IOELAB

## Mini Project Report On

**“Air Quality Monitoring System”**

## Submitted

#### by

Atharva Pradhan

Nishant Mane

Harsh Daphle

Under the guidance of

## Prof. Deepali Narkhede

****

**Department of Information Technology**

Shivajirao S. Jondhale College of Engineering

Dombivli (E)

## (Affiliated to University of Mumbai)

## (2025-26)

### Samarth Samaj’s

**SHIVAJIRAO S. JONDHALE COLLEGE OF ENGINEERING DOMBIVLI(E)**

**(Affiliated to University of Mumbai)**

**CERTIFICATE**

*This is to certify that the mini project report entitled “****Air Quality Monitoring System****” is submitted in final year sem – VII during the academic year 2025-26*

Atharva Pradhan

Nishant Mane

Harsh Daphle

|  |  |  |
| --- | --- | --- |
| Prof. Deepali Narkhede |  |  |
| **Subject Incharge** |  | **External Examiner** |
|  |  |  |
|  |  |  |
|  | Dr. Savita Sangam |  |
|  | **Head of Department** |  |

**Table of contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page No** |
|  | List of Figures | 1 |
|  | Abstract | 2 |
| **1** | Introduction | 3 |
| **2** | Literature Survey | 7 |
| **3** | Hardware and Software Requirement | 10 |
| **4** | Working of Project | 15 |
| **5** | Coding of Project | 18 |
| **6** | Applications | 25 |
| **7** | Conclusion | 27 |

**List of figures**

|  |  |  |
| --- | --- | --- |
| Sr no | List of figures | Page no |
| 1 | Fig 3.1 ESP32 Microcontroller | 11 |
| 2 | Fig 3.2 DHT11 Sensor | 11 |
| 3 | Fig 3.3 MQ135 Sensor | 12 |
| 4 | Fig 3.4 Breadboard | 13 |
| 5 | Fig 3.5 Jumper wires | 13 |
| 6 | Fig 4.2 Circuit Diagram | 17 |
| 7 | Fig 4.3 Project model | 17 |
| 8 | Fig 5.2.1 cloud dashboard Screenshots | 23 |
| 9 | Fig 5.2.2 cloud dashboard screenshots | 23 |
| 10 | Fig 5.3 Alert Notifications | 24 |

**Abstract**

This project presents the design and implementation of an IoE-based Air Quality Monitoring System using the ESP32 microcontroller for real-time environmental sensing and data transmission. The system integrates multiple sensors, including MQ135 for detecting harmful gases (such as CO₂, NH₃, and benzene), a PM2.5 sensor for measuring particulate matter, and a DHT11 sensor for temperature and humidity monitoring. The ESP32 collects sensor data and transmits it wirelessly to a cloud platform like Blynk cloud via Wi-Fi, enabling continuous remote monitoring and data visualization through a web or mobile interface. The system supports real-time alerts when pollution levels exceed safe thresholds and provides historical data for trend analysis. With its low power consumption, cloud integration, and IoE capabilities, this solution offers a scalable and cost-effective approach for improving public awareness, supporting health-related decision-making, and contributing to smart city initiatives.

**Chapter 1**

**Introduction**

* 1. **Air Quality Monitoring System**
  2. **Internet of Everything**

**Chapter 1, Introduction**

* 1. **Air Quality Monitoring System**

Air pollution has become a critical global issue, posing serious risks to human health and the environment. With increasing urbanization and industrial activities, the need for continuous air quality monitoring has become more important than ever. Traditional air monitoring systems are often expensive, complex, and limited in accessibility. To address these challenges, this project proposes an IoE (Internet of Everything)-based Air Quality Monitoring System that provides a low-cost, real-time, and remotely accessible solution using modern embedded technologies.

The system is built using the ESP32 microcontroller, which offers integrated Wi-Fi and Bluetooth capabilities, making it ideal for IoT applications. The air quality is monitored using an MQ gas sensor (such as MQ135), which detects harmful gases like carbon dioxide (CO₂), ammonia (NH₃), and benzene. Additionally, a DHT11 sensor is used to measure temperature and humidity, as these environmental factors also influence air quality and human comfort. All collected data is sent to the Blynk IoT Cloud platform through the ESP32’s Wi-Fi connection, allowing users to access real-time readings through a web or mobile dashboard.

This system not only alerts users when pollution levels exceed safe thresholds but also stores historical data for analysis and visualization. It is designed to be compact, energy-efficient, and scalable, making it suitable for use in homes, schools, offices, or public spaces. By leveraging IoE principles and cloud connectivity, this project demonstrates a practical approach to enhancing environmental awareness and supporting data-driven decisions for health and safety.

**Benefits of Air Quality Monitoring System**

* **Real-Timemonitoring**  
  The system provides continuous, real-time updates on air quality, enabling timely responses to pollution spikes or hazardous conditions.
* **PublicHealthProtection**  
  By detecting harmful gases and particulate matter, the system helps raise awareness

**Chapter 1, Introduction**

of air quality issues, reducing the risk of respiratory diseases and other health problems.

