

IoT cloud based noise intensity monitoring system

Tashfat Fatema, Md. Azizul Hakim, Tanjila Khan Mim, Mushrat Jahan Mitu, Bijan Paul

Department of Computer Science and Engineering, University of Liberal Arts Bangladesh, Dhaka, Bangladesh

Article Info

Article history:

Received Jul 13, 2022

Revised Nov 12, 2022

Accepted Nov 19, 2022

Keywords:

Arduino IoT Cloud

Decibel

ESP8266

Noise intensity

Sound pollution

Sound sensor

ABSTRACT

According to a survey published by the United Nations Environment Program (UNEP) last month, Dhaka, Bangladesh's capital, is the world's noisiest metropolis. Dhaka is consistently ranked as one of the most polluted cities in the world. The World Health Organization (WHO) has set a noise intensity limit of 55 decibels for a certain region. Dhaka, on the other hand, was determined to be twice as loud, at 110 to 132 decibels. As many as 66 percent of traffic cops work on the road to control traffic, and this high noise intensity noise is causing them hearing and sleeping problems on a daily basis. Furthermore, loud noises can increase blood pressure and pulse rates, produce mental stress, and interfere with amusement and personal interactions. Keeping all of this in mind, we designed a real-time noise pollution monitoring system that will assist in conducting experiments in this type of environment by evaluating noise intensity and automatically sending data to a IoT cloud-based platform.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Bijan Paul

Department of Computer Science and Engineering, University of Liberal Arts Bangladesh

Dhaka, Bangladesh

Email: bijan.paul@ulab.edu.bd

1. INTRODUCTION

Noise pollution is one of the most critical and overlooked environmental challenges. According to the World Health Organization (WHO), noise pollution from traffic or roads has a significant negative impact on people's health and living quality. An undesired sound made by activities that is damaging or destructive to one's health-quality of life is referred to as environmental noise. Noise is defined as an auditory pollution or a sound that is 'out of place' [1]. Despite all of the difficulties and negative consequences of noise pollution, we must eventually go about our daily lives. So, keeping all of these factors in mind, our system will operate in any setting and will be a highly portable device to carry around. As our system is Arduino IoT cloud based, we must acquire or collect data in real time and use it for future purposes. As a result, because our system is web based as well as mobile application based, it has actual data for subsequent technological use in any assessment process as well as easy access to real time noise monitoring analysis data, make up an integrated supervisory system for increased comfort and strategies to make it easier to manage noise levels [2], [3]. Children are becoming ill as a result of the high noise levels, and pregnant women are suffering as a result of hearing loss. There's also the possibility that this will have an impact on wildlife. If noise pollution is not successfully controlled, it will have substantial long-term public health repercussions. People become habituated to their home's noise levels and are unable to distinguish background noise pollution [4]. Our solution is IoT-based, which means it collects real-time data on decibel variables. The internet of things (IoT) concept aims to connect people, data, things, and processes in a global network that, when used wisely, can dramatically improve daily life, especially in smart cities [5]. In current modern cities, noise measurements are mostly done by local professional police officers. The officers primarily collect noise data in a specific location or area for examination and storage

using a sound measuring meter [6]. It will be difficult, but not impossible, to transform Dhaka into a noise pollution free city. We are actually using the Arduino IoT Cloud to capture real-time data on a daily basis. For our proposed system, Arduino IoT Cloud is primarily a software that can quickly retrieve data while also being simple and secure to utilize. The decibel unit is commonly used to measure sound. The loudness of the surroundings varies for the noise intensity level depending on the decibel range. Sound waves are produced by longitudinal waves moving in a To and Fr direction.

2. LITERATURE REVIEW

The human ear perceives sound as a change in pressure wave migrating through the air. However, noise may be quite unpleasant. The distinction between loudness and sound is identified by the receiver and the situation.

As indicated by a review led by Lapono and Pingak [7], the assessment among experimental instruments and selective laser melting (SLM) design of systems, exhibits that the results are not essentially different. Therefore, it was identified that the proposed equipment is able to proficiently perform well. The equipment might screen 44 and 94 dB sound reach and give admonitions through cautions.

In the work Soler-Llorens *et al.* [8], a minimal expense Arduino based seismometer was implemented for educational purposes. However, the inspection frequency and amplitude vary, making it unreliable for study. The precise synchronization among the monitors necessary for acoustic noise element observations is not properly addressed in this scenario.

In published research Fernandez-Prieto *et al.* [9]), some analyses have been initiated between the noise indices produced by the IoT devices. It includes those from a conventional noise level meter. The observations illustrate that the proposed technique is well equipped for use as a preliminary step for obtaining accurate noise maps in smart urban areas.

Rajagukguk and Sari [10] constructed a noise detection library implementing the Arduino mega micro-controller 8535. Its has a throughput already in the form of signal sounds. However, its loudness is just on a 7-segment, with such a decibel pressure of 43.4 dB in the space.

Jarupula [11] have shown a commotion recognizer. It uses an micro-controllers coded audio system is designed. This equipment alerts everybody when it detects exuberant speech sound levels exceeding 120 to 140 dB.

According to a study ShieldSquare Captcha [12], a large percentage of workers in the palm oil processing industry in Lampung, Indonesia. They suffer from hearing loss as a result of the high levels of noise in the area. There really is a loud intensity in that province.

Saki and Kehtarnavaz [13] devised a system that can record sound in actual time. The random forest classifier was primarily employed in the development of this system. The essential aspects and sub features of this system mostly state the sound signals.

Zimmerman and Robson [14] claim that they can send a short message service (SMS) text with the entire day's sound summary in less than a minute using their method. It is straightforward because the proposed microprocessor operates at a low level of cost. They developed the system cost-effectively.

Talari *et al.* [15] proposed in their study that their system can monitor sound levels in different regions. Their system offer data to climate noise pollution authority not only in real life but also in space time layout. Their work focused on data visualization of the sound levels.

Navarro *et al.* [16] on the suggested study largely focused on the environment noise pollution problem and how to reduce it in a solace by creating environmental acoustic in the most polluted cities. The writers saw noise pollution as a key concern in residential settings. They used an IoT sensor to acquire the data. Their work, on the other hand, focuses more on the acquisition of large datasets.

The proposal of Piczak [17] for a short work on sound level classification in the environment. With the expanded dataset, he applied a 1D neural network model. It has improved the system's performance.

In the work of Bartalucci *et al.* [18] describe a new smart noise monitoring system. The system was primarily created for the LIFE15 ENV/IT/000586 "Methodologies for noise low emission zones introduction and management" project. The prototype system was created with consideration for both the project's monitoring requirements and cutting-edge systems.

In order to automatically categorize the various noise sources, Manglani *et al.* [19] developed an acoustic pattern classification algorithm that runs in a wireless sensor. Their main objective was to monitor a

noise source classifier that was trained using only a few minimally annotated recordings. Over the course of 50 days on the work area, their technology performed consistently and accurately in detecting noise sources.

Czyzewski *et al.* [20] presented a concept and were able to put it into use for a dynamic acoustical map-based online system for monitoring urban noise. They show the outcomes of simulations and measurements made by the system prototype. The results of the noise measurements collected by their method and the resulting acoustic maps are evaluated with some of the other alternatives.

In order to solve the core anomalous issues in future advanced core operation, Mori *et al.* [21] developed a boiling water reactor (BWR) core noise monitoring system. Various modern signal processing techniques are introduced while they were monitoring in-core status from a small number of signals. By creating these algorithms, they aim to show that they work by analyzing actual plant data. A summary on some of the similar research work is shown in the Table 1. It presented what micro-controller (MCU), sensor, architecture, and some other features one researcher used to develop their system.

Table 1. Some of the similar research work summary on noise monitoring system vs our proposed system

Researcher	MCU	Sensor	Architecture	Connectivity	LCD	Data access	Easy installation
Maijala <i>et al.</i> [22]	Raspberry PI 3	Two microphones	IoT	3G/4G	NO	Web	NO
Alsina-Pagès <i>et al.</i> [23]	FRDMKL25Z and Teensy 3.2 Node	4544PF-W	IoT	Wi-Fi/3G	NO	NO	NO
Noriega-Linares and Ruiz [24]	Raspberry PI 2	T-Bone GC 100 USB	IoT	Ethernet	NO	NO	NO
Alsina-Pagès <i>et al.</i> [25]	Jetson TK1	4544PF-W	WSN	Ethernet	NO	NO	NO
Our proposed system	ESP8266 NODEMCU-12E	LM393 sound sensor module	IoT	Wi-Fi	YES	Web and mobile application	YES

Along with all of the related work, our work may be unique in that our system collects real-time noise intensity data from all types of environments with respect to date and time in decibels, making our system more adaptable and efficient than others. Our system collects data in real time utilizing the Arduino IoT cloud. We can also see the mobile application version of Arduino IoT cloud in live mode and get an idea of the noise level in that particular location by using this application. Our system is design is also incredibly user-friendly and portable. As a result, a noise measurement and recording device is highly suggested. More specifically, our effort intends to develop a noise monitoring system, and such a task can be accomplished if the decibel level is well beyond the threshold number. In addition to all of this, our system features an LCD display that will show the mode (quiet, moderate, or loud) based on the decibel noise intensity level specified in relation to the environment. Consequently, the output display information will change as the environment changes, as will the decibel values that have been pre-set. This method is generally used to protect against noise-induced hearing loss. This project will develop and build noise level monitoring measurement instruments that will employ a noise sensor device that will emit a signal when the loudness exceeds the normal hearing threshold, and it will also display inspection data in the form of digits with LED indicators for each noise exposure and through the gauge meter and graph it will show the noise intensity level.

3. SYSTEM MODEL AND METHODS

3.1. Hardware components used in proposed model

We utilize one micro-controller, one bread board, one noise detecting sensor for environmental noise detection, and an Ethernet networking device to monitor the noise strength. We also used an Android cellphone to provide updates to the user. The Table 2 shows a brief functionality of all the hardware component used in our project. It described the breadboard, LM393, ESP8266, and LCD1602 module. And, the Figure 1 shows the physical view of these hardware components.

Table 2. List of Hardware components used in our Proposed System

Hardware component	Functionality
Breadboard	The board will interconnect all necessary hardware components to produce a comprehensive prototype of our proposed system.
Microphone sound detection sensor module LM393	It will assess the level of noise in the environment. The sensor output will be than converted into decibels unit.
ESP8266 NODEMCU-12E	It will work as the primary micro-controller unit. Using this MCU, we send sensor data to the cloud which can be utilized to make noise intensity predictions.
LCD1602 I2C display	It will show the noise intensity level and the status on its screen.

3.2. Explaining hardware in details

3.2.1. Microphone sound detection sensor module

It has a fantastic dynamic microphone that measures the sound level from the source. The sound sensor which showed in Figure 1(a) is a compact circuit board with a microphone (50Hz-10kHz) and circuit connections that convert sound waves to electrical pulses. The LM393, a comparator IC, receives this electrical pulse. The signal is digitized by the LM393 IC, which is coupled to the OUT pin. A variable resistor on this noise sensor controls the intensity of the OUT output. OUT, VCC, and GND are the three pins on it.

3.2.2. ESP8266 NODEMCU-12E

The NodeMCU ESP8266 which showed in Figure 1(b) comes with the ESP-12E module, which has the ESP8266 chip and a Tensilica Xtensa 32-bit LX106 RISC microprocessor. This CPU supports RTOS and runs at a clock frequency of 80MHz to 160MHz. The NodeMCU has 128 KB of RAM and 4 MB of Flash memory to store data and programs. Because of its high computing power and built-in Wi-Fi or Bluetooth, it's ideal for IoT projects. The primary component of this system is the NodeMCU ESP8266 12E, which maintains the Sensor Module LM393 and LCD display by connecting the pins of all of these components together and keeping the system procedure smooth and maintainable. The ESP8266 is a fantastic component that allows you to keep the entire process on a single breadboard, logically connected with the relevant pins via wired connections.

3.2.3. LCD1602 I2C display

LCD1602 I2C Display which showed in Figure 1(c) is a 16x2 LCD display screen with I2C interface. The characters in this component are white on a blue background and may display 16x2 characters on two lines. The wire soldering and connection is also somewhat difficult.

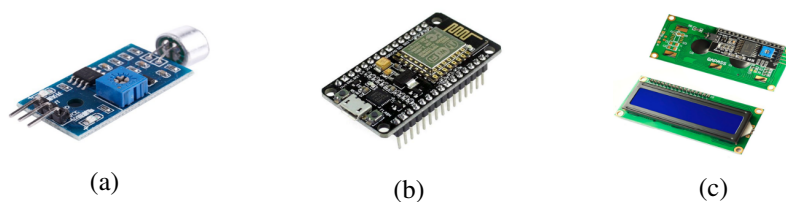


Figure 1. Hardware components used in the system (a) microphone sound detection sensor module, (b) ESP8266 NODEMCU-12E, and (c) LC1602 I2C display

4. IMPLEMENTATION OF THE SYSTEM

Our system was built using the LM393 sound sensor module and the ESP8266. To finish the connection, the GND of the Sensor Module is linked to the GND of the ESP8266. In addition, the VCC and OUT pins of the sensor module are connected to 3V3 and A0 on the ESP8266, respectively. The LCD1602 I2C Display GND pin, on the other hand, is connected to the ESP8266's GND pin. The LCD's VCC is also connected to the ESP8266's VIN. The LCD display's SDA and SCL pins are connected to the ESP8266's D2 and D1 pins, correspondingly. The block diagram is shown in the Figure 2. The pin connections between the three hardware components, ESP8266 NODEMCU-12E, microphone sound detection sensor module LM393 and LCD1602 I2C display respectively are shown in Table 3 as follows.

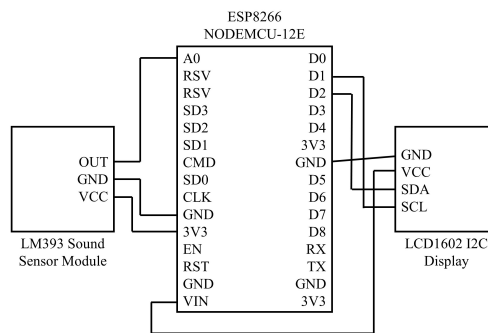


Figure 2. Block diagram of the proposed system

Table 3. Pin connection of NodeMCU-12E and LM393

ESP8266 NODEMCU-12E pin	Microphone sound detection sensor module LM939 Pin	LCD1602 I2C display pin
3V3	VCC	-
GND	GND	GND
A0	OUT	-
VIN	-	VCC
D1	-	SCL
D2	-	SDA

The Arduino IoT cloud must be connected to the hardware system. We must finish all of the settings in accordance with our system policy, such as setting the variable to decibel and collecting the ethernet to that cloud, after successfully logging into the software. Finally, we must set the board’s NodeMCU 1.0 (ESP-12E module) and port configurations. Before connecting our hardware configuration, we must additionally launch the Arduino agent from our PC. We must make alterations to the in the flavors area, immediately beside the boards, by selecting V2 IPV6 higher bandwidth and all flash content for further work. We must code and send our work to the program and the board after we have completed all of the steps. After we finish uploading, we will get the real-time data we need from the environment.

On the other hand, for the mobile application, go to the Google Play Store and download the Arduino IoT remote app. We must first log into the application using the provided information, or we can use Google to do it. We must configure the program after logging in by connecting to the server, which occurs automatically. When a user successfully logs into the application, data from that environment is retrieved, and the user’s dashboard displays the noise intensity level in that environment. Figure 3 depicts the whole architectural layout and setup of our proposed system where Figure 3(a) is the architectural diagram and Figure 3(b) is the hardware setup of the system.

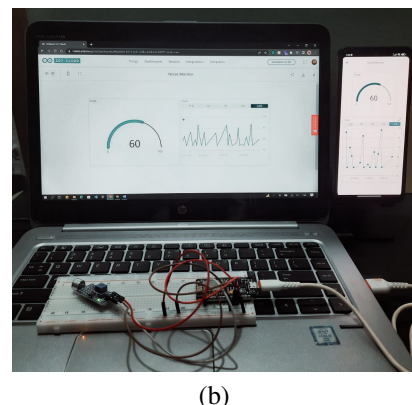
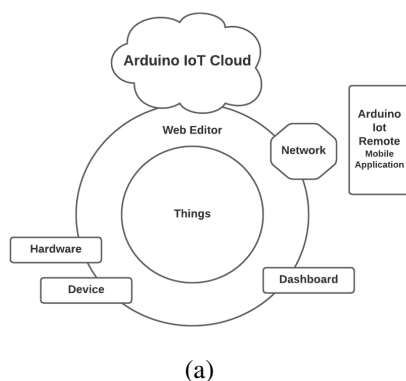


Figure 3. The design of the proposed system (a) architectural diagram and (b) the setup of the system

The suggested system's pseudo code is also shown as follows. To begin, we set up all of the variables we will require throughout the program using code. To get the single at the right spot, we set the signal maximum value to 0 and the signal minimum value to 1024. In order to receive the data, this will be reversed. The while loop was then utilized to evaluate the real signal's state.

The analog reading is set to 0 by default. The first if condition checks whether the sample in our input signal is smaller than 1024, if it is, it will go inside the loop and check the other condition, comparing the value sample to the maximum value, if it is, the value will be stored in the signal maximum variable, otherwise it will check the other condition, eventually storing the value in the signal minimum variable. After confirming all of the conditions, we will calculate the difference between the maximum and minimum signal. We save it in the *peakToPeak* variable. Then use calibration to transfer the value to db that is in decibel, the real variable.

```
peakToPeak = 0, signalMax = 0, signalMin = 1024
while (millis()- millis() < sampleWindow)
  sample = analogRead(0)
  if (sample < 1024)
    if (sample > signalMax) signalMax = sample
    else if (sample < signalMin) signalMin = sample
peakToPeak = signalMax - signalMin
db = map(peakToPeak, 20, 900, 49.5, 90)
```

To show the noise intensity level on the LCD screen, we used *LiquidCrystal_I2C* library for the I2C LCD screen module. We show the decibel value of the noise intensity in the first row of the LCD screen. And, in the second row of the LCD screen, we show the noise intensity status. The LCD works then proceed by initializing the values after doing so in code. As stated in the code, the cursor is set to (0,0). Then a print statement is given, which will print the output display when the system starts working according to the conditions set. After that, it will check the condition and display a *QUITE* message in the LCD display if the decibel value is less than or equal to 60 decibels. Again, if the decibel condition is more than or equal to 85 decibels, the output in display will be *HIGH*.

However, if both conditions are not met, it will output *MODERATE*, which is in the midst of the two above values. The moderate condition is defined as a decibel level of louder than 60 decibels but less than 85 decibels. Only if both conditions are true and met will it enter the loop. Otherwise, it will check the other condition. This status is showed in the right alignment of the second row of the screen. After delaying data collection for 1,500 milliseconds, new data is being collected.

```
lcd.setCursor(0, 0)
lcd.print("Noise: " + db + " dB")
if (db <= 60) lcd.print("QUITE")
else if (db > 60 && db < 85) lcd.print("MODERATE")
else if (db <= 85) lcd.print("HIGH")
delay(1500)
```

To show the noise intensity level on the LCD screen, we used *LiquidCrystalLiquidCrystal_I2C* library for the I2C LCD screen module. We show the decibel value of the noise intensity in the first row of the LCD screen. And, in the second row of the LCD screen, we show the noise intensity status. The LCD works then proceed by setting the cursor to (0,0). Then a print statement is given, which will print the output display when the system starts working according to the conditions set. After that, it will check the condition and display a *QUITE* message in the LCD display if the decibel value is less than or equal to 60 decibels. Again, if the decibel condition is more than or equal to 85 decibels, the output in display will be *HIGH*. However, if both conditions are not met, it will output *MODERATE*, which is in the midst of the two above values. The moderate condition is defined as a decibel level of louder than 60 decibels but less than 85 decibels. After delaying data collection for 1,500 milliseconds, new data is being collected.

4.1. Web data parsing

Our proposed system also works extremely well for web applications. To do so, a user must first register for the Arduino IoT Cloud software, or if they already have an account, they must login. After successfully

logging into the program, we must complete all of the settings in accordance with our system policy, such as setting the variable to decibel and collecting the ethernet to that cloud. Finally, we must configure the board to NodeMCU 1.0 (ESP-12E Module) and the port. We must also launch the Arduino agent from our PC before connecting our hardware configuration. In the flavors section, immediately beside the boards, we must make adjustments to the by selecting V2 IPV6 higher bandwidth and all flash content for further use. After completing all of the procedures, we must code and send our work to the software and the board. We will acquire our desired real-time data from the environment after we finish uploading.

4.2. Mobile data parsing

The technology collects data from the environment and displays it on a mobile app available on the Google Play Store. We must first log into the application using the specified credentials, or we can do it using Google. After logging in, we must configure the application by connecting to the server, which happens automatically. When a user successfully logs into the application, data from that environment begins to be retrieved, and the noise intensity level in that environment is displayed on the user’s dashboard.

5. RESULTS AND ANALYSIS

For the result and analysis section, we did our utmost to make the system run as smoothly as possible. Our goal was to achieve the highest possible level of success with our proposed project. We have tested our system in a variety of ways in order to preserve this. For instance, we might make a loud noise to see if our system is working properly. As a result, it produces the ideal result. On the other hand, by reducing noise intensity in a certain location, we were able to determine whether the sound level gap is acceptable to the system. The system also displays the expected outcome on this base station.

The output of the sensor reading shows in the LCD display and Android IoT Cloud dashboard; the output shows in the Figure 4. Regarding better results and visualizations, we utilized an LCD display to indicate the noise in a certain area, as well as the amount of noise mode, to make things more exact and understandable. For instance, the image of the LCD display in Figure 4(a) demonstrates the noise level in decibels and a sound level that is *QUITE* for that particular location.

To display the result in the dashboard, we have used a graph chart in relation to the data verses time after presenting the result in the gauge meter. The chart will primarily show the noise intensity level for various types of environments. Additionally, we used a gauge meter and labeled it with the variable decibel in Figure 4(b). So that the decibel output of the displayed noise intensity equals decibel.

In the Figure 4(c), the mobile app also displays the fine result of the noise intensity level, and the dashboard is set up as a gauge meter as well as a graph to make it easy to grasp. Because everyone has a smartphone in this highly technological age, having a cloud-based application that provides information about noise levels in a given area could be beneficial to the general public.

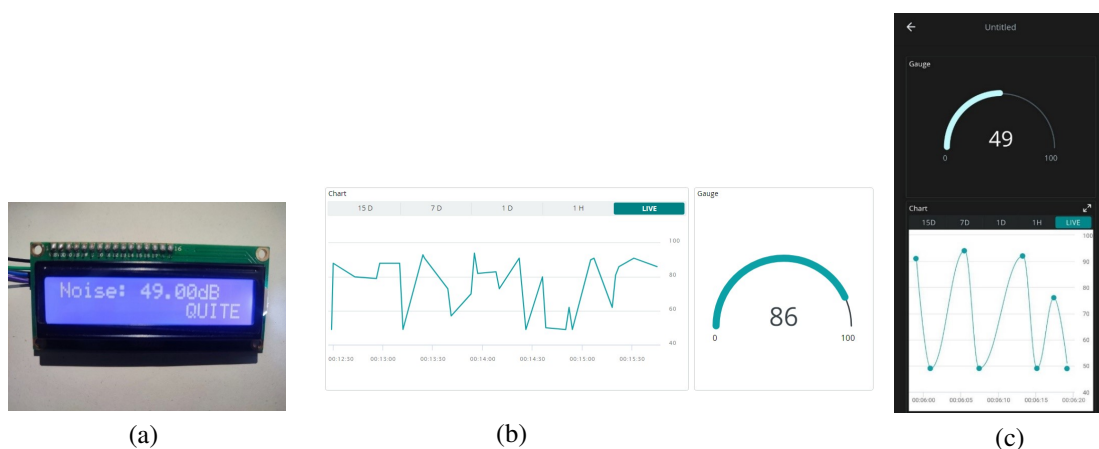


Figure 4. Output of the sensor reading (a) sensor reading and intensity status on the LCD module, (b) graph and gauge meter, and (c) mobile dashboard

We gathered real-time data for a week, from Sunday to Saturday. And, as shown in Figure 5, the average success rate of data collecting according to the day of the week. We've just calculated the average of the entire day's data to see if the system is operating as expected. The values ranged from 49 decibels to 86 decibels, with 86 decibels being the highest. On Saturday and Sunday, the noise level in the house was very low. On working days, which were Monday to Thursday, the values ranged from 61 to 72 decibels, respectively. We went out on Friday in multiple places, many of which were crowded, and because it was a holiday in our country, the average range was 86 decibels. All of this information was gathered at random in order to get a sense of the system's performance.

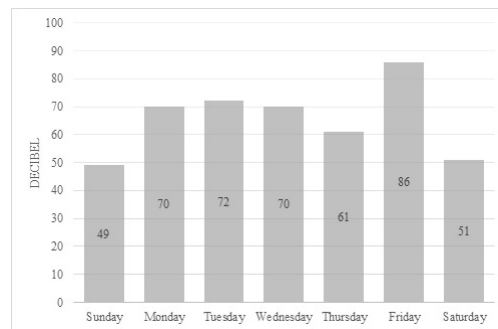


Figure 5. Graph of sound intensity levels averaged for each day of the week

6. APPLICATION

The prototype of our proposed design has some specific applications. The followings are few of the most important applications of the system. It will be very helpful to monitor the noise intensity in a particular situation.

- The proposed system can measure noise pollution is measured in a variety of environments. Through the system the noise level is mostly monitored outside, but it can also be done indoors.
- The proposed system is designed to measure the degree of noise intensity in any situation and relatively easy to use and portable.
- This system can also be used in robots because it is tiny and portable, and it can sensor noise levels by using data and able to control noise pollution in high-traffic areas by collecting real-time data and taking appropriate measures.
- The proposed approach can also be used in industrial regions, where noise pollution is a significant issue.
- An user can always have this system with them and use it to record their daily statistics. The data is easy to view because our system also operates as a mobile application. Additionally, as our system gathers real-time data, it may be applied to machine learning.

7. USER SATISFACTION LEVEL

We requested certain individuals to use our proposed system when it was completed and indicate their feeling of satisfaction with it, as well as provide some suggestions or evaluations, so that we might improve the system and make it more advanced and maintainable by addressing the issues. Table 4 displays the user's sense of satisfaction as well as some suggestions. However, customer satisfaction is extremely high and positive, indicating that the system is indeed a success.

Table 4. User satisfaction and suggestion

Features	Satisfied %	Suggestion
ESP8266 NODEMCU-12E	80	It works well; however, you could make it even better by adding the Bluetooth module.
Microphone sound detection sensor module LM393	70	It is quite effective.
LCD1602 I2C display	70	More information would be beneficial.
IoT Cloud dashboard	99	It was interesting to see something different.
Data access (web and mobile)	99	Could actually acquire data.

8. CONCLUSION

Our system primarily measures the noise levels in different environments and is able to identify the label of that environment by indicating a specific label, such as quite, moderate, or high, based on the level of noise in that specific region. This will make people aware of the possibility that their environment could lead to serious health problems. The main goal of our system is to gather data in the field so that more research can be done to help reduce this problem and enlighten people about such environments. To do this, we used the Arduino IoT cloud to collect data in real-time. With the use of this application, we can also view the Arduino IoT cloud's mobile application in live mode and get a sense of the level of noise in that particular area. The design of our system is also incredibly user-friendly and portable.

