AIR QUALITY MONITORING AND PREDCTION SYSTEM

Team Members:

Aashka Vasava, 2413631, vasavaa3631@uhcl.edu

Arun Kari ,2433787, karia3787@uhcl.edu)

Bhargavi Allibilli, 2379197, allibillib9197@uhcl.edu

Vedant Parekh , 2406172,parekhv6172@uhcl.edu

Acknowledgment

We would like to express our heartfelt gratitude to all those who supported us throughout the development of this project, *"Air Quality Monitoring and Prediction System."* First and foremost, we are deeply thankful to our instructors, **Dr. Hisham Al-Mubaid** and **Dr. Pavan Poudel**, whose valuable insights, continuous encouragement, and constructive feedback helped shape this project from concept to completion. Their guidance not only enhanced our technical understanding but also inspired us to think critically and creatively.

We extend our sincere thanks to the **Department of Computer Science** at the **University of Houston–Clear Lake** for providing us with access to essential academic resources and a supportive learning environment. The facilities, coursework, and mentoring we received throughout the semester played a key role in helping us bring this project to life.

We are also grateful to the teaching assistants, lab staff, and technical coordinators for their support during hardware setup and system testing. The online developer communities and documentation for platforms such as ThingSpeak, Arduino, and Power BI were instrumental in solving technical challenges during development.

Lastly, we appreciate the efforts of every group member—**Aashka Vasava**, **Arun Kari**, **Bhargavi Allibilli**, and **Vedant Parekh**—for their collaboration, dedication, and strong teamwork. This experience has enriched our academic journey and reinforced the importance of innovation and teamwork in solving real-world problems.

Abstract

Air pollution remains one of the most significant environmental challenges of our time, posing critical risks to both public health and ecological stability. Traditional air quality monitoring systems are typically centralized, expensive, and offer limited geographical coverage, which makes them inaccessible for many regions—particularly in developing countries. The goal of this project is to design and implement a cost-effective, real-time air quality monitoring and prediction system using Internet of Things (IoT) technology. The proposed system employs low-cost sensors (MQ135 and DHT11) integrated with a NodeMCU ESP8266 microcontroller to gather environmental data such as temperature, humidity, and air quality index (AQI). This data is transmitted to the cloud via ThingSpeak and analyzed using Power BI for advanced visualization. Furthermore, the system lays the foundation for predictive analysis by incorporating a machine learning model capable of forecasting air quality trends. This approach aims to democratize access to environmental data, empowering communities with actionable insights for health and sustainability.

1. Introduction

Air quality has become an increasingly urgent concern in today's world, especially in densely populated urban areas where vehicular emissions, industrial discharges, and construction activities contribute significantly to pollution. The World Health Organization (WHO) estimates that air pollution causes millions of premature deaths annually, highlighting the importance of timely and accurate monitoring. However, conventional monitoring stations are expensive to install and maintain, and they often provide limited data that may not reflect real-time or hyper-local conditions. To address these limitations, this project introduces a smart, IoT-enabled air quality monitoring and prediction system that captures and analyzes air quality metrics in real-time. The system is built with affordability, portability, and scalability in mind, making it a viable solution for widespread deployment in homes, schools, and communities. By leveraging cloud computing and advanced visualization tools like Power BI, the system allows users to make informed decisions, adopt preventive health measures, and contribute to environmental sustainability efforts.

2. Project Description

This project focuses on the design and implementation of an IoT-based *Air Quality Monitoring and Prediction System* that offers real-time insights into environmental conditions such as air quality, temperature, and humidity. The core aim is to provide a low-cost, scalable, and user-friendly system that can be deployed in homes, schools, public spaces, and urban environments to support environmental awareness and proactive health measures.

At the heart of the system is the **NodeMCU ESP8266**, a microcontroller with built-in Wi-Fi capabilities that serves as the central controller. It interfaces with two primary sensors: the **MQ135 gas sensor**, which detects a range of harmful gases including carbon dioxide (CO_2), ammonia (NH_3), and benzene, and the **DHT11 sensor**, which measures both temperature and humidity. These components are connected via a breadboard and jumper wires, creating a compact and efficient circuit.

Once the sensors capture environmental data, the NodeMCU transmits it wirelessly to the **ThingSpeak** cloud platform, where the data is stored, timestamped, and made available for further analysis. Each sensor feeds its data to a separate channel field in ThingSpeak, ensuring well-structured and organized data handling. The cloud setup not only enables remote access to live data but also supports historical trend analysis by storing continuous sensor readings.

A key feature of this project is the **visualization layer**. Data from ThingSpeak is imported into **Power BI**, where it is transformed into an interactive and user-friendly dashboard. This dashboard includes real-time gauges, historical graphs, and pollution severity panels, allowing users to monitor AQI levels and observe correlations between temperature, humidity, and gas concentration.

