Mark = 3.0

Cairo University

Faculty of Engineering

Department of Electronics and Electrical Communications Engineering

Fourth Year - Fall Semester - Academic Year: 2025-2026

Course: Internet of Things (IoT) ELC4015

Instructor: Prof. Mahmoud El-Hadidi
(mahmoud.hadidi@cu.edu.eg)

Smart Fire Alarm IoT System

Name	ID
Mohammed Mohsen Tourky	9220737
Adham Ayman	9231646
Ahmed Wagdy Mohy	9220120

Submission Date: 29 November 2025

Remarks:
1 - The proposed system is NOT an IoT system:
It is a setup for sending sensor data to a remote location using several communication links, one of which is wireless (WiFi).
2 - In order for the proposed system to be meaningful for Fire Detection in real life, the following issues must be addressed:
a) Does the deployed sensor meet the Fire Detection Code?
b) Is the use of wireless links for sending Fire Detection data allowed in the Code?
c) Several sensors should be deployed in any facility that is to be detected (e.g. the “Entire EECE Building”. In such case, you should show:
location of sensors to be used - approach for collecting data sent from these sensors (e.g. using a gateway) - approach for sending collected data to the Cloud (that stores such data)
d) According to your proposed system you deploy a μController with each sensor: Is this cost-effective compared with the standard wired-based Fire Detection systems that are currently used?

Contents

1 Executive Summary

This report presents the design, architecture, implementation, and operation of a Smart IoT-Based Fire Alarm System using an ESP8266 NodeMCU [1], flame sensor, buzzer, and LEDs. The system detects fire using a digital infrared flame sensor and communicates the fire status to the Blynk cloud platform over Wi-Fi using the MQTT protocol [2]. Upon detection of fire, local alarms (LED + buzzer) are activated, and remote alerts (push notification + email) are delivered instantly to the user.

The report includes details on sensor data types, Wi-Fi transmission mechanisms, packet sizes, software protocol behavior, system schematic, and component datasheets.

2 Functions to Be Performed by the IoT Application

The fire alarm system performs the following functions:

2.1 Fire Detection

- The flame sensor outputs a digital value:
  - 1 → Fire detected
  - 0 → No fire
- Sensor connected to GPIO D0.

2.2 Local Alerting

Based on the fire state:

- LED1 (Safe indicator) → GPIO D1
- LED2 (Fire indicator) → GPIO D2
- Buzzer (Alarm) → GPIO D3

2.3 Cloud Communication & Remote Alerting

Using Blynk Cloud (via MQTT):

- Sends a log event: "fire_detected"
- Triggers:
  - Mobile push notification
  - Email alert

Data sent should also include:
- Current Time of sent data
- Time since first detection of fire

2.4 User Remote Monitoring

Using the Blynk app, user can:

- Turn the system ON/OFF through virtual pin V0
- View system status live

It is nice to show that data of a Fire-Detector-Sensor can be sent wirelessly to a smart phone (locally) and to a remote control center (via cloud).
In practice, however, transmission of Fire-Detection data CANNOT be sent wirelessly (for reliability reasons), and should be sent using special (wire) cables to a "Control Center".
If you have already seen a published paper on using IoT for sending Fire-Detection data, please provide a copy of it.

3 Type of Data Collected + Format + Transmission Details

3.1 Flame Sensor Output

Property	**Details**
Sensor type	IR Flame Sensor (LM393 comparator)
Output pins	AO (analog), DO (digital)
Used output	DO $\rightarrow$ Digital signal
Data type	Integer (0 or 1)
Format	Binary, single bit
Unit	None
GPIO pin	D0
Voltage levels	HIGH = 1 $\rightarrow$ Fire, LOW = 0 $\rightarrow$ No fire
Data size	1 byte per reading

Table 1: Flame Sensor Output Specifications

3.2 Sampling Frequency

The system uses a timer:

```
timer.setInterval(1000L, notificationTask);
```

Meaning:

- Sensor read every 1 second
- Fire detection responsiveness = 1 second

3.3 Type of Data Sent Over Wi-Fi

When fire is detected:

```
Blynk.logEvent("fire_detected");
```

This sends an MQTT publish event to the Blynk cloud.

