**PROJECT TITLE: NOISE POLLUTION MONITORING**

**Phase 5: Project Documentation & Submission**

**INTRODUCTION:**

Noise pollution, a pervasive and often overlooked environmental issue, can have detrimental effects on health, well-being, and the quality of life in urban and even rural areas. To address this concern and promote awareness, we introduce a Raspberry Pi-based noise pollution monitoring project. This project utilizes the capabilities of the Raspberry Pi, coupled with a USB microphone and Python programming, to continuously monitor and analyze ambient noise levels.The Raspberry Pi, a versatile single-board computer, is an ideal platform for this endeavor due to its affordability, low power consumption, and ease of integration with various sensors and peripherals. By deploying this solution, individuals, communities, or organizations can gain valuable insights into the noise pollution in their surroundings, allowing them to make informed decisions, take corrective actions, and advocate for a quieter environment.

**OBJECTIVES:**

**Real-time Data Collection:** Raspberry Pi can collect noise data in real-time from various sensors, allowing for continuous monitoring of noise levels in different locations.

**Environmental Impact Assessment:** It helps in assessing the impact of noise pollution on the environment and public health. This data can be used for research and policy-making.

**Noise Level Mapping:** Create noise maps of an area to identify noise hotspots and patterns, which can be valuable for urban planning and zoning regulations.

**Anomaly Detection:** Detect unusual noise events or sudden spikes in noise levels, which can be indicative of emergencies or disturbances.

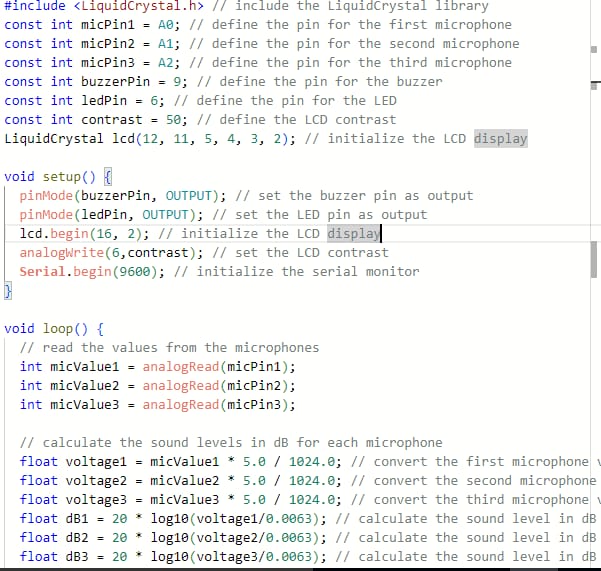
**Compliance Monitoring:** Ensure that noise levels in certain areas or industrial sites comply with legal limits and regulations.

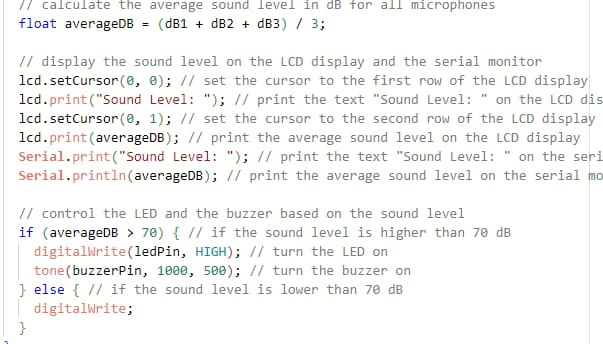
**Public Awareness:** Provide noise data to the public or local authorities to raise awareness about noise pollution issues.Historical Data Analysis: Accumulate historical data to analyze trends and patterns in noise pollution over time.

**Predictive Analysis:** Use collected data to predict potential noise pollution issues and take preventive measures.

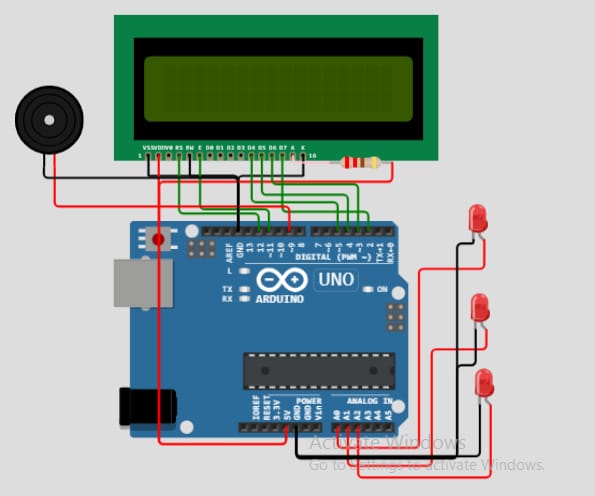
**Remote Access and Control:** Raspberry Pi and IoT technology allow remote access and control, enabling you to manage and monitor noise levels from anywhere.