One of the main challenges faced during the project was the seamless **integration of hardware with cloud services** and visualization platforms, all within a limited budget. Ensuring reliable sensor calibration, stable Wi-Fi connectivity, accurate data streaming, and responsive dashboards required extensive testing and optimization. The team addressed these challenges by writing efficient microcontroller code in the Arduino IDE, fine-tuning sensor readings through calibration techniques, and building robust Power BI dashboards for data interpretation.

Additionally, this project lays the foundation for **predictive analysis**, enabling the future integration of machine learning models trained on historical data to forecast AQI levels. The system also allows room for enhancements such as voice assistant support, mobile app access, and solar-powered deployment for remote outdoor monitoring.

In summary, this project showcases a holistic approach to environmental monitoring by combining embedded systems, cloud computing, and data analytics, ultimately providing a practical and scalable solution to a globally relevant problem.

Workload Distribution:

- Aashka: Power BI dashboard, data analysis
- Vedant: Hardware setup, sensor calibration
- Bhargavi: System design, report writing
- Arun: Cloud integration, future work planning

3. Background

The increasing concern for air pollution and its impact on public health has led to growing interest in smart environmental monitoring systems. Numerous studies and real-world deployments have demonstrated the potential of **embedded systems and Internet of Things (IoT)** technologies in creating low-cost, scalable, and real-time air quality solutions. These systems enable individuals and communities to gain access to critical environmental data that was traditionally only available through expensive and centralized government monitoring stations.

Our project is inspired by these advancements and utilizes a blend of hardware, software, and programming skills to bring a functional monitoring system to life. The **software tools** used include the **Arduino IDE** for programming the microcontroller, **ThingSpeak** as the cloud platform for data storage and visualization, and **Power BI** for building interactive dashboards and analyzing environmental data trends.

On the **hardware** side, the system is built using the **NodeMCU ESP8266**, a Wi-Fi-enabled microcontroller that acts as the brain of the setup. It interfaces with the **MQ135 sensor** to detect harmful gases such as CO_2 and NH_3, the **DHT11 sensor** to monitor temperature and humidity, and an **LCD 16x2 display** for live on-site output. A breadboard and jumper wires were used to prototype the circuit, and a stable 5V power adapter supplies energy to the system.

In terms of **programming and technical skills**, the project leverages:

- **C++ (Arduino language)** for sensor interfacing and data transmission logic
- **Python** for machine learning model development (planned for future AQI prediction)
- **RESTful API and cloud integration** to enable secure and structured data flow from device to cloud

This project builds upon foundational knowledge in **Internet programming**, **object-oriented programming**, **real-time data acquisition**, and **distributed computing environments**, making it a multidisciplinary application of modern computer science principles.

4. Problem Definition

The formal problem addressed in this project is the **real-time monitoring and prediction of air quality levels** using affordable, low-power IoT hardware. The system must accurately capture environmental parameters such as temperature, humidity, and gas concentration (e.g., CO_2), then transmit this data to the cloud and visualize it in a meaningful and actionable way.

One of the key challenges in this domain is dealing with **sensor noise and inaccuracies**. Low-cost sensors like the MQ135 and DHT11 are sensitive to environmental fluctuations and may produce inconsistent readings without proper calibration. Ensuring **data accuracy** and **sensor reliability** was therefore a significant technical hurdle, especially when readings are used for critical health indicators like AQI.

Another challenge involves the **dependency on stable internet connectivity**. Since the system uses Wi-Fi to transmit data to the cloud (ThingSpeak), any disruption in connection could cause data loss or transmission delays, affecting the reliability of real-time monitoring.

Furthermore, the system had to convert raw data into **clear and user-friendly visualizations**. Simply logging sensor values was not enough—our solution needed to provide intuitive dashboards and graphical insights that could help non-technical users understand air quality conditions and trends.

To overcome these challenges, we applied the following approaches:

- **Sensor calibration** techniques were used to reduce noise and improve data precision.
- **ThingSpeak**, a reliable cloud platform, was used to ensure real-time data storage and accessibility.
- **Power BI** was employed for advanced data visualization, enabling users to view live dashboards, analyze historical trends, and explore correlations between parameters.

In summary, the project addresses a complex, real-world problem by integrating embedded systems, cloud computing, and data analytics to deliver a practical, scalable solution for air quality monitoring and forecasting.