* **RemoteAccessibility**  
  With cloud integration (e.g., Arduino Cloud), users can monitor air quality data from anywhere using a smartphone or web dashboard.
* **Cost-EffectiveSolution**  
  Compared to traditional large-scale monitoring stations, IoE-based systems are affordable, portable, and easy to deploy in multiple locations.
* **DataLoggingandAnalysis**  
  Historical data stored in the cloud allows for trend analysis, pollution source identification, and better environmental planning.
  1. **Internet of Everything**

The Internet of Everything (IoE) is bringing together people, process, data, and things to make networked connections more relevant and valuable than ever before-turning information into actions that create new capabilities, richer experiences, and unprecedented economic opportunity for businesses, individuals, and countries. In simple terms: IoE is the intelligent connection of people, process, data and things. The Internet of Everything (IoE) describes a world where billions of objects have sensors to detect measure and assess their status; all connected over public or private networks using standard and proprietary protocols.

Pillars of The Internet of Everything (IoE)

• **People**: Connecting people in more relevant, valuable ways.

• **Data**: Converting data into intelligence to make better decisions.

• **Process**: Delivering the right information to the right person (or machine) at the right time.

• **Things**: Physical devices and objects connected to the Internet and each other for intelligent decision making; often called Internet of Things (IoT).

**Chapter 1, Introduction**

**The Internet of Things (IoT)**

The Internet of Things (IoT) is the network of physical objects accessed through the Internet.These objects contain embedded technology to interact with internal states or the

external environment. In other words, when objects can sense and communicate, it changes how and where decisions are made, and who makes them. For example, Nest thermostats.

**The difference between IoE and IoT**

The Internet of Everything (IoE) with four pillars: people, process, data, and things builds on top of The Internet of Things (IoT) with one pillar: things. In addition, IoE further advances the power of the Internet to improve business and industry outcomes, and ultimately make people’s lives better by adding to the progress of IoT. ( Dave Evans, Chief Futurist Cisco Consulting Services).

**Chapter 2**

**Literature Survey**

**2.1 Existing System**

**2.2 Problem Definition**

**Chapter 2, Literature Survey**

**2.1 Existing System**

Ritik Gupta and his co-authors designed a IOT based air pollution monitoring system which consists The level of pollution has increased with times by lot of factors like the increase in population, increased vehicle use, industrialization and urbanization which results in harmful effects on human wellbeing by directly affecting health of population exposed to it. In order to monitor In this project we are going to make an IOT Based Air Pollution Monitoring System in which we will monitor the Air Quality over a web server using internet and will trigger a alarm when the air quality goes down beyond a certain level, means when there are sufficient amount of harmful gases are present in the air like CO2, smoke, alcohol, benzene and NH3. It will show the air quality in PPM on the LCD and as well as on webpage so that we can monitor it very easily. In this IOT project, you can monitor the pollution level from anywhere using your computer or mobile.

Ch.V.Saikumar and his co-authors designed a IOT based air quality monitoring system where The main objective of this project is to monitor the air eminence in industrial and urban areas. The proposed outline includes a set of gas sensors (CO, and NO2) that are positioned on masses and structure of a IOT (Internet of things) and a dominant server to support both short-range realtime incident management and a continuing deliberate planning. In this Arduino platform is used to communicate the data simply and quickly. WSN (Wireless sensor network) acts as the trans receiver. This provide a real-time low rate monitoring system over the use of low rate, low information rate, and little control wireless communication technology. The projected monitoring system can be transferred to or shared by different applications. Through IOT we can able to visualize the values from the globe.

Lokesh kothari and his co-authors designed IOT based weather monitoring system This Paper makes use of 3 sensors to measure the weather/environment factors such as temperature, humidity, light intensity, dew point and heat index. The values read from the sensors are processed by the Arduino micro-controller and stored in a text file which can be processed upon to derive analysis. The readings are also displayed on an on board LCD for quick viewing. All these readings can be analyzed to get the weather characteristics of a particular area and record the weather pattern. These recorded parameters are essential and vary from places to places.

**Chapter 2, Literature Survey**

* 1. **Problem Definition**

Air pollution poses a serious threat to public health, the environment, and overall quality of life. Traditional air quality monitoring stations are expensive, bulky, and often limited in number, making it difficult to provide real-time, localized data for communities or individuals. As a result, people remain unaware of the air they breathe and are unable to take timely precautions against harmful pollutants like PM2.5, CO₂, CO, NO₂, and other gases.

