Building IoT

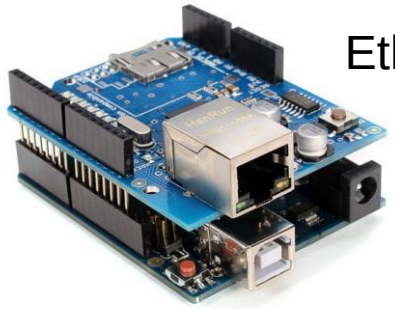
Feter Sitepu

Elyas Khorasani

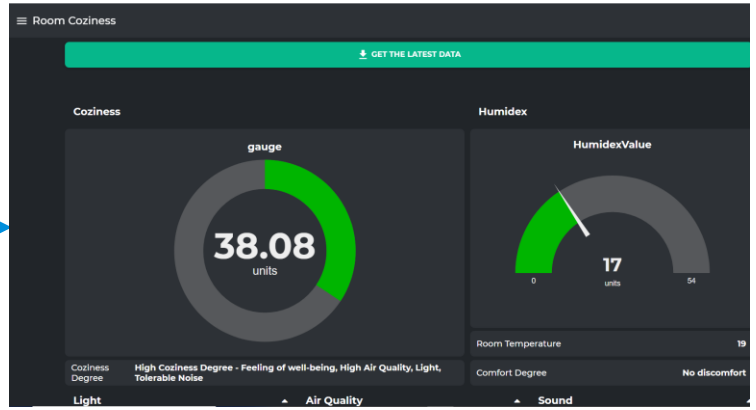
Jan
2021

Contents

Overview of the System



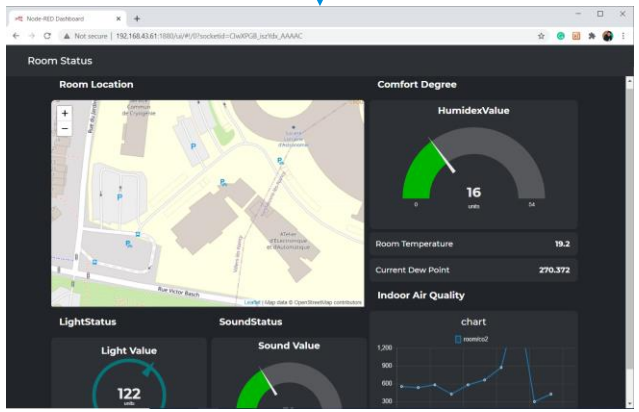
Ethernet Cable



User Interface

MQTT

Node-Red

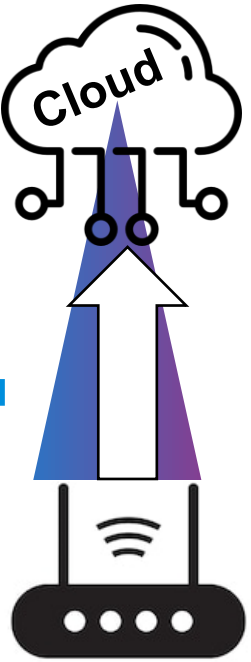


Admin Dashboard

Architecture

Web Client

Web Client



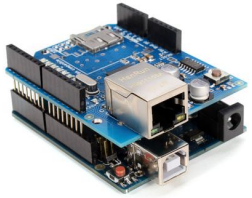
Raspberry Pi

Web Server

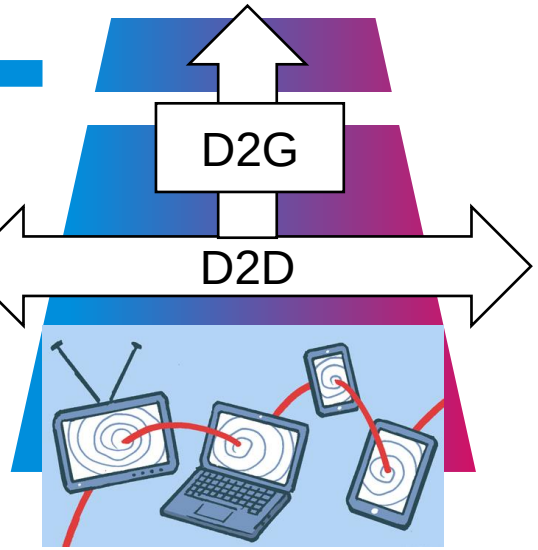
D2G

D2D

ARDUINO



Microcontroller
Sensor



Hardware Choices

Raspberry Pi

Arduino

Raspberry Pi

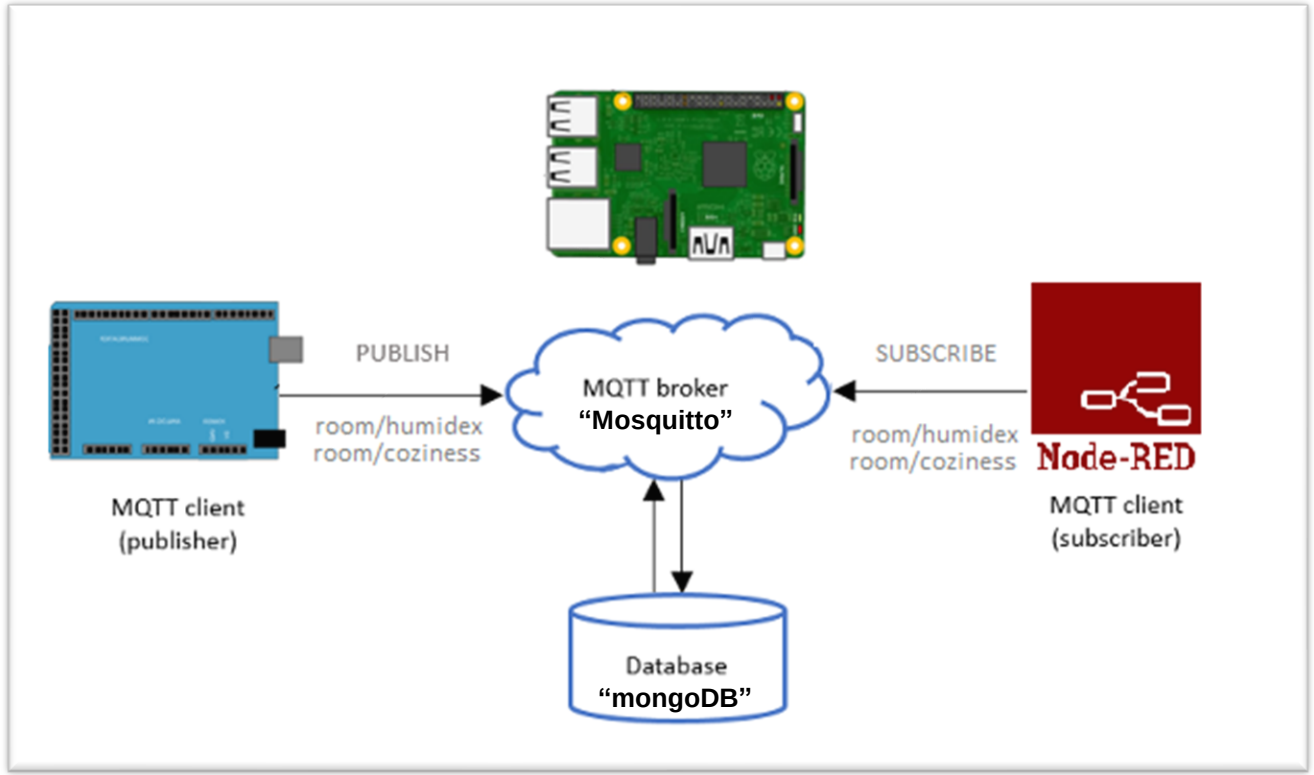
- It is small and portable minicomputer
- Can be used as a small server, make its good match for IoT usage

Arduino

It is a type of microcontroller-based kit.

- Used as a sensor
- Doesn’t require hardware programmer to burn the program
- Simple to use

Communication Protocols



MQTT

- For the communication between Arduino and Raspberry Pi, we used the MQTT protocol.
- Arduino publishes the Sensor data to the Broker (Mosquitto on Rapberry Pi).
- Then, the Node-Red on Raspberry subscribes to the Broker and transmits the data to the Database.

mosquitto

Mosquitto is lightweight and open source.

Design Choices

Database

- **MongoDB's** flexible document data model makes working with data unimaginably easy.
- **MongoDB** makes it easy to store a variety of heterogeneous sensor data in a natural, intuitive way.