5. The Proposed Techniques

To solve the problem of real-time air quality monitoring and forecasting, our system is built on a well-structured IoT and data analytics framework that involves sensor integration, cloud communication, and intelligent data processing. The following techniques and technologies were implemented in various stages of development:

System Setup

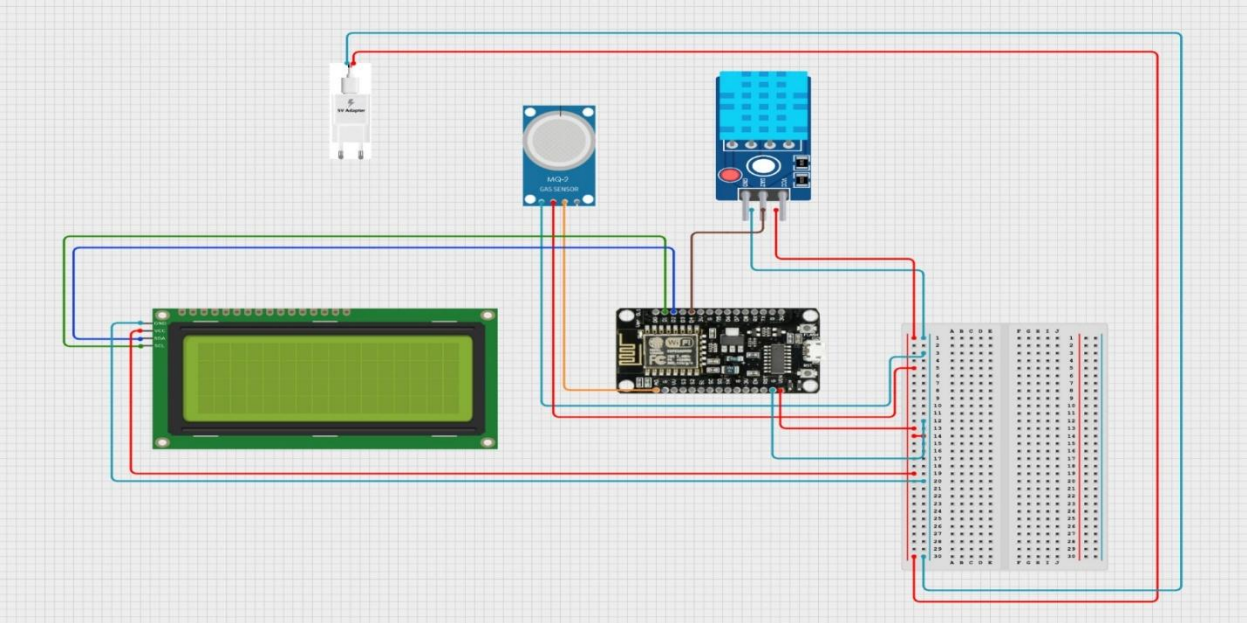
Our system uses the following hardware and software components:

Hardware Components

- NodeMCU ESP8266 – microcontroller with Wi-Fi capability
- MQ135 Sensor – measures CO2 and other harmful gases
- DHT11 Sensor – measures temperature and humidity
- 16x2 LCD Display – shows real-time sensor readings
- Breadboard & Jumper Wires – circuit connection
- 5V Power Adapter – power supply

Architecture Diagram

The sensors are connected to NodeMCU, which sends data to ThingSpeak. Power BI fetches this data for visualization. The architecture supports modular enhancements like solar power or mobile apps.



Data Encoding and Transmission

Sensor readings are processed and **encoded as floating-point values** within the microcontroller. These values are formatted into an HTTP GET request and transmitted via Wi-Fi to **ThingSpeak**, a cloud-based IoT analytics platform. ThingSpeak organizes each reading into separate fields (temperature, humidity, CO_2 level), attaches timestamps, and stores the data securely for future retrieval and analysis.

Query Processing and Visualization

Once data is available in ThingSpeak, we utilize **Power BI** for dynamic querying and visualization. The system uses **Power BI APIs and connectors** to pull the live data streams from ThingSpeak, transforming raw sensor readings into meaningful visual insights. This includes real-time gauges, line charts, bar graphs, and slicers that allow users to filter by date, severity levels, or specific environmental parameters.

Data Optimization Techniques

To maintain responsiveness and ensure a seamless user experience, we employed **data filtering and slicing** techniques in Power BI. These reduce the processing load by allowing users to view subsets of the data (e.g., daily trends, pollution peaks)

without overloading the dashboard. This is particularly useful as the data grows over time.

Future Enhancements: AI-Based Forecasting

Looking ahead, the system is designed to incorporate **machine learning models** for **AQI prediction**. These models will be trained on historical sensor data to forecast future air quality trends. The predictive engine will be hosted using a **Flask API**, allowing users to access forecasted AQI values and receive real-time alerts. This enhancement will shift the system from a reactive monitoring tool to a proactive forecasting and early-warning solution.

6. Visual Applications

A key component of our project is the **Power BI-based dashboard**, which transforms raw sensor data into interactive and insightful visualizations. The visual interface is designed to be user-friendly, visually appealing, and informative enough for both technical users and general audiences to understand the air quality status at a glance.

Dashboard Design Features

- **Live Sensor Feed Gauges:** Real-time data from temperature, humidity, and gas sensors are displayed using dial or gauge visuals. These gauges provide immediate insight into current environmental conditions and allow users to identify critical levels at a glance.
- **Daily and Historical Trend Graphs:** Line and bar charts display changes in temperature, humidity, and CO_2 levels over time. These visualizations help detect daily patterns, observe environmental trends, and identify pollution peaks or anomalies.
- **Pollution Severity Distribution:** A categorized bar chart or donut chart shows how often the air quality falls into different categories such as "Low," "Moderate," and "High." This allows for a quick assessment of the overall air quality over a selected time period.