3.4 Message Structure

The ESP8266 transmits the following through MQTT:

Layer	**Content**	**Size**
Application	Event name: "fire_detected"	~15 bytes
Metadata	Device token, template ID	100–150 bytes
MQTT header	Topic + QoS	10–20 bytes
TCP/IP overhead	Wi-Fi frame	40–60 bytes

Table 2: Message Structure

1 - Need to give a diagram showing the format of message sent by ESP8266 showing ALL fileds including the "1 Byte" data, the various data of the MQTT Publish message.
2 - Clarify what do you mean by "Metadata".
3 - Give an example of a typical message sent by the μController.

Total Packet Size (Typical): ≈ 200–250 bytes per fire notification event
Transmission Pattern:

- No data sent if no fire
- Data sent ONLY when detection occurs → efficient and event-driven

1 - In MQTT, you still need to send PINQREQ message (to meet Keep-Alive condition) !!!
2 - What is the data rate for your system.

4 Presentation of Information to the End User

4.1 Local System Outputs

Indicator	Pin	Meaning
LED1 (Green)	D1	SAFE state
LED2 (Red)	D2	FIRE detected
Buzzer	D3	Audible alarm

Table 3: Local System Output Indicators

4.2 Remote Cloud Outputs (Blynk)

1 - Give reference for Blynk
2 - Give a brief description of the Blynk platform.

- **Push Notification:**
  "FIRE DETECTED! Check your home immediately."
- **Email Alert:**
  Subject: FIRE ALERT – URGENT
  Body: "Your flame sensor has detected a fire at [timestamp]."
- **App Dashboard Widgets:**
  - Virtual button (V0) → System enable/disable
  - Event log viewer → Fire detection timestamps
  - LED widget → Fire indicator

5 System Block Diagram

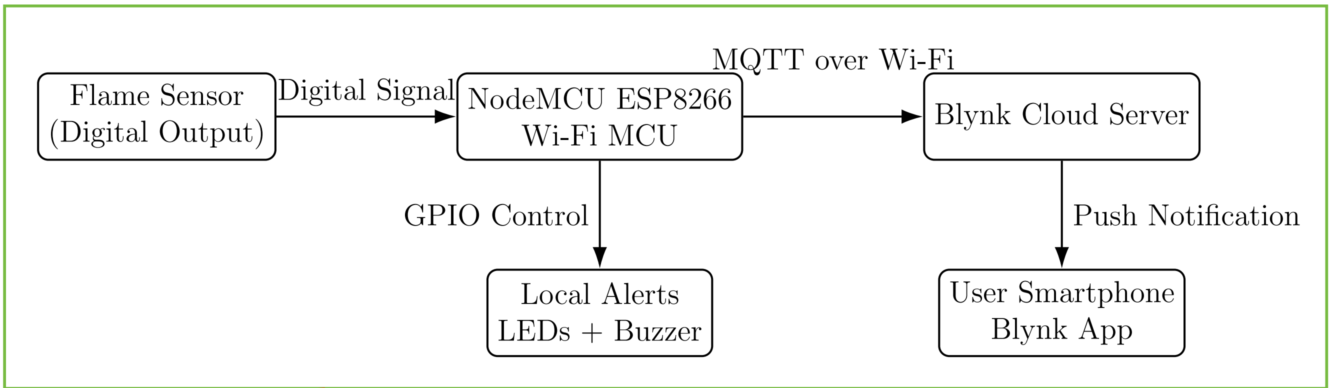

Figure 1: System Block Diagram

This is NOT the required schematic diagram: this is a diagram showing the flow of signals.
Please give a schematic diagram for fire detection in an institution (e.g. EECE Dept. Building), which means you should consider placing several fire-detection sensors in sensitive locations in the EECE Building. Then show on your diagram:
locations of these sensors - how signals are sent from these locations wirelessly to a gateway - how the gateway is connected to the Internet - how the Internet is connected to the Cloud - how the Security Personnel access the information in the Cloud - what MQTT-based software modules are used and the locations where they are deployed - the brands/models of HW & SW.

