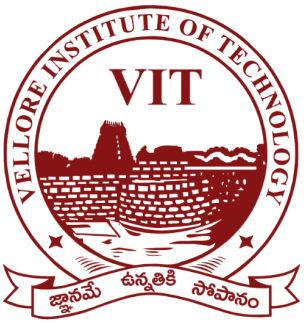


VIT-AP UNIVERSITY

VIT-AP University, G-30, Inavolu, Beside AP Secretariat Amaravati, Andhra Pradesh 522237

Tiltle

SilentZone : Sonic Sensor Library and Lab Management System

Submitted By :

1. Sahil Suman : 21BCE8897
2. R Hari Chandana : 21BCE8718
3. Shristi Mohanty : 21BCE8296
4. Batchu Ramya : 21BCE9106

Under the guidance of:

Dr. Ranjan Kumar,
Department of Mathematics,
School of Advanced Sciences (SAS),
VIT-AP University, Amaravati,
Andhra Pradesh

Abstract :

Sound detection is currently one of the most crucial subjects of study worldwide. Numerous activities take place when sound is present. One of the key areas of interest in the research has been the use of esp-8266, Ultrasonic sensors, passive infrared sensors, and many other devices to sense and quantify sound. The goal is to measure and observe human behavior remotely with the least amount of labor necessary. The objective of this research was to develop a sensor that can instantly determine changes in sound levels in a particular area and display the information on a liquid crystal display (LCD), offering both a light-coded signal and a sound alert. The esp-8266 was connected to (component names) on a breadboard, which was the hardware configuration.

The Arduino IDE program used to build and run the circuit is stored in the Node MCU's memory. The study demonstrated that the built-in sensor could be used in conjunction with the software developed to produce an automated library management system with regard to discipline to effectively handle the disciplinary issues in the library. The sensor concurrently emits audio signals in the form of a sound buzzer and displays visual LED signals that are configured and color-coded, for example, as sound levels greater than (value in db for each) corresponding to Green, Blue, and Red LED lights, respectively. Because of this, this approach to sound detection and measurement is efficient and guarantees precise readings of low sound levels.

When proximity detection is required, such as in hospitals, offices, and exam rooms, there are a number of applications for this sound detecting and measurement device.

Index

Introduction

The Sonic Sensor Library and Lab Management System is a project that was developed in response to the concerns about discipline that are taken into account when entering a library and lab. Our product is essentially a sound sensor that generates three different outputs depending on the decibel levels at each table. The use of IoT in this device's integration with software allows the library management to examine and administer the library with just their system. Each piece of data is transmitted via a network, fed into the backend, and dynamically displayed on the front end, resulting in an effective and real-time integration of hardware and software that yields reliable results.

Background

This project uses IoT technologies and is based on the ESP8266 platform. An efficient sound sensor system has been developed based on these two modules as well as additional modules like the sound sensor and Wi-Fi module, enabling the features required for the project's successful completion. The ESP8266 family of systems on a chip microcontrollers is low-cost and has integrated Wi-Fi and dual-mode Bluetooth. The ESP8266 series includes integrated antenna switches, RF baluns, power amplifiers, low-noise receive amplifiers, filters, and power-management modules in addition to the Tensilica Xtensa LX6 dual-core or single-core microprocessor, the Tensilica Xtensa LX7 dual-core, or a single-core RISC-V microprocessor.

The ESP8266 was created and built by the Shanghai-based Chinese company Espressif Systems, and it is manufactured by TSMC utilizing its 40 nm technology. The primary application CPU is not as taxed by communication stack overhead due to the ESP8266's ability to operate as a fully independent system or as a slave device to a host MCU. The ESP8266 may connect to other systems and provide Wi-Fi and Bluetooth functionality through its SPI/SDIO or I2C/UART interfaces.

The phrase "Internet of Things" (IoT) describes actual devices (or collections of such objects) that are outfitted with sensors, processing power, software, and other technologies and may exchange data with one another over the Internet or other communications networks. Devices just need to be individually addressable and connected to a network, not the entire internet, which has led to criticism of the name "internet of things."