We will going to design air quality monitoring system which will To develop a low-cost**,** IoE-based Air Quality Monitoring System using the ESP32 microcontroller and a set ofenvironmental sensors that canContinuously measure pollutantslike PM2.5**,** CO2**,** VOCs**,** temperature**,** and humidity**.** Transmit real-time data via Wi-Fi to a cloud platform for remote monitoring **.**Visualize the data through dashboards or mobile applicationsProvide alerts when air quality drops below healthy thresholds**.**

**Chapter 3**

**Hardware and Software Requirements**

**3.1 Hardware Requirement**

**3.2 Software Requirement**

**Chapter 3, Hardware and Software requirements**

**3.1 Hardware Requirement**

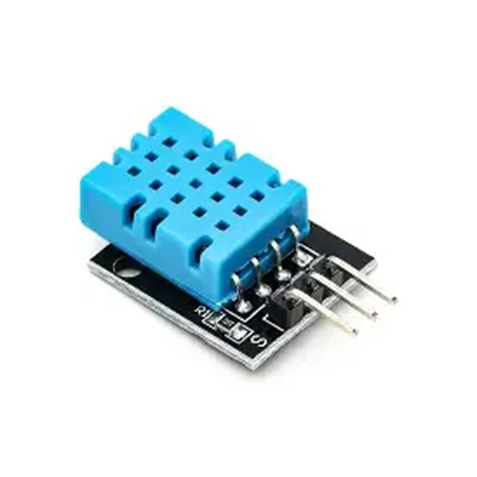
**1. ESP32 Microcontroller**



**Fig 3.1 ESP32 Microcontroller**

The ESP32 is a powerful, low-cost microcontroller developed by Espressif Systems, featuring built-in Wi-Fi and Bluetooth (classic and BLE) capabilities, making it ideal for IoT applications. It is based on a dual-core (or single-core) XtensaLX6/LX7 processor running up to 240 MHz, with 520 KB SRAM, support for external flash/PSRAM, and a wide range of peripherals including ADC, DAC, **PWM**, **SPI**, **I2C**, **UART**, CAN, and capacitive touch sensors. The ESP32 supports multiple power-saving modes, over-the-air (OTA) updates, and has built-in security features like secure boot and flash encryption.

**2. DHT11 Sensor**

****

**Fig 3.2 DHT11 Sensor**

**Chapter 3, Hardware and Software requirements**

The DHT11 is a low-cost digital sensor used for measuring temperature and humidity, widely used in basic weather monitoring and home automation projects. It features a thermistor for temperature sensing and a capacitive humidity sensor, providing readings with a temperature range of 0–50°C (±2°C accuracy) and a humidity range of 20–90% RH (±5% accuracy). The sensor communicates using a single-wire digital interface, making it easy to connect to microcontrollers like Arduino or ESP32. Though slower and less accurate compared to its counterpart DHT22, the DHT11 is popular for simple applications due to its low power consumption, small size, and ease of use.

**3. MQ 135 Sensor**

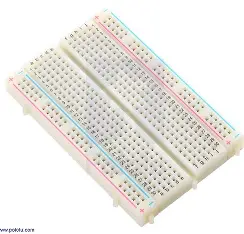
****

**Fig 3.3 MQ135 Sensor**

The MQ-135 is a popular gas sensor used for detecting a wide range of air pollutants, including ammonia (NH₃), nitrogen oxides (NOₓ), alcohol, benzene, smoke, and carbon dioxide (CO₂). It operates by using a sensitive material (SnO₂) whose resistance changes in the presence of gases, allowing it to output an analog signal corresponding to the gas concentration. The sensor requires a preheating time (usually 24–48 hours for accurate calibration) and operates on 5V, with both analog and digital output options. It's widely used in air quality monitoring, indoor pollution detection, and environmental sensing projects, often in combination with microcontrollers like Arduino or ESP32. However, for accurate readings, the MQ-135 must be properly calibrated and used in stable environmental conditions.

**Chapter 3, Hardware and Software Requirements**

**4. Breadboard**

****

**Fig 3.4 Breadboard**

A breadboard is a reusable prototyping tool used to build and test electronic circuits without soldering, making it ideal for beginners, hobbyists, and engineers. It consists of a grid of holes into which electronic components and jumper wires can be inserted, with internal metal strips connecting the holes in rows and columns to form circuits. Breadboards typically include two power rails on each side for supplying voltage and ground, and a central area (called the terminalstrip) for placing components like resistors, LEDs, ICs, and sensors. The layout allows easy modifications, troubleshooting, and expansion of circuits.

**5. Jumper wires**

****

**Fig 3.5 Jumper Wires**

**Chapter 3, Hardware and Software Requirements**

Jumper wires are flexible, insulated wires used to make temporary connections between components on a breadboard, development boards (like Arduino or ESP32), or other electronic circuits. They come in three main types: male-to-male, male-to-female, and female-to-female, depending on the connector pins at each end, making them versatile for various setups. Jumper wires are typically made of stranded or solid core wire, with plastic insulation and standardized pin headers for easy insertion. They come in different colors and lengths to help organize and troubleshoot complex circuits. Jumper wires are essential tools in prototyping, allowing users to quickly modify, test, and debug electronic projects without soldering.