Our system retrieves and stores data in real time, it could be highly valuable for future work in machine learning. It is also portable due to its simplicity and compact size. As a result, a user can easily collect noise intensity data using this method in their daily lives. Furthermore, because our system is web-based and mobile-based, the user may get a feel of the amount of noise intensity in various environments.




REFERENCES

- [1] E. Murphy and E. A. King, "Principles of environmental noise," in *Environmental Noise Pollution*, Elsevier, 2014, pp. 9–49 doi: 10.1016/B978-0-12-411595-8.00002-1.
- [2] D. Leaffer, C. Wolfe, S. Doroff, D. Gute, G. Wang, and P. Ryan, "Wearable ultrafine particle and noise monitoring sensors jointly measure personal co-exposures in a pediatric population," *International Journal of Environmental Research and Public Health*, vol. 16, no. 3, p. 308, 2019, doi: 10.3390/ijerph16030308.
- [3] H. Cho, "An air quality and event detection system with life logging for monitoring household environments," in *Smart Sensors at the IoT Frontier*, Cham: Springer International Publishing, 2017, pp. 251–270, doi: 10.1007/978-3-319-55345-0_10.
- [4] L. Jalali, P. Bigelow, M.-R. Nezhad-Ahmadi, M. Gohari, D. Williams, and S. McColl, "Before–after field study of effects of wind turbine noise on polysomnographic sleep parameters," *Noise and Health*, vol. 18, no. 83, p. 194, 2016, doi: 10.4103/1463-1741.189242.
- [5] P. J. S. Cardoso, J. Monteiro, J. Semião, and J. M. F. Rodrigues, Eds., *Harnessing the internet of everything (IoE) for accelerated innovation opportunities*. IGI Global, 2019, doi: 10.4018/978-1-5225-7332-6.
- [6] L. Filippini, S. Santini, and A. Vitaletti, "Data collection in wireless sensor networks for noise pollution monitoring," in *Distributed Computing in Sensor Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 492–497, doi: 10.1007/978-3-540-69170-9_35.
- [7] L. A. S. Lapono and R. K. Pingak, "Design of sound level meter using sound sensor based on Arduino Uno," *Jurnal ILMU DASAR*, vol. 19, no. 2, p. 111, 2018, doi: 10.19184/jid.v19i2.7268.
- [8] J. L. Soler-Llorens, J. J. Galiana-Merino, B. Y. Nassim-Benabdeloued, S. Rosa-Cintas, J. Ortiz Zamora, and J. J. Giner-Caturla, "Design and implementation of an Arduino-based plug-and-play acquisition system for seismic noise measurements," *Electronics*, vol. 8, no. 9, p. 1035, 2019, doi: 10.3390/electronics8091035.
- [9] J.-A. Fernandez-Prieto, J. Canada-Bago, and U. Birkel, "A fuzzy rule-based system to infer subjective noise annoyance using an experimental wireless acoustic sensor network," *Smart Cities*, vol. 5, no. 4, pp. 1574–1589, 2022, doi: 10.3390/smartcities5040080.
- [10] J. Rajagukguk and N. E. Sari, "Detection system of sound noise level (SNL) based on condenser microphone sensor," *Journal of Physics: Conference Series*, vol. 970, p. 012025, 2018, doi: 10.1088/1742-6596/970/1/012025.
- [11] V. Jarupula, "Noise detector with automatic recording system using Arduino with IoT," *SSRN Electronic Journal*, 2021, doi: 10.2139/ssrn.3919052.
- [12] "Arduino - Home." <https://www.arduino.cc> (accessed May 22, 2022).
- [13] F. Saki and N. Kehtarnavaz, "Real-time hierarchical classification of sound signals for hearing improvement devices," *Applied Acoustics*, vol. 132, pp. 26–32, 2018, doi: 10.1016/j.apacoust.2017.11.007.
- [14] T. Zimmerman and C. Robson, "Monitoring residential noise for prospective home owners and renters," in *Pervasive Computing. Pervasive 2011. Lecture Notes in Computer Science*, Berlin, Heidelberg: Springer, 2011, pp. 34–49, doi: 10.1007/978-3-642-21726-5_3.
- [15] S. Talari, M. Shafie-khah, P. Siano, V. Loia, A. Tommasetti, and J. Catalão, "A review of smart cities based on the internet of things concept," *Energies*, vol. 10, no. 4, p. 421, 2017, doi: 10.3390/en10040421.
- [16] J. M. Navarro, J. B. Tomas Gabarron, and J. Escolano, "On the application of big data techniques to noise monitoring of smart cities," in *Euroregion 2016*, 2016, pp. 1–10.
- [17] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6, doi: 10.1109/MLSP.2015.7324337.
- [18] C. Bartalucci, F. Borchi, M. Carfagni, R. Furferi, and L. Governi, "Design of a prototype of a smart noise monitoring system," in *Proceedings of the 24th International Congress on Sound and Vibration (ICSV24)*, 2017, pp. 23–27.
- [19] T. Manglani, A. Srivastava, A. Kumar, and R. Sharma, "IoT based air and sound pollution monitoring system for smart environment," *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, 2022, pp. 604–607, doi: 10.1109/ICEARS53579.2022.9752128..
- [20] A. Czyzewski, J. Kotus, and M. Szczodrak, "Online urban acoustic noise monitoring system," *Noise Control Engineering Journal*, vol. 60, no. 1, pp. 69–84, 2012, doi: 10.3397/1.3670102.
- [21] M. Mori, M. Kaino, S. Kanemoto, M. Enomoto, S. Ebata, and S. Tsunoyama, "Development of advanced core noise monitoring system for BWRS," *Progress in Nuclear Energy*, vol. 43, no. 1–4, pp. 43–49, 2003, doi: 10.1016/S0149-1970(03)00009-X.
- [22] P. Maijala, Z. Shuyang, T. Heittola, and T. Virtanen, "Environmental noise monitoring using source classification in sensors," *Applied Acoustics*, vol. 129, pp. 258–267, 2018, doi: 10.1016/j.apacoust.2017.08.006.




