NOISE POLLUTION MONITORING

Submitted by:

Name: Balamurugan A

Nm id:au812921106005

Email id:a.bala26032003@gmail.com

NOISE POLLUTION MONITORING

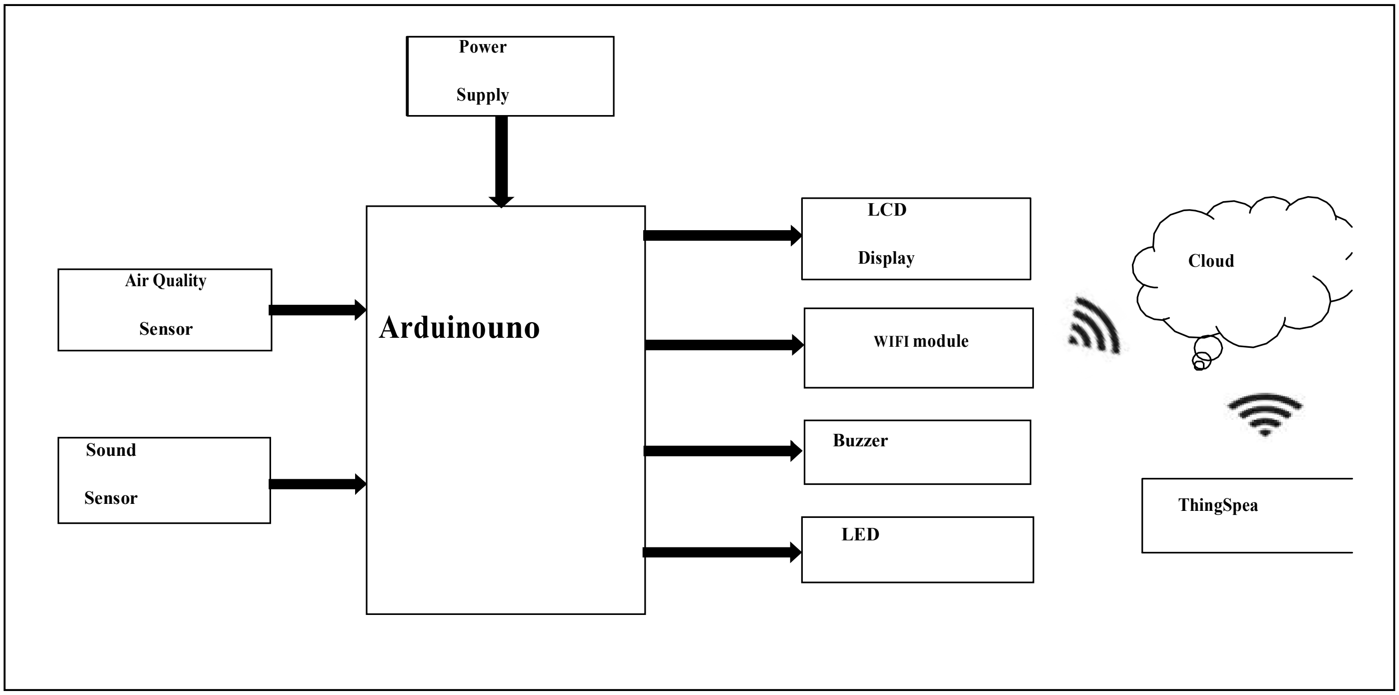
DEVELOPMENT PART 1:

Introduction:

This project presents an innovative approach to address noise pollution through an IoT-based monitoring system. By leveraging cutting-edge sensors and communication technologies, the system aims to provide real-time data on noise levels in urban environments. This data, processed through cloud-based analytics, will offer valuable insights for policymakers and urban planners to implement targeted interventions. The project's goal is to enhance public health and well-being by fostering a quieter and more sustainable living environment. Through the integration of IoT, this initiative underscores the potential of technology to mitigate environmental challenges and improve the quality of life in densely populated areas.

Block diagram:

COMPONENTS



Components:

1. ArduinoUNO
2. MQ135 (Gassensor)
3. LM393 (Noisesensor)
4. ESP8266 WIFIModule
5. 16*2 LCDDisplay
6. LED
7. Buzzer

- **Arduino UNO:**
  Arduino is 8 bit microcontroller board based on the ATmega328P. The operating voltage is 5V. It has 14 pins digital input output pins (Of which can be used 6 as PWM output
- **MQ135 Gas Sensor:**
  The MQ135 is a gas sensor it used for detecting or sensing harmful gases in the atmosphere. It has wide detecting scope. It gives fast response and also it it high sensitivity sensor. It is simple and long life device. They are used in air quality control equipment for building offices are suitable for detecting of NH3, alcohol, benzene, smoke CO2 etc.
- **LM393 Sound Sensor:**
  The sound sensor module provide an easy way to detect sound and it generally used for detecting sound intensity. Module detect the sound has exceeded a threshold value. Sound is detected via microphone and fed into an LM393 opamp.
- **ESP8266 WIFI Module:**
  The esp8266 WIFI module is a self containedsoc with integrated TCP/IP protocol stack that can give any microcontroller access to your WIFI network. The esp8266 is capable of either hosting an application or offloading all WIFI networking functions from another application processor
- **16*2 LCD Display:**
  LCD is used for to display the condition there are three conditions in air

pollution and three conditions in noise pollution means air and sound is clear, moderately polluted or highly polluted that is displayed on LED.

Program:

Coding:

To make and send alerting message facility via Telegram, we need to understand the skeleton of the coding. The whole program has two parts namely:

1. **Configuration code:** It consist all the backend details of Bolt IoT Wi-Fi Module and the Telegram.
2. **Main code:** It consists of the core coding of the facility.

To create above two mention files, executive the following steps:

Open the Digital Ocean Ubuntu server (For Windows /Mac Operating System) or open the terminal in the Ubuntu Operating System.

Step 1] We need to create a directory (folder) to store the two coding files mentioned above. To create a directory named alert, type out the following command

```
sudo mkdir alert
```

Step 2] Next we have to enter the directory that we just created. To enter the directory named alert type out the following command:

```
cd alert
```

Step 3] After entering the folder first we need to create the configuration python file which will hold all the backend details of the Bolt IoT Wi-Fi Module and Telegram.

To create the configuration python file in the folder which was created first type out the following command for creating the file with extension .py;

```
sudo nano configuration.py
```

Step 4] After the above mentioned file enter the following data into the file. Make sure that you add the updated Bolt API key, device id and Telegram details:

```
"""Configurations for Telegram alert message"""

BOLT_API_KEY = "XXXXXXXXX" #This is your Bolt cloud API
Key.
DEVICE_ID = "XXXXXXXXX" #This is the ID number of your
Bolt device.
TELEGRAM_CHAT_ID = "@XXXXXXXXX" #This is the channel ID of the
channel created in the
Telegram. Paste after @.
TELEGRAM_BOT_ID = "botXXXXXXXXX" #This is the bot ID of the bot
created in the Telegram. Paste
after bot.
THRESHOLD = 80 #Threshold beyond which the
alert should be sent.
```

Step 5] Save the file by clicking "ctrl+x" and press enter. Next create another file which will include the main coding of the facility.

```
sudo nano alert.py
```

Step 6] Enter the following code into the newly created file:

```
import requests                          #for making HTTP requests
import json                           #library for handling JSON data
import time                           #module for sleep operation
from boltiot import Bolt                      #importing Bolt from boltiot
                                 module
import configuration                    #configuration file
mybolt = Bolt(configuration.BOLT_API_KEY,configuration.DEVICE_ID)
def get_sound_sensor_value_from_pin(pin):
try:
response = mybolt.analogRead(pin)
data = json.loads(response)
      if data["success"] != 1:
            print("Request not successful")
            print("This is the response->", data)
          return -999
             sound_sensor_value = int(data["value"])
             return sound_sensor_value
except Exception as e:
print("Something went wrong when returning the sensor value")
             print(e)
     return -999
def send_telegram_message(message):
```

```
url = "https://api.telegram.org/" + configuration.TELEGRAM_BOT_ID +
"/sendMessage"
data = {
        "chat_id": configuration.TELEGRAM_CHAT_ID,
        "text": message
}
try:
        response = requests.request(
            "GET",
            url,
            params = data
        )
        print("This is the Telegram response")
        print(response.text)
        telegram_data = json.loads(response.text)
return telegram_data["OK"]
    except Exception as e:
print("An error occurred in sending the alert message via Telegram")
        print(e)
        return False
while True:
    #Step 1
    sound_sensor_value = get_sound_sensor_value_from_pin("A0")
    print("The current sensor reading is:", sound_sensor_value)
    #Step 2
    if sound_sensor_value == -999:
             print("Request was unsuccessful. Skipping.")
     time.sleep(10)
             continue
     #Step 3
    if sound_sensor_value >= configuration.THRESHOLD:
print("Sensor value has exceeded threshold")
message = "Alert! Noise disturbance around the XYZ Hospital. Random and
unidentified sound intensity has crossed " + str(configuration.THRESHOLD) +
str(“dB”) + \
“The current sound sensor reading is ” + str(sound_sensor_value) + str(“dB”) +
str(“To, The Police Incharge, Immediate action required. Thankyou.”)
telegram_status = send_telegram_message(message)
print("This is the Telegram status:", telegram_status)
```

```
# Step 4
time.sleep(50)   #Time interval to get the status update.
```

Step 7] Exit the code editor using "ctrl+x" and then run the code that you have written using the following command.

```
sudo python3 alert.py
```