The convergence of various technologies, including machine learning, ubiquitous computing, affordable sensors, and increasingly potent embedded systems, has altered the field. The traditional domains of embedded systems, wireless sensor networks, control systems, and automation (including home and building automation) enable the Internet of Things. In the

consumer market, IoT solutions are most frequently linked to the "smart home" or, in our case, the "smart library" notion.

These items consist of tools and appliances (such as lighting fixtures, thermostats, home security systems, cameras, and other home appliances) that work with one or more similar ecosystems and can be controlled by devices. These technologies serve as the foundation for our developed solution, which is further discussed in the study.

Problem Definition

Students typically head to the library to study or work quietly and effectively without distractions. It is ineffective, tiresome, and time-consuming to manage discipline in the library because there is usually still an issue with disturbance, and we observe the librarian moving from one table to another to keep order. We intend to create a method of operating the library using our gadget and its software in order to address the problems that the librarian and earnest students encounter.

Objectives

1. Noise free desk at library and lab.
2. Create a perfect study environment for students sitting in library and lab.
3. Creating a smart desk at library and in each lab
4. Designing a prototype for a lab and Library management system
5. Aim to find existing models in the current system.
6. Make simple yet efficient use of hardware with low cost requirements

Methodology/Procedure

Starting off with the Arduino IDE

Step 1:
To get started with Arduino IDE, go to https://www.arduino.cc/en/software and download the Arduino IDE.

Launch the Arduino IDE. Check that you are on version 1.8 or higher; if not, update your IDE to the most recent version.

1. Click on the File menu on the top menu bar.
2. Click on the Preferences menu item. This will open a Preferences dialog box.
3. You should be on the Settings tab in the Preferences dialog box by default.
4. Look for the textbox labeled “Additional Board Manager URLs”.
5. If there is already text in this box, add a comma at the end of it, then follow the next step.
6. Paste the following link into the text box – https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
7. Click the OK button to save the setting.

After we finish configuring the ESP8266 IDE, we will begin working on the project's logic.

Step 2:
The sensor ky-038 will now provide us with two types of inputs: digital and analog.

Step 3:
To change the sensitivity of the sensor, we need to rotate the pin on top of the potentiometer.

Step 4:
The sensor will respond to a specific range of sound, digitally with 0 and 1, and analogically with values ranging from 1600 to 2400.

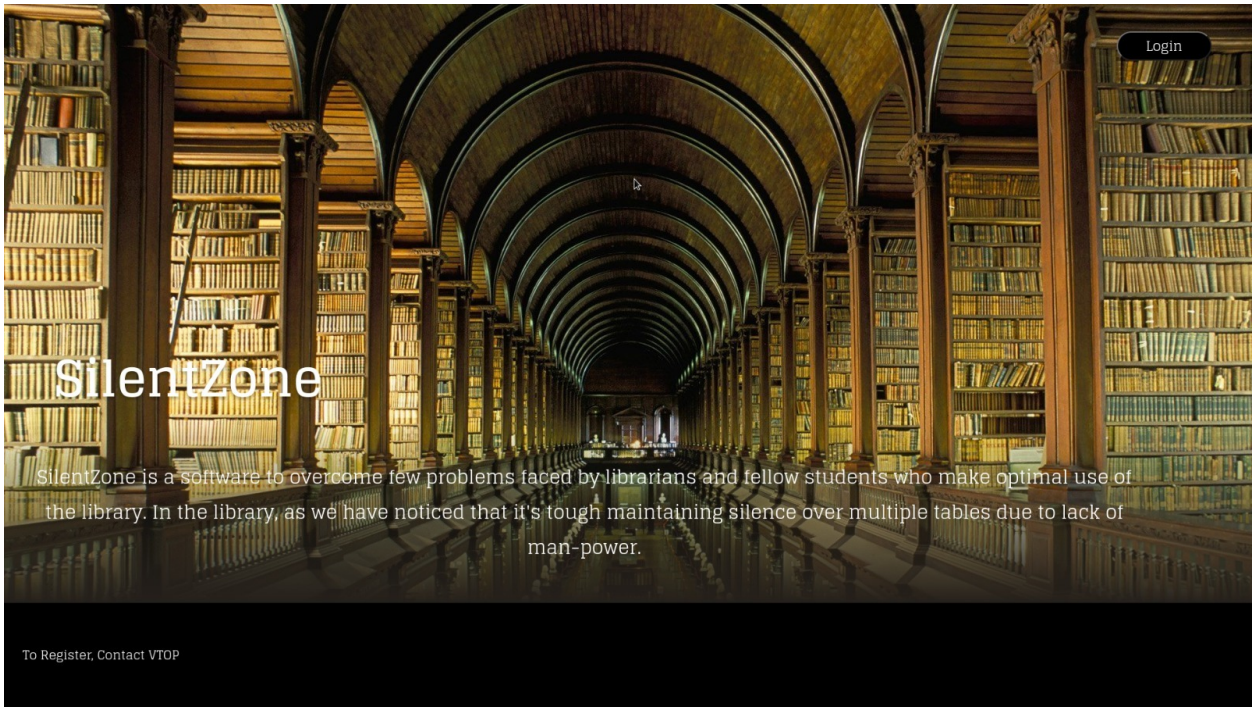
Step 5:
We will use a continuous sum array to process 200 inputs from the ky-038 sound sensor module's digital meter.