**3.2 Software Requirement**

**1. Blynk Cloud**

Blynk Cloud is a managed Internet of Things (IoT) platform that enables users to connect, manage, and control smart devices over the internet without the need to build or maintain their own server infrastructure. It acts as the backbone of the Blynk ecosystem, offering secure and scalable cloud hosting for IoT devices that communicate through standard protocols such as MQTT and HTTP. With Blynk Cloud, developers and businesses can easily provision devices, manage firmware updates, collect and analyze data, and build interactive dashboards for both mobile and web interfaces. The platform ensures high reliability with a global network of servers, offering 99.995% uptime and low latency across multiple regions. Security is a top priority—communication between devices and the cloud is encrypted using TLS 1.2 or TLS 1.3, and each device is authenticated with a unique token. Blynk Cloud supports features like real-time monitoring, role-based access control, and integration with third-party services. It is suitable for both hobbyists building prototypes and enterprises deploying large-scale IoT solutions, thanks to its flexibility, white-label branding, and private cloud options. Overall, Blynk Cloud simplifies IoT development by providing a complete, secure, and easy-to-use backend for connecting devices and managing data from anywhere.

**Chapter 4**

**Working of project**

**4.1 Conceptual overview**

**4.2 Circuit Diagram**

**4.3 project models**

**Chapter 4, working of project**

**4.1 Conceptual Overview**

An air quality monitoring system using ESP32 as part of an Internet of Everything (IoE) project integrates low-cost, smart sensing with wireless communication to monitor and share real-time air pollution data. This system is built around the ESP32 microcontroller, which is well-suited for IoE applications due to its built-in Wi-Fi and Bluetooth capabilities, low power consumption, and ability to interface with a variety of environmental sensors. The primary goal of this project is to create an affordable, scalable, and connected solution for tracking air quality in different environments—urban areas, industrial zones, schools, or even within homes.

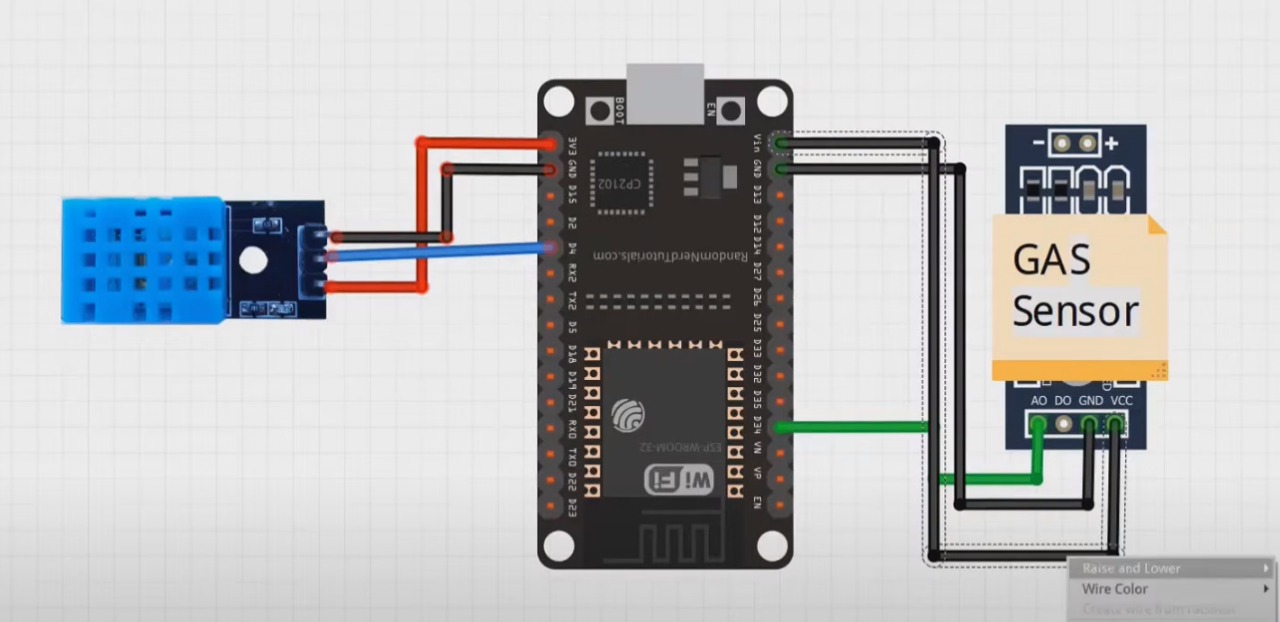
In this setup, various air quality sensors—such as the MQ-series gas sensors (e.g., MQ-135 for detecting CO₂, NH₃, benzene), PM sensors like the PMS5003 or SDS011 (for PM2.5 and PM10), and environmental sensors like DHT11 or BME280 (for temperature and humidity)—are connected to the ESP32. These sensors continuously collect atmospheric data, which is processed locally by the ESP32. The microcontroller then sends this data wirelessly to a cloud server or IoT platform (such as ThingSpeak, Firebase, or Blynk) using Wi-Fi, enabling remote access and real-time monitoring through web or mobile dashboards.