Design Choices

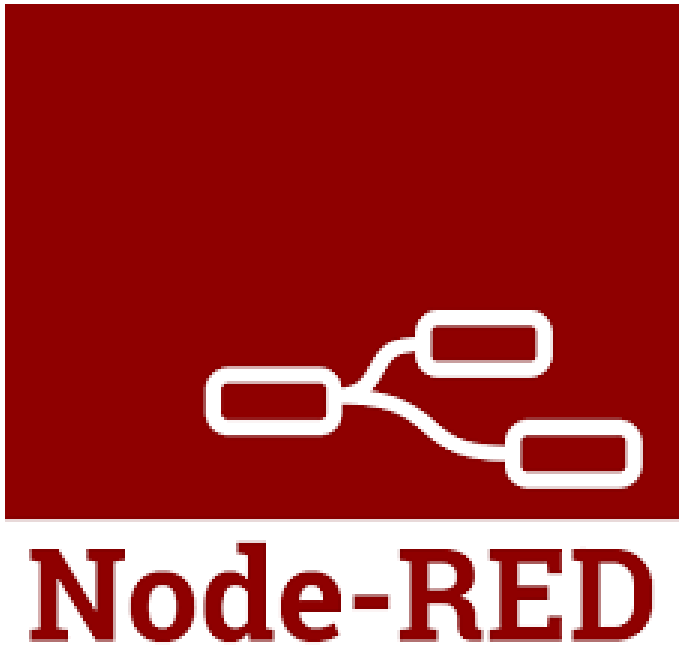
Back-end/Front-end

It’s a programming tool for wiring hardware devices, APIs and online services. It provides browser-based flow editing tool.

- Easy to use

Provides:

- Database connection
- API
- Dashboard User Interface



Arduino Code

```
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>
#include <math.h>


const int B = 4275;               // B value of the thermistor
const int R0 = 100000;            // R0 = 100k
const int pinTempSensor = A0;     // Grove - Temperature Sensor connected to A0
const int pinSoundSensor = A1;    // Grove - Sound Sensor connected to A1
const int pinLightSensor = A3;    // Grove - Light Sensor connected to A2



byte mac[]    = {  0x90, 0xA2, 0xBA, 0x10, 0xA8, 0x2D };
IPAddress ip(192, 168, 1, 30);
IPAddress server(192, 168, 1, 31);

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();
}

EthernetClient ethClient;
PubSubClient client(ethClient);
```

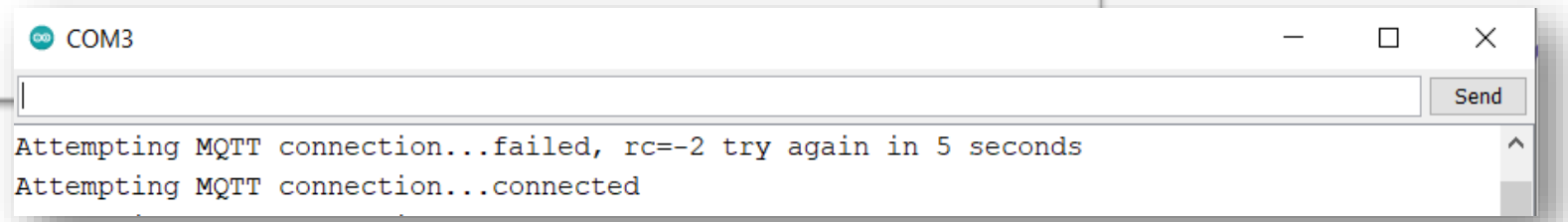
```
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("arduinoClient")) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("room/connection", "Connected");
      // ... and resubscribe

    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

```
void setup()
{
  Serial.begin(9600);

  client.setServer(server, 1883);
  client.setCallback(callback);

  Ethernet.begin(mac, ip);
  // Allow the hardware to sort itself out
  delay(1500);
}
```



```
void loop()
{
  if (!client.connected()) {
    reconnect();
  }

  //---------TheTempSensor-----------------------------
  int a = analogRead(pinTempSensor);
  float R = 1023.0 / a - 1.0;
  char temp[5];
  R = R0 * R;
  float temperature = 1.0 / (log(R / R0) / B + 1 / 298.15) - 273.15; // convert to temperature via datasheet
  String temp0 = String(temperature, 2);
  temp0.toCharArray(temp, 5);
  delay(500);
  //---------------------------------------------------