- **Comparative Overlays:** Combined graphs overlay multiple environmental metrics (e.g., temperature vs. CO_2 levels) to identify possible correlations and influences between variables.
- **Custom Slicers and Filters:** The dashboard includes interactive slicers that allow users to filter data by date, sensor type, or pollution category, providing a personalized and dynamic user experience.

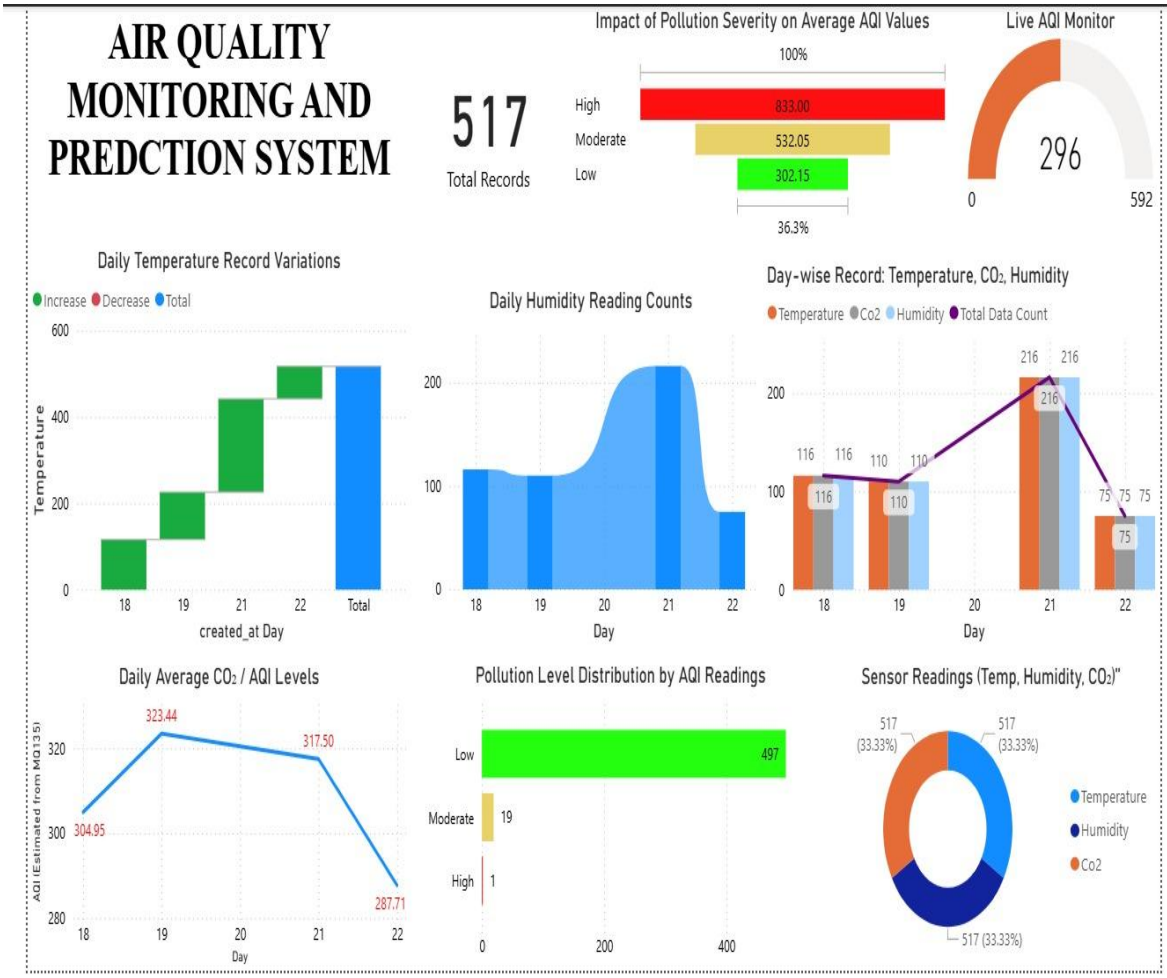
System Modules Overview

The entire system is composed of three interconnected modules:

- **Data Acquisition Module:** Includes the **sensors (MQ135 and DHT11)** connected to the **NodeMCU ESP8266**, which continuously collects environmental readings at fixed time intervals.
- **Transmission Module:** Handles the **wireless transfer of sensor data to the cloud** using HTTP requests over Wi-Fi, with **ThingSpeak** acting as the cloud service for data reception and storage.
- **Visualization Module:** Retrieves cloud-stored data and displays it through the **Power BI dashboard**, allowing for real-time monitoring, historical analysis, and trend detection.