**Integration with Other IoT Devices:** Integrate noise monitoring with other IoT devices for a more comprehensive smart city or environmental monitoring system.

**CODE:**



**SETUP:**



**PROJECT DESCRIPTION:**

Noise pollution is a growing concern in urban and industrial areas, affecting the quality of life and public health. To address this issue, this project aims to create a Noise Pollution Monitoring System using Raspberry Pi in the Internet of Things (IoT) framework. The system will continuously monitor noise levels in specific locations and provide real-time data for analysis and decision-making.Project Objectives:Design and assemble a Raspberry Pi-based sensor node capable of capturing ambient noise levels.Develop a user-friendly IoT interface to display real-time noise data.Implement data storage and analysis capabilities to identify noise trends and patterns.Incorporate alert mechanisms for exceeding predefined noise thresholds.Enable remote access for monitoring and control of the system.

**PYTHON CODE:**

import RPi.GPIO as GPIO

import time

import requests

# Replace with your IoT platform's endpoint for data transmission

IOT\_ENDPOINT = "https://your-iot-endpoint.com/data"

# Replace with your noise sensor pin configuration

NOISE\_SENSOR\_PIN = 18

GPIO.setmode(GPIO.BCM)

GPIO.setup(NOISE\_SENSOR\_PIN, GPIO.IN)

def capture\_noise\_data():

try:

while True:

if GPIO.input(NOISE\_SENSOR\_PIN) == GPIO.HIGH:

# Noise level detected

send\_data\_to\_iot\_platform(1) # You can customize this data format

else:

# No noise detected

send\_data\_to\_iot\_platform(0) # You can customize this data format

time.sleep(1) # Adjust the time interval as needed

except KeyboardInterrupt:

GPIO.cleanup()

def send\_data\_to\_iot\_platform(noise\_level):

data = {

"sensor\_id": "noise\_sensor\_1", # Replace with your sensor ID

"noise\_level": noise\_level,

"timestamp": int(time.time())

}

# Send data to the IoT platform

response = requests.post(IOT\_ENDPOINT, json=data)

if response.status\_code == 200:

print("Data sent successfully")

else:

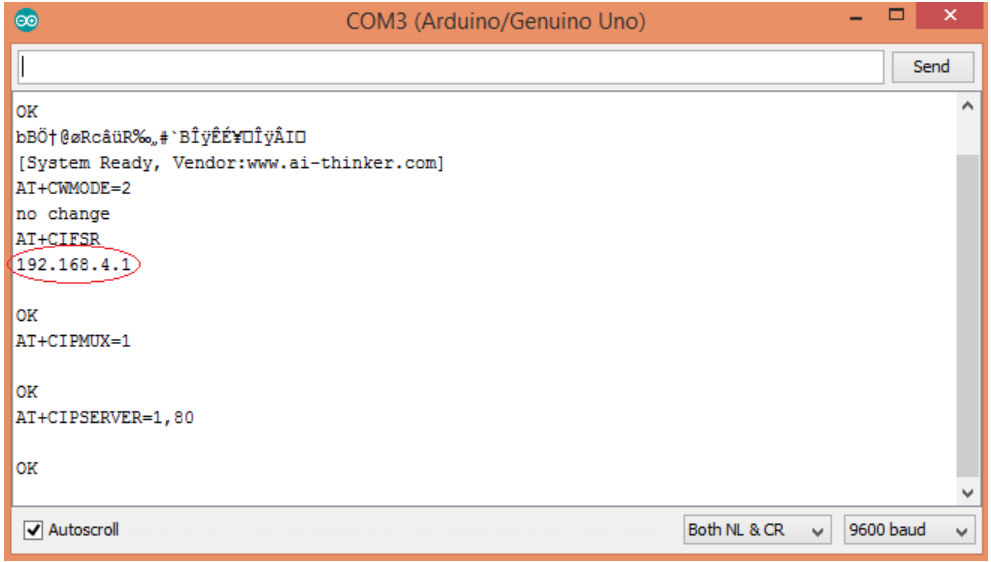
print("Failed to send data")

if \_\_name\_\_ == "\_\_main\_\_":

capture\_noise\_data()