  //---------TheLightSensor----------------------------
  int light0 = analogRead(pinLightSensor);
  float resistance = (1023 - light0) * 10 / light0;
  char light[4];
  String light1 = String(resistance, 2);
  light1.toCharArray(light, 5);
  delay(500);
  //---------------------------------------------------

  client.publish("temp", temp);
  client.publish("light", light);
  delay(58000);

  delay(1000);

  client.loop();
}
```

Indexes

Air Quality

Catergory	Description	CO2 level ppm
IDA 1	High indoor air quality	< 400
IDA 2	Medium indoor air quality	400-600
IDA 3	Moderate indoor air quality	600-1000
IDA 4	Low indoor air quality	> 1000

EN13779 is a European Standard for **ventilation and air conditioning in buildings**. The standard aids ventilation designers to address factors inside and outside of a building that effect the environmental conditions for the occupants

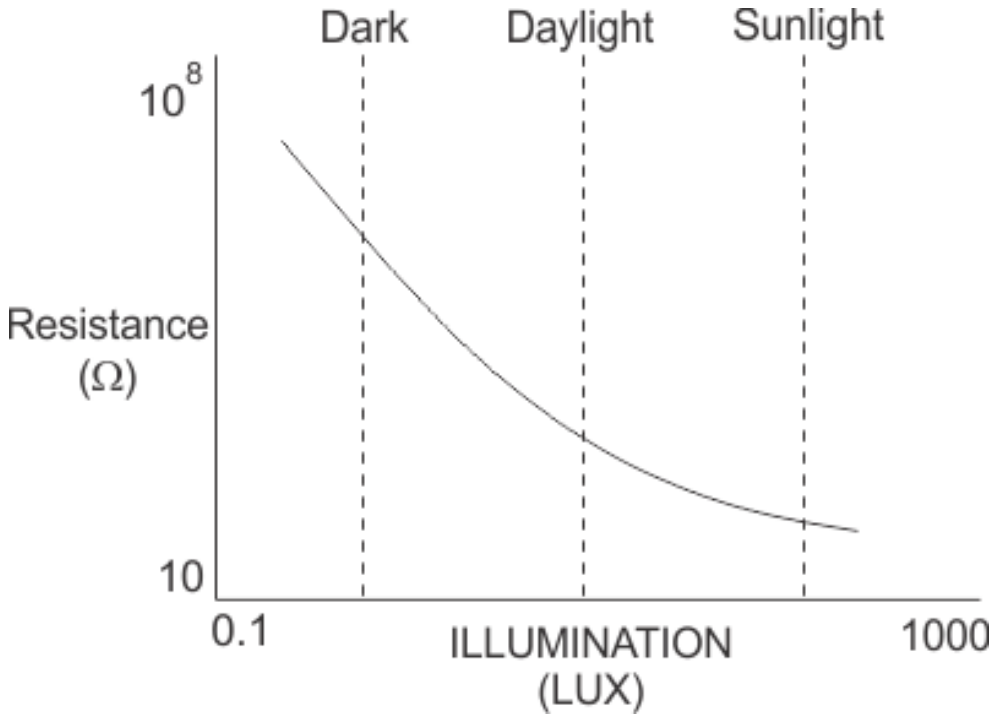
- Indoor air quality (IDA) is classified by levels of carbon dioxide present in the air and fresh air levels being introduced per person.