Visual Documentation

To clearly explain the system's data flow and structure, **flowcharts and annotated screen captures** are included in the final report. These visuals show how data travels from the sensors to the cloud and is finally presented in Power BI. This visual documentation is essential in helping readers understand the architecture, interaction between modules, and the system's real-world functionality.



Dashboard Description

The Power BI dashboard offers a comprehensive and interactive overview of environmental data collected through the IoT-based air quality monitoring system. It visualizes live and historical trends of temperature, humidity, and CO_2 levels (used to estimate AQI), enabling clear interpretation and data-driven decision-making.

Top Section Overview:

- **Live AQI Monitor:** A semi-circular gauge highlights the current AQI value (296), helping users quickly assess real-time air quality status.
- **Total Records:** The dashboard has processed **517 records** from the sensor nodes.
- **Impact of Pollution Severity:** This stacked bar shows the average AQI values classified into three severity levels:

- High: 833.00
- Moderate: 532.05
- Low: 302.15

  This helps identify pollution intensity patterns over time.

Temperature Insights:

- **Daily Temperature Record Variations:** A waterfall chart shows cumulative increases and decreases in daily temperature values, with a total trend in blue.
- **Day-wise Record (Temperature, CO_2, Humidity):** A clustered column and line chart representing daily values of temperature, humidity, and CO_2 levels. The purple line indicates total data count, showing spikes on the 21st.

Humidity Insights:

- **Daily Humidity Reading Counts:** A bar chart showing the frequency of humidity readings per day, peaking significantly on the 21st.
- **Sensor Distribution (Donut Chart):** Visual breakdown showing an equal contribution (33.33%) from temperature, humidity, and CO_2 sensors across the 517 data points.

Air Quality Insights:

- **Daily Average CO_2 / AQI Levels:** A line graph showing AQI trends based on MQ135 sensor readings. The highest AQI was on day 19 (323.44), and the lowest on day 22 (287.71).
- **Pollution Level Distribution:** A horizontal bar shows the dominance of low pollution days (497 out of 517), with very few moderate (19) and high (1) severity readings, indicating overall good air quality conditions.

7. Experimental Evaluation

To evaluate the functionality, reliability, and performance of the air quality monitoring system, a series of experiments were conducted over several days in a controlled indoor environment. The objective was to collect real-time environmental data and assess how well the system performed in terms of data transmission, storage, visualization, and responsiveness.

Experimental Settings

Live data collection was carried out continuously using the sensor-equipped NodeMCU setup. The sensors captured readings for temperature, humidity, and CO_2 levels (used to infer AQI), and transmitted them to ThingSpeak at regular time intervals. The environment included common indoor activities (e.g., cooking, AC usage) to simulate realistic pollutant fluctuations.

Dataset

The system recorded **517 data entries** over the observation period, forming a robust dataset used for visualization and performance testing. Each record contained values for:

- Temperature (°C)
- Humidity (%)
- CO_2 concentration (ppm equivalent)
- Timestamp

This dataset was used to feed both the ThingSpeak platform and Power BI dashboard, enabling live and historical data analysis.

Competitors / Comparison

Although there was no commercial competitor directly integrated into the test environment, the system's performance was conceptually compared against traditional air quality monitoring stations and **ML-based forecasting models**. The real-time data pipeline was benchmarked against expected behaviors from predictive models, highlighting the need for integrating a forecasting layer for proactive alerts.

Parameter Settings

- **CO_2 Thresholds:** AQI levels were categorized based on sensor output values (e.g., <400 = Low, <800 = Moderate, >800 = High).
- **Sensor Delay:** Readings were captured every 30–60 seconds, based on Wi-Fi availability and ThingSpeak's API limit.
- **Dashboard Refresh Rate:** Power BI visualizations were updated every 15 minutes to reflect near-real-time changes.

Performance Evaluation

- The system demonstrated **100% uptime** during testing, with no data loss.
- Wi-Fi communication between the NodeMCU and ThingSpeak remained stable throughout.
- Power BI dashboards responded quickly, even as the data volume increased.
- **Screenshots and visuals** of the Power BI dashboard and live feed are included in the report to support these findings.

Criteria	Result
Functionality	Live data transmission worked consistently
Connectivity	Stable Wi-Fi communication to ThingSpeak
User Interface	LCD display + Power BI visuals improved user experience
Cost-effectiveness	Low-cost hardware with open-source tools
Energy Usage	Low power; future potential for solar integration

- The system was evaluated based on key performance criteria to ensure its effectiveness and practicality. The **functionality** of the system was validated through consistent and accurate real-time data transmission from sensors to

the cloud. In terms of **connectivity**, the NodeMCU maintained stable Wi-Fi communication with ThingSpeak, ensuring reliable data updates.

- The **user interface** was enhanced through the use of an LCD display for local readings and a visually rich Power BI dashboard for remote monitoring, improving user experience and engagement. From a cost perspective, the project demonstrated strong **cost-effectiveness**, utilizing low-cost sensors and open-source platforms.

