Title: Noise pollution monitoring

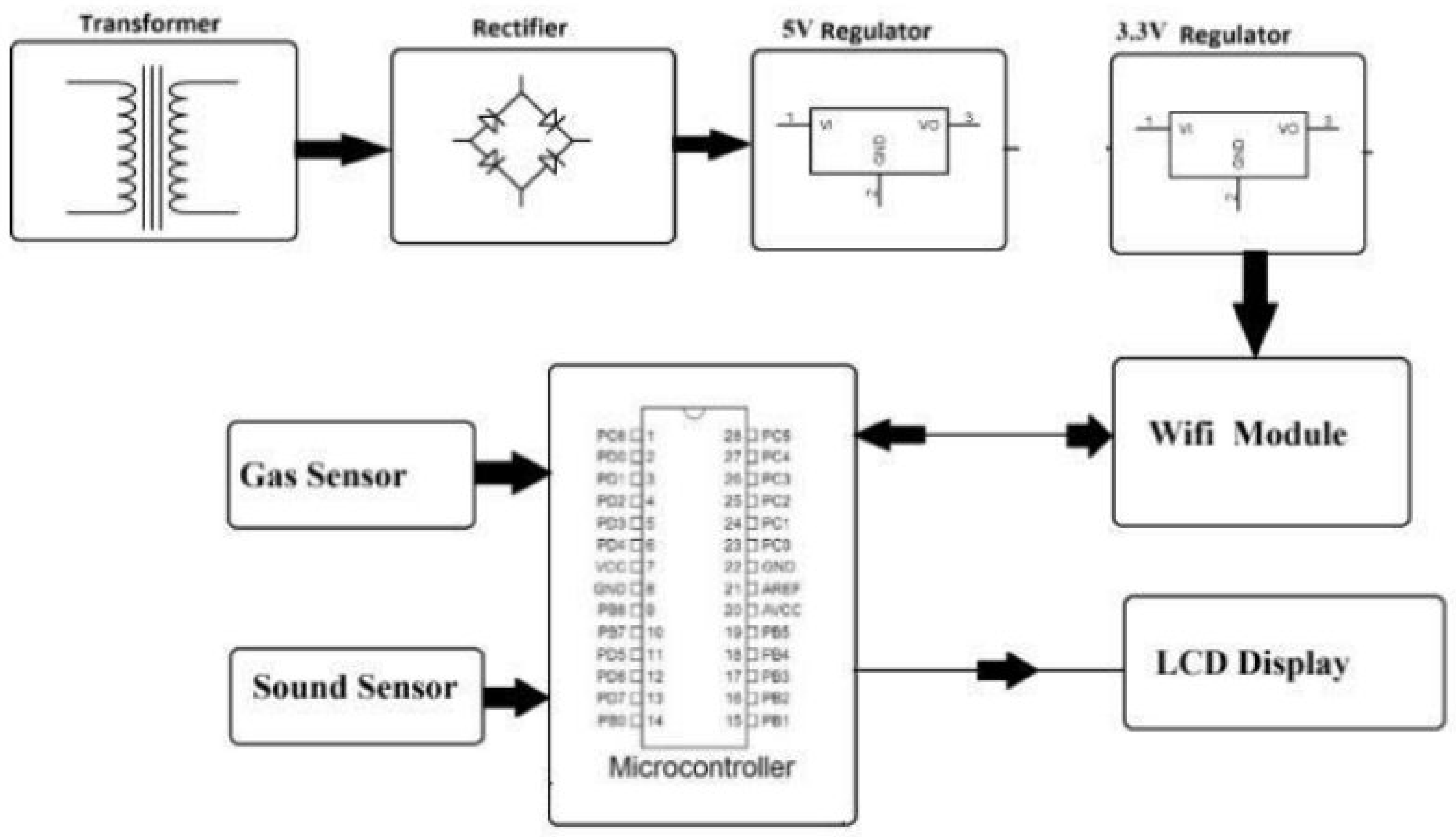
Problem statement:

Problems related to noise include stress related illnesses, high blood pressure, speech interference, hearing loss, sleep disruption, and lost productivity.

Problem Definition:

The solution focuses on traffic junctions throughout the city and plans to deploy a noise monitoring node at every junction. As a part of the solution to this project, we have implemented a "**Punishing Signal**" to create awareness about noise pollution among commuters. The implementation includes a noise pollution monitoring node that reads the noise levels at a given traffic junction and also the prototype of a traffic light controller which controls the flow of traffic at a junction. The idea is to discourage commuters from honking unnecessarily at a traffic light.