Step 6:
If sum >50, then noise level is very frequent and high
else if <50 && >10. then noise level is medium
else if <10. then noise level is low

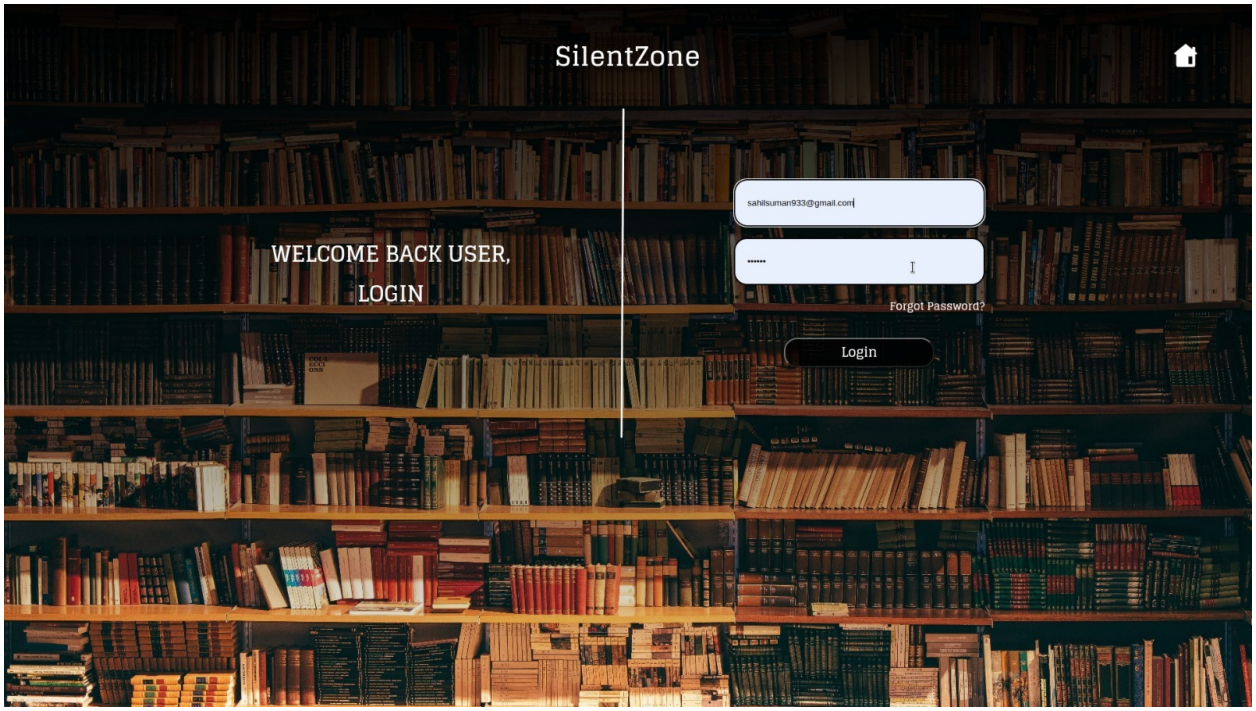
Step 7:
The values will now be sent to a server in the form of JSON data, and the server will send a request to the frontend to process those values and display them on the client side, informing the client which tables need to be supervised or monitored.