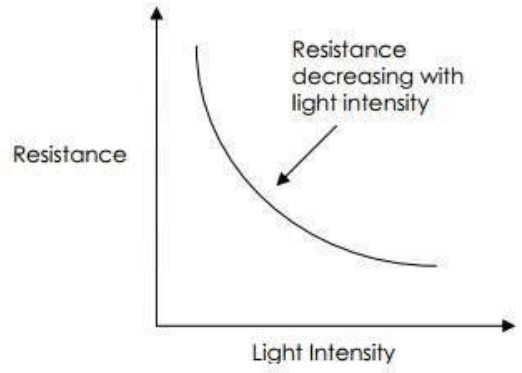
Indexes

Light Resistance

Energy Conservation in Smart Home Using Wireless Sensor Network.

Marypraveena, S & A K, Kavitha & Rajamanickam, Kanmani & Professor, Associate & Professor, Asssitant. (2016).

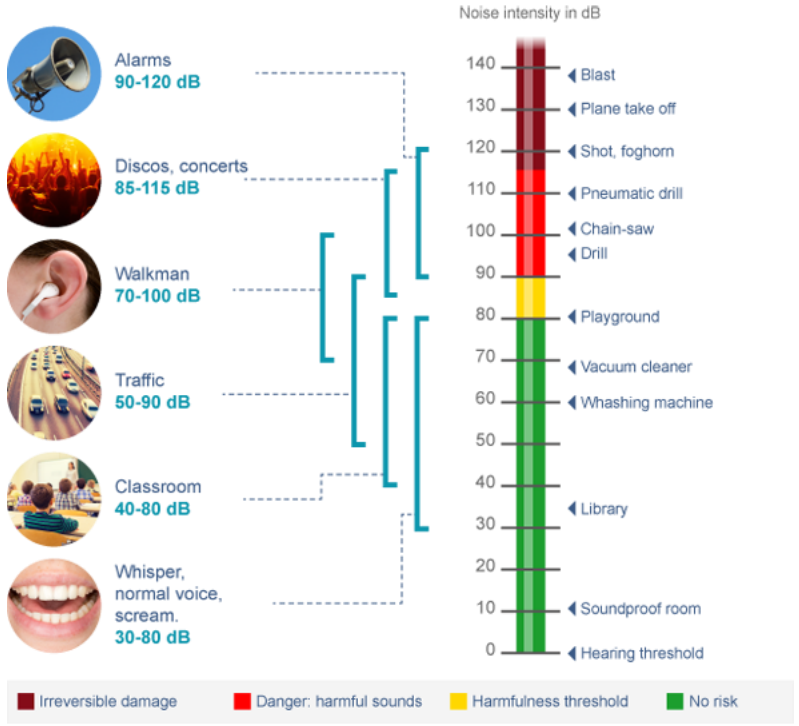


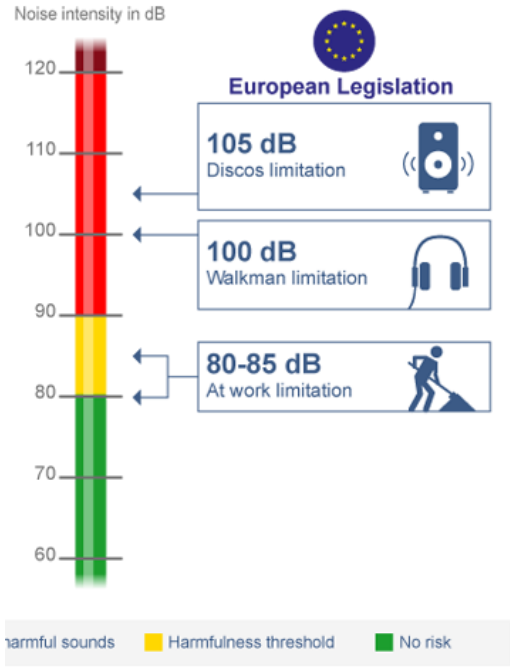


Indexes

Sound (Noise)

DIRECTIVE 2003/10/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 6 February 2003 on





the minimum health and safety requirements regarding the exposure of workers to the risks arising from physical agents (noise)

Indexes

Humidex

Humidex value	Degree of comfort
Under 15	Feeling cool or cold
From 15 to 19	No discomfort
From 20 to 29	Feeling of well-being
From 30 to 34	Feeling of greater or lesser discomfort
From 35 to 39	Rather great feeling of discomfort. Caution. Slow down certain outdoor activities.
From 40 to 45	Generalized feeling of discomfort. Danger. Avoid effort.
