# IOT_Phase 5

# Noise Pollution Monitoring

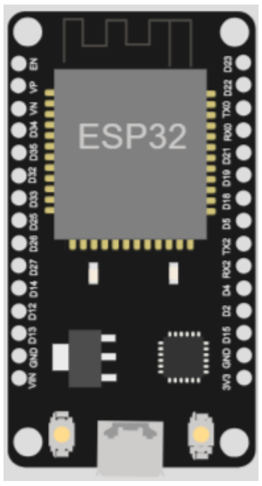
## Objectives:

The main objective of this project is to assess, measure, and analyze the levels of unwanted or harmful sound in a particular environment.

## Noise Sensor deployment:

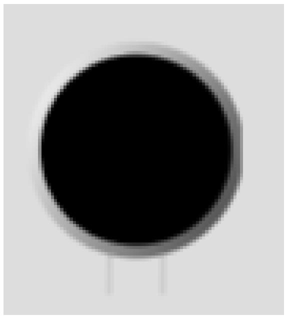
In wokwi simulator ESP32, Microphone and HC-SR04 Ultrasonic Distance Sensors are used

1. **ESP32:**
   The ESP32 is a dual-core microcontroller that integrates WiFi and Bluetooth connectivity. It's an improvement over the ESP8266 and offers a more powerful CPU, more GPIOs, various interfaces, and enhanced capabilities.



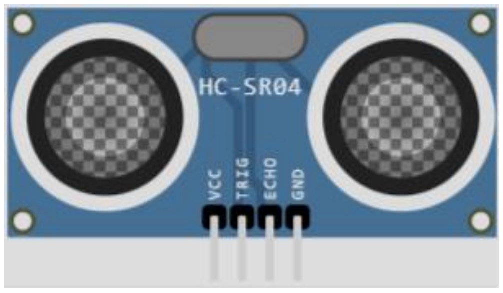
**ESP32**

2. **Microphone:**
   The "Microphone" in Wokwi likely refers to a simulated component or module that emulates the functionality of a microphone in a circuit. It's often used in conjunction with other electronic components to create and test projects that involve sound input or voice recognition systems.



**Microphone**

3. **HC-SR04 Ultrasonic Distance Sensor:**
   The HC-SR04 is a popular ultrasonic distance sensor module used for measuring distances and obstacle detection. It's widely utilized in various projects due to its simplicity and reliability.



**HC-SR04 Ultrasonic Distance Sensor**

## Steps Used to Deploy Noise Monitoring Sensor with Code Implementation in Wokwi

### 1. Circuit Setup:

Connect the Microphone and ultrasonic sensor to the ESP32.

**Microphone:** Microphone have two pins—Mic1 and Mic 2. Connect Mic1 to GPIO 2 pin on the ESP32 and Mic 2 to GND pin on the ESP32.

**HC-SR04 Ultrasonic Distance Sensor:** Ultrasonic sensors have trig (trigger) and echo pins. Connect trig to a GPIO 15, echo to another GPIO 4 pin, VCC to 3V3 and GND to GND.

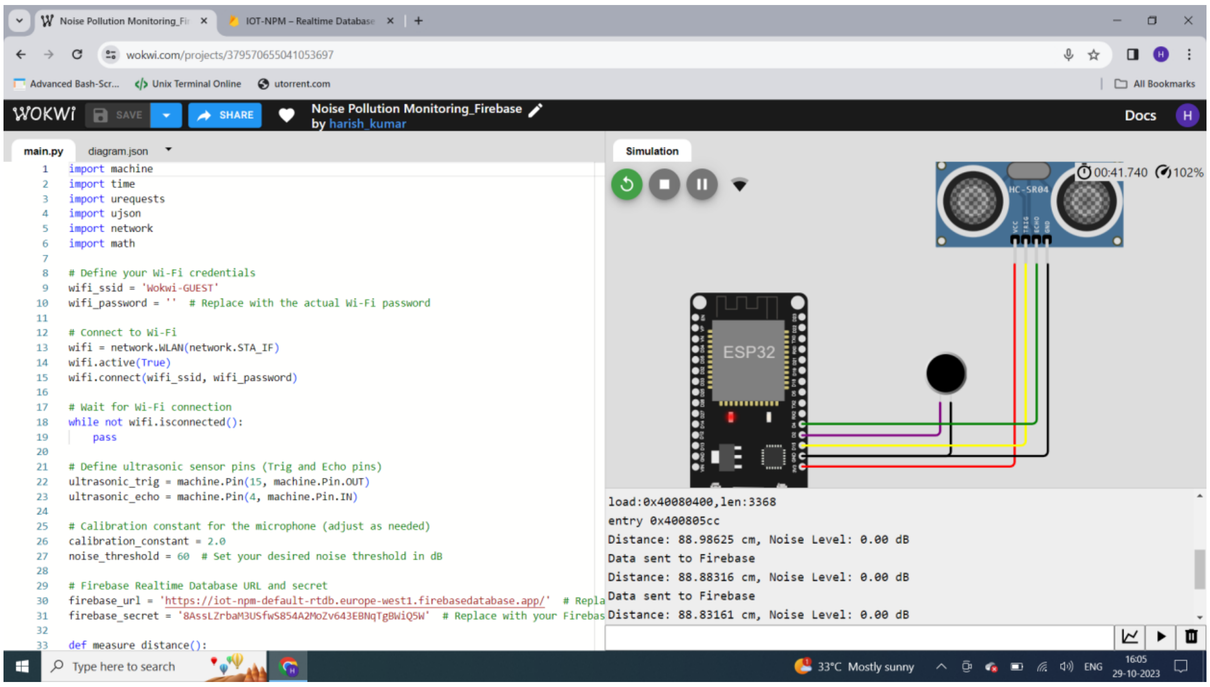
### 2. Software Setup:

#### main.py

```
import machine
import time
import urequests
import ujson
import network
import math

# Define the Wi-Fi credentials
wifi_ssid = 'Wokwi-GUEST'
wifi_password = ''  # Replace with the actual Wi-Fi password

# Connect to Wi-Fi
wifi = network.WLAN(network.STA_IF)
wifi.active(True)
wifi.connect(wifi_ssid, wifi_password)

# Wait for Wi-Fi connection
while not wifi.isconnected():
    pass

# Define ultrasonic sensor pins (Trig and Echo pins)
ultrasonic_trig = machine.Pin(15, machine.Pin.OUT)
ultrasonic_echo = machine.Pin(4, machine.Pin.IN)

# Calibration constant for the microphone (adjust as needed)
calibration_constant = 2.0
noise_threshold = 60  # Set the desired noise threshold in dB

# Firebase Realtime Database URL and secret
firebase_url = 'https://iot-npm-default-rtdb.europe-west1.firebasedatabase.app/'  #
Replace with the Firebase URL
firebase_secret = '8AssLZrbaM3USfwS854A2MoZv643EBNqTgBWiQ5W'  #
Replace with the Firebase secret

def measure_distance():
    # Trigger the ultrasonic sensor
    ultrasonic_trig.value(1)
    time.sleep_us(10)
    ultrasonic_trig.value(0)

    # Measure the pulse width of the echo signal
    pulse_time = machine.time_pulse_us(ultrasonic_echo, 1, 30000)

    # Calculate distance in centimeters
    distance_cm = (pulse_time / 2) / 29.1
    return distance_cm

# Function to send data to Firebase
def send_data_to_firebase(distance, noise_level_db):
    data = {
        "Distance": distance,
        "NoiseLevelDB": noise_level_db
    }
    url = f'{firebase_url}/sensor_data.json?auth={firebase_secret}'

    try:
        response = urequests.patch(url, json=data)  # Use 'patch' instead of 'put'
        if response.status_code == 200:
            print("Data sent to Firebase")
        else:
            print(f"Failed to send data to Firebase. Status code: {response.status_code}")
    except Exception as e:
        print(f"Error sending data to Firebase: {    str(e)}")

try:
    while True:
        distance = measure_distance()
        noise_level_db = 0  # Replace with the actual noise level measurement
        print("Distance: {} cm, Noise Level: {:.2f} dB".format(distance,
noise_level_db))

        if noise_level_db > noise_threshold:
            print("Warning: Noise pollution exceeds threshold!")

        # Send data to Firebase
        send_data_to_firebase(distance, noise_level_db)

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

except KeyboardInterrupt:
    print("Monitoring stopped")
```

## Analysis:

The readings from the microphone and the ultrasonic sensor, allowing to capture sound levels and distance measurements.



```
import machine
import time
import urequests
import ujson
import network
import math

# Define your Wi-Fi credentials
wifi_ssid = 'Wokwi-GUEST'
wifi_password = ''  # Replace with the actual Wi-Fi password

# Connect to Wi-Fi
wifi = network.WLAN(network.STA_IF)
wifi.active(True)
wifi.connect(wifi_ssid, wifi_password)

# Wait for Wi-Fi connection
while not wifi.isconnected():
    pass

# Define ultrasonic sensor pins (Trig and Echo pins)
ultrasonic_trig = machine.Pin(15, machine.Pin.OUT)
ultrasonic_echo = machine.Pin(4, machine.Pin.IN)

# Calibration constant for the microphone (adjust as needed)
calibration_constant = 2.0
noise_threshold = 60  # Set your desired noise threshold in dB

# Firebase Realtime Database URL and secret
firebase_url = 'https://iot-npm-default-rtdb.europe-west1.firebasedatabase.app/'  # Repla
firebase_secret = '8AssLZrbaM3USfwS854A2MoZv643EBNqTgBWiQ5W'  # Replace with your Firebas

def measure_distance():
```

```
load:0x40080400,len:3368
entry 0x400805cc
Distance: 88.98625 cm, Noise Level: 0.00 dB
Data sent to Firebase
Distance: 88.88316 cm, Noise Level: 0.00 dB
Data sent to Firebase
Distance: 88.83161 cm, Noise Level: 0.00 dB
```

wokwi.com/projects/379570655041053697
Noise Pollution Monitoring_Firebase
by harish_kumar
main.py
diagram.json
Simulation
00:41.740
102%
HC-SR04
ESP32

**Wowki simulator Output**

## Platform and Mobile app development:

In Noise pollution monitoring project **Firebase platform** used for real-time database and **Visual studio** with voltbuilder for Mobile app development

## Firebase Platform:

**Firebase platform** for Real-time Database which allows to store and synchronize data between users in real-time. The tools used for developing a database are Authentication, App Check, Real-Time Database, Storage and Hosting.

- **Firebase Authentication:**

  Firebase Authentication is a service that provides easy-to-use authentication methods, allowing users to sign in to the app using different authentication providers like email/password, phone number, Google, Facebook, Twitter, and more. It handles the entire authentication flow, including user management, identity verification, and security controls.

- **App check:**

  Firebase App Check is a security service that helps protect the backend resources from abuse, such as by verifying the authenticity of the incoming traffic from the app. It's designed to work with Firebase services like Real-time Database, Fire store, and Cloud Functions to help prevent unauthorized usage.

- **Real-Time Database:**

  Firebase Real-time Database is a NoSQL cloud database that allow to store and sync data between users in real-time. It's a JSON-based, cloud-hosted database where data is synchronized across all connected clients in milliseconds. Real-time Database provides real-time data updates and offline support, enabling smooth and responsive app experiences.

- **Storage:**

  Firebase Storage is a secure cloud storage service for storing user-generated content such as images, videos, audio, or other files. It offers powerful features, including secure file uploads and downloads, scalability, and integration with Firebase Authentication for access control.

- **Hosting:**

  Firebase Hosting provides fast and secure way to host web apps and static content. It enables the hosting of the entire web application, including HTML, CSS, JavaScript, and other assets. Firebase Hosting also includes features like SSL (Secure Socket Layer) and content delivery networks for fast global delivery of web content.

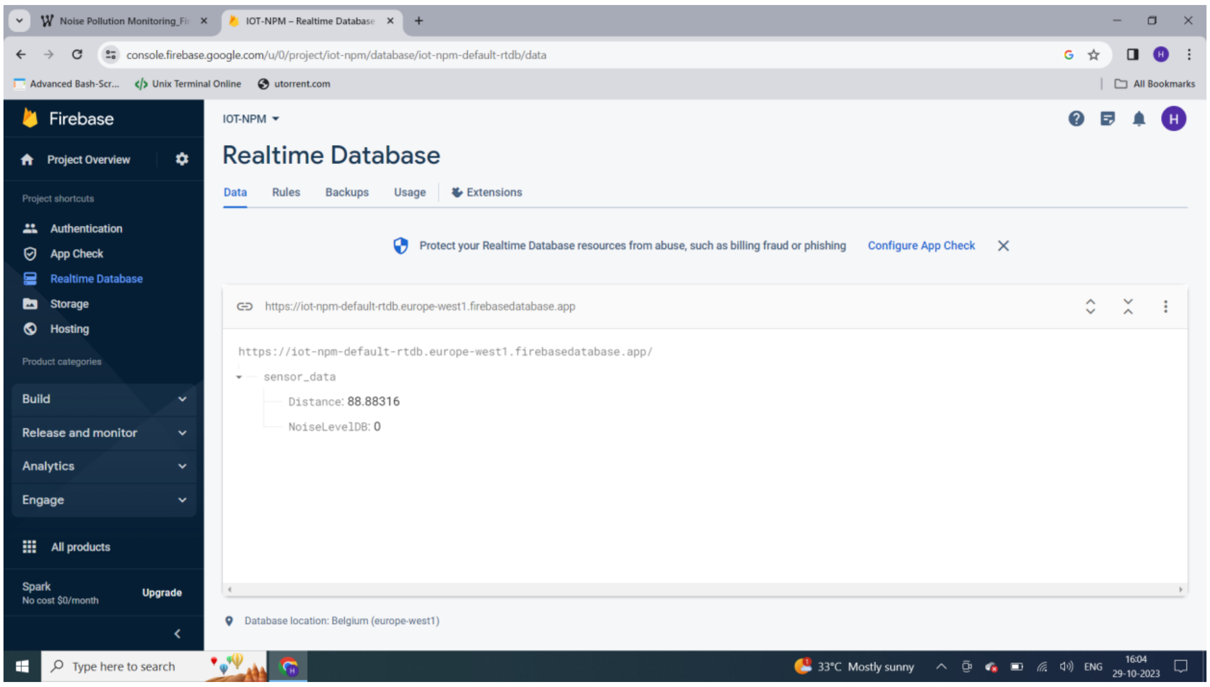
## Steps used to create a real time database:

### 1. Setting up firebase project:

- Go to the Firebase Console and Create a New Project (IOT-NPM)
- Click on "Add Project" button and enter the project name and select the country/region.
- After creating new project, Authentication, App Check, Real-Time Database, Storage and Hosting services chosen for used in the project.

### 2. Setting up Firebase in Web app:

- In Firebase project, click on the web icon (</>) to add a web app and register the app by giving it a nickname and click "Register app."
- After creating the web app, copy the real-time database URL and API key from IOT-NPM project for wokwi simulator.



```
https://iot-npm-default-rtdb.europe-west1.firebasedatabase.app/
▾ sensor_data
    Distance: 88.88316
    NoiseLevelDB: 0
```

**Firebase Real-time database**

## Visual Studio for Mobile App Development with Code Implementation:

Visual Studio is a powerful integrated development environment (IDE) created by Microsoft, primarily used for software development. With the help of various extensions and plugins, Visual Studio used for a wide range of programming languages and frameworks, including C#, C++, JavaScript, Python, and more. For mobile app development, Visual Studio offers tools and integrations that allow to create applications for various platforms such as Android and iOS.

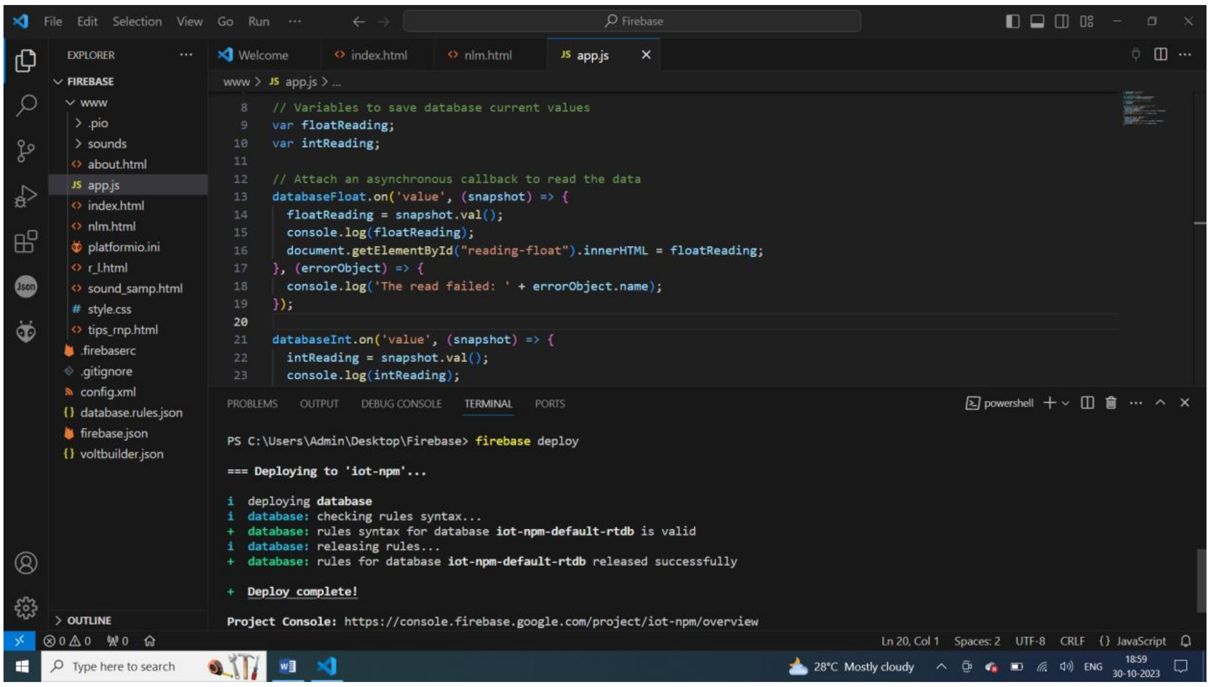
## Steps used to create a Mobile app:

### 1. Setting up the development environment:

- ❖ Setting up Visual Studio and configuring it for mobile app development.
- ❖ This involves installing HTML, CSS, JavaScript, JSON and PlatformIO Extensions and Firebase Arduino Client Library for ESP8266 and ESP32**.**

### 2. Creating the Web page:

- ❖ Create a folder and deploy the firebase, inside the firebase files create another folder for the web page.
- ❖ Create the web page using required HTML and CSS coding
- ❖ Combine both the firebase and web page using required HTML and JavaScript
- ❖ Check whether the firebase and visual studio are connected using "firebase deploy"



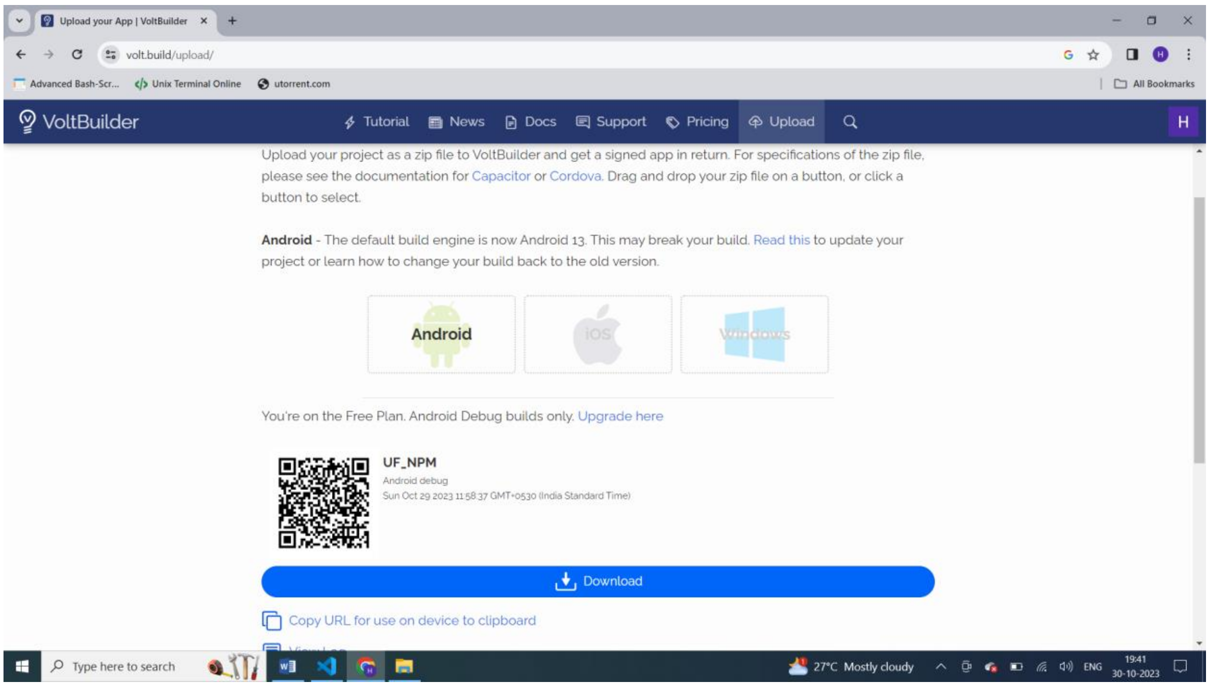
```javascript
// Variables to save database current values
var floatReading;
var intReading;

// Attach an asynchronous callback to read the data
databaseFloat.on('value', (snapshot) => {
  floatReading = snapshot.val();
  console.log(floatReading);
  document.getElementById("reading-float").innerHTML = floatReading;
}, (errorObject) => {
  console.log('The read failed: ' + errorObject.name);
});

databaseInt.on('value', (snapshot) => {
  intReading = snapshot.val();
  console.log(intReading);
```

```
PS C:\Users\Admin\Desktop\Firebase> firebase deploy

=== Deploying to 'iot-npm'...

i  deploying database
i  database: checking rules syntax...
+  database: rules syntax for database iot-npm-default-rtdb is valid
i  database: releasing rules...
+  database: rules for database iot-npm-default-rtdb released successfully

+  Deploy complete!

Project Console: https://console.firebase.google.com/project/iot-npm/overview
```

**Deployment of firebase and voltbuilder in visual studio**

## 3. Integration with VoltBuilder:

- Once the app development is complete use VoltBuilder to compile the app.
- Convert the project folder into zip file and upload it into the voltbuilder
- VoltBuilder integrates with Visual Studio, enabling to send the app project to the VoltBuilder cloud service directly from the IDE.
- The Mobile app can be installed using QR code or Copy the URL.



Upload your project as a zip file to VoltBuilder and get a signed app in return. For specifications of the zip file, please see the documentation for Capacitor or Cordova. Drag and drop your zip file on a button, or click a button to select.

**Android** - The default build engine is now Android 13. This may break your build. Read this to update your project or learn how to change your build back to the old version.

You're on the Free Plan. Android Debug builds only. Upgrade here

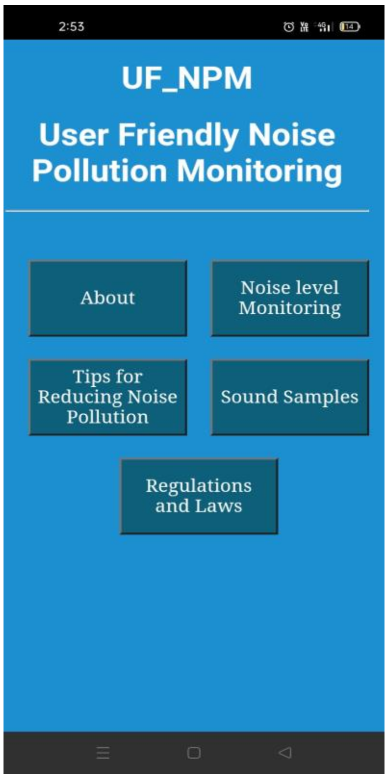
UF_NPM
Android debug
Sun Oct 29 2023 11:58:37 GMT+0530 (India Standard Time)

Copy URL for use on device to clipboard

**Conversion of Web page into Mobile app**

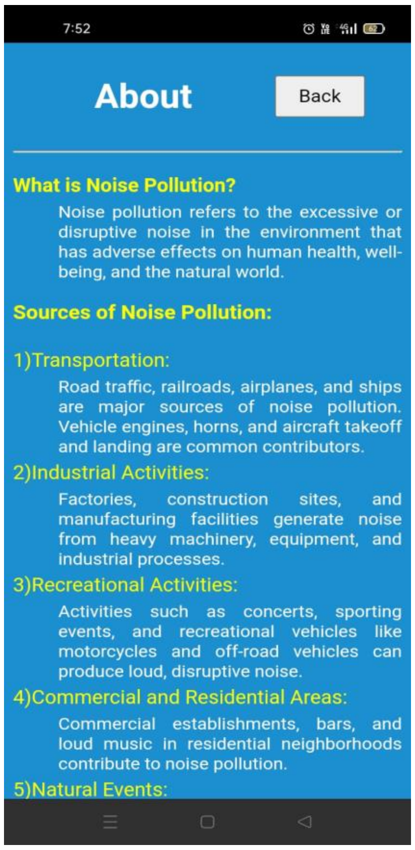
## Working of UF_NPM (User Friendly Noise Pollution Monitoring) App

### Home Screen:


2:53
UF_NPM
User Friendly Noise Pollution Monitoring
About
Noise level Monitoring
Tips for Reducing Noise Pollution
Sound Samples
Regulations and Laws

**Home page**

**About:** Shows users about what is noise pollution, its sources and ways to migrate it.



7:52

About

Back

What is Noise Pollution?

Noise pollution refers to the excessive or disruptive noise in the environment that has adverse effects on human health, well-being, and the natural world.

Sources of Noise Pollution:

1)Transportation:

Road traffic, railroads, airplanes, and ships are major sources of noise pollution. Vehicle engines, horns, and aircraft takeoff and landing are common contributors.

2)Industrial Activities:

Factories, construction sites, and manufacturing facilities generate noise from heavy machinery, equipment, and industrial processes.

3)Recreational Activities:

Activities such as concerts, sporting events, and recreational vehicles like motorcycles and off-road vehicles can produce loud, disruptive noise.

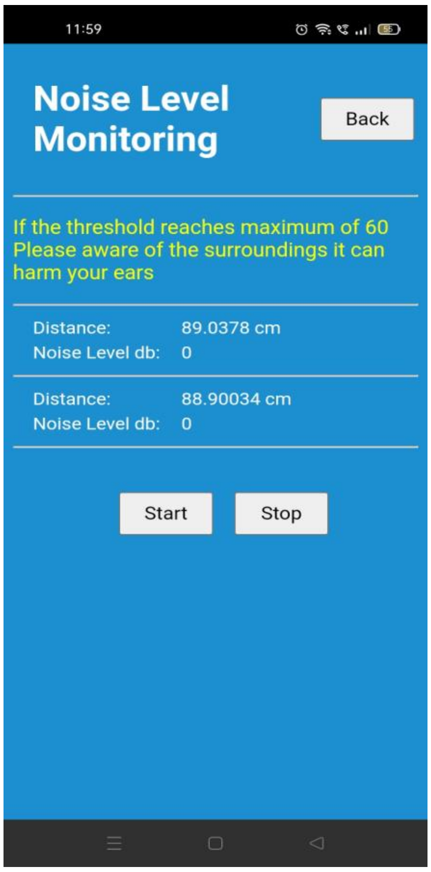
4)Commercial and Residential Areas:

Commercial establishments, bars, and loud music in residential neighborhoods contribute to noise pollution.

5)Natural Events:

**About page**

**Noise level monitoring**: Shows the Noise level around them with the distance. Start button activate the sensor and stop button deactivate the sensor.



11:59

Noise Level Monitoring

Back

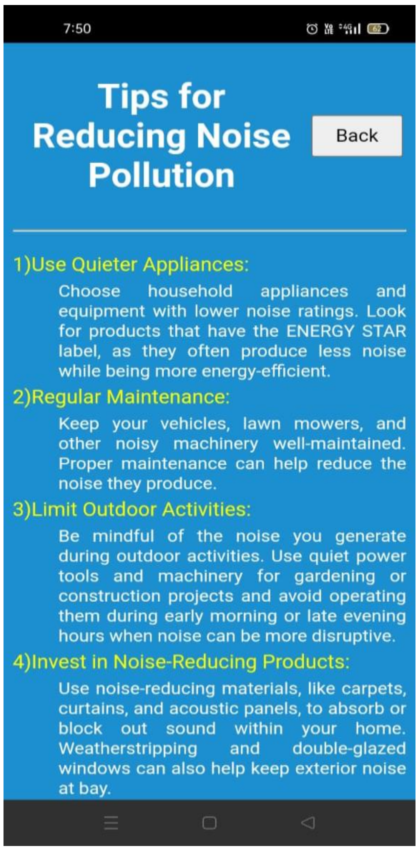
If the threshold reaches maximum of 60 Please aware of the surroundings it can harm your ears

|  |  |
|---|---|
| Distance: | 89.0378 cm |
| Noise Level db: | 0 |
| Distance: | 88.90034 cm |
| Noise Level db: | 0 |

Start Stop

**Noise level monitoring page**

**Tips for Reducing Noise Pollution**: Shows how to reduce noise pollution by eco-friendly.


7:50

**Tips for Reducing Noise Pollution**

Back

1)Use Quieter Appliances:

Choose household appliances and equipment with lower noise ratings. Look for products that have the ENERGY STAR label, as they often produce less noise while being more energy-efficient.

2)Regular Maintenance:

Keep your vehicles, lawn mowers, and other noisy machinery well-maintained. Proper maintenance can help reduce the noise they produce.

3)Limit Outdoor Activities:

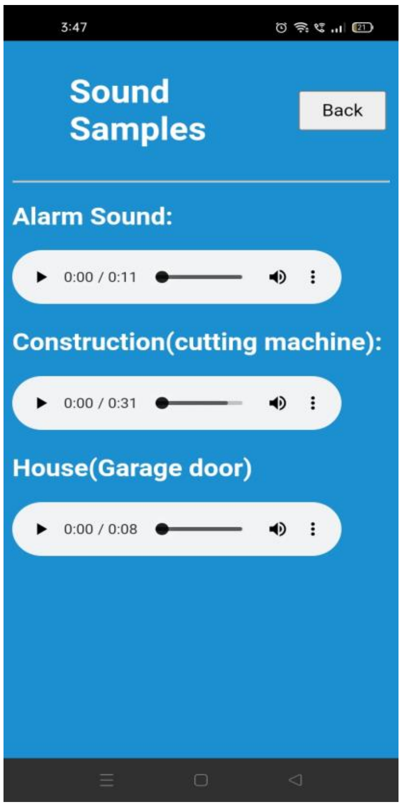
Be mindful of the noise you generate during outdoor activities. Use quiet power tools and machinery for gardening or construction projects and avoid operating them during early morning or late evening hours when noise can be more disruptive.

4)Invest in Noise-Reducing Products:

Use noise-reducing materials, like carpets, curtains, and acoustic panels, to absorb or block out sound within your home. Weatherstripping and double-glazed windows can also help keep exterior noise at bay.

**Tips for Reducing Noise Pollution page**

**Sound Sample:** Shows some sample sounds to users **for** their easy understanding. (e.g. Noises made during house works and in construction sites)


3:47

**Sound Samples**

Back

Alarm Sound:

0:00 / 0:11

Construction(cutting machine):

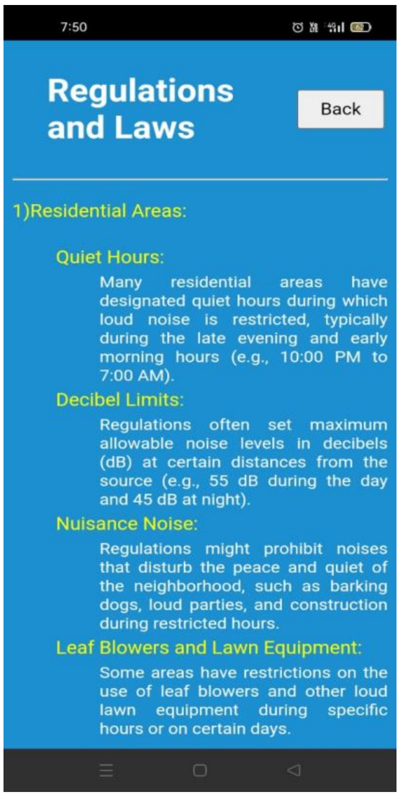
0:00 / 0:31

House(Garage door)

0:00 / 0:08

**Sound Sample page**

**Regulations and Laws**: Shows the basis rules about noise pollution that happens in residential areas, entertainment districts, Special events, Commercial and industrial areas.


7:50

Regulations and Laws

Back

1)Residential Areas:

Quiet Hours:

Many residential areas have designated quiet hours during which loud noise is restricted, typically during the late evening and early morning hours (e.g., 10:00 PM to 7:00 AM).

Decibel Limits:

Regulations often set maximum allowable noise levels in decibels (dB) at certain distances from the source (e.g., 55 dB during the day and 45 dB at night).

Nuisance Noise:

Regulations might prohibit noises that disturb the peace and quiet of the neighborhood, such as barking dogs, loud parties, and construction during restricted hours.

Leaf Blowers and Lawn Equipment:

Some areas have restrictions on the use of leaf blowers and other loud lawn equipment during specific hours or on certain days.

**Regulations and Laws page**

## A Real-Time Noise Level Monitoring System Plays a Crucial Role in Promoting Public Awareness and Contributing to Noise Pollution Mitigation in Several Ways:

- **Immediate Awareness:**

  Real-time monitoring provides instant feedback on noise levels in specific areas, making people aware of the current noise pollution status. This immediate awareness can help individuals and authorities understand the severity of noise pollution at any given time.

- **Education and Information:**

  Access to real-time noise data can educate the public about noise levels in their surroundings. When people have access to this information, they become more informed about the impact of noise pollution on their health and well-being. This knowledge often leads to more responsible and considerate behaviour in terms of noise generation.

- **Behavioural Changes:**

  Awareness of noise levels in real-time encourages people and businesses to alter their behaviours. For instance, if a specific area is experiencing high noise levels, individuals or organizations might make adjustments by reducing loud activities, adjusting equipment, or modifying schedules to minimize noise output.

- **Community Engagement:**

  Real-time noise monitoring often involves community engagement. Residents and stakeholders can actively participate in monitoring and reporting noise levels in their vicinity, fostering a sense of collective responsibility. This engagement can lead to collaborative efforts in identifying noise sources and finding solutions to mitigate them.

- **Policy and Planning:**

  Data collected through real-time monitoring can inform policymakers and urban planners. This information helps in creating effective policies, zoning regulations, and urban planning strategies aimed at reducing noise pollution. For instance, city planners can use this data to designate quiet zones, adjust traffic flow, or regulate noise-emitting activities.

- **Compliance and Enforcement:**

  Real-time monitoring data can be used as evidence for enforcing noise regulations. Authorities can take action against noise violators based on the real-time data, ensuring compliance with established noise level standards.

- **Continuous Improvement:**

  Continuous monitoring allows for the assessment of the effectiveness of noise reduction measures. It provides feedback on the impact of implemented changes and helps in refining strategies for noise mitigation.

By enhancing public awareness and facilitating active participation in noise pollution management, real-time noise level monitoring systems contribute significantly to creating quieter, healthier, and more liveable environments. The combination of informed individuals, engaged communities, and responsive authorities can lead to more effective and sustainable noise pollution mitigation strategies.