**Testing and Output of the Project:**

Monitoring noise pollution using a Raspberry Pi in an IoT (Internet of Things) project involves several steps. Here's a high-level overview of the process and potential outputs:Hardware Setup:Raspberry Pi: You'll need a Raspberry Pi board (e.g., Raspberry Pi 4), power supply, and an SD card with the operating system.Microphone: Connect a USB microphone to the Raspberry Pi to capture ambient sound.Software Setup: 3. Operating System: Install a suitable operating system (e.g., Raspberry Pi OS) on the SD card and configure your Raspberry Pi.Python Programming: Write Python scripts to record audio using the microphone.Noise Analysis: Use libraries like NumPy and SciPy to analyze the recorded audio data for noise levels, frequency analysis, or specific sound patterns.Data Storage and Transmission: 6. Data Storage: Store noise data in a database or local storage on the Raspberry Pi for later analysis.IoT Communication: Implement IoT communication protocols (e.g., MQTT) to transmit data to a remote server or cloud platform for real-time monitoring.



Type this IP address in your browser, it will show you the output as shown below. You will have to refresh the page again if you want to see the current Air Quality Value in PPM.

**Output:** Data Visualization: Display noise pollution data in real-time or historical charts and graphs using tools like Matplotlib or web-based dashboards.

Alerts: Set up alerts based on predefined noise thresholds to notify relevant parties if noise pollution exceeds acceptable levels.

Historical Analysis: Analyze historical data to identify trends, peak noise times, or problematic areas.

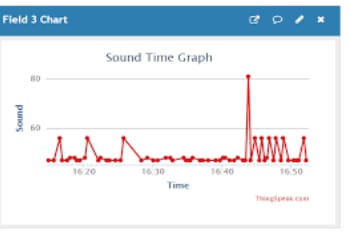
Reports: Generate reports summarizing noise pollution statistics over time.

User Interface: Create a user-friendly interface to access noise data and control the system remotely.

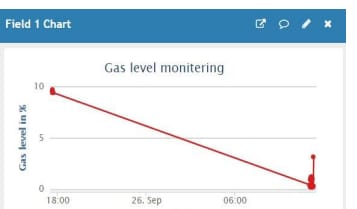
**OVERVIEW:**

This Shows the complete setup of the Noise Pollution Monitoring System Based on the Iot Using Raspberry pi that contains Mq-2 Gas sensor , Mq-7 Gas Sensor and Mq-135 Gas Sensor and finally placed on the board for easy to use and Convince

⁹







**APPLICATIONS:**

Monitoring noise pollution using a Raspberry Pi in an IoT (Internet of Things) setup is a valuable application. Here are some key applications and ideas:

**Urban Noise Monitoring:** Install Raspberry Pi devices in various urban locations to continuously monitor noise levels. This data can be used for city planning, understanding noise patterns, and identifying areas with high noise pollution.

**Traffic Noise Analysis:** Deploy Raspberry Pi sensors near highways or busy intersections to track traffic noise. This information can help with traffic management and assessing the impact of noise on nearby residents.

**Industrial Zones:** Monitor noise levels around industrial areas to ensure compliance with noise regulations and to protect the health of workers and residents in the vicinity.

**Community Noise Complaints:** Create a system where residents can report noise complaints via a web interface or mobile app, and Raspberry Pi devices can corroborate these complaints with real-time data.

**Soundscape Mapping:** Use multiple Raspberry Pi devices in a network to create a soundscape map of an area, identifying sources of noise and their intensity. This can be useful for urban planners and architects.

**Wildlife Conservation:** Install Raspberry Pi devices in wildlife conservation areas to monitor noise levels and their impact on wildlife. This can help in taking conservation measures to reduce disturbances.

**Smart Home Integration:** Integrate Raspberry Pi-based noise monitoring with smart home systems to trigger actions like closing windows, turning on white noise machines, or sending notifications when noise levels exceed a certain threshold.

**Noise in Public Transport:** Install Raspberry Pi devices in buses, trams, or trains to monitor noise levels and ensure that they meet safety and comfort standards for passengers.

**Educational Projects:** Raspberry Pi noise monitoring projects can be used in educational settings to teach students about environmental monitoring, data analysis, and the impact of noise pollution.

**CONCLUSION:**

In conclusion, noise pollution monitoring using a Raspberry Pi is a feasible and cost-effective solution for assessing and managing noise levels in various environments. This approach offers several advantages, including flexibility, scalability, and the ability to integrate.

README LINK:

https://github.com/Yamuna-60/yamu/blob/main/README.md