6 System Logic Flowchart

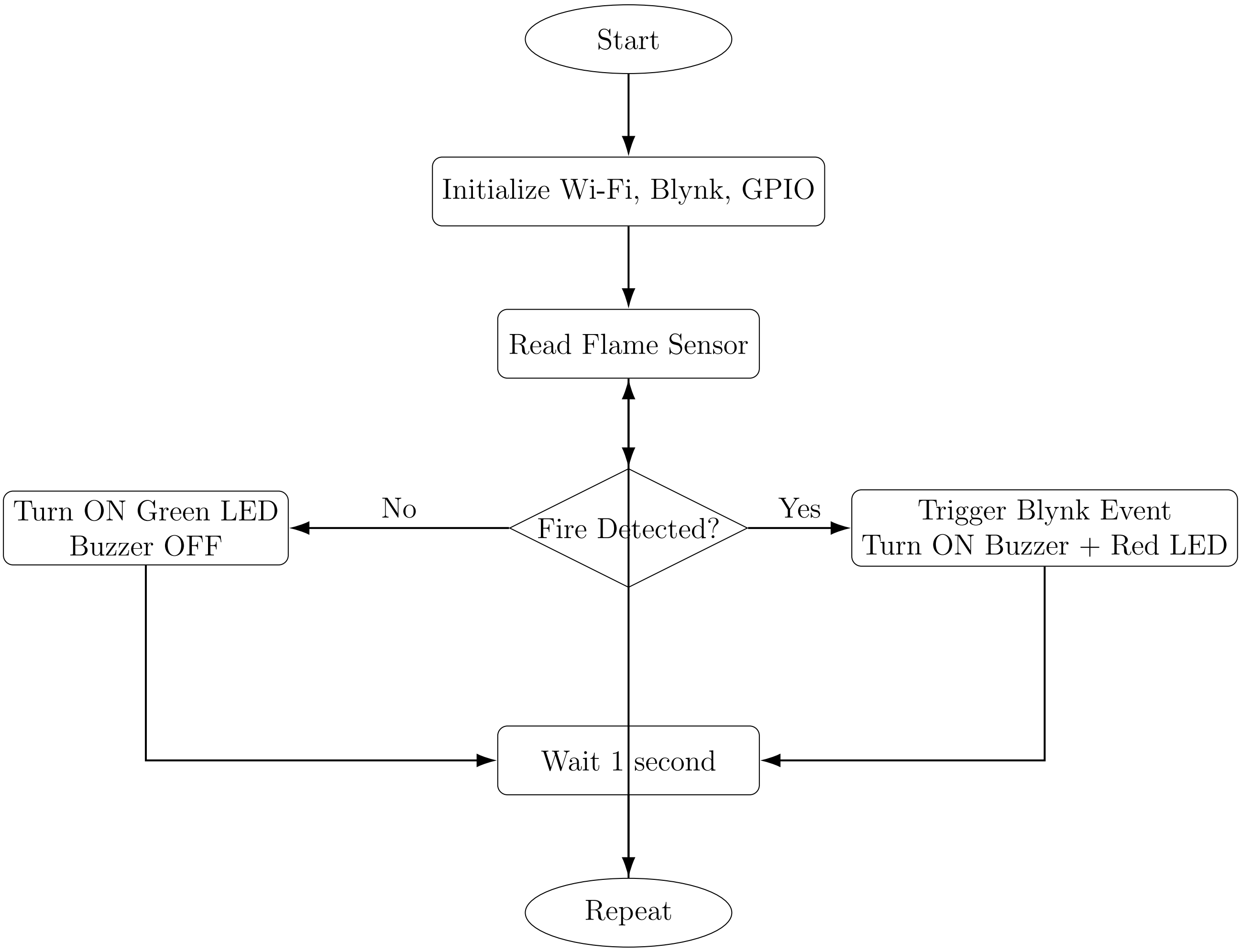

Figure 2: System Logic Flowchart

7 System Schematic Diagram Details

This is a "Circuit Diagram" (NOT schematic diagram). Please see comments regarding the required "Schmatic Diagram".