Results and Discussion:

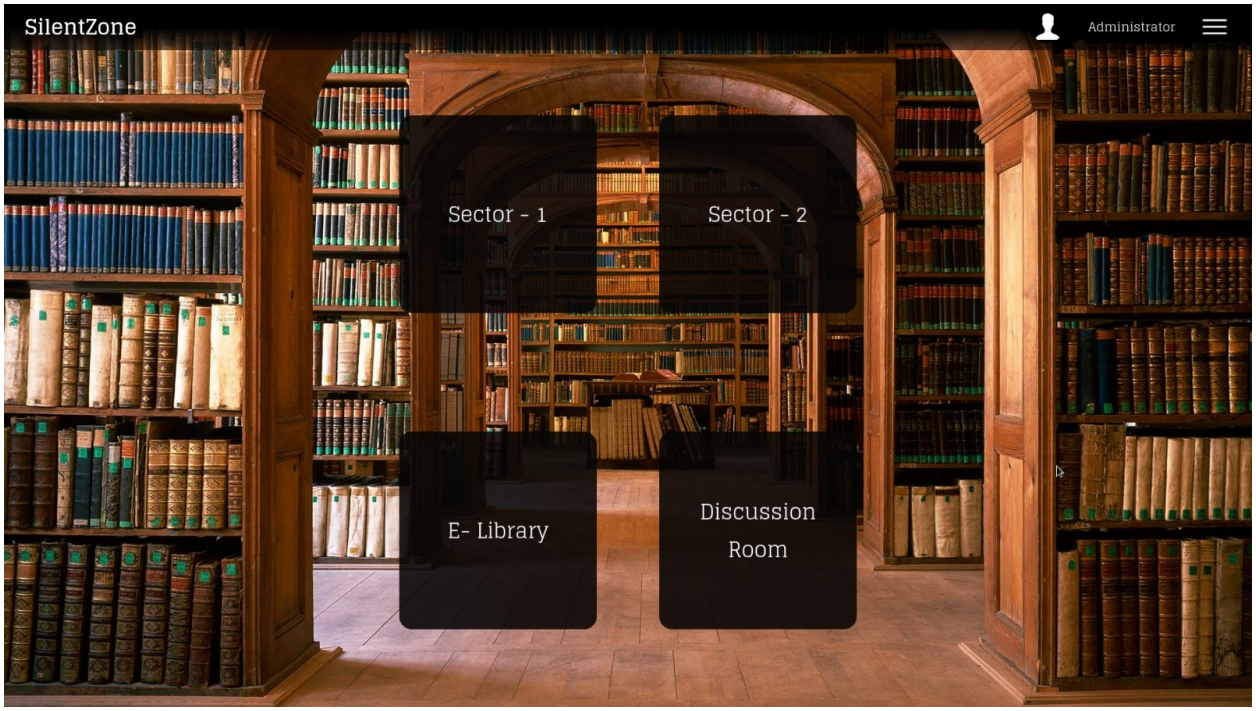
1) Homepage



