



INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Flame Detection System

Bachelors of Engineering

1. Falak Naaz Khan – 242P002
2. Mariam Shaikh – 242P001
3. Shifa Shaikh – 242P004
4. Aafiya Saudagar – 242P015

Guide:

Prof. Manila Gupta

Abstract

In a world where fire safety and real-time response are critical, the Flame Sensor-based IoT System offers an intelligent and responsive solution for early flame detection and remote alerting. Designed using the ESP8266 NodeMCU microcontroller and integrated with the Blynk IoT platform, this system provides a cost-effective yet efficient method for real-time fire monitoring in both residential and industrial settings.

The system leverages a digital flame sensor to detect the presence of fire, triggering immediate local alerts through a buzzer and remote notifications via push alerts and email. Core components such as Wi-Fi connectivity and Blynk cloud integration enable seamless communication and allow users to monitor their environment remotely through a smartphone or computer.

Key features of the system include:

- **Real-Time Flame Detection:** The sensor detects fire instantly and initiates an immediate alert response.
- **Blynk Notification System:** Sends notifications and email alerts to the owner's registered devices, even when off-site.
- **Visual and Audio Feedback:** LED indicators and a buzzer provide immediate, on-the-spot alerts when flames are detected.
- **Notification Management:** The use of BlynkTimer helps prevent alert spamming by managing the frequency of notifications.

This system is not only easy to set up but also scalable for various use cases—ranging from home safety to industrial fire monitoring. With its combination of fast response, remote accessibility, and reliable performance, the flame detection system stands out as a robust solution for modern fire safety applications.

Chapter 1

Introduction

Introduction to Flame Sensor-Based IoT Fire Detection System in an era where technology is rapidly redefining safety standards, the need for intelligent, real-time fire detection systems has never been more critical. Traditional fire alarms and smoke detectors, while still valuable, often fall short in delivering the speed, accuracy, and remote accessibility demanded by today's interconnected environments. This gap has led to the development of innovative, IoT-based fire detection solutions that promise to revolutionize the way we detect and respond to fire hazards—whether in homes, industries, or remote facilities.

At the forefront of this innovation is the Flame Sensor-Based IoT Fire Detection System, powered by the ESP8266 NodeMCU microcontroller and seamlessly integrated with the Blynk IoT platform. This project represents a significant leap forward in combining hardware efficiency with cloud-based monitoring, offering a comprehensive safety mechanism that not only detects the presence of flame but also delivers real-time alerts to users anywhere in the world.

- Real-Time Flame Detection:** Precision Meets Speed
 Utilizing a digital flame sensor, the system detects the presence of fire by responding instantly to infrared light emitted from flames. Upon detection, a buzzer is activated, signaling nearby occupants while the microcontroller processes and communicates the event.
- Blynk Integration:** Remote Awareness and Alerts
 Through the integration with the Blynk platform, users receive push notifications and email alerts directly on their registered devices. This ensures that no fire incident goes unnoticed, even when the user is away from the monitored location.
- Notification Management:** Smart, Non-Intrusive Alerts
 To prevent false alarms and alert fatigue, a BlynkTimer is used to smartly manage notification frequency—ensuring that only consistent flame detection triggers alerts, and transient readings are intelligently filtered out.

In conclusion, the Flame Sensor-Based IoT Fire Detection System is a powerful, compact, and reliable solution aimed at enhancing fire safety through real-time detection and remote notification. By harnessing the capabilities of modern microcontrollers and IoT platforms, it offers an accessible yet highly effective tool for safeguarding lives and property—ushering in a smarter, safer future.

Chapter 2

Review of Literature

- Fire Detection and Early Warning Systems:** Literature on fire safety consistently highlights the importance of early detection in preventing fire-related accidents and minimizing damage. Flame sensors integrated with microcontrollers like the ESP8266 offer real-time flame detection capabilities, which are critical in both residential and industrial safety applications.
- IoT-Based Monitoring and Notification:** Studies emphasize the effectiveness of IoT platforms, such as Blynk, in enhancing remote monitoring. By providing push notifications and email alerts, these systems significantly reduce response time and enable timely interventions during fire emergencies.
- System Reliability and Redundancy:** Flame detection systems often face the challenge of false positives due to environmental noise or lighting. The use of timers, such as BlynkTimer, has been documented in literature as an effective method to filter transient detections and minimize false alarms, ensuring only valid alerts are communicated.

