WALCHAND COLLEGE OF ENGINEERING, SANGLI

(Government-Aided Autonomous Institute)

Department of Electronics Engineering

Third Year B. Tech.

Mini-Project-1 Synopsis (5EN345)

"Automated Smoking Zone Monitoring And Alerting System"

Submitted By

Group-2 of Batch EN4

Submitted By

Tanuj Joshi	21410051
Yash Bagul	21410057
Shreyas Shevale	21410061
Devyani Patil	21410067

Under the guidance of

Dr. B. G. Patil

TABLE OF CONTENTS

1. ABSTRACT

This project presents an enhanced automated smoking zone monitoring system, incorporating an MQ-2 smoke sensor interfaced with an ESP32 microcontroller. In addition to the core functionality of smoke detection and real-time alerting using a buzzer, the system now integrates a comprehensive fire threat analysis feature. The smoke values captured by the sensor are stored in a SQL database.

Implemented using Java Spring Boot, the system leverages the ESP32's capabilities to continuously read smoke sensor inputs, process the data, and store it in the SQL database. The system monitors smoke levels and, upon surpassing predefined thresholds, triggers both local alerts and remote notifications. The remote alerts are sent to a cloud server for archival and analysis.

The Java Spring Boot framework facilitates robust coding practices, ensuring scalability, and ease of maintenance. The stored smoke data enables detailed observations and trend analysis, allowing users to assess the evolving threat of fire in smoking zones. The system's effectiveness has been validated through extensive testing in diverse environments, showcasing its reliability and potential for widespread implementation.

This low-cost solution not only serves as an efficient smoking zone monitoring system but also contributes to fire prevention by providing insights into smoke concentration patterns over time.

2. INTRODUCTION

Smoking in public places poses a significant health risk to individuals in the vicinity. In response to this concern, our project introduces a system designed to automatically detect and alert people to smoking activities in these areas. Utilizing components such as the ESP32 microcontroller, an MQ-2 smoke sensor, and a buzzer, the microcontroller serves as the central processing unit, whether it's an ESP32, Arduino UNO, or any other microcontroller, contributing to the formation of the entire system.

Going beyond smoke detection, our automated smoking zone monitoring and alerting system includes a pivotal feature for smoke analysis. The system not only monitors smoking zones but also stores data on smoke values over time. By implementing data analytics, this stored information enables a comprehensive analysis of the smoke detection in the monitored area. This innovative approach provides valuable insights into the evolving patterns of smoke concentration, enhancing the system's capacity to assess and mitigate potential fire risks.

To optimize the hardware setup, we incorporate a motor driver that efficiently decreases the input voltage from 12V to 5V. This ensures compatibility and seamless operation of the system components, further contributing to its reliability and longevity. The hardware design remains both simple and easy to comprehend, making it an accessible solution for public spaces. The development process involves selecting components, designing the system architecture, and writing/debugging the software program using the Arduino IDE. Furthermore, the system is configured to send alerts promptly when the defined smoke level threshold is crossed. This multi-faceted approach not only promotes public health but also

contributes to fire prevention, reinforcing the system's effectiveness in ensuring the safety of public spaces.

Overview

Our mini project is designed to work at many public places as well as various private places where smoking is prohibited.

Our motto is to build the device which will help the people and society to discourage smoking and also prevent fire accidents due to smoking activities.

Problem Statement

The aim of this project is to design and implement and automated smoking zone monitoring and alerting system.

The expected outcomes of project are –

- The device will sense the smoking and alert the public/authorized person.
- The smoking in public places will eventually decrease.
- Our project aims to prevent public hazard caused by smoking through proactive measures enabled by data analytics of previously stored smoke level data.

3. OBJECTIVE

1. Develop a Robust Sensor Network: Create a reliable sensor network comprising smoke detectors strategically placed in designated smoking zones. The system's effectiveness hinges on a well-designed sensor network that accurately detects the presence of smoke.

2. Implement Real-time Data Transmission: Enable real-time data transmission from sensors to a central monitoring unit using IoT technology. Timely data transmission ensures that alerts are generated promptly, allowing for swift response.

3. Design an Intuitive User Interface: Create an intuitive and user-friendly graphical user interface (GUI) for monitoring and managing the system. A user-friendly interface simplifies system operation, making it accessible to personnel responsible for monitoring.

4. Develop a Smoke Detection Algorithm: Design and implement a smoke detection algorithm capable of distinguishing between normal air quality and the presence of smoke. The accuracy of the algorithm is critical in preventing false alarms and ensuring reliable smoke detection.

5. Establish Alerting Mechanism: Develop a robust alerting mechanism that notifies relevant personnel, such as security personnel or administrators, when smoke is detected. Alerts must be prompt and reliable to ensure quick response and compliance with smoking regulations.

6. Conduct Rigorous Testing: Conduct comprehensive testing to evaluate the system's performance under various conditions, including controlled smoke simulations. Testing is crucial to validate the system's functionality, accuracy, and reliability.

7. Ensure Data Privacy and Compliance: Implement measures to protect user privacy and ensure compliance with data protection regulations while monitoring smoking zones. Ethical and legal considerations are essential to maintain the system's integrity and user trust.

8. Analyze Data for Insights: Analyze collected data to identify trends and patterns in smoke concentration levels. Data analysis provides insights that can inform decision-making and system optimization.

9. Provide Room for Scalability: Design the system with scalability in mind, allowing for the integration of additional sensors or expansion to more smoking zones. Scalability ensures that the system can adapt to changing needs and growing facilities.

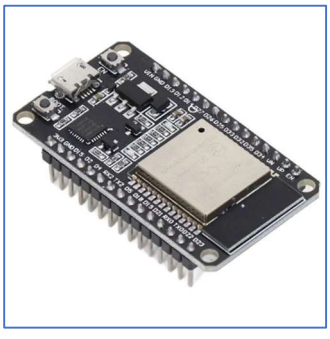
10. Efficient Resource Utilization: The automated system optimizes resource utilization by eliminating the need for dedicated personnel for constant monitoring. Human-based monitoring may require additional staff dedicated to surveillance, leading to higher labor costs.

4. COMPONENTS

1. ESP32 Microcontroller
   The ESP32 is a versatile microcontroller known for its robust wireless capabilities, dual-core processor, and extensive set of features. It is widely used in IoT applications and offers a balance between performance and power efficiency.

   ESP32 Specifications:

- Processor: Dual Core Tensilica LX6
- Operating Frequency: Adjustable upto 240 MHz
- Wireless Connectivity: Wi-Fi and Bluetooth
- Flash Memory: Varies (typically 4 MB)
- RAM: Varies (typically 520 Kb)
- Operating Voltage: 3.3V
- Analog Inputs: 18
- Digital I/O: 34 (of which 26 support PWM)
- GPIO Pins: 40
- I2C,SPI,UART,I2S: Yes, multiple interfaces
- Low Power Modes: Yes



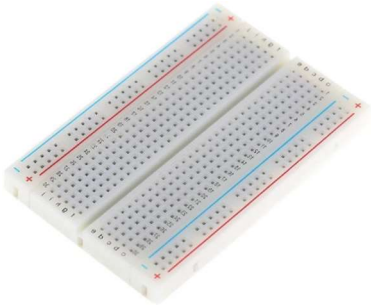
2. MQ-2 Smoke Sensor

The MQ-2 smoke sensor is a gas detection module widely used for detecting various gases, including smoke, methane, propane, and other flammable gases. It is commonly employed in applications where the monitoring of air quality and the detection of potential fire hazards are essential.

MQ-2 Sensor Specifications:

- Sensitivity Adjustment: Yes
- Operating Voltage:5V
- Heating Power Consumption: 800mW
- Analog Output Voltage(Clean Air): Varies (typically around 1.4V)
- Analog Output Voltage(Presence of Gas): Increases
- Digital Output: Yes
- Warm-up Time: Several Seconds

3. Bread Board: It is used to connect all components together.



4. Connecting Wires:

5. Buzzer 5V:

6. 5V rechargeable battery:

7. Motor Driver(L298N)

A motor driver is an electronic component or circuit that manages the operation and control of electric motors. It is integral in applications requiring precise motor control, such as robotics, automation systems, and electric vehicles.

Motor Driver Specifications:

- Driver Model: L298N 2A
- Driver Chip: Double H Bridge L298N
- Motor Supply Voltage (Maximum): 46V
- Motor Supply Current (Maximum): 2A
- Logic Voltage: 5V
- Driver Voltage: 5-35V
- Driver Current:2A
- Logical Current:0-36mA
- Maximum Power (W): 25W
- Current Sense for each motor
- Heatsink for better performance
- Power-On LED indicator

5. METHODOLOGY

1)Gather the required components:

- ESP32 Microcontroller
- MQ-2 Smoke Sensor
- Bread Board
- Connecting Wires
- Buzzer 5V
- 12V Rechargeable Battery
- L298N Motor Driver

2) Powering of esp32, Sensors and buzzers:

The operation voltage of the esp32 is 3.3V- 5V and therefore too low for the MQ2 gas sensor, that has an operation voltage of 5V. We power the esp32 board and the MQ-2 via 5V output of motor driver,

3)Connections for data transfer between peripherals and esp32:

1. Connect pin 34 to the analog output of the gas sensor
2. Connect the pin 25 of esp32 to the buzzer (To ring the buzzer when required)

4)Software part of the project (Writing and uploading code in esp32):

Set Up Your Development Environment:

1. Install the Arduino IDE
2. Install the ESP32 board package.
3. Connect Your ESP32:

Connect your ESP32 board to your computer using a USB cable. Make sure the board is powered on.

1. Select the Right Board and Port
2. In your Arduino IDE select the correct board model and port:
3. Select your ESP32 board model and select the appropriate COM port
4. Write or Open Your Code:
5. Write your code or open an existing project that you want to upload to the ESP32.
6. Compile Your Code
7. Upload Your Code
8. Monitor the Serial Output
9. Open the Serial Monitor in your IDE to view the output. Set the baud rate in your code and in the Serial Monitor to match.

5) Create a backend service for real time data storage:

We create a restful API in spring boot, which we add the smoke values and the metadata at real-time into our SQL database, which can be used by authorities for monitoring and analysis.

1. Make Spring Boot REST API.
2. Call to our Thing-Speak service to retrieve the data.
3. Validate the data is correct or not and avoid duplication of data.
4. Store the data into PostgreSQL database.

The GitHub link for source code: [Spring-Book-Backend-Integration]

6. ALGORITHM AND CODE

ALGORITHUM

1. Connect all peripherals and connect ESP32 to Wi-Fi.
2. Get Data from MQ-2.
3. When data is higher than 300 PPM send it to ThingSpeak client.
4. Get the data from thing-Speak client
5. Validate the data using spring-boot backend system
6. If data received is correct, then store it in PostgreSQL database

CODE

```
#include <WiFi.h>

#include "secrets.h"

#include "ThingSpeak.h" // always include thingspeak header file after other header files and custom macros

char ssid[] = SECRET_SSID;   // your network SSID (name)

char pass[] = SECRET_PASS;   // your network password

WiFiClient  client;

unsigned long myChannelNumber = SECRET_CH_ID;

const char * myWriteAPIKey = SECRET_WRITE_APIKEY;

#define sensorPin 34

#define buzzerPin 25

void setup() {

  Serial.begin(115200);  //Initialize serial
```

```
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo native USB port only
  }
  WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);  // Initialize ThingSpeak
}
void loop() {
  // Connect or reconnect to WiFi
  if(WiFi.status() != WL_CONNECTED){
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(SECRET_SSID);
    while(WiFi.status() != WL_CONNECTED){
      WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network. Change this line if using open or WEP network
      Serial.print(".");
      delay(5000);
    }
    Serial.println("\nConnected.");
  }
  int smokeValue = 0;
  smokeValue = analogRead(sensorPin);
  Serial.println("Smoke value : ");
  Serial.println(smokeValue);
  // Write to ThingSpeak. There are up to 8 fields in a channel, allowing you to store up to 8 different
  // pieces of information in a channel.  Here, we write to field 1.
```

```
  if(smokeValue > 2000) {
    tone(buzzerPin, 1200, 3000);
    delay(2000);
  }
  if(smokeValue > 300) {
    int x = ThingSpeak.writeField(myChannelNumber, 1, smokeValue, myWriteAPIKey);
    if(x == 200){
      Serial.println("Channel update successful. Code: " + String(x));
    }
    else{
      Serial.println("Problem updating channel. HTTP error code " + String(x));
    }
    delay(20000);
  }
  delay(1000);
}
```

Spring-Boot-backend: [Backend-Github-link]

7. APPLICATIONS AND BENEFITS

APPLICATIONS

1)Public Spaces: Implementing this system in public areas like airports, railway stations, bus terminals, and malls can help in creating designated smoking zones and ensuring that people adhere to the rules.

2)Educational Institutions: Universities and schools can use this technology to enforce no-smoking policies within their campuses. It can also serve as an educational tool to discourage smoking among students.

3)Workplaces: Offices and factories can use this system to monitor and enforce no-smoking policies in designated areas. It can contribute to a healthier and safer work environment.

4)Healthcare Facilities: Hospitals and healthcare facilities can implement this technology to ensure that there is no smoking in and around their premises, contributing to patient well-being.

5)Residential Buildings: Apartment complexes and residential communities can deploy this system in common areas to discourage smoking, creating a cleaner and healthier environment for residents.

BENEFITS / ADVANTAGES

1)Health and Safety: The primary benefit is the improvement of health and safety. Smoking is a major health hazard, and enforcing no-smoking policies reduces exposure to harmful secondhand smoke.

2)Real-time Monitoring: The IoT system provides real-time monitoring of smoking levels in public areas, allowing authorities to receive instant alerts when smoke is detected. Traditional monitoring by people may rely on periodic checks, and there could be delays in identifying smoking incidents.

3)Continuous Operation: The system operates continuously, 24/7, providing constant surveillance without human intervention. Traditional monitoring may be subject to human limitations such as breaks, shift changes, and potential lapses in attention.

4)Accuracy and Consistency: IoT sensors are designed for accuracy and consistency in detecting smoke levels, reducing the likelihood of false alarms or missed incidents. Human-based monitoring may vary in terms of accuracy and consistency due to factors like fatigue, distraction, or differing interpretations.

5)Efficient Resource Utilization: The automated system optimizes resource utilization by eliminating the need for dedicated personnel for constant monitoring. Human-based monitoring may require additional staff dedicated to surveillance, leading to higher labor costs.

6)Immediate Alerts and Response: Authorities receive immediate alerts when smoking is detected, enabling swift response and intervention. Traditional monitoring may rely on someone noticing smoking activity and reporting it, introducing delays in response time.

7)Data Logging and Analysis: The system logs and stores data on smoking incidents, providing a valuable historical record for analysis and reporting.

traditional monitoring might lack a systematic and centralized approach to record-keeping, making it harder to analyze trends and patterns.

8)Cost-Effective: Over the long term, an automated IoT system can be cost-effective compared to employing personnel for continuous monitoring. While there are upfront costs for setting up the IoT infrastructure, it can lead to savings in the form of reduced labor costs over time.

9)Scalability: The IoT-based system is scalable, allowing for easy expansion to cover larger areas or additional monitoring points. Traditional monitoring might face challenges in scaling up without a proportional increase in human resources.

10)Integration with Other Systems: The IoT system can be integrated with other smart city technologies or emergency response systems for a more comprehensive solution. Traditional monitoring may operate in isolation and lack seamless integration with other city infrastructure.

11)Enhanced Public Safety: The system contributes to enhanced public safety by detecting and addressing smoking incidents promptly, minimizing risks associated with smoking in prohibited areas. Traditional monitoring may have limitations in providing timely alerts and interventions.

7. FUTURE SCOPE

1. Enhanced Sensing Technology: Smoke detectors may become more sensitive to detect smoke particles at an early stage, reducing false alarms and improving fire detection accuracy.

2. Connectivity: Integration with the Internet of Things (IoT) will enable smoke detectors to communicate with other devices and emergency services for faster response times.

3. Can determine the leakage and send the data over to a website, where it can be monitored and corrective actions can be taken.

4. Fire Prevention Technologies: Future smoke detectors may integrate with fire suppression systems, such as automated sprinklers, to mitigate fires more effectively.

5. The presence of automatic fire &smoke detector is very useful to send the message to fire station to prevent the unnecessary fire accidents.

6. If appropriate measures are taken quickly after it is reported over the IOT, it can help in saving the loss of lives and property.

7. We can enhance the gas leakage detection system project to detect toxic gases. Further, we can add Smoke and Fire Detectors to detect fire

8. CONCLUSION:

1. Functionality: The ES32-based smoke detector effectively detects smoke and triggers an alarm when smoke is detected, ensuring the safety of the environment.

2. Sensor Performance: The choice of smoke sensor (e.g., MQ-2, MQ-7) significantly impacts the system's sensitivity and false alarm rate. Calibration and testing are essential to optimize sensor performance.

3. Alarm Mechanism: We have successfully integrated an audible alarm (e.g., buzzer) to alert users when smoke is detected.

4. Threshold Settings: Adjusting the smoke detection threshold is crucial to balance between early detection and false alarms. This requires experimentation and fine-tuning.

5. Using Real-time data process provides fast and accurate feedback, and gives room for further analysis of data.