The IoE aspect of the project comes into play by connecting not just sensors and devices, but also users and data services. For instance, the collected data can be analyzed to detect trends, send alerts when pollution levels exceed safe limits, or trigger automated actions (like turning on air purifiers or sending notifications to smartphones). This creates a smart, responsive system where both people and machines can interact with environmental data in real time. Additionally, multiple ESP32-based nodes can be deployed in different locations to create a distributed sensing network, making the system suitable for community-level or city-wide air quality monitoring.

In summary, an ESP32-based air quality monitoring system for an IoE project demonstrates how embedded systems, cloud computing, and wireless communication can be combined to create intelligent environmental monitoring solutions. It offers a cost-effective, flexible, and scalable approach to improving awareness about air pollution and enabling data-driven decisions for health, safety, and sustainability.

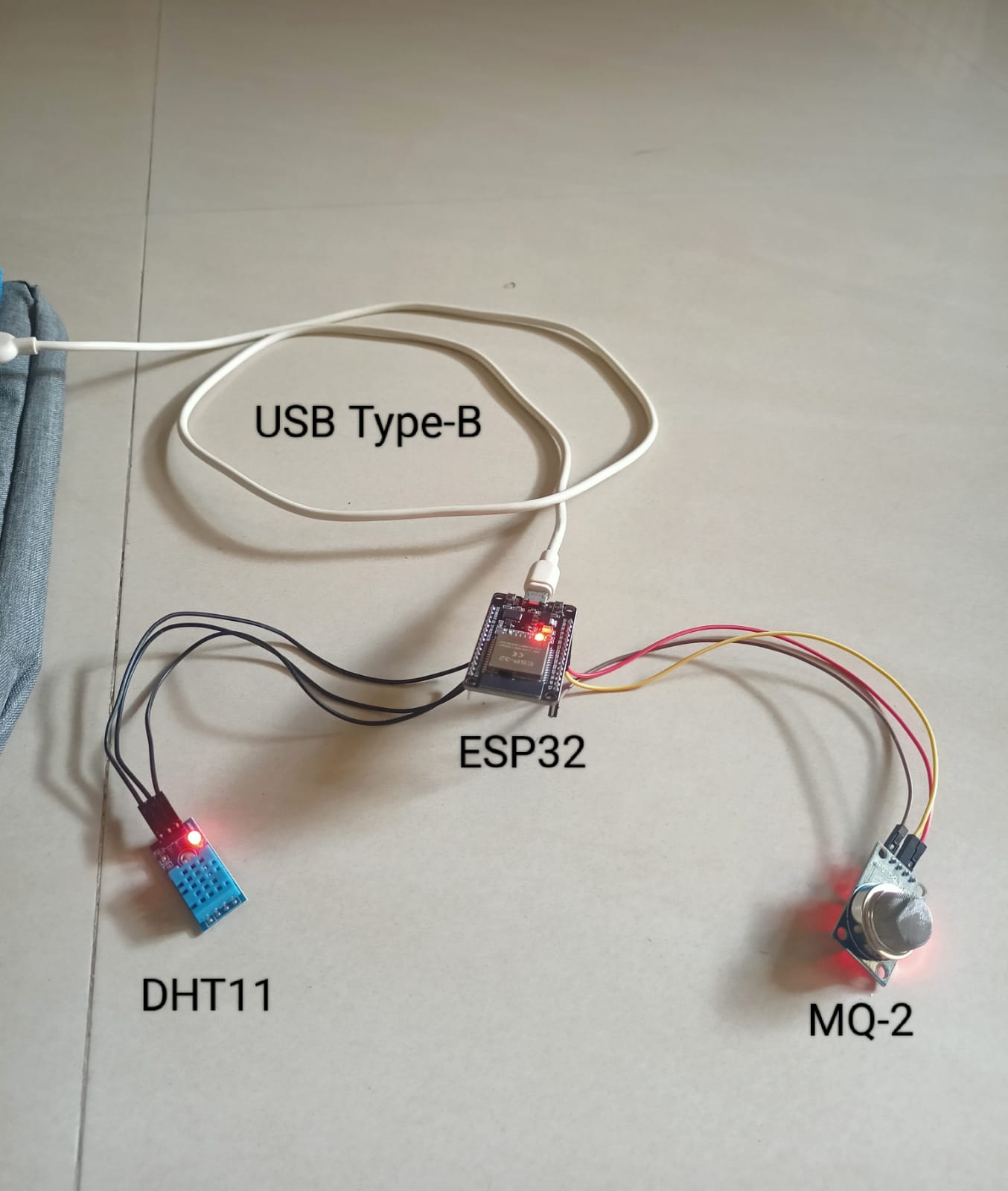
**Chapter 4, Working of project**

**4.2 Circuit Diagram**

****

**Fig 4.2 Circuit diagram for air quality monitoring system**

**4.3 Project model**



**Fig 4.3 project model**

**Chapter 5**

**Coding of project**

**5.1 Programs**

**5.2 Blynk cloud Screenshots**

**5.3 Alert Notifications**

**Chapter 5, Coding of project**

#define BLYNK\_TEMPLATE\_ID "TMPL3KAIESZ2"