4. **Real-Time Feedback Mechanisms:** The integration of visual (LED indicators) and auditory (buzzers) feedback has been shown to enhance user awareness in local environments. These mechanisms allow users to take immediate action, even before receiving mobile notifications, thereby adding an extra layer of safety.
5. **User Experience and Accessibility:** Previous case studies underscore the importance of ease of setup and use in IoT systems. Systems built with the ESP8266 and Blynk are widely acknowledged for their user-friendly interfaces, making them accessible to both technical and non-technical users alike.
6. **Challenges and Future Improvements:** While flame sensor systems provide effective warnings, limitations such as environmental interference and notification delays are commonly discussed. Future advancements are recommended to focus on reducing latency, improving sensor accuracy, and enhancing multi-sensor integration for broader fire detection coverage.

Chapter 3

Report on the Present Investigation

Different Types of Methodologies for Flame Detection.

3.1.1. Sensor-Based Approach: This method uses a flame sensor that detects infrared (IR) radiation emitted by fire. The sensor provides a digital output (HIGH or LOW), which is processed by the ESP8266 microcontroller. When a flame is detected, the system triggers an alarm using a buzzer and sends a notification via Blynk IoT. This approach is cost-effective and has a fast response time, but it may be affected by external light sources.

3.1.2. IoT-Based Monitoring Approach: This approach integrates the Blynk IoT platform with the ESP8266 to enable remote monitoring and notifications. When a fire is detected, the system logs an event in Blynk, which sends push notifications and email alerts to notify the user in real time. This methodology ensures that users can take immediate action, even when they are not physically present.

3.2. Technologies that can be used are discussed as:

3.2.1. ESP8266WiFi Library: This library enables the ESP8266 microcontroller to connect to a Wi-Fi network for internet-based communication. It allows the system to interact with the Blynk IoT platform and transmit real-time data.

3.2.2. Blynk IoT Platform: Blynk is a cloud-based IoT platform that facilitates remote monitoring and control of smart devices. In this project, Blynk allows the ESP8266 to send notifications upon detecting fire. It supports event logging, ensuring alerts are efficiently managed without spamming the user. This makes it a powerful tool for real-time fire detection applications.

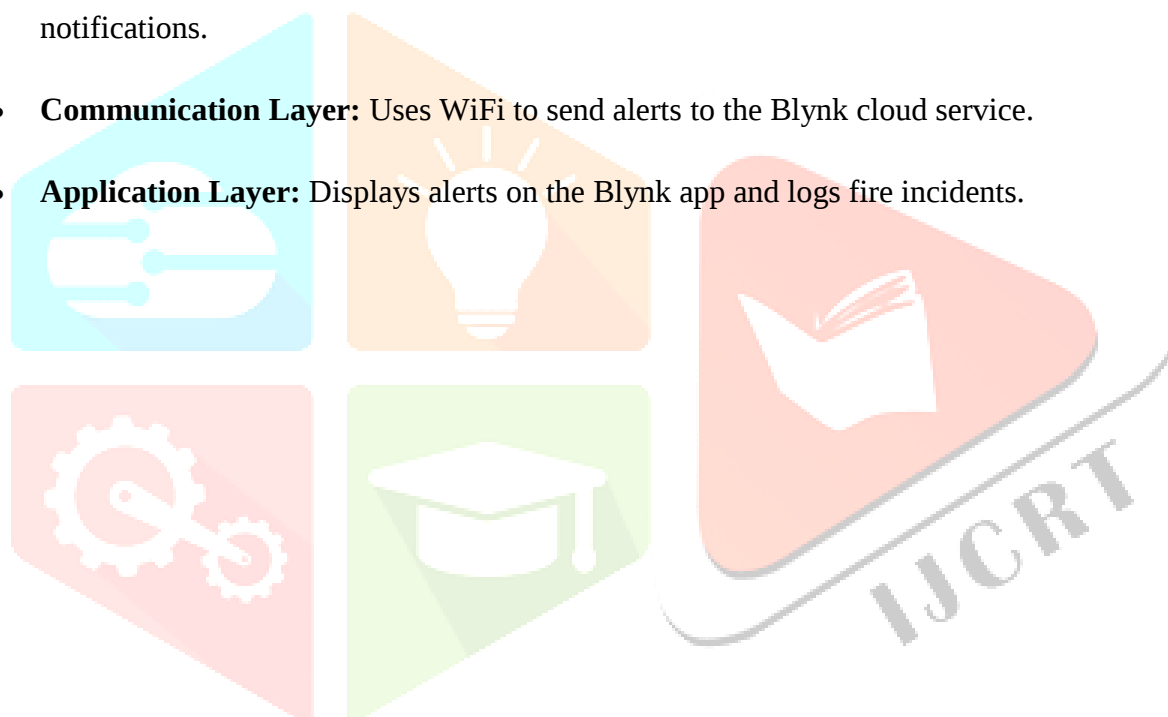
3.2.3. BlynkTimer: To avoid spam alerts, a BlynkTimer is implemented to check the sensor readings every 10 seconds. This prevents the system from continuously sending alerts due to momentary flame detections or sensor noise.