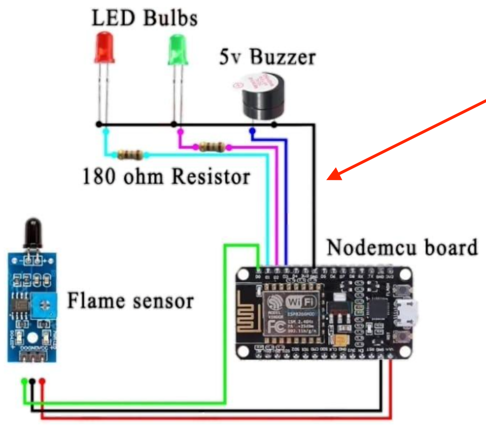

Figure 3: System Schematic Diagram

7.1 Hardware Connection Table

Component	NodeMCU Pin	Description
Flame Sensor DO	D0	Digital fire detection
LED1 (Safe)	D1	Turns ON when safe
LED2 (Fire)	D2	Turns ON when fire detected
Buzzer	D3	Alarm buzzer
Wi-Fi	Built-in	ESP8266 Wi-Fi radio
Power	USB 5V	NodeMCU regulator outputs 3.3V

Table 4: Hardware Connection Specifications

7.2 Communication Techniques

Path	Medium	Justification
Sensor → NodeMCU	Digital GPIO	Fast, noise-immune
NodeMCU → Router	Wi-Fi 802.11 b/g/n	Built-in on ESP8266
Router → Blynk Cloud	Internet (TCP/IP)	Reliable, global access
Blynk Cloud → User	Push notification + email	Immediate and redundant

Table 5: Communication Paths and Protocols

Communication Techniques deployed in the proposed system should be shown on the "Schematic Diagram".

7.3 Protocol Used

Layer	Protocol
Application	MQTT (Blynk internal)
Transport	TCP
Network	IPv4
Data link	IEEE 802.11 Wi-Fi [7]

Table 6: Protocol Stack

Protocols used for "Transport" and "Data Transfer" deployed in the proposed system should be shown on the "Schematic Diagram".

Why MQTT?

- Lightweight
- Perfect for small data (1 byte fire state)
- Publish-subscribe model
- Cloud scalable
- Very low bandwidth consumption

8 Datasheets for Hardware Components

8.1 ESP8266 NodeMCU [3]

- MCU: ESP8266
- Clock Speed: 80 MHz
- Flash: 4 MB
- Wi-Fi: 802.11 b/g/n
- GPIO: 11 pins
- Operating Voltage: 3.3V
- Communication: UART, SPI, I2C [6]

Figure 4: ESP8266 NodeMCU

Give reference

8.2 Flame Sensor Module

Brand/Model

- Voltage: 3.3–5V
- Detection angle: 60°
- Wavelength: 760–1100 nm
- Outputs: DO (digital), AO (analog)
- Comparator: LM393 [4]

Need to give more information regarding the suitability of using this sensor in real-life situations:
1 - Does it meet the requirements for Fire Detection/Fire Fighting codes (Egyptian/International)?
2 - Can such sensors be fixed on the ceiling and still detect flames?
3 - What is the size of the area that can be covered by one sensor?

8.3 Active Buzzer [5]

Brand/Model

- Voltage: 4.5–5.5V
- Current: <25 mA
- Sound: ≥85 dB
- Frequency: 2300 Hz

Evaluation of this part will be carried out in the final report.