#define BLYNK\_TEMPLATE\_NAME "Air Quality Monitor"

#define BLYNK\_AUTH\_TOKEN "Sld0J0eTa8Gi11AUESiDVqsITAXU\_TeF"

// Wi-Fi credentials

const char\* ssid = "Aditya";

const char\* pass = "Anuj@0373";

// Includes

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include "DHT.h"

// -------------------- SENSOR PINS & CONSTANTS --------------------

#define DHTPIN 4

#define DHTTYPE DHT11

#define MQ2\_PIN 34

DHT dht(DHTPIN, DHTTYPE);

BlynkTimer timer;

#define GAS\_ALERT\_THRESHOLD 200

#define EVENT\_CODE "high\_gas\_alert"

**Chapter 5, Coding of project**

// -------------------- FUNCTION TO SEND DATA --------------------

void sendSensorData() {

float h = dht.readHumidity();

float t = dht.readTemperature();

int gasValue = analogRead(MQ2\_PIN);

if (isnan(h) || isnan(t)) {

Serial.println(" DHT11 read failed!");

return;

}

Serial.println(" Sending data to Blynk...");

Serial.print(" Temp: "); Serial.print(t); Serial.println(" °C");

Serial.print(" Humidity: "); Serial.print(h); Serial.println(" %");

Serial.print(" Gas: "); Serial.println(gasValue);

Blynk.virtualWrite(V0, t);

Blynk.virtualWrite(V1, h);

Blynk.virtualWrite(V2, gasValue);

if (gasValue > GAS\_ALERT\_THRESHOLD) {

Blynk.logEvent(EVENT\_CODE);

Serial.println(" HIGH GAS ALERT!");

**Chapter 5, Coding of project**

}

Serial.println("-------------------------------------");

}

void setup() {

Serial.begin(115200);

delay(1000);

Serial.println(" Initializing Air Quality Monitor...");

WiFi.begin(ssid, pass);

Serial.print(" Connecting to Wi-Fi");

int retries = 0;

while (WiFi.status() != WL\_CONNECTED && retries < 30) {

delay(500);

Serial.print(".");

retries++;

}

if (WiFi.status() == WL\_CONNECTED) {

Serial.println("\n Wi-Fi connected!");

Serial.print(" IP Address: ");

Serial.println(WiFi.localIP());

} else {

Serial.println("\n Wi-Fi connection failed!");

**Chapter 5 , Coding of project**

}

Blynk.config(BLYNK\_AUTH\_TOKEN);

if (Blynk.connect()) {

Serial.println(" Connected to Blynk Cloud!");

} else {

Serial.println(" Failed to connect to Blynk Cloud!");

}

dht.begin();

timer.setInterval(5000L, sendSensorData);

}

void loop() {

if (WiFi.status() != WL\_CONNECTED) {

Serial.println(" Wi-Fi lost! Reconnecting...");

WiFi.reconnect();

delay(1000);

}

if (!Blynk.connected()) {

Serial.println(" Blynk disconnected! Trying again...");

Blynk.connect();

}

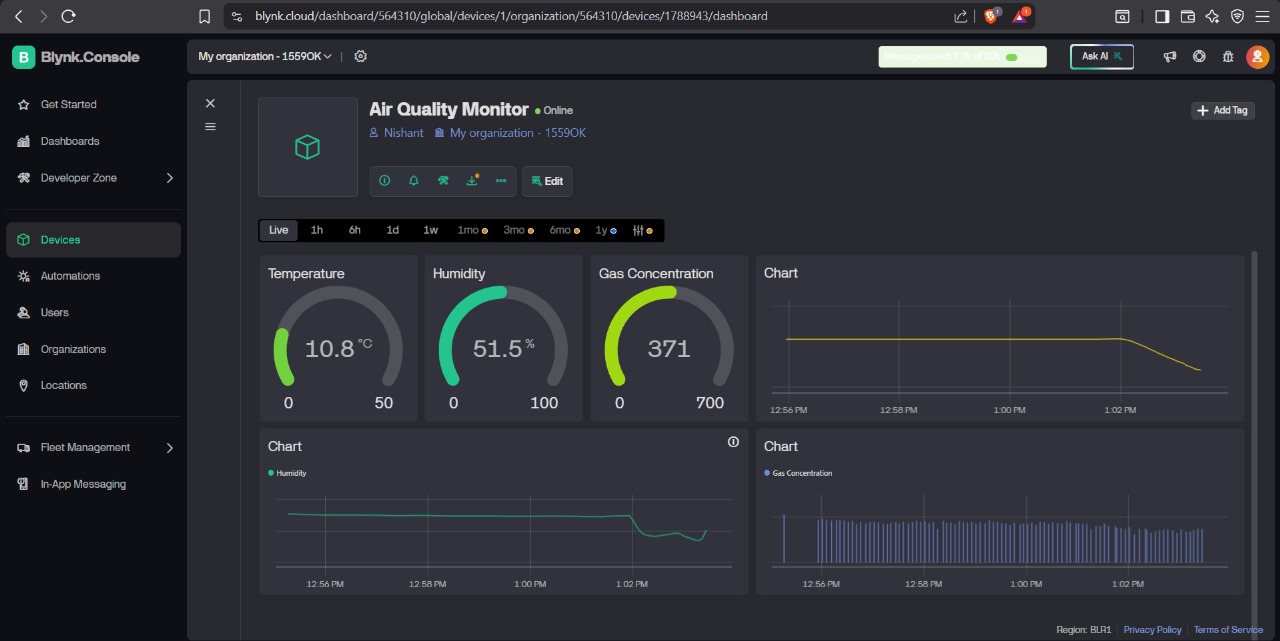
Blynk.run();