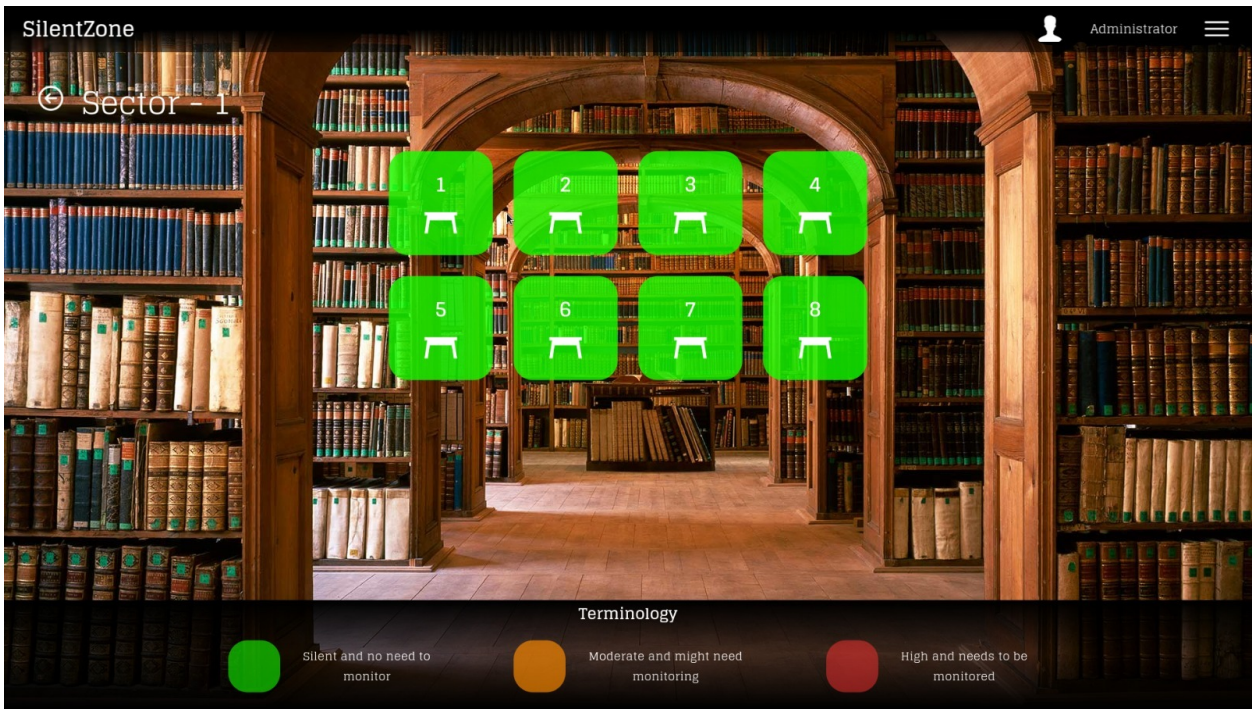
2) Login Page :



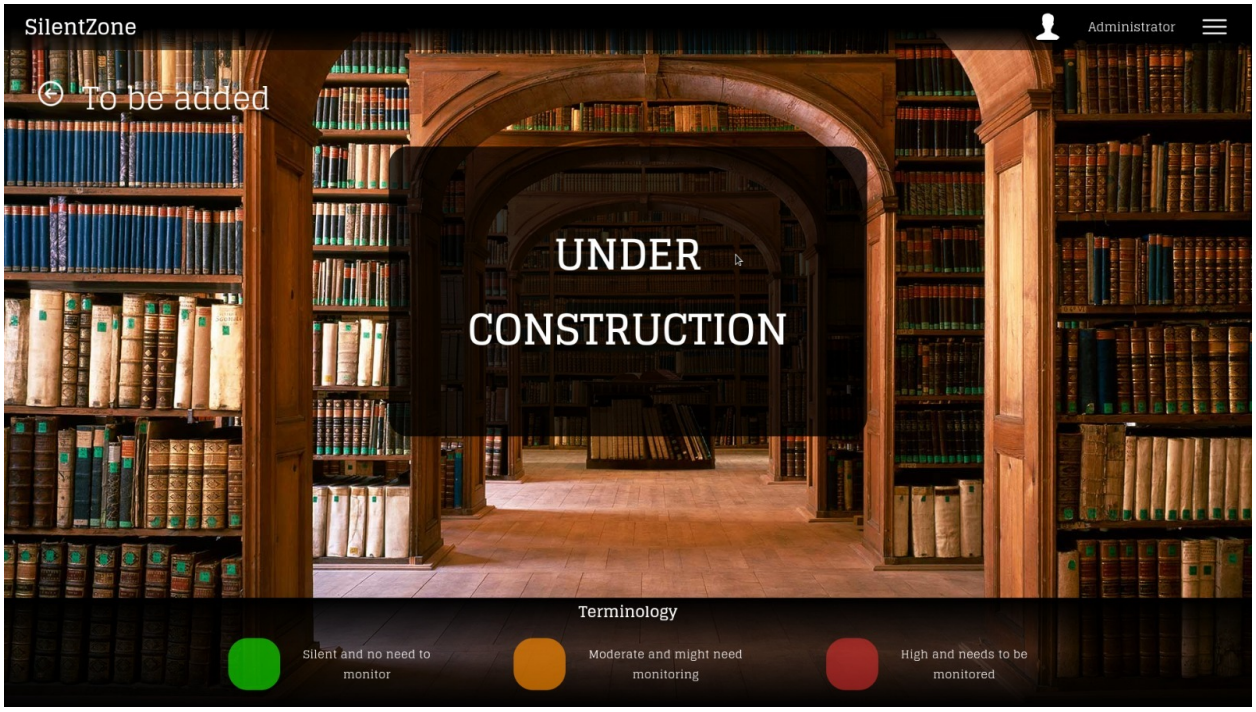
3) Dashboard :