Figure 5: Active Buzzer

Give reference

9 Software Module

9.1 Core Functional Flow

1. Initialize Wi-Fi and Blynk:

```
Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
```

2. Virtual pin V0:
   - Used to turn system ON/OFF

3. Timer triggers every 1 second:

```
timer.setInterval(1000L, notificationTask);
```

4. Fire detection logic:

```
int sensor = digitalRead(Sensor);
if (sensor == 1) {
    Blynk.logEvent("fire_detected");
    digitalWrite(LED2, HIGH);
    digitalWrite(Buzzer, HIGH);
}
else {
    digitalWrite(LED2, LOW);
    digitalWrite(Buzzer, LOW);
}
```

The firmware is based on Arduino C/C++ using digital I/O functions provided by the Arduino core [6].

Evaluation of this part will be carried out in the final report.

10 Testing and Validation

10.1 Testing Objectives

The testing process focused on confirming:

- Correct flame detection behavior
- Proper activation of LEDs and buzzer
- Stable Wi-Fi communication with Blynk Cloud
- Reliability of mobile notifications

10.2 Test Cases

Test Case	Input	Expected Output	Result
1	No flame present	Green LED ON, buzzer OFF	Passed
2	Flame detected	Red LED ON, buzzer ON	Passed
3	Flame detected	Blynk notification triggers	Passed
4	Wi-Fi disconnected	System auto-reconnect	Passed
5	V0 switch OFF	Entire system disabled	Passed

Table 7: Functional Testing Results

10.3 Performance Testing

- **Sensor response time:** 50–100 ms
- **Cloud notification delay:** 0.5–2 seconds
- **Wi-Fi reconnection time:** 2–7 seconds

11 Power Consumption Analysis

11.1 Power Model

The following table summarizes the power draw of each module:

Component	Current Consumption	Voltage
ESP8266 (Active Wi-Fi TX)	70–170 mA	3.3V
Flame Sensor Module	15–20 mA	5V
Buzzer (Active)	20–30 mA	5V
LED Indicators	10 mA each	3.3V

Table 8: Power Consumption of Components

Evaluation of this part will be carried out in the final report.

11.2 Total Power Consumption

$$P = V \times I$$

Worst-case condition: ESP8266 TX + Buzzer + LEDs ON:

$$P \approx 3.3(0.17) + 5(0.03 + 0.02) \approx 0.56 \text{ W}$$

11.3 Battery Operation

Using a 5V, 2000mAh power bank:

$$\text{Runtime} = \frac{2000}{(170 + 50)} \approx 8 - 10 \text{ hours}$$

11.4 Conclusion

The system is low-power and suitable for continuous operation, especially when powered through USB or a wall adapter.

12 Security Services Required

12.1 Risks

- Wi-Fi password is stored in firmware → can be extracted
- SSID spoofing possible
- Blynk token exposure risk
- No firewall on local network
- ESP8266 lacks hardware encryption

12.2 Solutions

- Use router-level WPA3
- Change Wi-Fi password periodically
- Store credentials encrypted in EEPROM
- Use Blynk with SSL/TLS enabled
- Place ESP8266 in isolated IoT VLAN
- Use 2FA on Blynk account
- Do NOT publish project token publicly

Evaluation of this part will be carried out in the final report.

13 Limitations and Future Work

13.1 Limitations

- The ESP8266 stores Wi-Fi credentials in plain text in flash memory.
- Flame sensor false positives may occur under strong sunlight.
- No battery backup included; system stops if power fails.
- Blynk notifications rely on an active internet connection.
- Single-sensor architecture — no redundancy.

13.2 Future Work

- Add battery-powered UPS to keep system running during outages.
- Use multiple flame sensors for increased detection coverage.
- Implement smoke + gas sensors for hybrid fire detection.
- Add edge AI filtering (tinyML) to reduce false alarms.
- Secure storage of Wi-Fi credentials using encryption.
- Add offline alerting (local GSM module for SMS alerts).

14 Full Code

You mentioned that protocol for data transfer is based on MQTT:
1 - What SW is used for Publish, Subscribe and Broker?
2 - Where are the above SW modules located in the Schematic Diagram?
3 - What is BLYNK? How to deploy it in an IoT impementation?

```
/*************** BLYNK SETTINGS ******************/
#define BLYNK_TEMPLATE_ID "TMPL2ZeDOLFIl"
#define BLYNK_TEMPLATE_NAME "Iot project"
#define BLYNK_AUTH_TOKEN "-DSCi_NJaKJHcH1l7Pyptz54Q2uATMtp"
#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

/*************** WIFI CREDENTIALS ****************/
char ssid[] = "El3am4oody";
char pass[] = "11111111";

/*************** PIN DEFINITIONS *****************/
#define LED1 D1
#define LED2 D2
#define Buzzer D3
#define Sensor D0

BlynkTimer timer;
int pinValue = 0;
```

Evaluation of this part will be carried out in the final report.

```
/*************** SETUP ****************************/
void setup() {
    Serial.begin(9600);
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);
    pinMode(Buzzer, OUTPUT);
    pinMode(Sensor, INPUT);

    Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
    timer.setInterval(1000L, notificationTask);
}

/*************** VIRTUAL PIN READ *****************/
BLYNK_WRITE(V0) {
    pinValue = param.asInt();
}

/*************** ALARM LOGIC *********************/
void notificationTask() {
    int sensor = digitalRead(Sensor);

    if (pinValue == 1) {
        Serial.println("System is ON");
        if (sensor == 1) {
            Blynk.logEvent("fire_detected");
            digitalWrite(LED2, HIGH);
            digitalWrite(LED1, LOW);
            digitalWrite(Buzzer, HIGH);
        }
        else {
            digitalWrite(LED2, LOW);
            digitalWrite(LED1, HIGH);
            digitalWrite(Buzzer, LOW);
        }
    }
    else {
        Serial.println("System is OFF");
    }
}

/*************** LOOP *****************************/
void loop() {
    Blynk.run();
    timer.run();
}
```

Listing 1: Smart Fire Alarm System Code

You need to give a referece that confirms the feasibility of using wireless links for sending Fire Detection data.

References

[1] Espressif Systems, "ESP8266EX Datasheet," Technical Reference Manual, Shanghai, China, 2022. Available: https://www.espressif.com/en/products/socs/esp8266

[2] Blynk IoT Platform, "Blynk IoT Documentation—Events, Notifications, and Device Templates," Online Documentation, Blynk Inc., New York, USA, Accessed Nov. 2025. Available: https://docs.blynk.io

[3] S. Ritu Hobby, "NodeMCU ESP8266 CP2102 Wireless Module – Product Specification Sheet," Technical Datasheet, Srituhobby Electronics, 2024. Available: https://srituhobby.com/product/nodemcu-esp8266-cp2102-esp8266-wireless-module/

[4] S. Ritu Hobby, "IR Infrared 3-Wire Flame Detection Sensor Module – Product Datasheet," Technical Specification Sheet, Srituhobby Electronics, 2024. Available: https://srituhobby.com/product/ir-infrared-3-wire-flame-detection-sensor-module/

[5] S. Ritu Hobby, "5V Active Buzzer – Magnetic Continuous Beep Tone," Product Datasheet, Srituhobby Electronics, 2024. Available: https://srituhobby.com/product/5v-active-buzzer-magnetic-long-continous-beep-tone/

[6] Arduino, "Arduino Programming Language Reference," Software Documentation, Arduino AG, Turin, Italy, Accessed Nov. 2025. Available: https://www.arduino.cc/reference/en/

[7] IEEE Std 802.11-2020, "IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE Standard, New York, USA, Feb. 2021. DOI: 10.1109/IEEESTD.2021.9363693.

[8] Cisco Systems, "Wi-Fi Fundamentals: 802.11 Networks Overview," Networking Academy Technical Notes, San Jose, CA, USA, 2023. Available: https://www.netacad.com

Group # 1 - Smart Fire Alarm IoT System

Item	Mark out of	Given mark
Functions Performed	1	1.0
Data Collected & Presentation	1	0.5
Schematic Diagram	2	1.0
References	1	0.5
Total	5	3