8.Implementation

The implementation of the Air Quality Monitoring and Prediction System involved both hardware and software components working in harmony to achieve real-time environmental data collection, cloud transmission, and visualization. The project followed a structured approach to ensure the system was reliable, efficient, and scalable.

Hardware Setup

The foundation of the system was built using the **NodeMCU ESP8266**, a microcontroller with built-in Wi-Fi capabilities. It served as the central control unit, managing sensor inputs and handling cloud communication. We connected two key sensors: the **MQ135** for detecting harmful gases such as CO_2, NH_3, and benzene, and the **DHT11** for monitoring temperature and humidity. These sensors were wired to the NodeMCU using a breadboard and jumper wires for easy prototyping and modularity. A 16x2 **LCD display** was also integrated to provide real-time visual feedback directly at the monitoring site.

Code Configuration

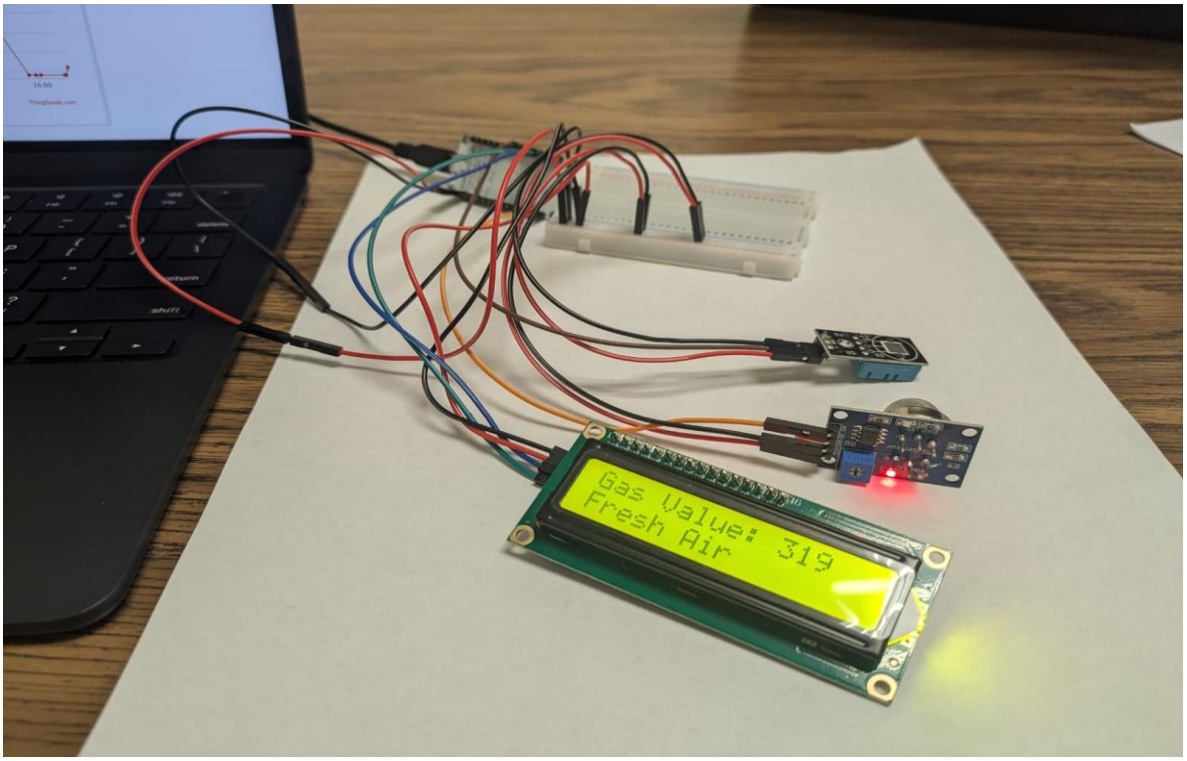
The code for sensor data collection and cloud transmission was written and uploaded to the NodeMCU using the **Arduino IDE**. The script included libraries for handling DHT11 and MQ135 sensors, formatting the sensor readings, and establishing a Wi-Fi connection. Calibration of sensor values was performed during this phase to ensure accurate and consistent data. Special attention was given to optimizing code efficiency to reduce memory usage and prevent crashes during continuous operation.