4) Sector 1 :



5) Added Features Under construction:



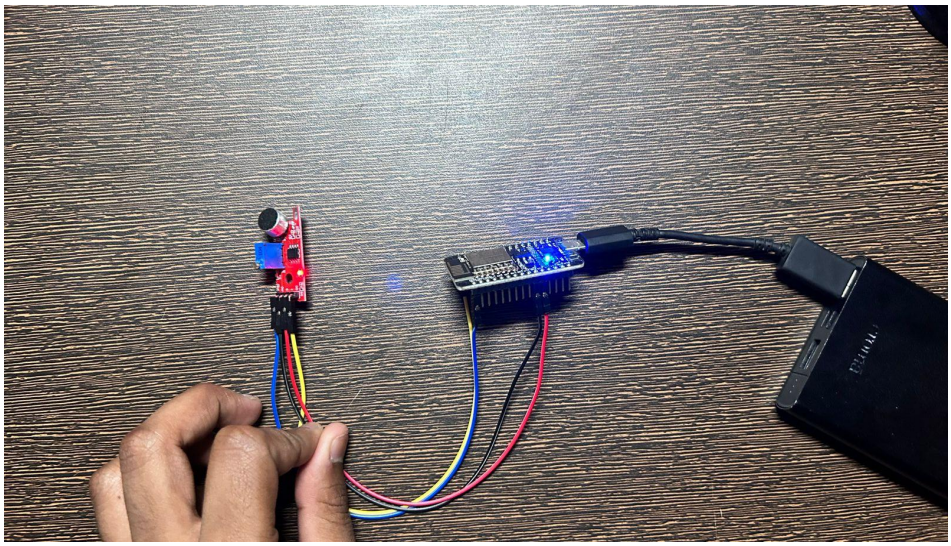
Video Demonstration

Video Link :
https://drive.google.com/file/d/1ZV2VuQDSdgYRi0aqrgiTgOFvsScc3FeK/view?usp=sharing

Initially, the device was not connected, so we were not receiving any data. However, after connecting it to the power supply, we started receiving data, which is displayed in the terminal. When some noise is made near the microphone sensor, it detects the noise and sends the data to the server. You can see in the terminal that the noise level changed from Low to High.

Moving on to the SilentZone dashboard, we added a new section called "All" and used the terminology "occupied," as per instructions. Additionally, the color of the table changes to red immediately after making noise, indicating that the noise level on that table is high.

Complete Model



Conclusion and Future Scope

In conclusion, this approach is an ideal and effective replacement for manual disciplinary administration in libraries. It is also an attractively affordable and flexible solution. Additionally, extra features like award points for the tables that maintain the highest levels of discipline over the course of the week can be added, and students who have utilized them can be monitored using their IDs. Another advantage is that if an e-library with on-table screens is enabled using the Wi-Fi module, pupils will be able to scan QR codes to download the selected book to their devices. With this technology as its base, we may practically see a totally automated library of the future.

