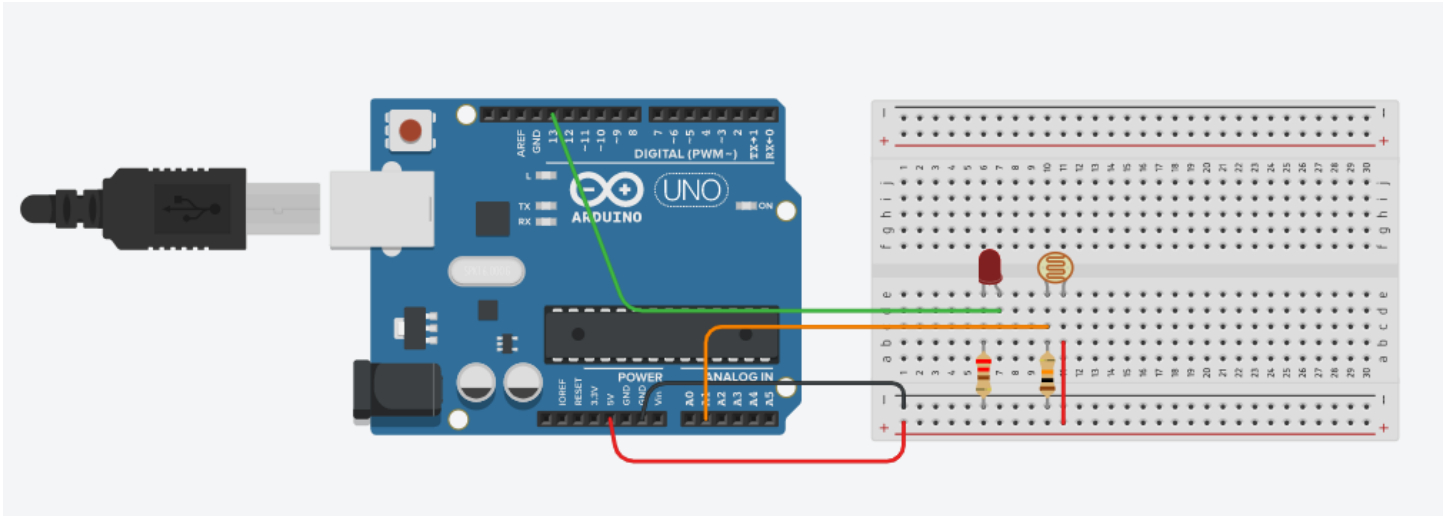
```
//Helper Method 2: Convert Celsius to Fareheign
int celsius_to_fahrenheit(int temp) {
    return (temp * 9.0 / 5.0) + 32.0;
}
```

e/ Testing:

- Software:
 - + Copy the code to Arduino IDE
 - + Choose Port and Verify
 - + Go to Tool => Serial Monitor
 - + Upload to see the real-time temperature
- Hardware:
 - + If the temperature goes above 67, then AC (LED) is on, else it will go off.

IV/ Automated Night Light:

a/ Arduino Assembly:

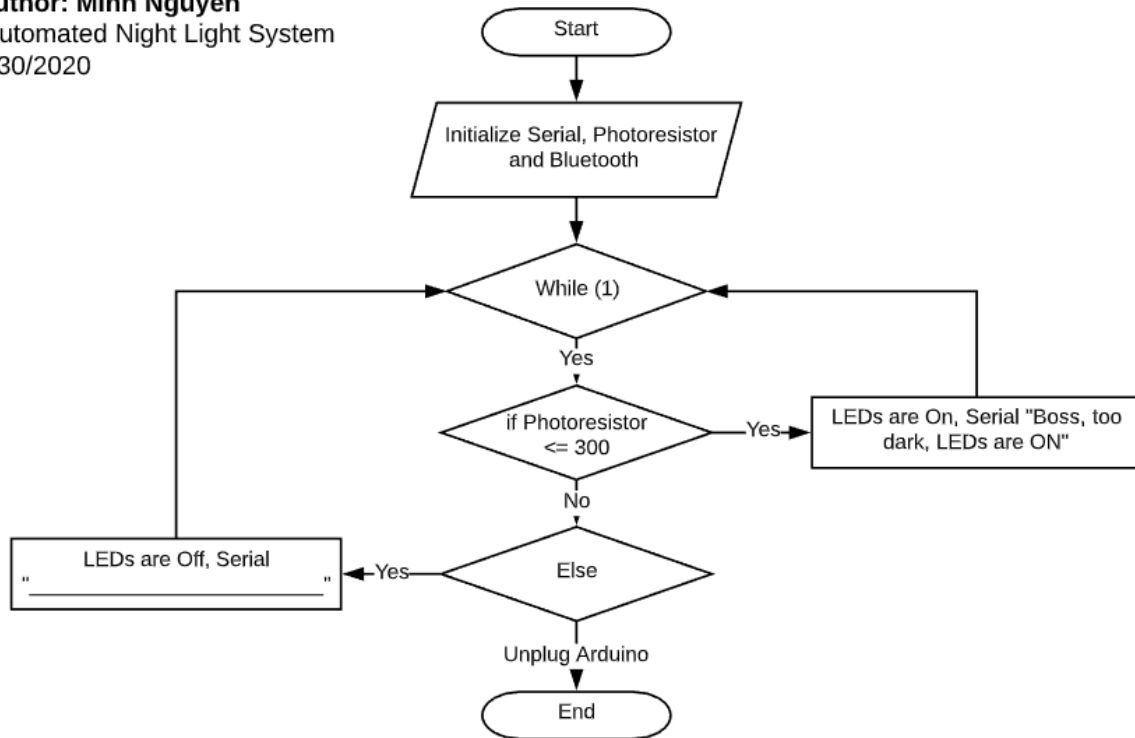


b/ Components:

- Photoresistor
- LED
- Resistor Calculation:
 - + Vcc of Arduino pin = 5V
 - + V needed for LED = 0.7V
 - + Efficient 20mA
 - + $R (LED) = (5 - 0.7) / 0.02 = 235 \text{ Ohm} \Rightarrow$ Use 270 Ohm in the ECE 1004 Kit
 - + $R (\text{Photoresistor}) = (5 - 1.8) / 0.02 \Rightarrow 160 \text{ Ohm}$

c/ Flowchart:

*Author: Minh Nguyen
 * Automated Night Light System
 *5/30/2020



d/ Code:

```

/**
 * Author: Minh Nguyen
 * 5/29/2020
 * Automated Night Light
 */
const int ledPinRED = 13;
const int prPin = A1; // Photoresistor Pin

void setup() {
    Serial.begin(9600);
    pinMode(ledPinRED, OUTPUT);
    pinMode(prPin, INPUT);
}

void loop() {
    int prStatus = analogRead(prPin);
    
```

```
// Check if the prStatur <= 300
// If it is, LED is High
if (prStatus >= 300){
  digitalWrite(ledPinRED, HIGH);
  Serial.println("Boss, too dark, LEDs are ON");
}
else{
  digitalWrite(ledPinRED, LOW);
  Serial.println("_____");
}
}
```

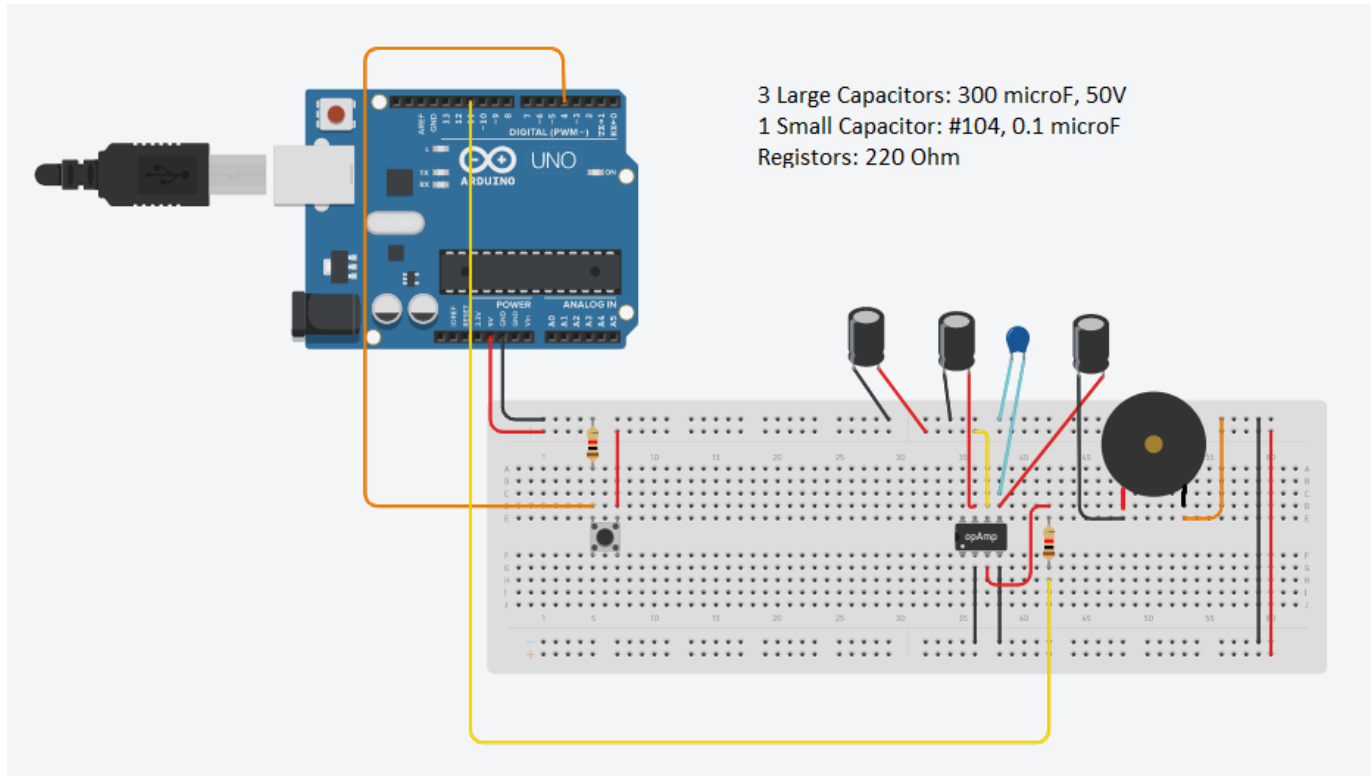
e/ Testing:

- Software:
 - + Copy the code to Arduino IDE
 - + Choose Port and Verify
 - + Go to Tool => Serial Monitor
 - + Upload to see the LED Status
- Hardware:
 - + Test by covering and uncover the Photoresistor.