The connections are pretty simple, we just have to connect the sound sensor to one of the Analog pin and the LCD to the I2C pins



we propose an air quality as well as sound pollution monitoring system that allows us to monitor and check live air quality as well as sound pollution in a particular areas through IOT. System uses air sensors to sense presence of harmful gases/compounds in the air and constantly transmit this data to microcontroller. Also system keeps measuring sound level and reports it to the online server over IOT. The sensors interact with microcontroller which processes this data and transmits it over internet. This allows authorities to monitor air pollution in different areas and take action against it. Also authorities can keep a watch on the noise pollution near schools, hospitals and no honking areas, and if system detects air quality and noise issues it alerts authorities so they can take measures to control the issue.

An IoT-based air and sound pollution monitoring system is implemented using a network of sensors, connectivity technologies, and data analytics platforms. Air quality sensors are deployed in strategic locations to measure pollutant levels such as particulate matter, gases, and volatile organic compounds (VOCs).

Design thinking:

Empathize: Start by understanding the stakeholders involved, including residents, local authorities, and environmental agencies.

Conduct interviews, surveys, and observations to gather insights into their experiences with noise pollution and their needs.

Define: Clearly define the problem by creating a problem statement, such as "How might we effectively monitor and mitigate noise pollution in urban areas?"

Identify specific pain points and challenges faced by different stakeholders.

Ideate:

Brainstorm solutions collaboratively with a diverse team. Encourage creativity and think beyond traditional monitoring methods.

Use techniques like mind mapping, brainstorming sessions, or the "How Might We" method to generate a wide range of ideas.

Prototype:

Select a few promising ideas and create low-fidelity prototypes to visualize how they might work.

For noise pollution monitoring, this could involve designing physical prototypes, software interfaces, or conceptual models.

Test:

Collect feedback on your prototypes from stakeholders, including residents, experts, and policymakers.

Refine your prototypes based on the feedback and iterate on your designs.

Develop:

Once you have a well-refined prototype, start developing a functional version of your noise pollution monitoring solution.

This may involve designing hardware (e.g., noise sensors), software (e.g., data analysis and visualization tools), and user interfaces.

Implement:

Deploy your noise pollution monitoring system in a real-world setting, such as a pilot project in a specific urban area.

Collaborate with local authorities and communities to ensure a successful implementation.

Evaluate:

Continuously monitor and evaluate the effectiveness of your solution.

Gather data on noise levels, community satisfaction, and any improvements in noise pollution mitigation.

Iterate:

Based on ongoing evaluation and feedback, make necessary adjustments and improvements to your noise pollution monitoring system.

Continue to iterate and evolve the solution to better meet the needs of the community and stakeholders.

Scale:

Once you have a proven and effective solution, consider scaling it to other urban areas or regions facing similar noise pollution challenges.

Seek partnerships and funding opportunities to support expansion.

Prerequisites:

- Arduino IDE 1.8.5 or later (To compile and upload code on sensor nodes)
- JAVA 1.8 (To run the OneM2m platform)
- Eclipse OM2M v1.4.1 (implementation of the standard used in the project)
- NodeJS v14
- Postman
- Grafana v9.2 or later

Sensor design:

System uses air sensors to sense the presence of harmful gases/compounds in the air and constantly transmit this data to microcontroller. Also, the system keeps measuring sound level and reports it to the online server over IOT. The sensors interact with microcontrollers which process this data and transmit it over the internet.

sound pollution monitoring system is implemented using a network of sensors, connectivity technologies, and data analytics platforms

TEAM:

Project Manager: Gokul raja

Data Engineer: Seeni Bala Sivanesh

Data Scientist: Jayanthan

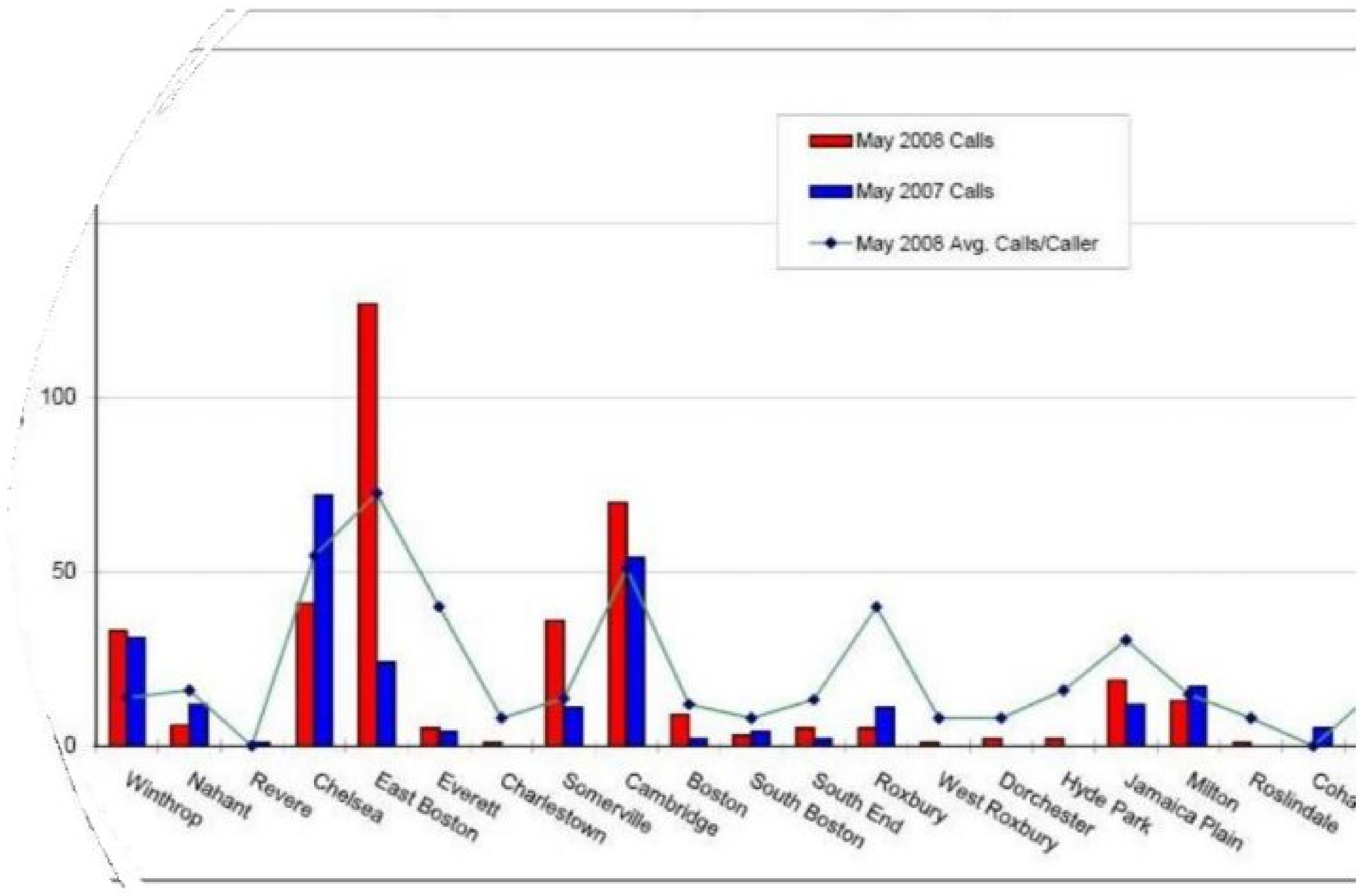
Quality inspector: yogamathavan

Conclusion:

conclusion, implementing noise pollution monitoring using the Internet of Things (IoT) offers a promising and effective solution to address the growing problem of noise pollution in urban environments. IoT-based noise monitoring systems leverage the power of connected sensors, data analytics, and real-time communication to provide numerous benefits and improve our understanding and management of noise pollution.

PROBLEM

"The problem of noise pollution involves the presence of disruptive and excessive sounds, often originating from human activities, which adversely affect human health, well- being, and the environment



INNOVATIVE SOLUTION :

Ai Power noise monitoring

How it works :

- **Noise Sensors:** Outfit the drones with high-quality noise sensors capable of capturing sound data with precision.

- **Real-time Analysis:** Use AI algorithms to process the noise data in real-time. These algorithms can detect patterns, identify sources of noise pollution, and differentiate between different types of noise (e.g., traffic, construction, industrialmachinery).
- **Mapping and Visualization:** The drones can create real-time noise pollution maps of a specific area. These maps can provide a visual representation of noise levels, allowing authorities and researchers to identify hotspots of noise pollution.

- Alerts and Reporting: Implement an alert system that can notify authorities and the public when noise levels exceed acceptable limits. Automated reports can be generated for regulatory agencies.
- Data Integration: Integrate the drone-collected noise data with other sources of information, such as weather conditions, traffic data, and land use patterns. This holistic approach can provide valuable insights into the causes and impacts of noise pollution.