References :

1. https://www.electronicshub.org/basics-of-embedded-c-program/
2. https://www.javatpoint.com/embedded-system-c-programming
3. https://www.allaboutcircuits.com/technical-articles/introduction-to-the-c-programming-
4. https://www.arduino.cc/en/Guide/Windows
5. https://github.com/abobija/esp-idf-vscode-boilerplate
6. https://docs.arduino.cc/software/ide-v2
7. https://www.upesy.com/blogs/tutorials/how-to-connect-wifi-acces-point-with-esp32
8. https://www.youtube.com/watch?v=Celxv69ycIA
9. https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/all
10. https://esp32io.com/tutorials/esp32-http-request
11. https://www.upesy.com/blogs/tutorials/how-to-connect-wifi-acces-point-with-esp32
12. https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/
13. https://esp32io.com/tutorials/esp32-http-request
14. https://randomnerdtutorials.com/esp32-http-get-post-arduino/
15. https://stackoverflow.com/questions/2194645/how-to-implement-websockets-on-an-
16. https://github.com/cjhdev/wic

Appendix

C++ code of Arduino

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <ArduinoJson.h>

WiFiClient wifiClient;

int blue = 16;
int green = 17;
int red = 18;
int iterator = 0;
int count = 0;
int digitalInput = 4; // Change this to the correct GPIO pin number for your
digital input
int digitalInputVal = 0;
int analogInput = A0; // Change this to the correct analog pin connected to the
microphone
int analogInputVal = 0;
int sum = 0;

const char* ssid = "WhoAmI_2.4GHz";
const char* password = "kingmonty@886";
const char* endPoint = "http://192.168.1.204:8000/";

void setup() {
Serial.begin(115200);
pinMode(green, OUTPUT);
pinMode(red, OUTPUT);
pinMode(blue, OUTPUT);
pinMode(digitalInput, INPUT);
pinMode(analogInput, INPUT);
}

void establishConnection() {
digitalWrite(green, LOW);
```

```
digitalWrite(blue, HIGH);
WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
Serial.println("\nConnecting");
while (WiFi.status() != WL_CONNECTED) {
if (count == 10) {
WiFi.begin(ssid, password);
count = 0;
}
digitalWrite(blue, HIGH);
Serial.print(".");
delay(300);
digitalWrite(blue, LOW);
delay(300);
Count++;
}

Serial.println("\nConnected to the WiFi network");
Serial.print("Local ESP32 IP: ");
Serial.println(WiFi.localIP());
delay(5000);
}

void apiCall() {
HTTPClient http;
http.begin(wifiClient, endPoint);
http.addHeader("Content-Type", "application/json");

// Serial.println("\nHTTP request Sent waiting for Response : ");
StaticJsonDocument<200> doc;
String noiseLevel = "LOW";
Serial.println("SUM VALUE:" + String(sum));

if (sum >= 25){
noiseLevel = "HIGH";
}else if (sum >= 20){
noiseLevel = "MID";
}
```

```
doc["tableNo."] = 2;
doc["noise"] = sum;
doc["noiseLevel"] = noiseLevel;
String requestBody;
serializeJson(doc, requestBody);
int httpResponseCode = http.POST(requestBody);
if (httpResponseCode == 200) {
String response = http.getString();
// Serial.println(httpResponseCode);
// Serial.println(response);
}
}

void loop() {
Iterator++;
if (WiFi.status() != WL_CONNECTED)
establishConnection();
delay(20);
digitalWrite(red, LOW);
digitalWrite(green, HIGH);
digitalInputVal = digitalRead(digitalInput);
analogInputVal = analogRead(analogInput);
// Serial.println("Microphone Value: " + String(analogInputVal));
//Serial.println(digitalInputVal);
sum = analogInputVal;
if (digitalInputVal == 1 || analogInputVal > 1850) {
digitalWrite(green, LOW);
digitalWrite(red, HIGH);
// Serial.println("RED Color");
}

if (iterator == 200) {
apiCall();
sum = 0;
iterator = 0;
}
}
```