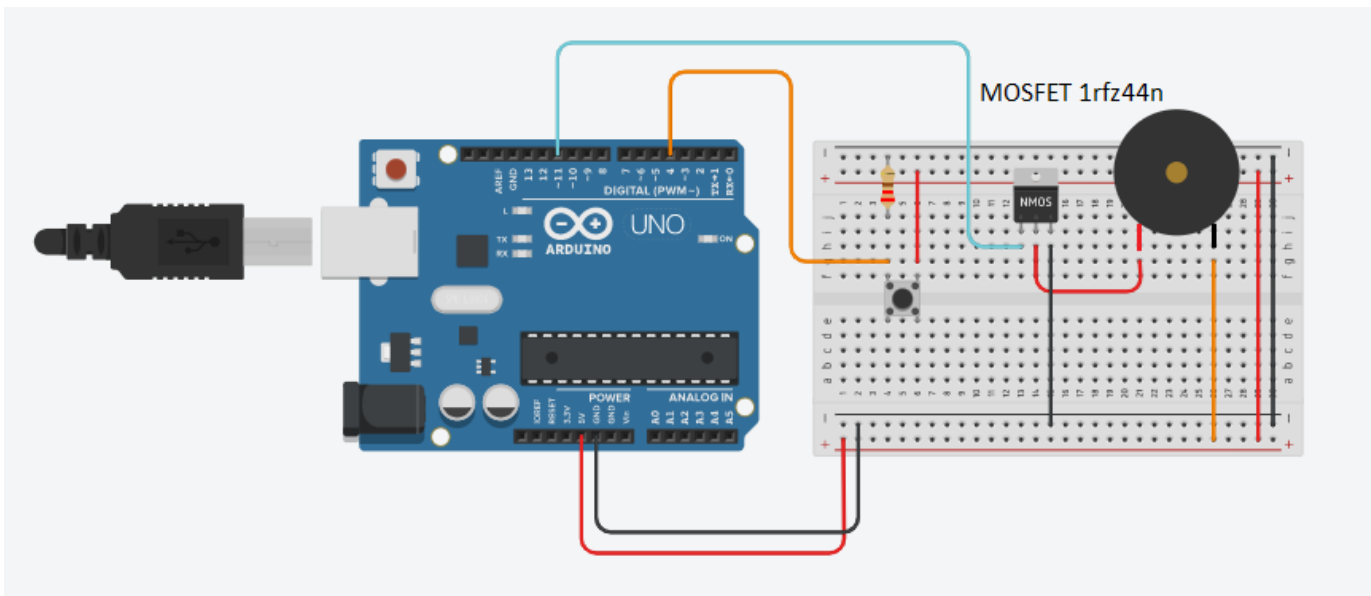
V/ Fire Detection System:

a/ Arduino Assembly:

Method 1: Using LM386 Amplifier (Provided)



Method 2: Using MOSFET irfz44n (ECE 1004 Kit)



b/ Components:

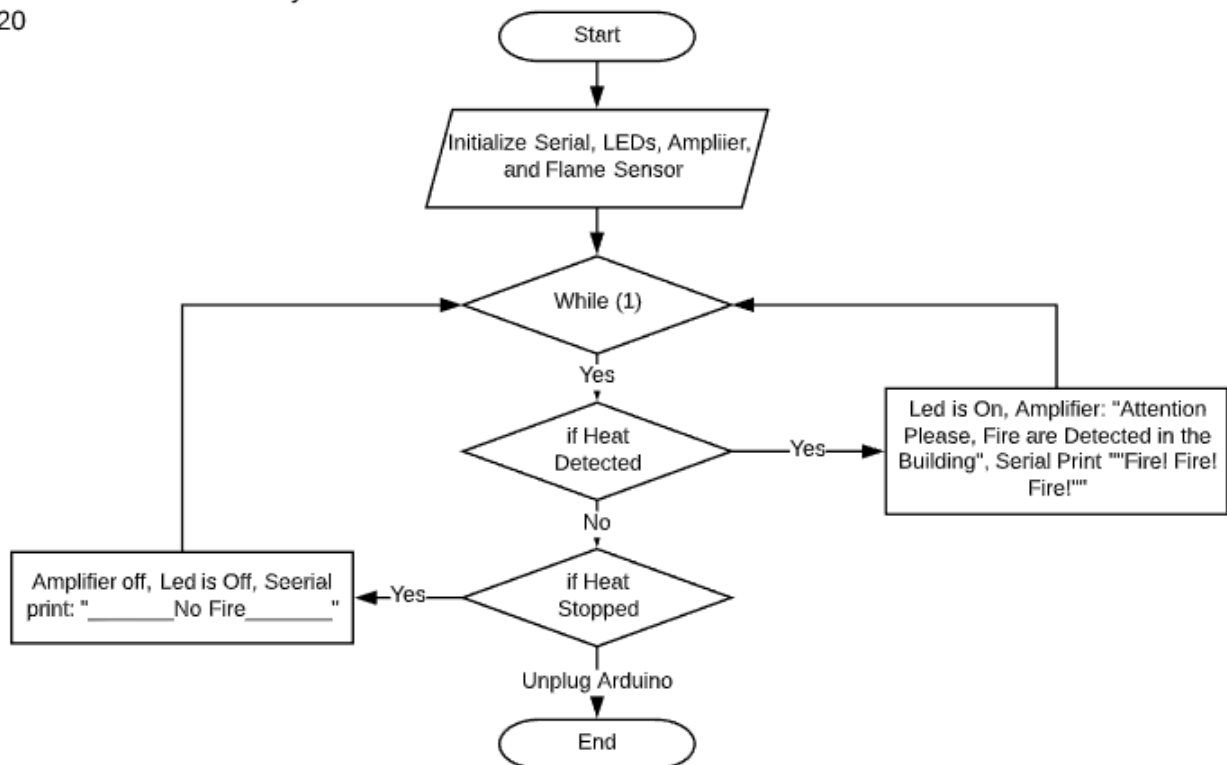
- Button Press.
- Amplifier 1RFZ44n (provided in the ECE 1004 Kit) or LM386 (Provided).
- Speaker (not a buzzer)
 - + Resistor Calculation:
 - + Vcc of Arduino pin = 5V
 - + V needed for LED = 0.7V
 - + => Efficient 20mA
 - + $R = (5-0.7)/0.02 = 235 \text{ Ohm} \Rightarrow$ Use 270 Ohm in the ECE 1004 Kit

c/ Flowchart:

***Author: Minh Nguyen**

***Flowchart for Flame Detection System**

***5/30/2020**



d/ Code:

Method 1: Using Talkie Library

```
// Author: Minh T. Nguyen.
// Method 1:Talkie Library
// 6/9/2020
// Source for MOSFET Amplifier: https://www.youtube.com/watch?v=aaqaAXIZbuc
// LM384 schematic: https://www.youtube.com/watch?v=4ObzEft2R\_g&t=163s
// Button Tutorial: https://www.arduino.cc/en/tutorial/button
// Talkie Library: https://www.arduinolibraries.info/libraries/talkie
```

```
#include <Arduino.h>
#include "Talkie.h"
#include "Vocab_US_Large.h"
```

Talkie voice;

```
int buttonPin = 4;
int buttonState = 0;    // variable for reading the pushbutton status
```

```
// This is message will run only 1 time.
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);

  pinMode(11, OUTPUT);
  digitalWrite(11, HIGH); //Enable Amplified PROP shield
}
```

```
void loop()
{
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    Serial.println("Attention please, fire has been reported in the building!");
    voice.say(sp2_DANGER);
    voice.say(sp2_DANGER);
    voice.say(sp2_RED);
  }
}
```

// Author: Minh T. Nguyen.

// 6/9/2020

// LM384 schematic: https://www.youtube.com/watch?v=4ObzEft2R_g&t=163s

/**

* Have the audio MP4, then convert it to WAV file.

* Note that the audio should not be more than 2s. Else the message will not be fully spoken

* Then drane the WAV file to Audacity to 8000Hz and 16 bit.

* Use EncoderAudio to convert sounds to bit. Then paste that bit to arduino uno below.

*/

```
//Initialize the sounds in bit
```

```
const unsigned char sample[] PROGMEM = {
```

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#)

Nguyen 16

[illegible]
$$\} ;$$

```
int buttonState = 0;    // variable for reading the pushbutton status
```

```
void setup()
```

```
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}
```

```
void loop()
```

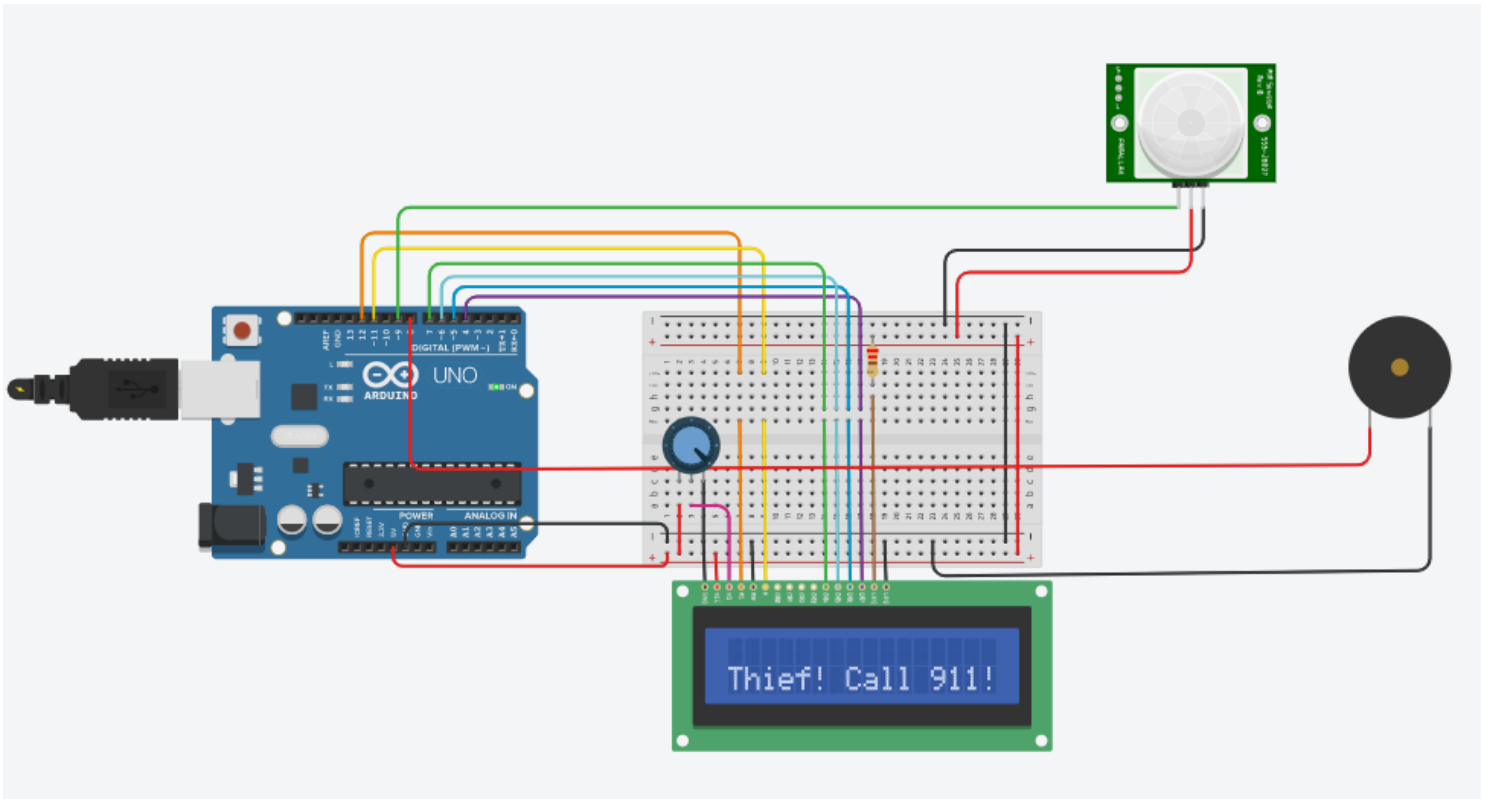
```
{
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    Serial.println("Attention please, fire has been reported in the building!");
    startPlayback(sample, sizeof(sample));
    delay(2000);
  }
}
```

e/ Testing:

- Software:
 - + Copy the code to Arduino IDE
 - + Choose Port and Verify
 - + Go to Tool => Serial Monitor
 - + Upload to see the Fire Detection Status
- Hardware:
 - + Test by press the button to see if the amplifier work or not

VI/ Intruder Detection System:

a/ Arduino Assembly:



b/ Components:

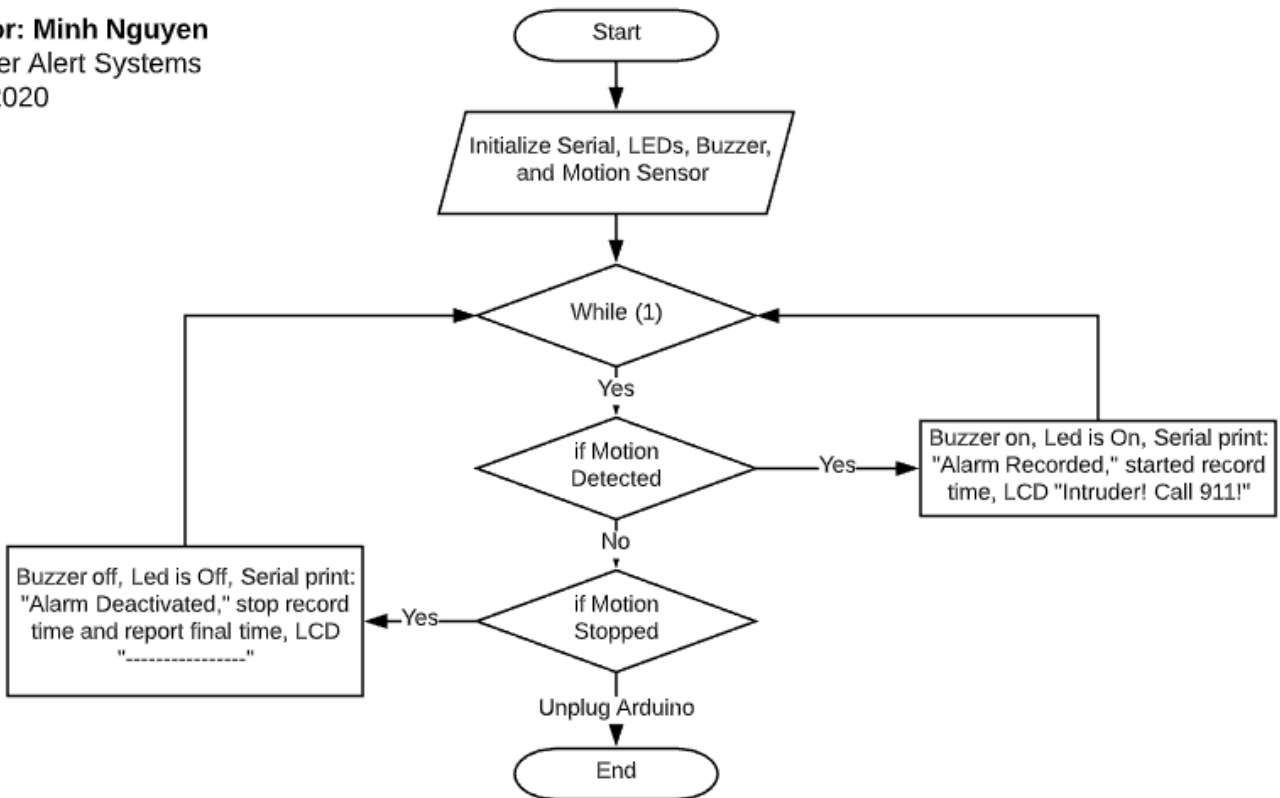
- LED
- 3-Pin Buzzer (not a speaker, we can use 2 pin buzzer alternatively)
- Motion Sensor
- 16x2 LCD Screen
- Resistor Calculation (Not Really need a resistor in real world, but required for TinkerCAD):
 - + V_{cc} of Arduino pin = 5V
 - + V needed for LED = 0.7V
 - + Efficient 20mA
 - + $R = (5-0.7)/0.02 = 235 \text{ Ohm} \Rightarrow$ Use 270 Ohm in the ECE 1004 Kit

c/ Flowchart:

***Author: Minh Nguyen**

***Intruder Alert Systems**

***5/30/2020**



d/ Code:

/**

* Author: Minh Nguyen

* Motion Sensor Control

* 6/5/2020

*/

#include <LiquidCrystal.h>

#include <SoftwareSerial.h>

//Initialize the library by associating ICD interface pin with the arduino pin number

const int rs = 12, en = 11, d4 = 7, d5 = 6, d6 = 5, d7 = 4;

LiquidCrystal lcd(rs,en, d4, d5, d6, d7);

//the time when the sensor outputs a low impulse

long unsigned int lowIn;

```
//the amount of milliseconds the sensor has to be low before we assume all motion has stopped
int pause = 5000;
```

```
boolean lockLow = true;
boolean takeLowTime;
```

```
int pirPin = 9; //the digital pin connected to the PIR sensor's output
int buzzerPin = 8;
```

```
//////////
```

```
void setup(){
  Serial.begin(9600);
  pinMode(pirPin, INPUT);
  digitalWrite(pirPin, LOW);
  pinMode(buzzerPin, OUTPUT);
  lcd.begin(16,4); // Initialize the LCD with column and row
  lcd.clear();
```

```
  //give the sensor some time to calibrate
  // Serial.print("calibrating sensor ");
  // for(int i = 0; i < calibrationTime; i++){
  //   Serial.print(".");
  //   delay(1000);
  // }
  // Serial.println(" done");
  // delay(50);
  }
```

```
//////////
```

```
void loop(){
  lcd.setCursor(0,1);
  if(digitalRead(pirPin) == HIGH){
    tone(buzzerPin, 150);
    if(lockLow){
      //makes sure we wait for a transition to LOW before any further output is made:
      lockLow = false; // reset lock low
```

```

    lcd.print("Thief! Call 911!");
    Serial.print("Intruder detected at ");
    Serial.print(millis()/1000);
    Serial.println(" sec");
    delay(100);
}
takeLowTime = true;
}

if(digitalRead(pirPin) == LOW){
    noTone(buzzerPin);
    if(takeLowTime){
        lowIn = millis();    //save the time of the transition from high to LOW
        takeLowTime = false;    //make sure this is only done at the start of a LOW phase
    }
    //if the sensor is low for more than the given pause,
    //we assume that no more motion is going to happen
    if(!lockLow && millis() - lowIn > pause){
        //makes sure this block of code is only executed again after
        //a new motion sequence has been detected
        lockLow = true;
        lcd.print("-----");
        Serial.print("Intruder stopped at ");    //output
        Serial.print((millis() - pause)/1000);
        Serial.println(" sec");
        delay(100);
    }
}
}
}

```

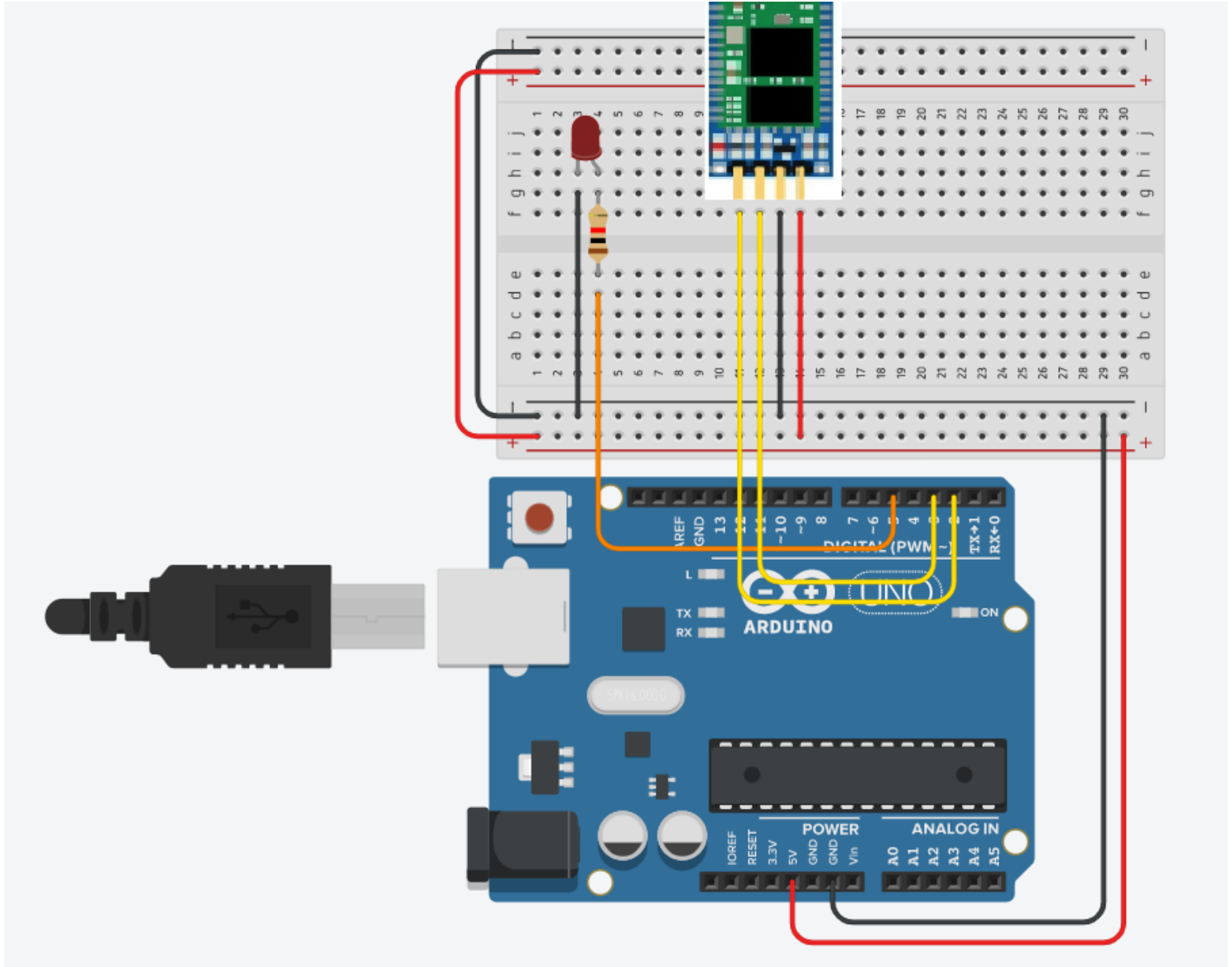
e/ Testing:

- Software:
 - + Copy the code to Arduino IDE
 - + Choose Port and Verify
 - + Go to Tool => Serial Monitor
 - + Upload to see the Intruder Status
- Hardware:
 - + Test by moving your hand around the motion sensor. If there is motion, Buzzer and LED will be on, Serial monitor will print out the time, and LCD print out the alert.

VII/ Indoor Light/Stove/Computer...etc (Bluetooth)

(MIT App is the below section)

a/ Arduino Assembly:



b/ Components:

- Arduino Bluetooth module hc-06
- LED
- Resistor Calculation:
 - + Vcc of Arduino pin = 5V
 - + V needed for LED = 0.7V

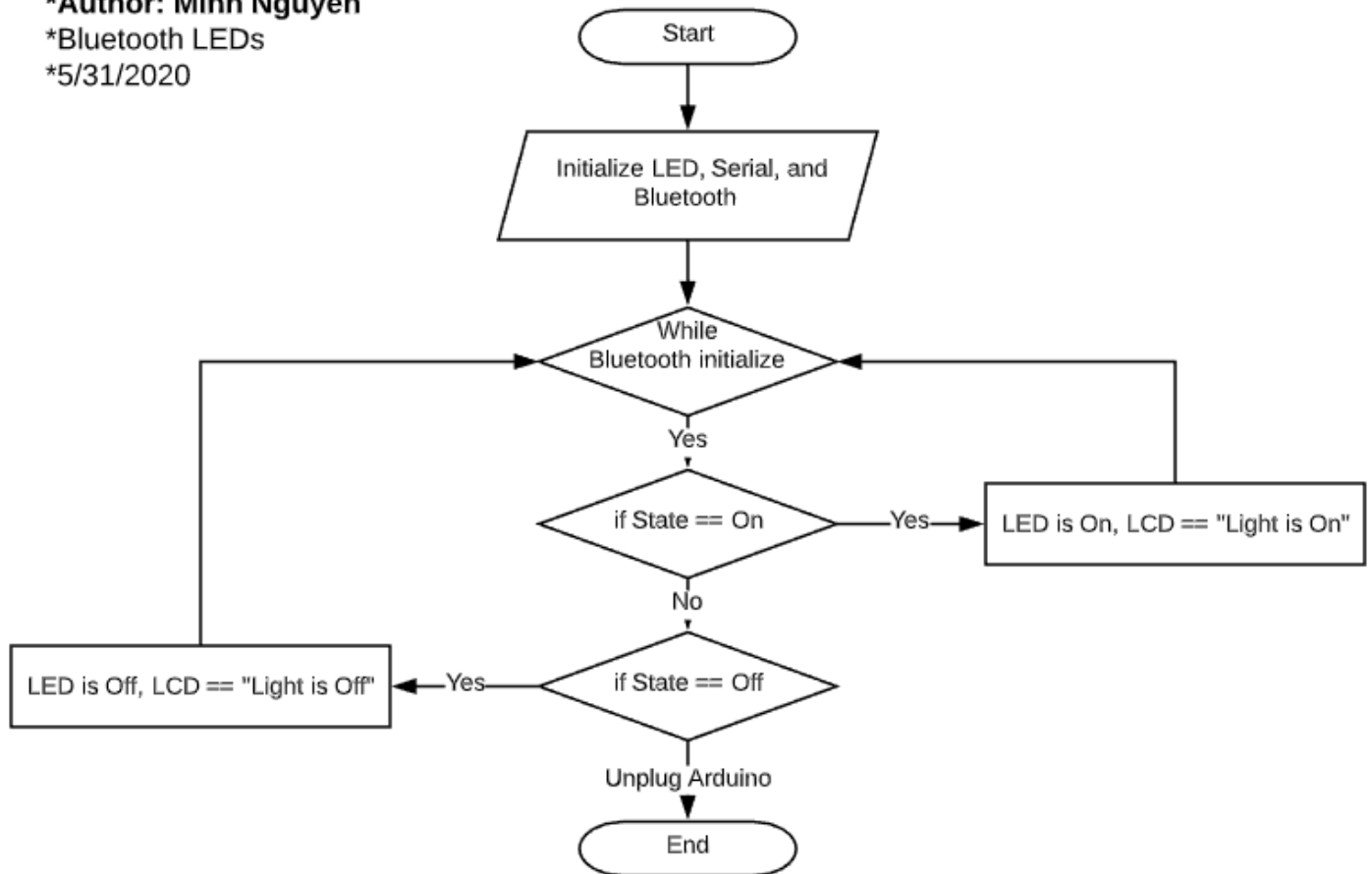
- + Efficient 20mA
- + $R = (5 - 0.7) / 0.02 = 235 \text{ Ohm} \Rightarrow$ Use 270 Ohm in the ECE 1004 Kit

c/ Flowchart:

***Author: Minh Nguyen**

***Bluetooth LEDs**

***5/31/2020**



d/ Code:

// Author: Minh T. Nguyen

// 5/31/2020

#include <SoftwareSerial.h>

SoftwareSerial BT(3,2); //TX, RX of the BT respectively.

String state;// string to store the incoming message from Bluetooth

void setup() {

BT.begin(9600);// bluetooth serial communication will happen on pin 10 and 11

Serial.begin(9600); // serial communication to check the data on serial monitor

```
pinMode(5, OUTPUT); // LED connected to pin 5
}
//-----//
void loop() {
  while (BT.available()){ //Check if there is an available byte to read
    delay(10); //Delay added to make thing stable
    char c = BT.read(); //Conduct a serial read
    state += c; //build the string- either "On" or "off"
  }
  if (state.length() > 0) {
    Serial.println(state);
    if(state == "On") // Check
    {
      digitalWrite(5, HIGH);
    }
    else if(state == "Off")
    {
      digitalWrite(5, LOW);
    }
    state = "";
  }
}
```

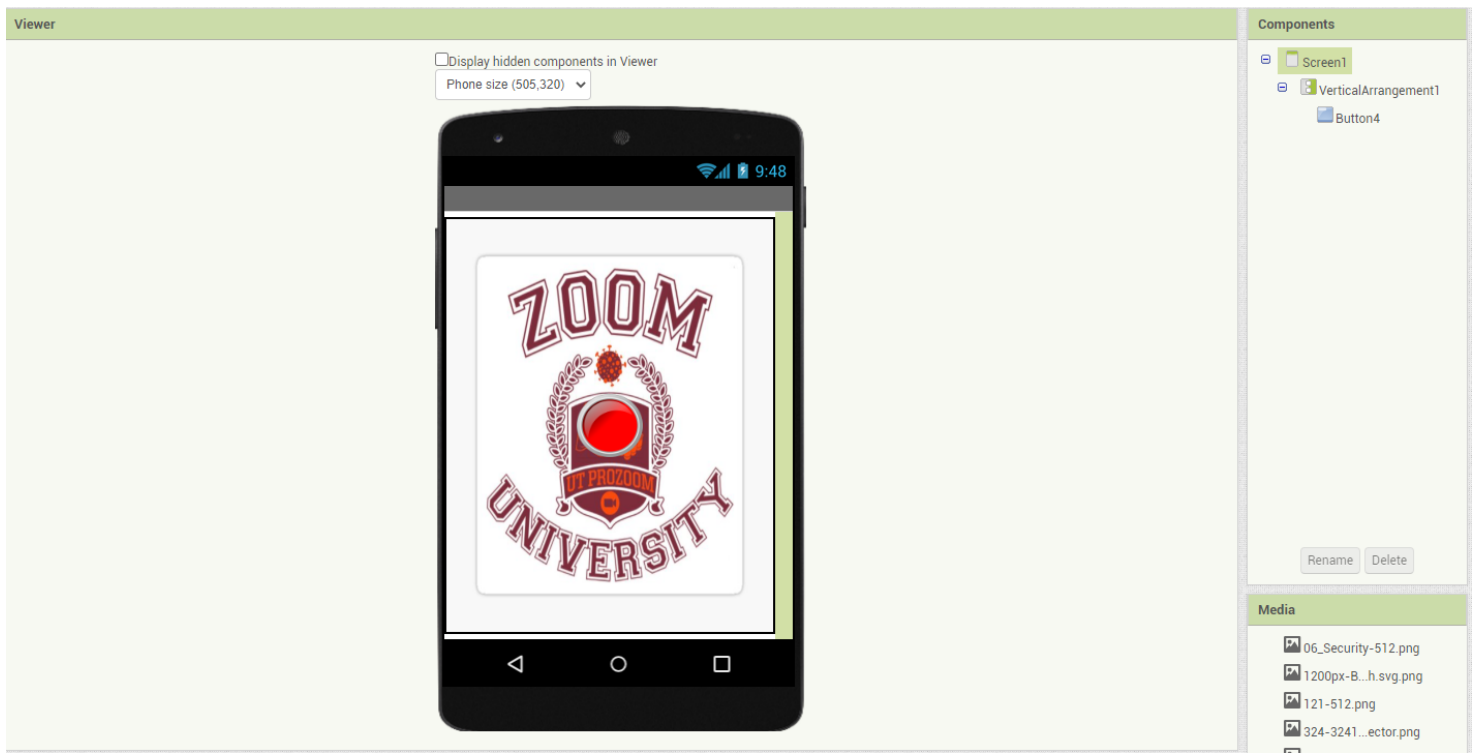
e/ Testing:

- Software:
 - + Copy the code to Arduino IDE
 - + Choose Port and Verify
 - + Go to Tool => Serial Monitor
 - + Upload to see the LED Status
- Hardware:
 - + Connect the App and the Bluetooth, then turn on and off the phone to see if it actually work
 - + Link for the App:

<http://ai2.appinventor.mit.edu/b/71ld>

VIII/ MIT App Inventor: (Link: <http://ai2.appinventor.mit.edu/b/71ld>)

a/ Flash Screen + Code:



*Logistic:

- This is just a flash screen for decoration, you don't need to have it. (Screen is named "Screen1").
- When the button is pressed, then the app go to the main screen called "Light"

*Implementation in Designer section (Follow the Components column above):

- First, drag in the Vertical Arrangement, adjust the properties below, and choose any image. Then, drag the Button into the Vertical Arrangement, choose any image, and adjust the properties below.
- The Vertical Arrangement can have any name, but make sure the button name Button4.

Properties

VerticalArrangement1


AlignHorizontal

Center : 3 ▾

AlignVertical

Center : 2 ▾

BackgroundColor

 Default

Height

Fill parent...

Width

Fill parent...

Image

Bruh.jpg...

Visible

☒

Properties

Button4

BackgroundColor

☐ None

Enabled

☒

FontBold

☒

FontItalic

☐

FontSize

15

FontTypeface

default ▾

Height

15 percent...

Width

20 percent...

Image

button-clipart-push-button.p

Shape

default ▾

ShowFeedback

☒

Text

TextAlignment

center : 1 ▾

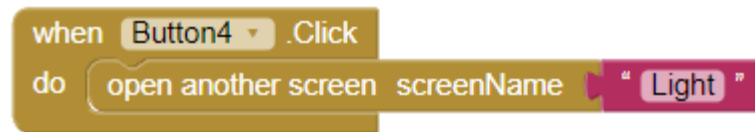
TextColor

☐ White

Visible

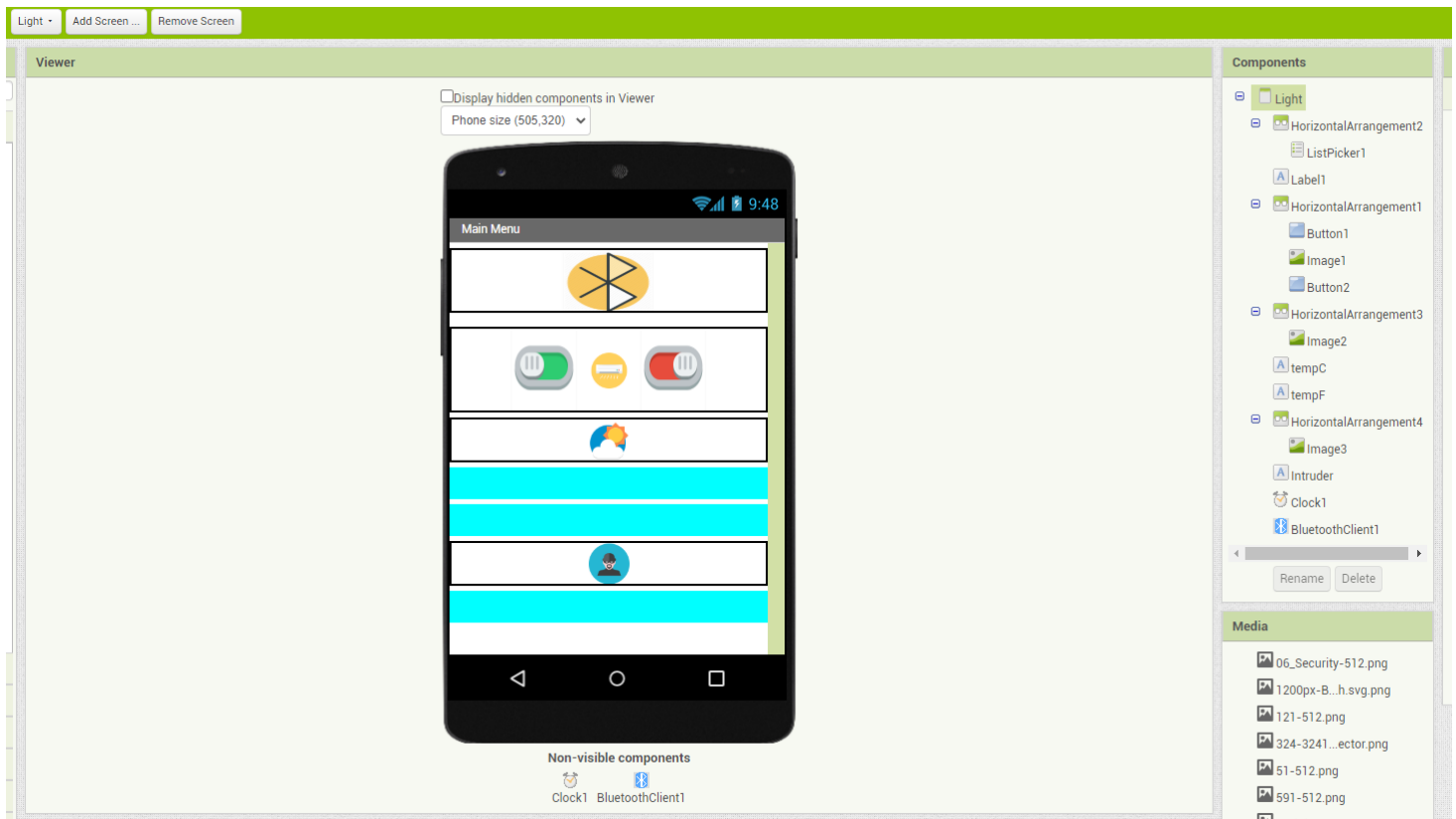
☒

***Implementation in the Blocks Section:**



In the Blocks Section, do:

b/ Main Screen + Code:



***Logistics:**

- This is the main screen of the App which displays the usage of Indoor Light System, Weather Station, and the Intruder Detection System. The first section allows the user to connect the phone with the Arduino Uno via Bluetooth device. The second section shows the control of the AC Unit/ Indoor light on/off state. The third section outputs the temperature in F and C in real time. The last section provides the recorded time of the intruder's motion.
- Screen is named "Light"

***Implementation in Designer section (Follow the Components column above):**

- Firstly, drag in the Horizontal Arrangement and do the properties below. Then drag a ListPicker into the Horizontal Arrangement and do the properties below. This is a section for Bluetooth. Choose any image of the Bluetooth you preferred. The Horizontal Arrangement can have any name, but the List Picker must be named "ListPicker1."

Properties

HorizontalArrangement2


AlignHorizontal

Center : 3 ▾

AlignVertical

Top : 1 ▾

BackgroundColor

 None

Height

15 percent...

Width

Fill parent...

Image

None...


Visible

☒

Properties

ListPicker1

BackgroundColor

 Default

ElementsFromString

Enabled

☒

FontBold

☒

FontItalic

☐

FontSize

18

FontTypeface

default ▾

Height

15 percent...


Width

28 percent...

Image

121-512.png...

ItemBackgroundColor

 Default

ItemTextColor

☐ Default

Selection

Shape

default ▾

ShowFeedback

☒

ShowFilterBar


☐

Text

TextAlignment

center : 1 ▾

TextColor

 Default

Title

Visible

☒

- Secondly, drag in the Label below the Horizontal Arrangement (not inside) and name it Label1. Adjust the properties below. The Label will let the user if the Bluetooth is connected or not.

Properties

Label1

BackgroundColor

☐ None

FontBold

☐

FontItalic

☐

FontSize

14.0

FontTypeface

default ▾

HTMLFormat

☐

HasMargins

☒

Height

Automatic...

Width

Fill parent...

Text

TextAlignment

center : 1 ▾

TextColor

☒ Default

Visible

☒

- Thirdly, Drag in the Horizontal Arrangement (can change it to any name). Inside the Horizontal Arrangement, add in 2 buttons named "Button1" and "Button2." Choose any Image for the Button1, Button2, and Image1 Adjust the Horizontal Arrangement, Button1, Button2, and Image1 with the properties below. This section allows the user to control the LED's on/off state via Bluetooth.

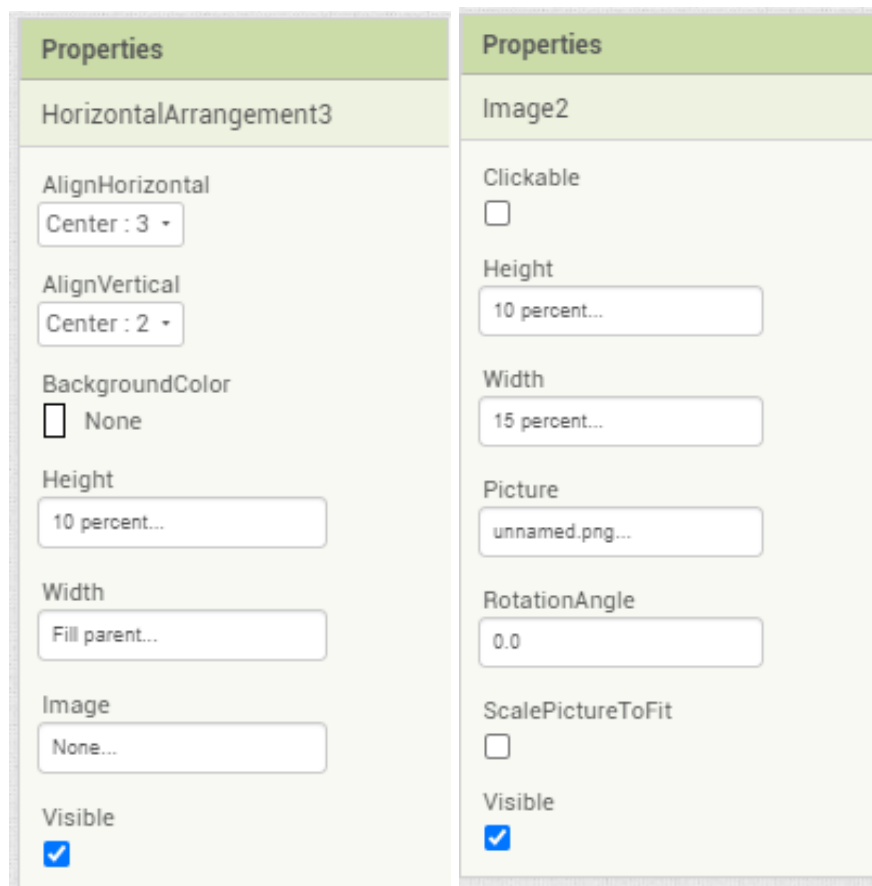
Properties
HorizontalArrangement1
AlignHorizontal Center : 3 ▾
AlignVertical Center : 2 ▾
BackgroundColor <input type="checkbox"/> White
Height 20 percent...
Width Fill parent...
Image None...
Visible <input checked="" type="checkbox"/>

Properties
Button1
BackgroundColor <input type="checkbox"/> White
Enabled <input checked="" type="checkbox"/>
FontBold <input checked="" type="checkbox"/>
FontItalic <input type="checkbox"/>
FontSize 20
FontTypeface default ▾
Height 20 percent...
Width 20 percent...
Image switch-on-icon.png...
Shape default ▾
ShowFeedback <input checked="" type="checkbox"/>
Text On
TextAlignment center : 1 ▾
TextColor <input type="checkbox"/> None
Visible <input checked="" type="checkbox"/>

Properties
Image1
Clickable <input type="checkbox"/>
Height 10 percent...
Width 15 percent...
Picture unnamed(1).png...
RotationAngle 0.0
ScalePictureToFit <input type="checkbox"/>
Visible <input checked="" type="checkbox"/>

Properties
Button2
BackgroundColor <input checked="" type="checkbox"/> Default
Enabled <input checked="" type="checkbox"/>
FontBold <input checked="" type="checkbox"/>
FontItalic <input type="checkbox"/>
FontSize 20
FontTypeface default ▾
Height 20 percent...
Width 20 percent...
Image switch-off-icon.png...
Shape rectangular ▾
ShowFeedback <input checked="" type="checkbox"/>
Text Off
TextAlignment center : 1 ▾
TextColor <input type="checkbox"/> None
Visible <input checked="" type="checkbox"/>

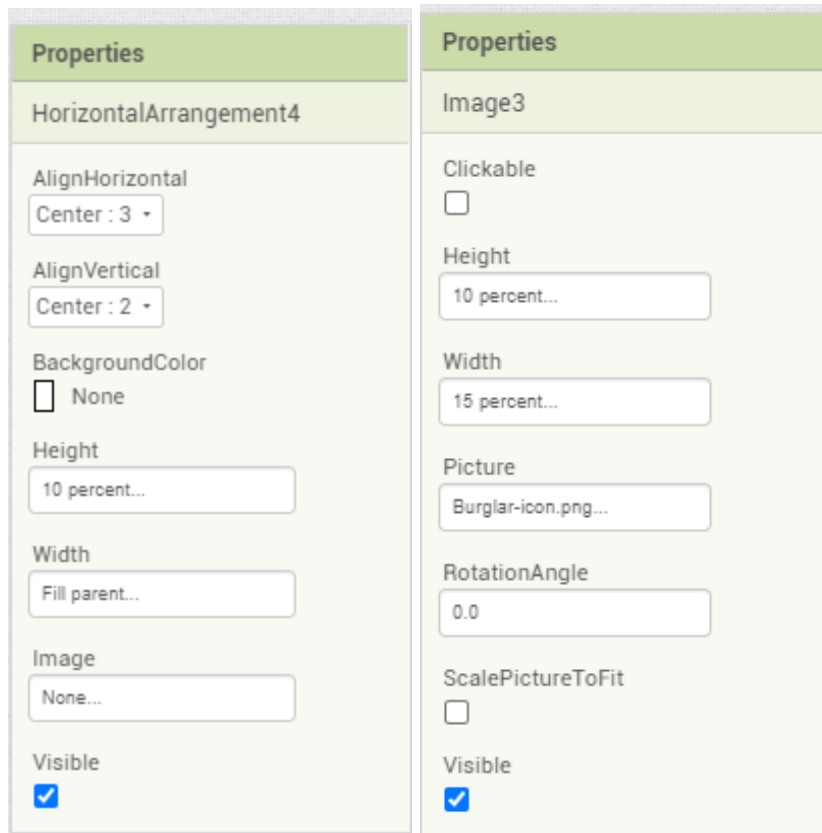
- Fourthly, drag in another Horizontal Arrangement. Then drag in an image block of the Weather Station in. Choose any image you prefer. This block is just for decoration. Adjust the properties of the Horizontal Arrangement and the Image like below.



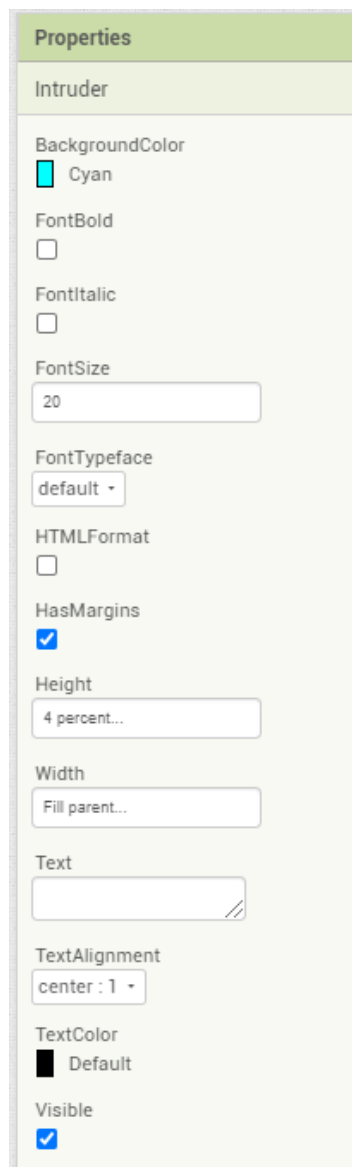
- Fifthly, drag in 2 labels below the Horizontal Arrangement and name them “tempC” and “tempF” respectively. Adjust the properties of the 2 labels like below. These section allo

Properties	Properties
tempC	tempF
BackgroundColor <input type="color"/> Cyan	BackgroundColor <input type="color"/> Cyan
FontBold <input type="checkbox"/>	FontBold <input type="checkbox"/>
FontItalic <input type="checkbox"/>	FontItalic <input type="checkbox"/>
FontSize <input type="text" value="20"/>	FontSize <input type="text" value="20"/>
FontTypeface <input type="text" value="default"/>	FontTypeface <input type="text" value="default"/>
HTMLFormat <input type="checkbox"/>	HTMLFormat <input type="checkbox"/>
HasMargins <input checked="" type="checkbox"/>	HasMargins <input checked="" type="checkbox"/>
Height <input type="text" value="5 percent..."/>	Height <input type="text" value="5 percent..."/>
Width <input type="text" value="Fill parent..."/>	Width <input type="text" value="Fill parent..."/>
Text <input type="text"/>	Text <input type="text"/>
TextAlignment <input type="text" value="center : 1"/>	TextAlignment <input type="text" value="center : 1"/>
TextColor <input type="color"/> Default	TextColor <input type="color"/> Default
Visible <input checked="" type="checkbox"/>	Visible <input checked="" type="checkbox"/>

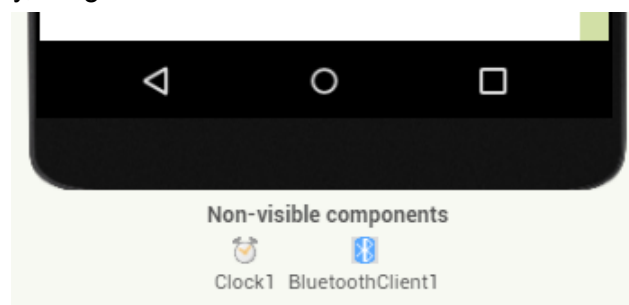
- Sixthly, drag in the Horizontal Arrangement right below the two labels. Then, drag the image block inside the Horizontal Arrangement and choose any preferred image. This section is just for decoration for the Intruder System. Adjust the properties of the Horizontal Arrangement and the Image block like below.



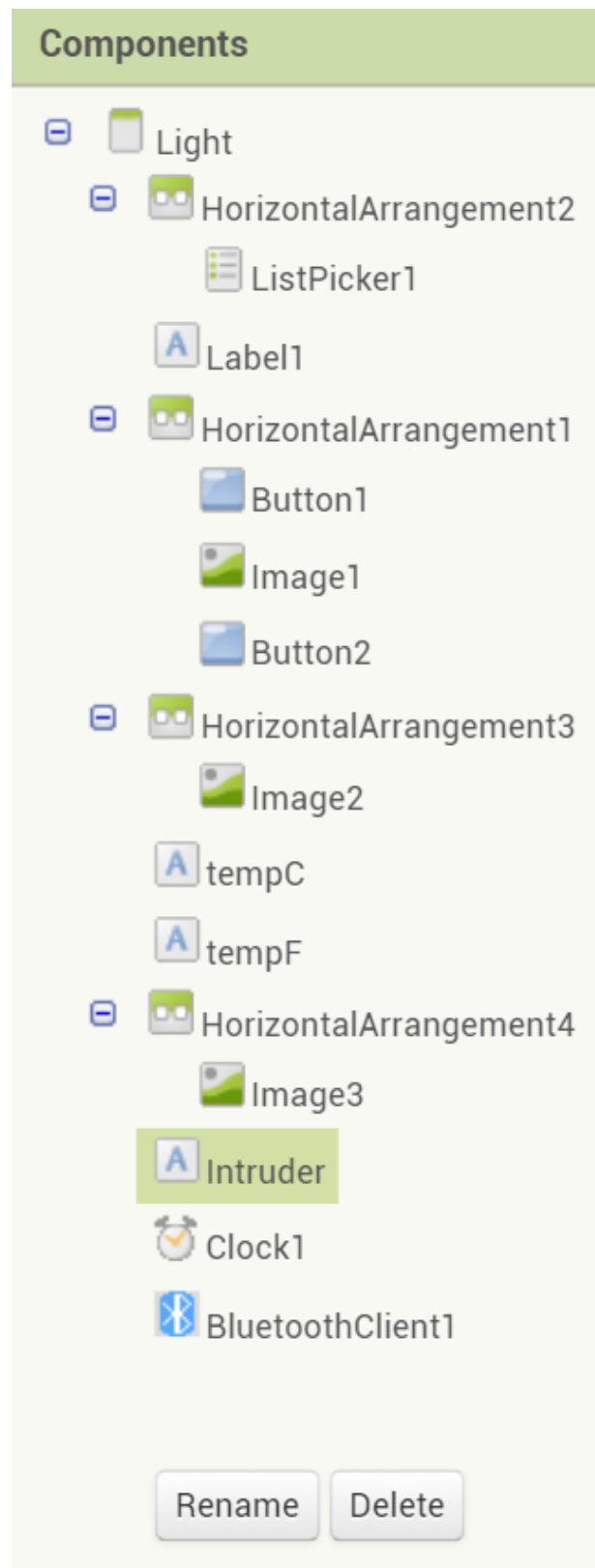
- Sevenly, drag in a label below the Horizontal Arrangement and adjust the properties like below. This section prints out the recorder time of the Intruder and alert to the user. Make sure it is named "Intruder"



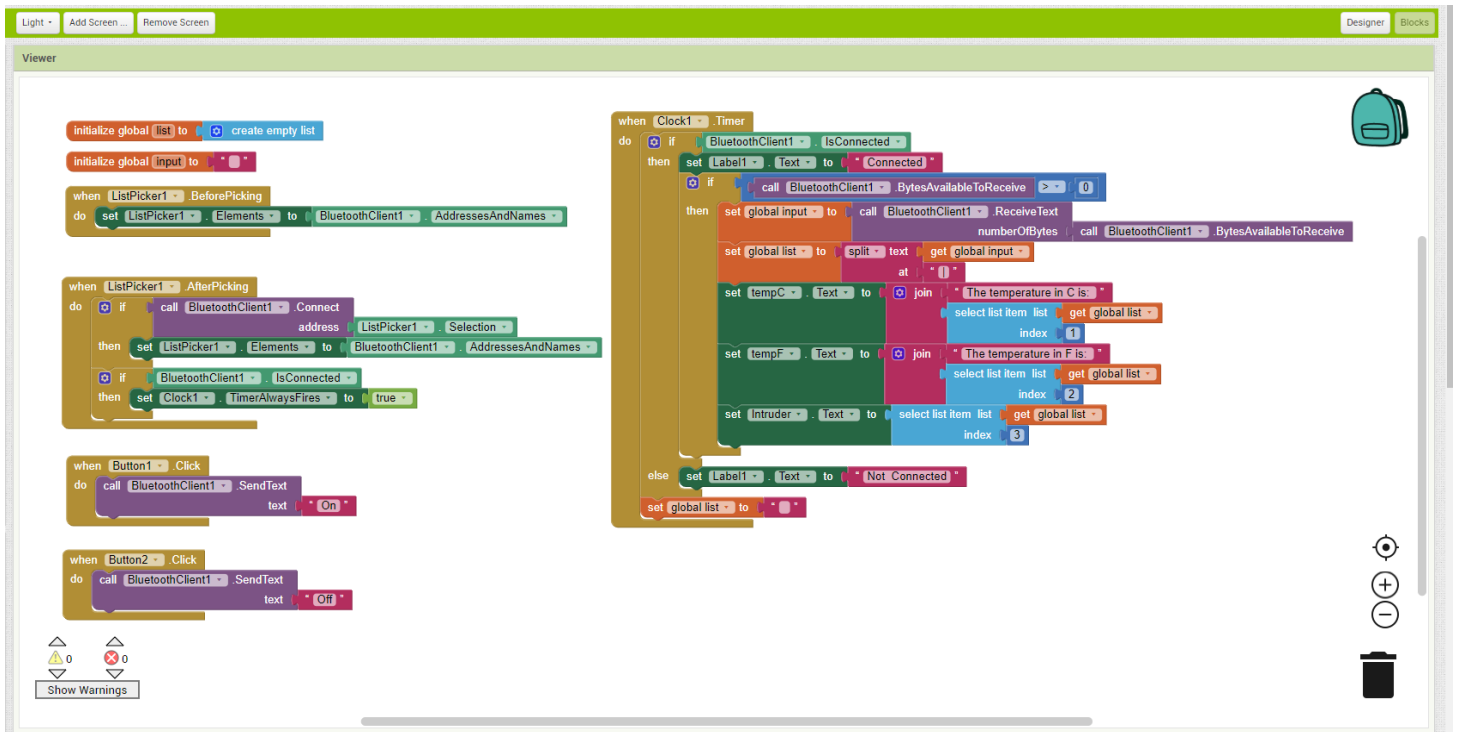
- Eightly, drag in a Clock and A BluetoothClient to the screen.