- [23] R. Alsina-Pagès, U. Hernandez-Jayo, F. Alías, and I. Angulo, "Design of a mobile low-cost sensor network using urban buses for real-time ubiquitous noise monitoring," *Sensors*, vol. 17, no. 12, p. 57, 2016, doi: 10.3390/s17010057.
- [24] J. Noriega-Linares and J. N. Ruiz, "On the application of the Raspberry Pi as an advanced acoustic sensor network for noise monitoring," *Electronics*, vol. 5, no. 4, p. 74, 2016, doi: 10.3390/electronics5040074.
- [25] R. Alsina-Pagès, J. Navarro, F. Alías, and M. Hervás, "homeSound: Real-time audio event detection based on high performance computing for behaviour and surveillance remote monitoring," *Sensors*, vol. 17, no. 4, p. 854, 2017, doi: 10.3390/s17040854.

BIOGRAPHIES OF AUTHORS






Tashfat Fatema    is a University of Liberal Arts Bangladesh undergraduate student pursuing a Bachelor of Science in CSE. She works as a Teaching Assistant at the University of Liberal Arts Bangladesh in addition to her studies. Her university also awarded her a Dean scholarship in recognition of her outstanding performance. She attended Mohammadpur Preparatory Girls' School for her elementary and secondary schooling. She is interested in researching machine learning, cybersecurity, and the internet of things. She can be contacted at email: tashfat.fatema.cse@ulab.edu.bd.






Md. Azizul Hakim    is currently a undergraduate student of Computer Science and Engineering at University of Liberal Arts Bangladesh. In addition to his studies, he works as a Teaching Assistant at his department. During his bachelor studies, he explored the competitive programming arena and attended few national and international programming contests. Besides, he is exploring his preferred research arena, Machine Learning, Deep Learning and Internet of Things. He can be contacted at email: azizul.hakim.cse@ulab.edu.bd.






Tanjila Khan Mim    is an undergraduate student pursuing Bachelor of Science in Computer Science and Engineering from the University of Liberal Arts Bangladesh. She was born and raised in Dhaka, the capital city of Bangladesh. She completed her higher secondary education at Dhanmondi Ideal college in 2018. Her research interests include sound intensity level measurement and Cloud Iot based applications. She can be contacted at email: tanjila.khan.cse@ulab.edu.bd.



Mushrat Jahan Mitu    is a Computer Science and Engineering student at the University of Liberal Arts in Bangladesh, as well as an IEEE and WIE member. She is currently enrolled in the 12th semester of BSC Engineering. She obtained her Secondary School Certificate in 2014 and Higher Secondary Certificate in 2016 from Bir Shrestha Noor Mohammad Public College. She enjoys designing software and working in teams, and she hope to work in project management in the future. She is interested in learning about automation systems and machine learning, and she is working on an IoT project. She can be reached at the email address: mushrat.jahan.cse@ulab.edu.bd.



Bijan Paul    is currently working as a senior lecturer at the Department of Computer Science and Engineering at the University of Liberal Arts Bangladesh. He has completed his B.Sc and M.Sc in CSE from Shahjalal University of Science and Technology. He was awarded the "ICT Special Award" for his work on Mongol Dip (A Bilingual Screen Reading Software for Visually Impaired People) from the Honorable Prime Minister of the Govt. of the Peoples Republic of Bangladesh for his tremendous effort and involvement in building a system for the visually impaired people. His research interest includes Deep Learning, the Internet of Things, Vehicular Adhoc Network, and Wireless Sensor Network. He can be reached at the email address: bijan.paul@ulab.edu.bd.