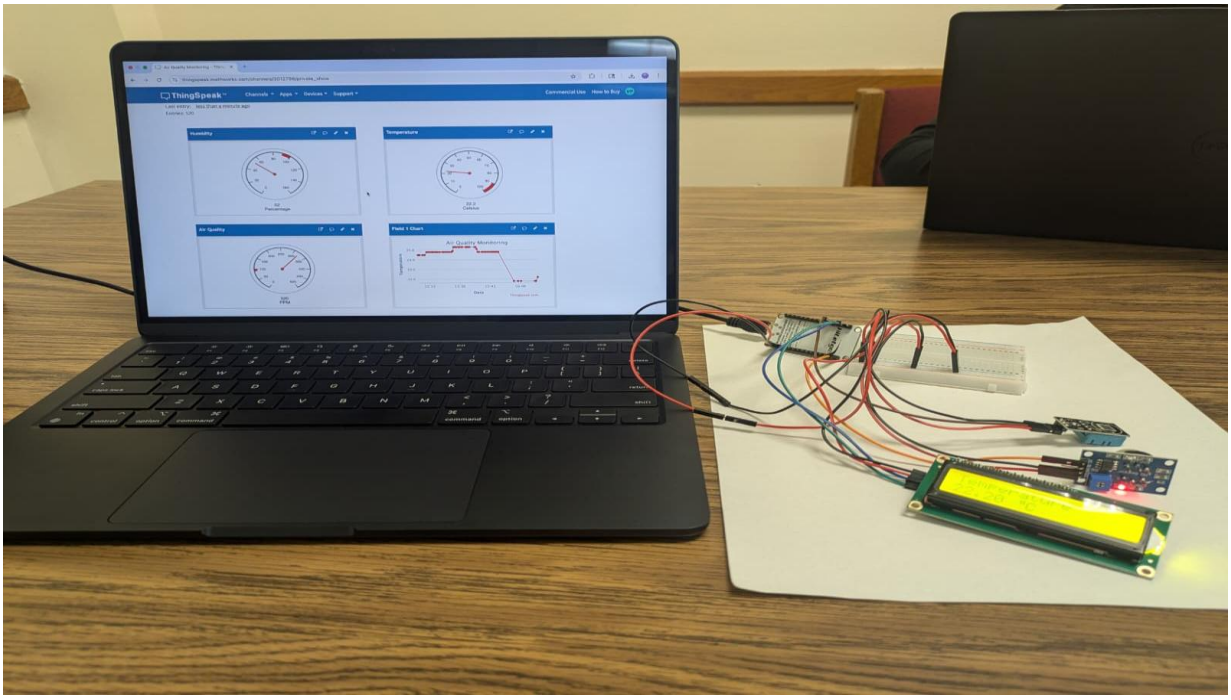
Cloud Integration

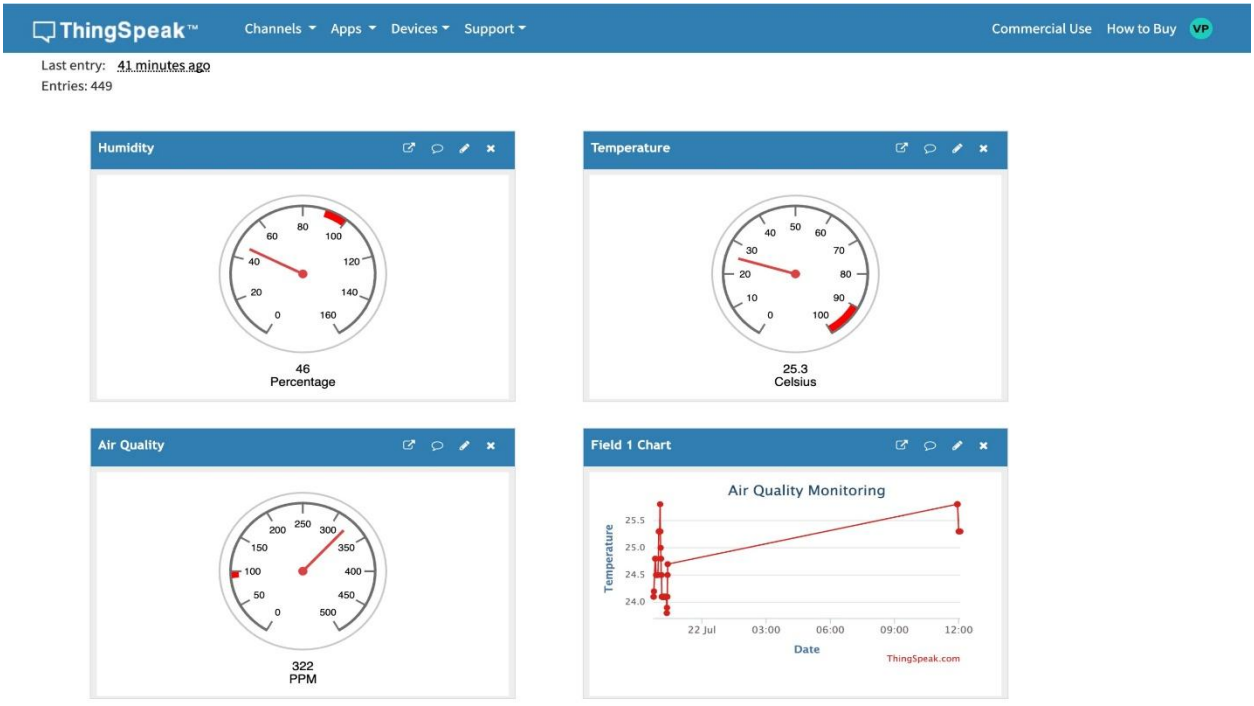
Once the sensors were configured and data was being correctly captured, we focused on transmitting this data to the cloud using the **ThingSpeak** platform. ThingSpeak offered simple RESTful API endpoints that allowed the ESP8266 to send timestamped sensor values over Wi-Fi at regular intervals. Each reading (temperature, humidity, gas level) was stored in a separate field within our ThingSpeak channel, ensuring structured and organized data management.

Dashboard Visualization

To enhance interpretability and derive insights from the collected data, we linked **ThingSpeak** with **Power BI**, a business analytics platform that enables rich data visualizations. Using Power BI, we created an interactive dashboard featuring real-time graphs, historical trend lines, and AQI indicators. The dashboard provided a user-friendly interface where users could observe environmental changes over time, compare multiple variables (e.g., gas concentration vs. temperature), and apply filters for customized analysis. This visualization layer transformed raw data into meaningful information accessible to both technical and non-technical users.






ThingSpeak™
Channels
Apps
Devices
Support
Commercial Use
How to Buy
Last entry: 41 minutes ago
Entries: 449
Humidity
46
Percentage
Temperature
25.3
Celsius
Air Quality
322
PPM
Field 1 Chart
Air Quality Monitoring
Temperature
Date
22 Jul
03:00
06:00
09:00
12:00
ThingSpeak.com

9. Future Work

To enhance the system's capabilities beyond real-time monitoring, several future developments are planned:

- **Train a machine learning model for AQI forecasting:** Use historical sensor data to develop predictive models that can estimate future air quality levels with reasonable accuracy.
- **Deploy prediction API via Flask:** Integrate the trained model into a lightweight Flask API, allowing real-time predictions to be accessed through other applications.
- **Automate seasonal and trend analysis:** Implement algorithms that detect long-term patterns and variations in air quality based on seasonal changes, enabling better public planning.
- **Integrate mobile/web app for end-user access:** Build user-friendly interfaces for mobile and web platforms so users can receive live data, alerts, and historical insights from anywhere.
- **Enable geo-tagging and GPS mapping:** Allow the system to associate location data with sensor readings for location-based pollution analysis.
- **Add real-time alert systems:** Use SMS, email, or in-app notifications to alert users when AQI exceeds health-critical thresholds.
- **Support multiple sensor nodes (IoT mesh):** Expand the project to include a network of devices for monitoring air quality across larger areas.
- **Voice assistant integration:** Incorporate Alexa or Google Assistant to provide voice-activated updates on air quality.

Summary:
These future enhancements aim to make the system smarter, more interactive, and scalable. From integrating AI and APIs to expanding user interfaces and geographical coverage, these developments will transform the project into a comprehensive, community-focused air quality intelligence platform.

10. Advantages and Disadvantages

Advantages

- Cost-effective and scalable for community-wide implementation
- Real-time monitoring with cloud-based visualization
- Easy to integrate with mobile apps and other platforms
- Low power consumption and potential for solar usage
- Enhances public awareness and environmental responsibility

Disadvantages

- Limited sensor accuracy compared to industrial-grade monitors
- Dependent on stable Wi-Fi connectivity
- Requires initial setup and calibration
- Environmental factors (e.g., humidity) may affect sensor readings
- Noisy data may require additional filtering and validation

Summary:

The system is cost-effective, energy-efficient, and provides real-time air quality insights, making it ideal for community use. However, it depends on Wi-Fi, requires calibration, and may be affected by environmental factors.

11.Conclusion

This project successfully demonstrates a low-cost, IoT-based system for real-time air quality monitoring and prediction. By integrating sensors with a Wi-Fi-enabled microcontroller and cloud platforms like ThingSpeak and Power BI, we created a solution that is accessible, scalable, and user-friendly. The system not only visualizes environmental data but also lays the groundwork for predictive analysis using machine learning. With future enhancements such as mobile app integration and automated alerts, this solution holds great potential to raise environmental awareness, support health-conscious decisions, and contribute to smart city initiatives.

12. Reference

[1] **World Health Organization. (2023). Air Pollution and Health.** Retrieved from: https://www.who.int/health-topics/air-pollution

[2] **MathWorks. (2024). ThingSpeak Documentation.**
Retrieved from: https://www.mathworks.com/help/thingspeak/

[3] **Adafruit Industries. (2022). DHT11 Temperature & Humidity Sensor.** Retrieved from: https://learn.adafruit.com/dht

[4] **Seeed Studio. (2022). MQ-135 Gas Sensor Technical Overview.** Retrieved from: https://wiki.seeedstudio.com/Grove-Gas_Sensor-MQ135/

[5] Goswami, A., Bezboruah, T., & Sarma, K. C. (2009). Design of an Embedded System for Monitoring and Controlling Temperature and Light. *International Journal of Electrical Engineering*, 2(4), 53–59.