From 46 to 53	Extreme danger. Work stoppage in many areas.
Above 54	Imminent heat stroke (danger of death).

The humidex is an index number used by Canadian meteorologists to describe how hot the weather feels to the average person, by combining the effect of heat and humidity.

$$H = T_{\text{air}} + \frac{5}{9}\left[6.11 \times e^{5417.7530\left(\frac{1}{273.16} - \frac{1}{273.15+T_{\text{dew}}}\right)} - 10\right]$$

AHP

Analytic hierarchy process

We used AHP to generate wights for the four initial indexes that we had (Humidex, Light, Noise and Air Quality) to make a general Index called Room Coziness.

		Criteria		more important ?	Scale
i	j	A		B	A or B	(1-9)
1	2	Humidex		AirQuality	A	3
1	3			LightStatus	A	5
1	4			Noise	A	1
1	5					
1	6					
1	7					
1	8					
2	3	AirQuality		LightStatus	A	5
2	4			Noise	B	3
2	5					
2	6					
2	7					
2	8					
3	4	LightStatus		Noise	B	5
3	5					
3	6					
3	7					
3	8					

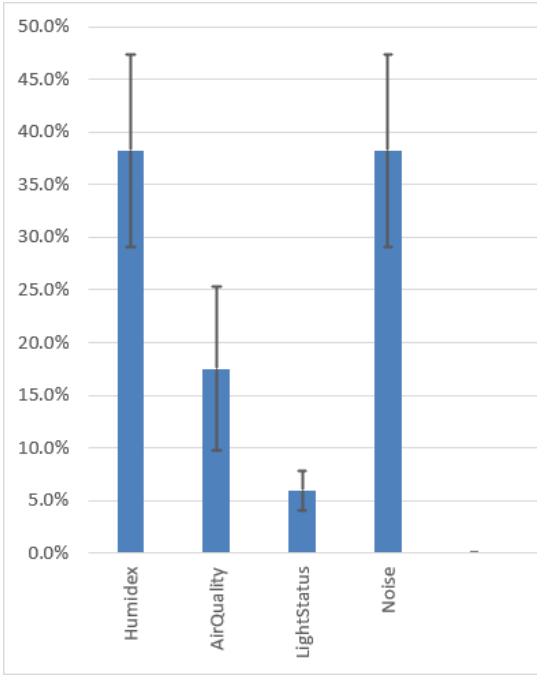
Intensity of	Definition	Explanation
1	Equal importance	Two elements contribute equally to the objective