3.2.4. Embedded Systems & Microcontrollers: Embedded systems like ESP8266 provide a low-cost, efficient, and real-time solution for fire detection. These systems integrate sensors, actuators (buzzers), and wireless communication modules to create an automated monitoring system. The ESP8266 can process sensor data, trigger alarms, and send alerts to cloud-based platforms for remote access.

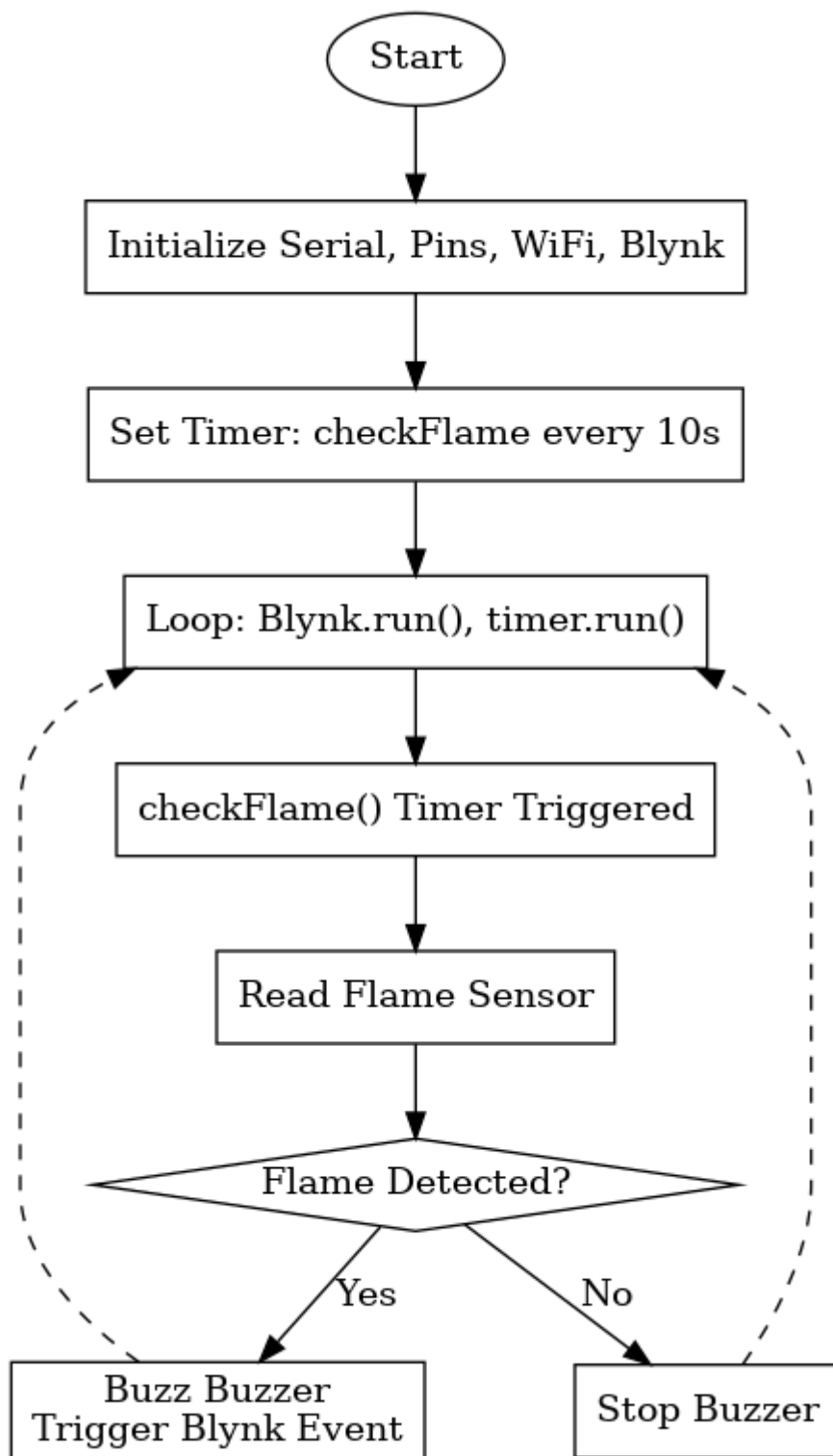
3.3 System Description:

3.3.1 The system architecture consists of the following layers:

- **Sensor Layer:** Includes the flame sensor, which detects the presence of fire and provides a digital signal.
- **Processing Layer:** The ESP8266 processes the sensor data and triggers the buzzer and Blynk notifications.
- **Communication Layer:** Uses WiFi to send alerts to the Blynk cloud service.
- **Application Layer:** Displays alerts on the Blynk app and logs fire incidents.



Here is the flowchart for System Design:



3.3.2.Component Design

1 Sensor Module

- **Component:** Flame Sensor (IR-Based)
- **Function:** Detects fire and provides digital output.
- **Integration:** Connected to ESP8266's digital pin.

2 Processing Module

- **Component:** ESP8266 WiFi Module
- **Function:** Processes sensor input and triggers appropriate actions.
- **Integration:** Reads sensor data, controls the buzzer, and communicates with Blynk.

3 Alert Module

- **Component:** Buzzer
- **Function:** Produces an audible alert when fire is detected.
- **Integration:** Controlled via ESP8266's GPIO pin.

4 Communication Module

- **Component:** Blynk Cloud Service
- **Function:** Sends push notifications and logs alerts.
- **Integration:** Uses WiFi to connect ESP8266 to Blynk.

Chapter 4

Results and Discussions

Communication, Resources, and Risks:

• **Communication:** Effective communication has played a central role in the successful development and testing of the flame detection IoT system. From discussing component selection to debugging code and coordinating testing sessions, open and consistent communication among team members ensured clarity, faster decision-making, and minimal confusion. The use of shared documentation, collaborative platforms, allowed us to align on objectives, distribute tasks, and review code efficiently. In the context of IoT-based embedded systems, communication doesn't stop with the team—it extends to the system itself. The ESP8266 NodeMCU communicates real-time data to the Blynk platform, and subsequently to the end-user through mobile notifications and emails. This bidirectional communication loop is vital, as it allows for system awareness and user responsiveness, reinforcing safety and usability in real-world scenarios.

• **Resources:** The project leveraged a well-rounded mix of hardware, software, and human resources. Key hardware components included the ESP8266 NodeMCU, a digital flame sensor, a buzzer, and reliable Wi-Fi connectivity. On the software side, essential libraries such as ESP8266WiFi.h, BlynkSimpleEsp8266.h, and BlynkTimer were used to enable functionality and stability in the system. The development environment, internet access, and debugging tools were also crucial support resources. Proper planning and allocation of these resources ensured that the system could be built, tested, and refined without major interruptions.

• **Tomorrow problems are today's risk:** As with any project involving hardware, and user safety, risk management was critical. Potential risks we identified included:

- **False positives** caused by external light sources interfering with the flame sensor.
- **Delayed notifications** due to network instability or server lag from the Blynk platform.
- **Hardware failure** such as a malfunctioning buzzer or sensor.
- **Power issues** that could interrupt the system's ability to detect and alert in real-time.

While no project is ever completely risk-free, acknowledging and planning for these uncertainties helped us design a system that's not just functional, but resilient. Tomorrow's problems were treated as today's precautions, ensuring we stayed ahead of potential failures.