  timer.run();

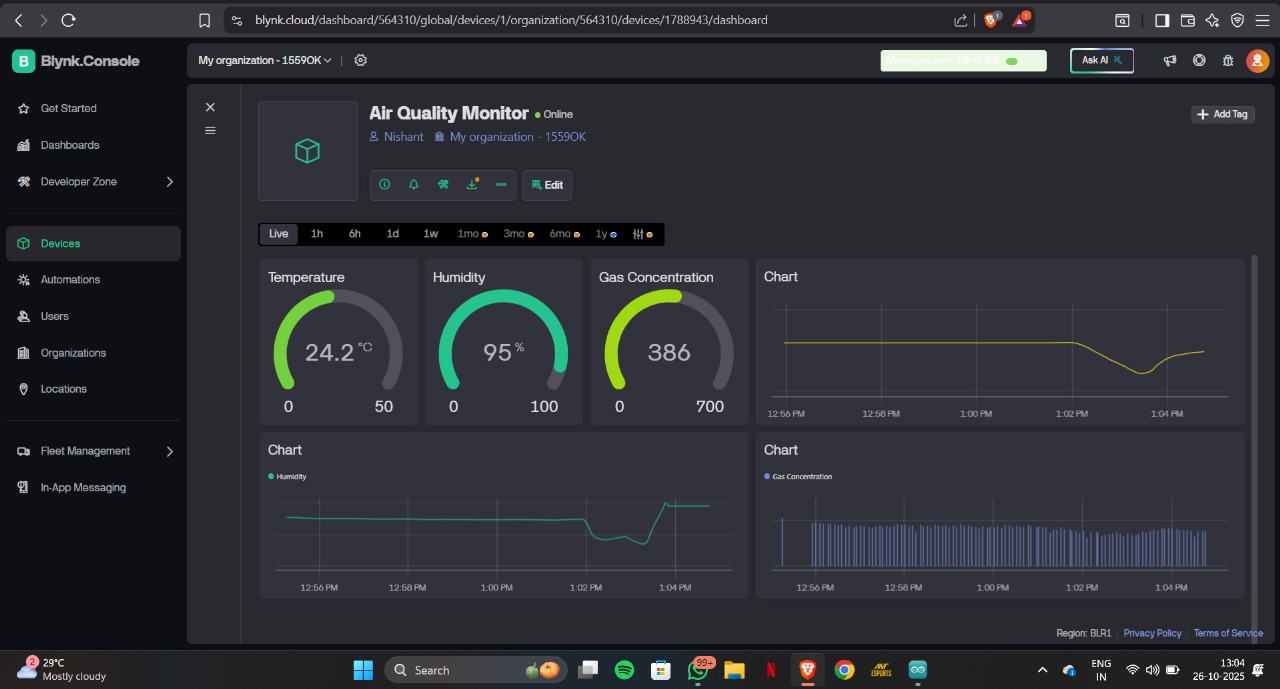
}

**Chapter 5, Coding of project**

**5.2 Blynk Cloud Screenshots**



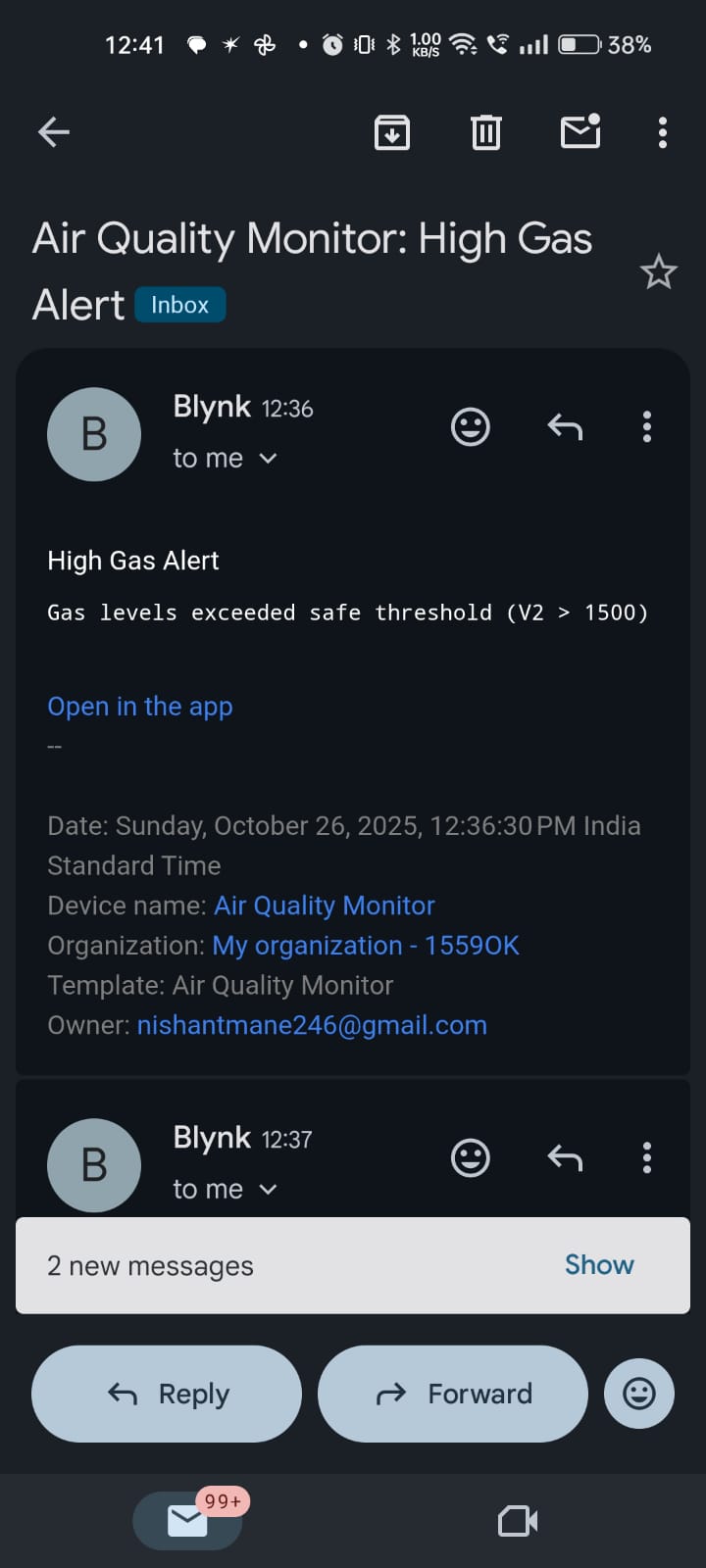
**Fig 5.2.1 Cloud dashboard Screenshots**

****

**Fig 5.2.2 Cloud dashboard Screenshots**

**Chapter 5, Coding of project**

**5.3 Alert Notifications**

****

**Fig 5.3 Alert Notifications**

**Chapter 6**

**Application**

**Chapter 6, Application**

An Air Quality Monitoring System is a smart IoT-based application designed to measure and track pollutants in the air—such as PM2.5, PM10, CO2, CO, and VOCs—and transmit real-time data to the Blynk Cloud for monitoring, logging, and remote access. It helps in understanding air pollution levels, identifying hazardous environments, and promoting better health and environmental awareness

* Indoor Air Quality Monitoring (Homes & Offices)
* Schools and Educational Institutions
* Hospitals and Healthcare Facilities
* Smart Cities
* Industrial Areas
* Traffic & Vehicle Monitoring
* Environmental Research
* Shopping Malls and hotels

The list is still not exhaustive and will evolve over the time to accommodate new IoT use cases.

**Conclusion**

The Air Quality Monitoring System is a practical and essential solution for addressing the growing concerns about air pollution and its impact on health and the environment. By using sensors and IoT-enabled Arduino platforms, this system can detect harmful gases, particulate matter, temperature, and humidity levels in real-time. The collected data is transmitted to the Blynk Cloud, where it can be monitored, analyzed, and used to trigger alerts or actions, such as turning on ventilation systems or notifying users. This enables timely responses to deteriorating air conditions, ensuring safer environments in homes, schools, workplaces, and public spaces.

Overall, the system promotes awareness and proactive management of air quality, contributing to improved public health and environmental sustainability. It also supports larger initiatives like smart cities, disaster preparedness, and climate research by providing valuable data insights. With its scalability, low cost, and remote monitoring capabilities, the Air Quality Monitoring System is a versatile and impactful tool for individuals, communities, and governments aiming to reduce exposure to air pollutants and make informed decisions for a cleaner, healthier future.