3	Moderate importance	Experience and judgment slightly favor one element over another
5	Strong Importance	Experience and judgment strongly favor one element over another
7	Very strong importance	One element is favored very strongly over another, it dominance is demonstrated in practice
9	Extreme importance	The evidence favoring one element over another is of the highest possible order of affirmation
2,4,6,8 can be used to express intermediate values		

Matrix

		Humidex	AirQuality	LightStatus	Noise	0	0	0	0	0	0
		1	2	3	4	5	6	7	8	9	10
Humidex	1	1	3	5	1	-	-	-	-	-	-
AirQuality	2	1/3	1	5	1/3	-	-	-	-	-	-
LightStatus	3	1/5	1/5	1	1/5	-	-	-	-	-	-
Noise	4	1	3	5	1	-	-	-	-	-	-
0	5	-	-	-	-	1	-	-	-	-	-
0	6	-	-	-	-	-	1	-	-	-	-
0	7	-	-	-	-	-	-	1	-	-	-
0	8	-	-	-	-	-	-	-	1	-	-
0	9	-	-	-	-	-	-	-	-	1	-
0	10	-	-	-	-	-	-	-	-	-	1

normalized principal Eigenvector

38.25%
17.53%
5.96%
38.25%
0.00%
0.00%
0.00%
0.00%
0.00%
0.00%



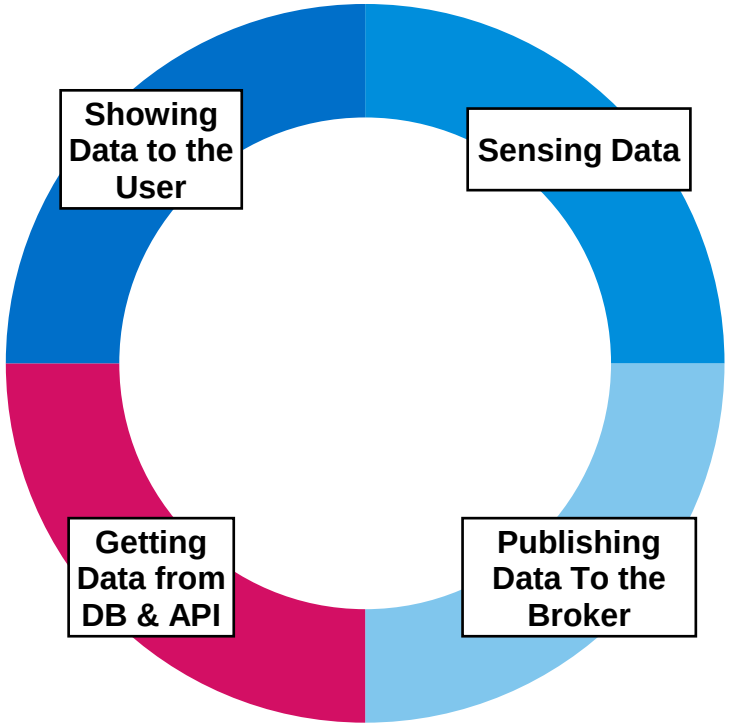
Coziness Formula