Implementing AI-powered noise monitoring drones requires collaboration between technology companies, environmental agencies, and local governments.

Benefits

- ***Efficiency:* Drones can cover large areas quickly and easily, making it possible to monitor noise pollution in diverse urbanenvironments.**
- Real-time Monitoring: Provides real-time noise data, enabling rapid response to noise disturbances andmore effective enforcement of noiseregulations.
- **Cost-effectiveness:** Drones can be more cost-effective than stationary noise monitoring stations because they can be deployed as needed and do not require significant infrastructure.
- STEP1:INSTALL(pipinstallpaho-mqttsounddevice)
-
- STEP 2:reading noise level data from amicrophonesensorand publishitto anMQTT broker

importpaho.mqtt.clientasmqtt importsounddevice assd

- importnumpyasnp
-
- #MQTTsettings

BROKER_ADDRESS="YOUR_BR0KER_ADDRESS"USERNAME="Y0UR_USERNANE"

-

- `#Callback whenthecli ent connects totheMQTT broker defon_con nect(clie nt,userda ta,flags, rc):`
- `print("Connectedwithresultcode"+str(rc))`
-
-
- #CreateanMQTTclient

client=mqtt.Client() client.username_pw_set(USERNAME,PASSWORD) client.on_connect=on_connect

-
- # Connect to the broker client.connect(BROKER_ADDRESS,1883,60)
-
- `#Callbackfunctionforaudiostream`
- `def audio_callback(indata,frames, time, status): if status:`

print(f"Error:{status}") ifany(indata):
rms=np.sqrt(np.mean(indata**2)) pnint(f"SoundLevel:{nms:.2f}")

- `clieut.publish(TOPIC,f"SoundLevel:{rms:.2f}dB")`
-
- #Setupaudiostreamconfiguration
- sample_rate=44100

```
            #Adjustasperyourmi
            crophone'ssamplera
            te block_size = 1024
sd.default.samplenate=sample_rate sd.default.blocksize=block_size


withsd.InputStream(callback=audio_callback):pnint("Listeningfonnoise levels. PressCtnl+C toexit.")
sd.sleep(duration=3600)#Runfor1hour(adjustasneeded) exceptKeyboardInterrupt:
pass

#Disconnectfromthebroker client.disconnect()
```

IOTDEVICES1.Microphone Sensor 2.Microcontroller3.Power Supply4.ConnectivityModule 5.Data Storage andAnalysis Platform 6.Housing and Weatherproofing 7.Power Management Circuitry

DataCollection:Usethemicrophonesensontocapturesoundlevels.Convertanalogdatatodigitalusingt hemicnocontroller.

DataProcessing:Analyzethedata,applyfilters,orcalculatesoundlevelsindecibels(dB).Youcanalsorecor dtimestampsandGPScoordiua
DataTransmission:Sendtheprocesseddatatoacentralserverorcloudplatform. YoucanuseMQTT,HTTP,orothercommunication protocols.
DataStorage:Storethecollecteddatainadatabaseforfuturereferenceandanalysis.

DataVisualization:Cneatedashboandsorneportstovisualizethenoisepollutiondata.Youcanuselibraries likeMatplotliborweb-basedto
AlertsandNotifications:Implementalertingmechanismstouotifyrelevantpartieswhennoiselevelsexcee dpredefinedthresholds.

Integration:YoucanintegratethenoisepollutionmonitoringsystemwithotherIoTdevicesorsystemstoenhance itsfunctionality.

```
<!DOCTYPE html>

<html>

<head>

<title>Noise Pollution Monitoring</title>

<script>

    // JavaScript code for noise monitoring goes here

    // You'd need to use libraries like Chart.js or D3.js for data visualization.

    // Also, connect to sensors or data sources to get noise level data.

    // This is a placeholder for demonstration purposes.
```

```
functionstartMonitoring() {
        // Start monitoring noise levels
        // Update the chart or data in real-time
      }

functionstopMonitoring() {
        // Stop noise monitoring
      }
</script>
</head>
<body>
<h1>Noise Pollution Monitoring</h1>

<div>
<button onclick="startMonitoring()">Start Monitoring</button>
<button onclick="stopMonitoring()">Stop Monitoring</button>
</div>

<div>
<h2>Noise Level</h2>
<p id="noise-level">0 dB</p>
</div>

<div>
<h2>Noise History</h2>
<!-- You may need a chart or table to display historical noise data -->
</div>
```

```
<!-- Include JavaScript libraries for data visualization here -->

<!--<script src="chart.js"></script> -->


</body>

</html>
```

Output: