**Internet of Things Fundamentals**

*Subject Project*

BS AI 6th Semester SP-25 (AIE-3079)

**Date:** 26-06-2025

**Project Title:** Smart Environment Monitor: IoT Smoke Detector & Air Index Calculator

**Group Name/no.:** AI\_ZONE /10

**Team Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| Members | Registration no | Name | Signature |
| **Member-1 (Leader)** | **22-NTU-CS-1336** | **ABDUL RAFAY** |  |
| **Member-2** | **22-NTU-CS-1355** | **M. FAISAL** |  |
| **Member-3** | **22-NTU-CS-1375** | **SUFHAN SIDDIQUE** |  |
| **Member-4** |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Contributions in % of each Team Members for each component | | | | | |
|  | | Member-1 | Member-2 | Member-3 | Member-4 |
| Distribution Components | | RAFAY | FAISAL | SUFHAN | Name |
| Coding | ESP32-coding | 40% | 30% | 30% |  |
| Python Coding | 30% | 40% | 30% |  |
| UI Design | | 30% | 40% | 30% |  |
| Database | | 30% | 40% | 30% |  |
| Cloud Integration | | 40% | 30% | 30% |  |
| IoT Gateway | | 40% | 30% | 30% |  |
| Edge Processing | | 30% | 40% | 30% |  |
| Documentation | | 30% | 30% | 40% |  |
| Presentation  Design | | 30% | 30% | 40% |  |
| Replace for other contribution | |  |  |  |  |
| Replace for other contribution | |  |  |  |  |
| Replace for other contribution | |  |  |  |  |
| Replace for other contribution | |  |  |  |  |

*To be filled by the evaluator*

# Team-Based Evaluation (60 Marks)

|  |  |  |
| --- | --- | --- |
| Criteria | Obtained Marks | Out of |
| System Design & Architecture |  | 10 |
| Hardware Integration & Circuit Setup |  | 10 |
| IoT Gateway and Cloud Communication |  | 10 |
| Working Prototype Demonstration |  | 10 |
| Performance & Reliability Testing |  | 10 |
| Presentation |  | 10 |
| Total (Team-Based) |  | 60 |

# Individual-Based Evaluation (40 Marks per Member)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Member 1 | Member 2 | Member 3 | Member 4 |
| Criteria |  |  |  |  |
| Understanding of the Project & Role | /10 | /10 | /10 | /10 |
| Code Contribution and Explanation | /10 | /10 | /10 | /10 |
| Q/A VIVA | /10 | /10 | /10 | /10 |
| Documentation/Reporting & Communication | /10 | /10 | /10 | /10 |
| Total (Individual-Based) | /40 | /40 | /40 | /40 |
| Total Overall (60+40) | /100 | /100 | /100 | /100 |
| Weightage Lab Grade (50) |  |  |  |  |

**1. Abstract / Executive Summary**

This project presents a smart IoT-based Smoke and Air Quality Monitoring System using an ESP32 microcontroller. It integrates a smoke sensor (MQ-2) and an air index sensor (MQ-135) to detect harmful gases and pollution in real time. The system alerts users through LED indicators and sends live data to the Blynk Cloud, where it is visualized via a mobile application. This solution promotes environmental monitoring and safety, especially in indoor settings prone to fire or poor air quality.

**2. Table of Contents**

Abstract / Executive Summary

Table of Contents

Introduction

Literature Review

Methodology / System Design

Implementation

Results & Discussion

Testing & Validation

Conclusion & Future Work

References

Links

**3. Introduction**

**Background & Motivation**

Air pollution and fire hazards pose significant health and safety threats. Many indoor spaces lack real-time monitoring tools for smoke and air quality.

**Problem Statement**

There is a lack of affordable, real-time air quality and smoke detection solutions that also provide remote monitoring capabilities.

**Project Goals**

Detect harmful smoke and poor air quality

Alert locally with LEDs

Send live data to the cloud via ESP32

View data on the Blynk mobile app

**4. Literature Review**

**Relevant Concepts:**

ESP32: Wi-Fi enabled microcontroller ideal for IoT

MQ-2 & MQ-135: Sensors used to detect smoke and gas pollutants

**Similar Projects:**

Fire detection systems with Arduino

Indoor AQI monitors with Blynk

IoT-based smart homes using ESP modules

**5. Methodology / System Design**

**5.1 Hardware Components:**

ESP32 microcontroller

MQ-2 smoke sensor

MQ-135 air quality sensor

LEDs (Red/Green)

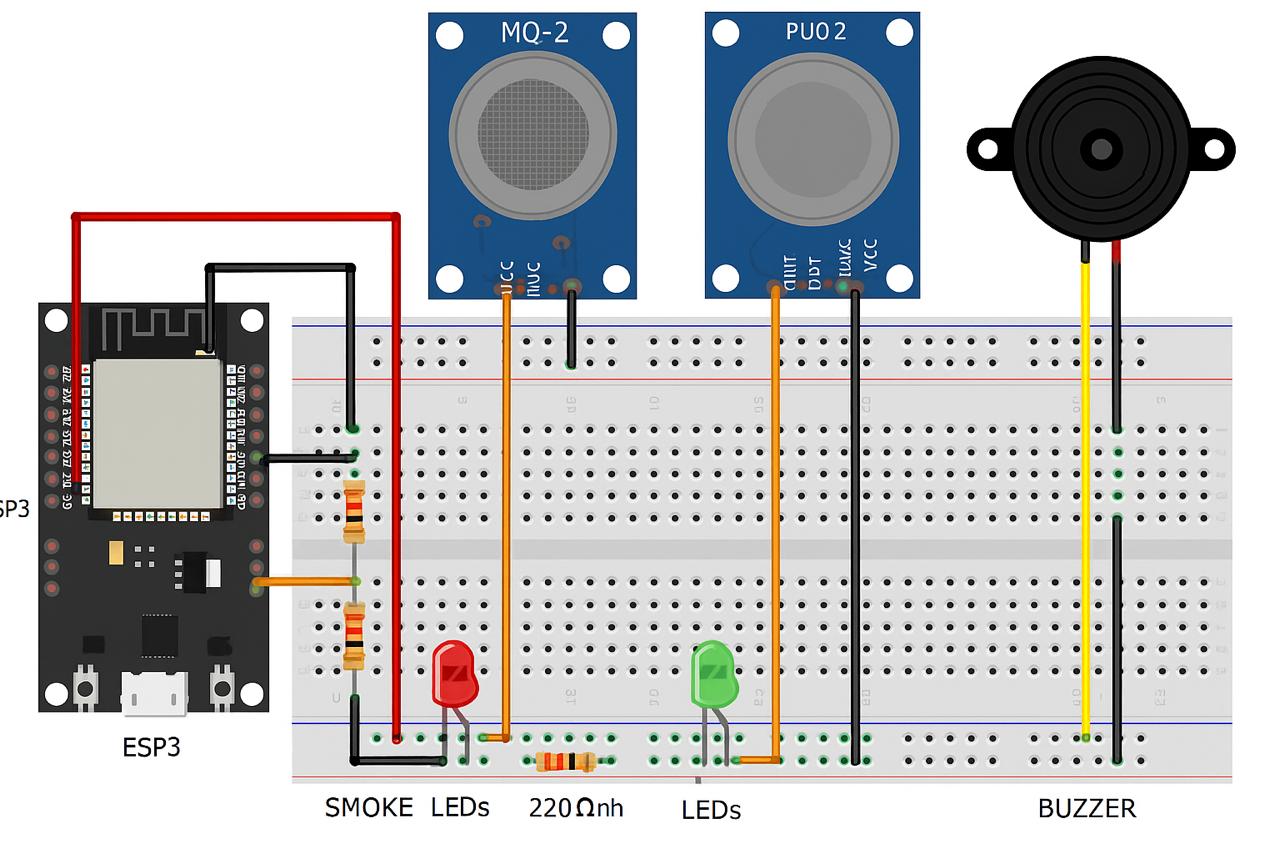
220-ohm resistors

Jumper wires & Breadboard

5V USB supply

buzzer

**Circuit Diagram:**



MQ-2 and MQ-135 connected to ADC pins (A0, A1)

LEDs connected to GPIOs via resistors

**5.2 Software Design:**

**Libraries/Tools:**

MicroPython via Thonny

urequests (for Blynk HTTP API)

Blynk Cloud and Mobile App

**Flowchart/System Logic:**

Connect to Wi-Fi

Read sensor values

Turn ON LED if unsafe

Send data to Blynk

Loop

Pseudocode:

connect\_to\_wifi()

while True:

smoke = read\_mq2()

air = read\_mq135()

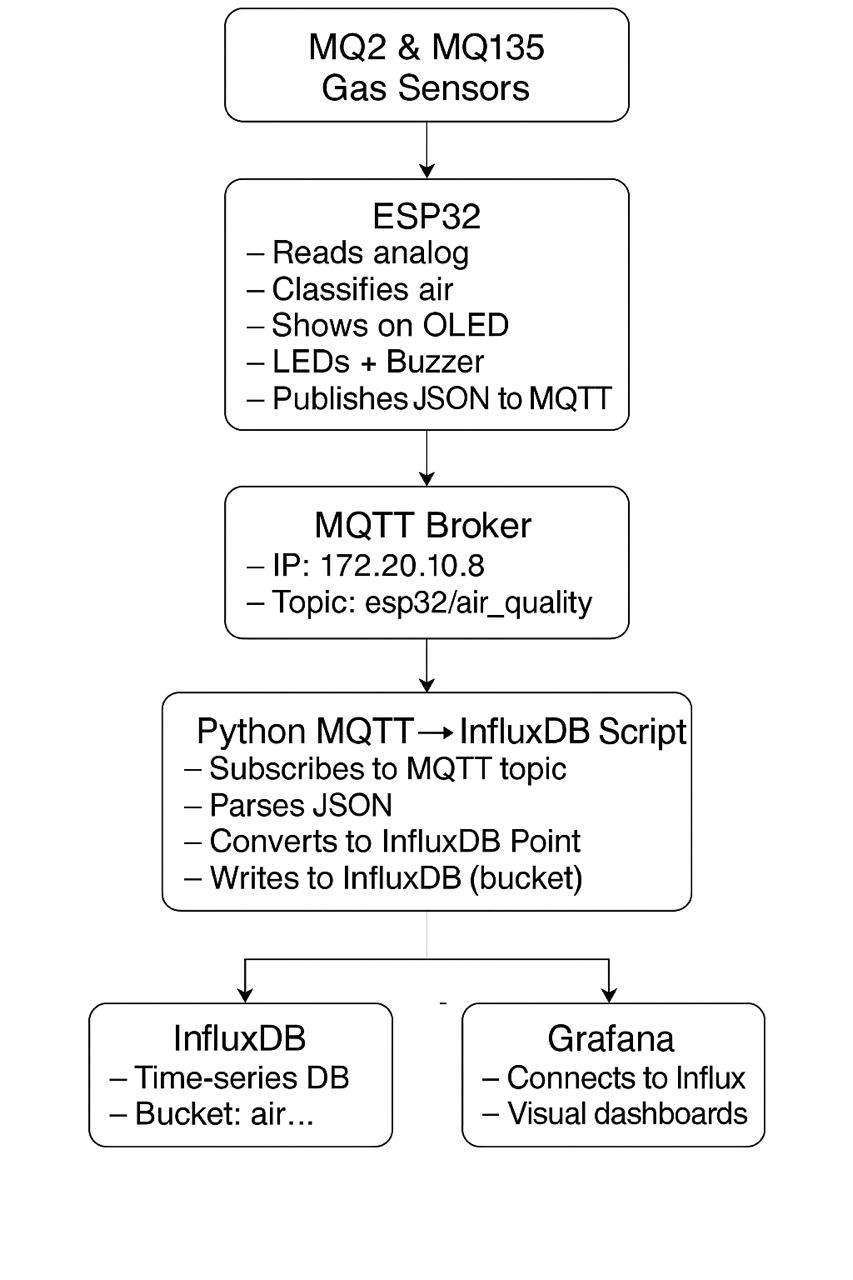
if smoke > threshold or air > threshold:

turn\_on\_led()

send\_to\_blynk(smoke, air)

sleep(5)

**6. Implementation**



**Arduino Code:**

// WiFi and MQTT details

const char\* ssid = "iPhone 13 Pro";

const char\* password = "11223344";

const char\* mqtt\_server = "172.20.10.8";

const int mqtt\_port = 1883;

const char\* topic = "esp32/air\_quality";

void reconnect\_mqtt() {

while (!client.connected()) {

Serial.print("Connecting to MQTT...");

if (client.connect("ESP32\_AirQuality")) {

Serial.println("connected");

} else {

Serial.print("failed, rc=");

Serial.print(client.state());

delay(2000);

}

setup\_wifi();

client.setServer(mqtt\_server, mqtt\_port);

}

void loop() {

if (!client.connected()) {

reconnect\_mqtt();

}

client.loop();

int mq2Value = analogRead(MQ2\_PIN);

int mq135Value = analogRead(MQ135\_PIN);

if (mq2Value < 800) {

digitalWrite(GREEN\_LED, HIGH);

quality = "GOOD";

} else if (mq2Value < 1800) {

digitalWrite(BLUE\_LED, HIGH);

quality = "MODERATE";

} else {

digitalWrite(RED\_LED, HIGH);

ledcWrite(buzzerChannel, 128); // Half duty (50%)

quality = "BAD";

}

// MQTT Publish

char payload[150];

snprintf(payload, sizeof(payload),

"{\"mq2\": %d, \"mq135\": %d, \"air\_quality\": \"%s\"}",

mq2Value, mq135Value, quality.c\_str());

client.publish(topic, payload);

Serial.print("Published: ");

Serial.println(payload);

delay(5000);

**Python Script:**

# MQTT broker config

broker = '172.20.10.8' # HiveMQ IP

port = 1883

topic = "esp32/air\_quality"

client\_id = 'mqtt\_influx\_bridge'

# InfluxDB config

influx\_url = "http://localhost:8086"

token = "R5pxqAZrrxiCgsCexGALmwG2PeVaGyqNsMpPAfSy89ufz6z8QuvZCKvz3qlHKaGGILdxkqvf17FkIIlvY9hetw=="

org = "ntu"

bucket = "air\_quality\_monitor"

def connect\_mqtt():

def on\_connect(client, userdata, flags, rc):

if rc == 0:

print("Connected to MQTT Broker!")

else:

print(f"Failed to connect, return code {rc}")

client = mqtt\_client.Client(client\_id=client\_id)

client.on\_connect = on\_connect

client.connect(broker, port)

return client

# Write to InfluxDB

write\_api.write(bucket=bucket, org=org, record=point)

print("Written to InfluxDB")

except Exception as e:

print(f"Error processing message: {e}")

def run():

client = connect\_mqtt()

client.subscribe(topic)

client.on\_message = on\_message

client.loop\_forever()

**Challenges:**

Sensor calibration was tricky

urequests sometimes failed due to weak Wi-Fi

Debouncing LED flickers

ESP32 faces challenges in sensor accuracy, data handling, and integrating with other systems

5v Power supply to sensors was tricky

**7. Results & Discussion**

**Outputs:**

Sensor readings in serial monitor

Data updates in Blynk app (Gauge, Label widgets)

**Performance:**

Real-time response ~1s delay

Blynk update stable under 2.4GHz Wi-Fi

MQ sensors show reliable trend changes

**Comparison:**

Smoke alerts matched real ignition sources

AQI levels increased with lighter spray/smoke test

**8. Testing & Validation / Limitations**

**Test Cases:**

Test with candle smoke: LED turns on, Blynk updates

Test with air spray: AQI spike observed

Disconnect Wi-Fi: ESP fails to send, handles exception

**Limitations:**

No GSM/Offline support

Blynk free plan has data rate limits

Sensors need calibration for real AQI

**9. Conclusion & Future Work**

**Key Takeaways:**

Low-cost, real-time detection of air and smoke

Simple Blynk-based dashboard for remote access

MicroPython is suitable for rapid IoT development

**Future Improvements:**

Add SMS alert

Integrate temperature/humidity sensors

Use AI/ML for pattern detection (fire prediction)

**10. References**

MQ-2 Datasheet

MQ-135 Info

<https://www.youtube.com/watch?v=4cv-6hddJdY>

<https://www.youtube.com/watch?v=kJ0vG7ZFZ5I>

[ESP32 MicroPython Guide](https://randomnerdtutorials.com/)

<https://docs.blynk.io/en/blynk.cloud/blynk-cloud-api>

**11. Links:**