$$Coziness = (Humidex \times 0.382) + (\frac{0.176}{(AirQuality + 1)}) + (\frac{0.60}{light}) + (Noise \times 0.382)$$

Coziness

It determines the general status of a room in terms of being comfortable for a human being.

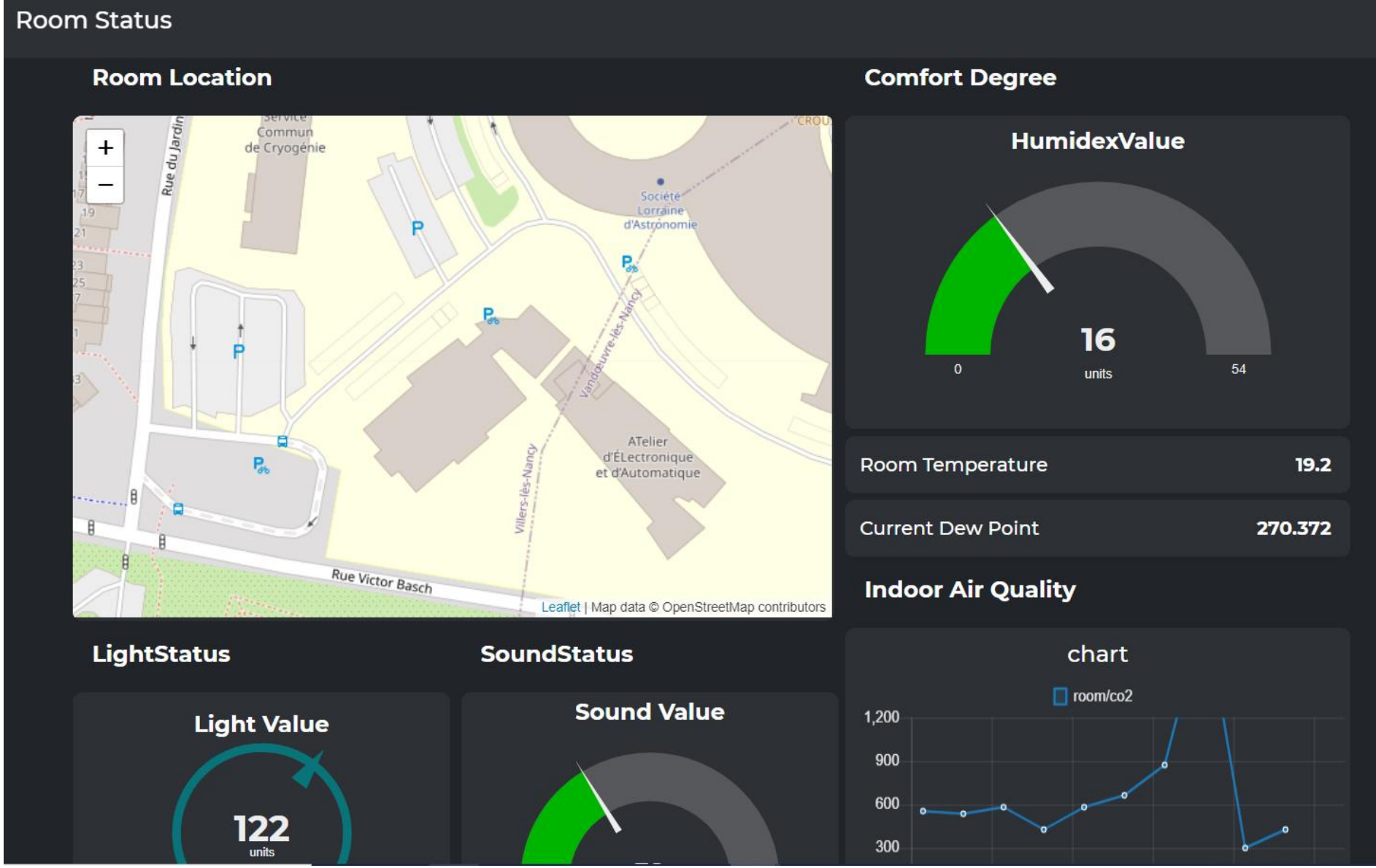
Coziness Value	Degree of Coziness
< 5.73	Low Coziness - Feeling cold, Bad Air Quality, Dark
5.73 – 41.65	High Coziness - Feeling of well-being, High Air Quality, Light, Tolerable Noise
41.65 – 48.15	Medium Coziness – Slight Feeling of Discomfort, Very Light, Noisy
> 48.15	Low Coziness - Great feeling of Discomfort, Annoying Light, Very Noisy

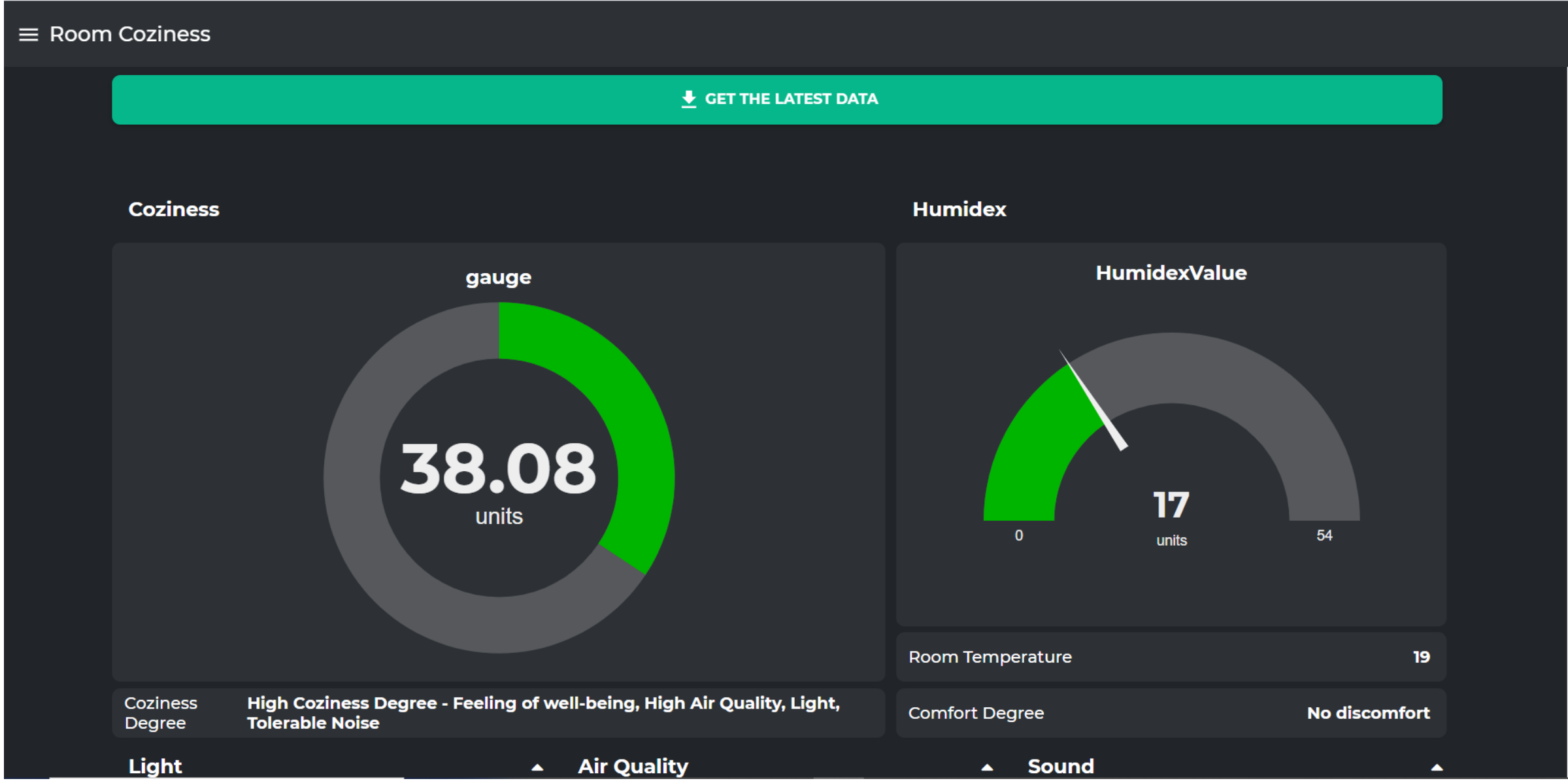


Life Cycle of the System

1. Sensing Data
2. Publishing Data to the Broker
3. Getting Data from Database and put it in an API
4. Showing Data to the User

Admin Dashboard





User Interface

Next Steps…

Securing the Node-Red

- Enabling HTTPS access
- Securing the editor and admin API
- Securing the HTTP Nodes and Node-RED Dashboard

THANK YOU!

Feter Sitepu

Elyas Khorasani