- Lastly, check the names and position of each component.

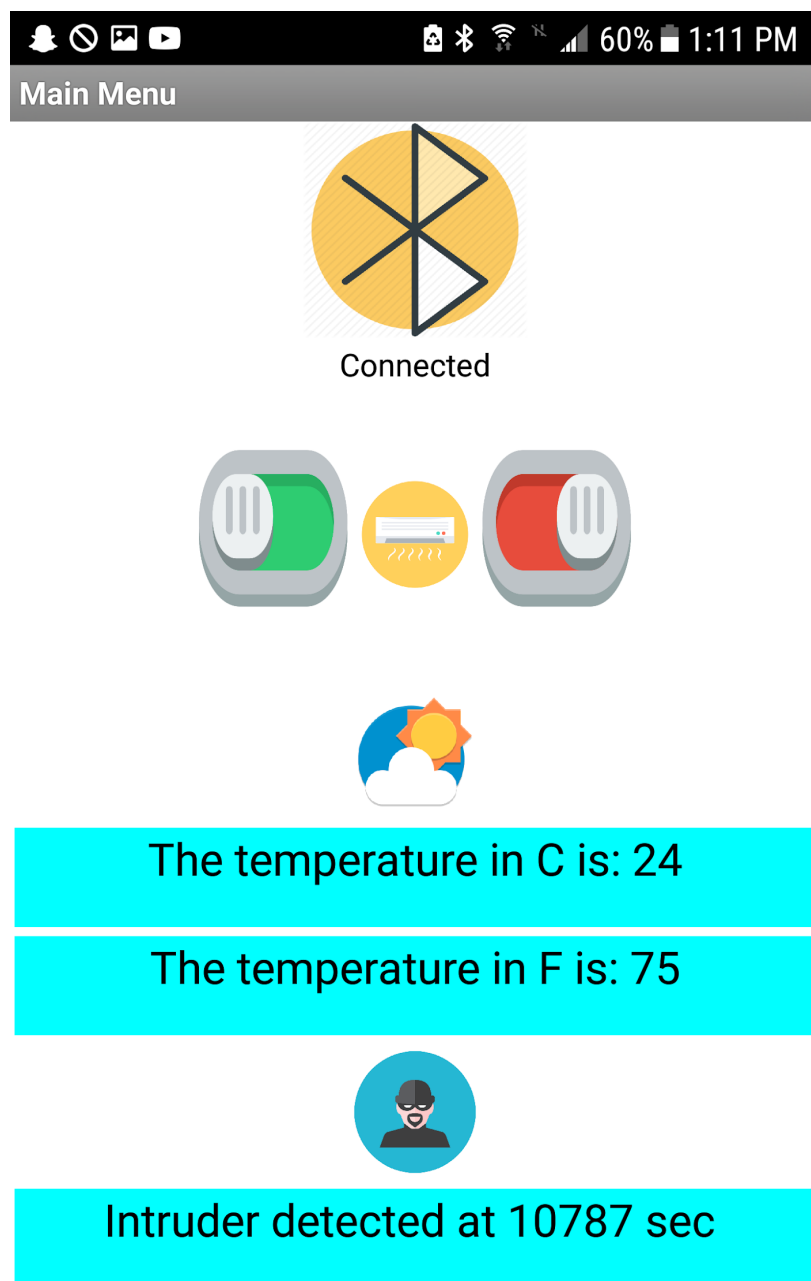


***Implementation in the Blocks Section:**



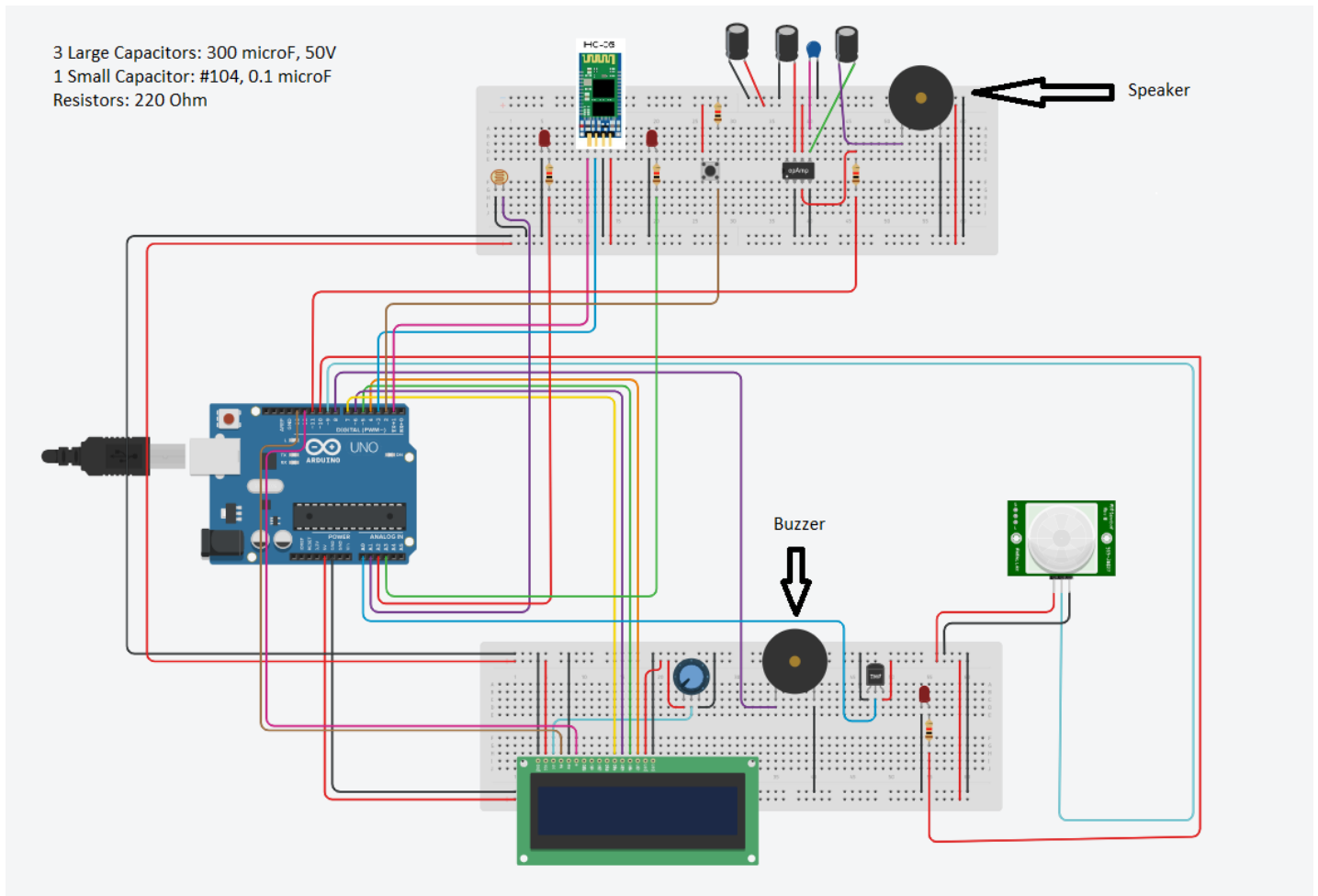
- To begin, we declare a global empty list and a global input.
- Then, set up the connection of ListPicker1 and the BluetoothClient1 before picking (or connecting)
- Then, set up the connection of ListPicker and the BluetoothClient1 after picking. If the Bluetooth of the phone is paired with the Arduino Bluetooth when the ListPicker is selected, then the phone and microcontroller are paired. Then, if the system is paired, then the Clock will be started (real-time runner for TX-RX connection).
- Then, set up the ON/OFF state, if the Button1 is clicked, then the Bluetooth will send the "On" string to the Serial Monitor which then turns on the LED. If the Button2 is clicked, then the Bluetooth will send the "Off" string to the Serial Monitor which then turns off the LED. (This is a transmitting data from MIT App to Arduino Uno via Bluetooth process.)
- Then, set up the receiving data from the Arduino Uno to the MIT App via Bluetooth process. When the clock is run and the BluetoothClient1 is connected, then Label1 prints "Connected" on the MIT App, else, the Label1 prints "Not Connected" on the MIT App. Next, add in a condition if the global input receive any data from the Arduino Uno, then convert it to a list which contains 3 pieces of informations splitted by "|".
 - + Label "tempC" print "The temperature in C is" + the 1st piece of data from global list
 - + Label "tempF" print "The temperature in F is" + the 2nd piece of data from global list
 - + Label "Intruder" print the 3rd piece of data from the global list
- Set the global list to be empty to receive a new global list.

c/ Result:



IX/ Final Assembly:

a/ Arduino Assembly:



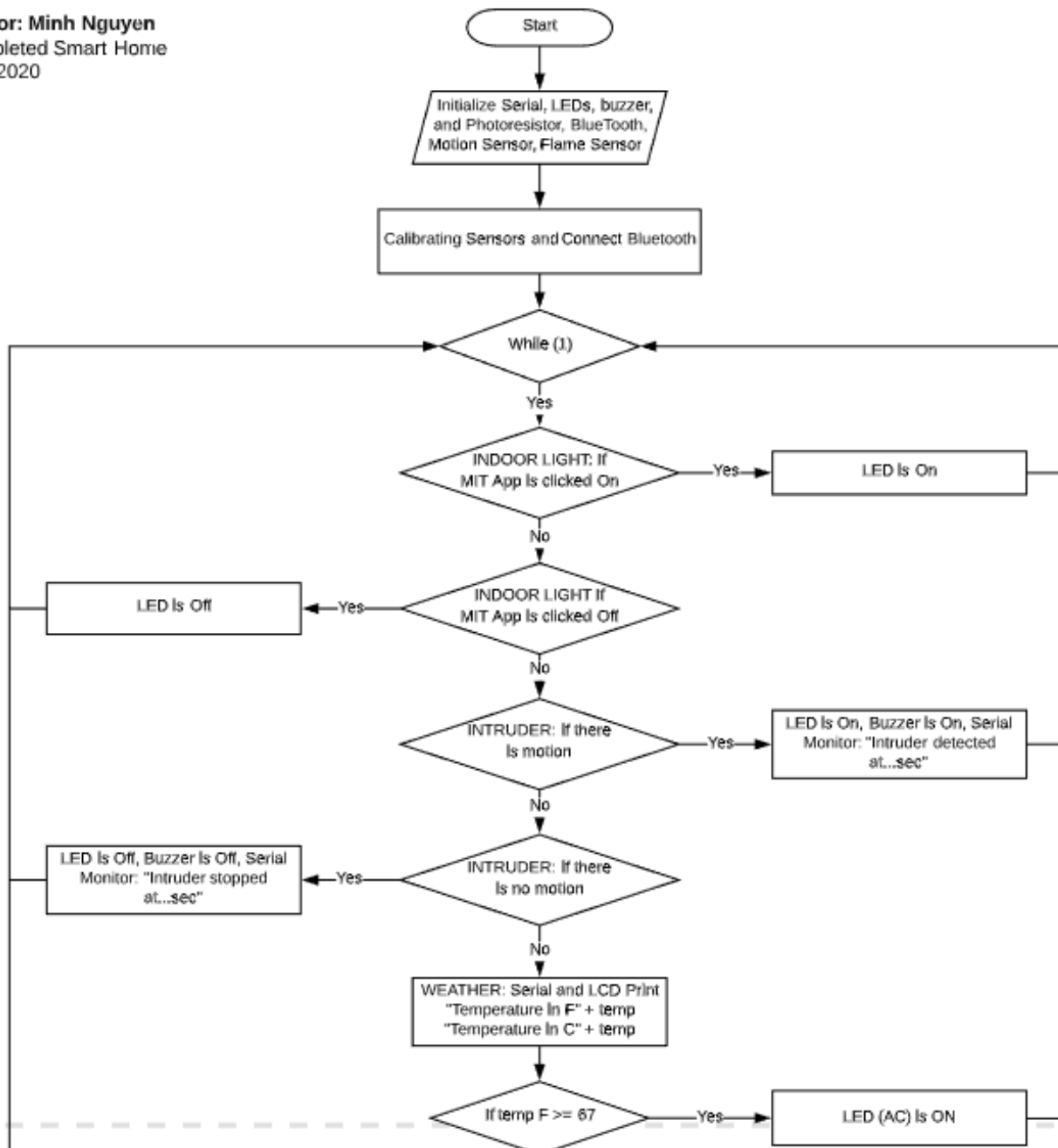
b/ Components need for a complete project:

Functionalities	Components	Quantities
General	Arduino Uno	1
	Soldered 16x2 LCD	1
	Breadboard	2
	Potentiometer	1
	Capacitor (Different depend on calculation and application)	4

	Resistors (Different depend on calculation and application)	5
Weather Station + Auto AC	LED	1
	Temperature Sensor	1
Automated Night Light	Photoresistor	1
	LED	1
Bluetooth Indoor System	LED	1
	Bluetooth model hc-06 (the model you provided)	1
Intruder System	Motion Sensor model HC-SR501	1
	Arduino Active Buzzer model KY-012 or regular Arduino buzzer	1
Fire Detection System	Amplifier LM386	1
	Button	1
	Speaker	1

c/ Flowchart:

*Author: Minh Nguyen
 *Completed Smart Home
 *5/30/2020





- * Author: Minh Nguyen
- * 6/9/2020
- * Ultimate Merge for
 - * Intruder Detection System
 - * Weather Station + AC Unit
 - * Automated Night Light
 - * Fire Alarm System
 - * Indoor Light
- * Tip for Intruder: <https://www.youtube.com/watch?v=FxaTDvs34mM>
- * /

////////////////////////////////////

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

```
#include <Arduino.h>
#include "Talkie.h"
#include "Vocab_US_Large.h"
#include <PCM.h>
```

```
/**-----
```

```
 * Initialization for Intruder Detection System
```

```
 */
```

```
//Initialize the library by associating ICD interface pin with the arduino pin number
```

```
const int rs = 13, en = 12, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
```

```
LiquidCrystal lcd(rs,en, d4, d5, d6, d7);
```

```
//the time when the sensor outputs a low impulse
```

```
long unsigned int lowIn;
```

```
//the amount of milliseconds the sensor has to be low before we assume all motion has stopped
```

```
int pause = 5000;
```

```
boolean lockLow = true;
```

```
boolean takeLowTime;
```

```
int pirPin = 9; //the digital pin connected to the PIR sensor's output
```

```
int buzzerPin = 8;
```

```
/**-----
```

```
 * Initialization for Weather Station + AC Unit
```

```
 */
```

```
int sensorPin = A0; // Adjustment for what Analog Pin, currently A0
```

```
int tempC, tempF;
```

```
int AC = 10;
```

```
int get_temperature(int pin) {
```

```
    int temperature = analogRead(pin);
```

```
    float voltage = temperature * 5.0;
```

```
    voltage = voltage / 1024.0;
```

```
    return ((voltage - 0.5) * 100);
```

```
}  
int celsius_to_fahrenheit(int temp) {  
    return (temp * 9.0 / 5.0) + 32.0;  
}
```

```
/**-----
 * Initialization Automated Night Light
 */
```

```
int ledPinRED = A2;  
int prPin = A1; // Photoresistor Pin
```

```
/**-----
 * Initialization for the BT Indoor Light
 */
```

```
SoftwareSerial BT(3,1); //TX, RX of the BT respectively connect to RX, TX of Arduino
//(Can't use pin 0 instead of 3 as Arduino can't receive it for some reason).
String state;// string to store incoming message from bluetooth
```

```
/**-----
 * Initialization for Push Fire Alert
 */
```

```
//Talkie voice;
```

```
int buttonPin = 2;  
int buttonState = 0;    // variable for reading the pushbutton status
```

```
const unsigned char sample[] PROGMEM = {
```

[illegible]

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 84

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 84

}.

```
void setup(){
```

/ **

* Set up for Intruder Detection System

*/

```
Serial.begin(9600);
```

```
pinMode(pirPin, INPUT);
```

```
digitalWrite(pirPin, LOW);
```

```
pinMode(buzzerPin, OUTPUT);
```

```
lcd.begin(16,2); // Initialize the LCD with column and row
```

```
lcd.clear();
```

/ **

* Set up for Weather Station + AC Unit


```

*/
pinMode(AC, OUTPUT);

/**-----
 * Set up for Automated Night Light
 */
pinMode(ledPinRED, OUTPUT);
pinMode(prPin, INPUT);

/**-----
 * Set up for BT Indoor Light
 */
BT.begin(9600); // bluetooth serial communication will happen on pin 10 and 11
pinMode(A3, OUTPUT); // LED connected to 13th pin

/**-----
 * Set Up for Fire Alert System
 */
pinMode(buttonPin, INPUT);
// pinMode(11, OUTPUT);
// digitalWrite(11, HIGH); //Enable Amplified PROP shield
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void loop(){
    String ret = "";

    /**-----
    * Loop for BT Indoor Light
    */
    while (BT.available()){ //Check if there is an available byte to read
        delay(10); //Delay added to make thing stable
        char c = BT.read(); //Conduct a serial read
        state += c; //build the string- either "On" or "off"
    }
    if (state.length() > 0) {
        if(state == "On") // Check
        {
            digitalWrite(A3, HIGH);

```

```

    }
    else if(state == "Off")
    {
        digitalWrite(A3, LOW);
    }
    state = "";
}

/**-----
 * Loop for Weather Station + AC Unit
 */
tempC = get_temperature(sensorPin);
tempF = celsius_to_fahrenheit(tempC);

lcd.setCursor(0,0);
lcd.print("Temp: ");

lcd.setCursor(6,0);
lcd.print(tempF); lcd.print((char)223); lcd.print("F");

lcd.setCursor(12,0);
lcd.print(tempC); lcd.print((char)223); lcd.print("C");

//This is a condition to turn the AC on.
if(tempF >= 67)
{
    digitalWrite(AC, HIGH);
}
else if(tempF < 67)
{
    digitalWrite(AC, LOW);
}
ret += tempC;
ret += "|";
ret += tempF;
ret += "|";
delay(200);

```

```

/**-----
 * Loop for Intruder System
 */
lcd.setCursor(0,1);
if(digitalRead(pirPin) == HIGH){
  tone(buzzerPin, 150);
  if(lockLow){
    //makes sure we wait for a transition to LOW before any further output is made:
    lockLow = false; // reset lock low
    lcd.print("Thief! Call 911!");
    ret += "Intruder detected at ";
    ret += millis()/1000;
    ret += " sec";
    delay(100);
  }
  takeLowTime = true;
}

if(digitalRead(pirPin) == LOW){
  noTone(buzzerPin);
  if(takeLowTime){
    lowIn = millis();    //save the time of the transition from high to LOW
    takeLowTime = false; //make sure this is only done at the start of a LOW phase
  }
  //if the sensor is low for more than the given pause,
  //we assume that no more motion is going to happen
  if(!lockLow && millis() - lowIn > pause){
    //makes sure this block of code is only executed again after
    //a new motion sequence has been detected
    lockLow = true;
    lcd.print("-----");
    ret += "Intruder stopped at ";
    ret += (millis() - pause)/1000;
    ret += " sec";
    delay(100);
  }
}
Serial.println(ret);
ret = "";

```

```

/**-----
 * Loop for Automated Night Light
 */
int prStatus = analogRead(prPin);
if (prStatus >= 300){
    digitalWrite(ledPinRED, HIGH);
}
else{
    digitalWrite(ledPinRED, LOW);
}

/**-----
 * Loop for Fire Alert System.
 */
// buttonState = digitalRead(buttonPin);
// if (buttonState == HIGH) {
//     voice.say(sp2_DANGER);
//     voice.say(sp2_DANGER);
//     voice.say(sp2_RED);
//     voice.say(sp2_ALERT);
//     voice.say(sp2_FIRE);
//     voice.say(sp2_FIRE);
//     voice.say(sp2_FIRE);
// }
buttonState = digitalRead(buttonPin);
if (buttonState == HIGH) {
    startPlayback(sample, sizeof(sample));
    delay(2000);
}
}

```

e/ Testing:

- Software:
 - + Copy the code to Arduino IDE
 - + Choose Port and Verify
 - + Go to Tool => Serial Monitor
 - + Upload Code to Arduino
 - + Connect to MIT app via Bluetooth
- Hardware:
 - + Test the Weather Station with different temperature => Updates on App and LCD
 - + Test the Automated Night Light by covering and uncovering the Photoresistor
 - + Test the Bluetooth indoor system by controlling on and off on the App.
 - + Test the Motion Sensor by moving and unmoving motion in front of the motion sensor => Updates should be on the App and LCD
 - + Test the Fire Alert System by pressing the button.

*Serial Monitor Prints:

```

32|89|
32|89|Intruder stopped at 7 sec
32|89|
32|89|
32|89|

```