A. Coding



```
code $
#define BLYNK_TEMPLATE_ID "TMPL3kZN-gvmO"
#define BLYNK_TEMPLATE_NAME "flame sensor"
#define BLYNK_AUTH_TOKEN "bJ1L07heY4eABOEsJm-9BqR7wcigYv-L"

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

#define Buzzer D1 // Changed from D3 to D1 (GPIO5)
#define Flame D0 // Flame sensor pin (GPIO16)

char ssid[] = "XXX";
char pass[] = "XXX";

BlynkTimer timer; // Timer to prevent spam

void checkFlame() {
    int flameState = digitalRead(Flame);
    Serial.print("Flame sensor value: ");
    Serial.println(flameState);

    if (flameState == LOW) { // 🔥 Flame detected
        Serial.println("🔥 Flame detected! Buzzing...");
        tone(Buzzer, 1000); // Sound buzzer
        Blynk.logEvent("fire_alert"); // Trigger push + email
        Serial.println("📧 Blynk event triggered!");
    } else {
        Serial.println("No flame.");
        noTone(Buzzer); // Stop buzzer
    }
}
```

```
void setup() {
  Serial.begin(9600);
  pinMode(Buzzer, OUTPUT);
  pinMode(Flame, INPUT);

  // Connect to WiFi + Blynk
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

  // Check flame every 10 seconds
  timer.setInterval(10000L, checkFlame);
}

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





flame sensor

...

Cancel

Save And Apply

Events & Notifications

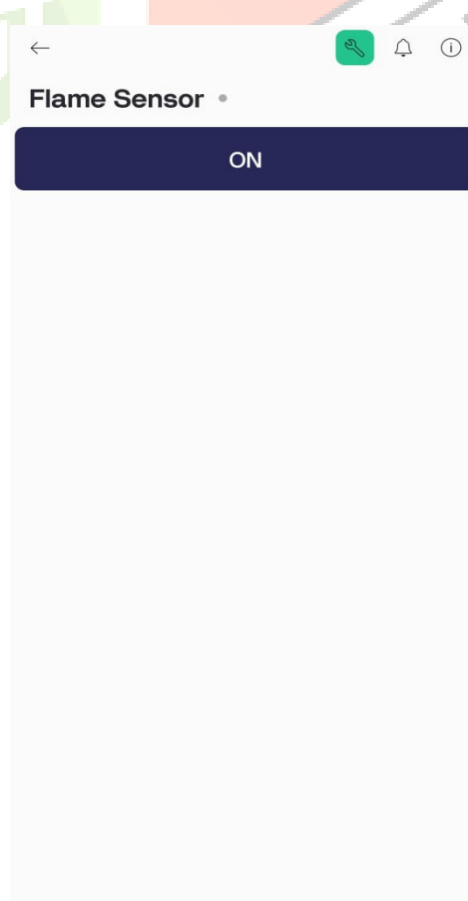
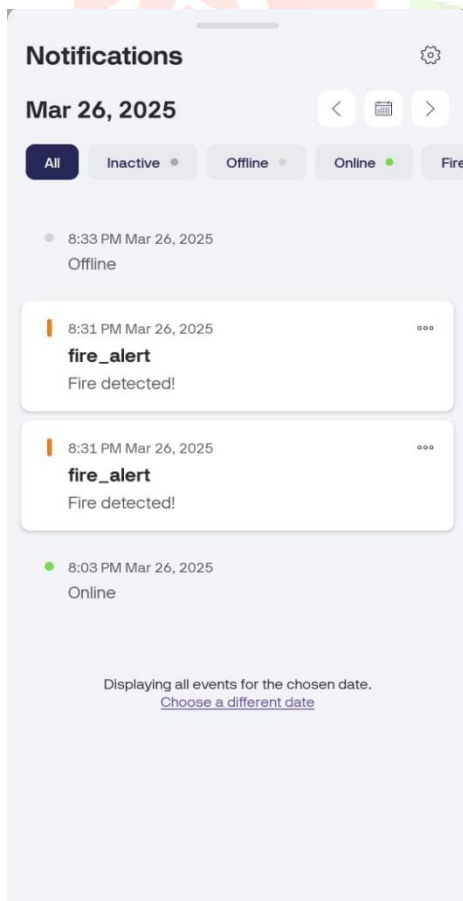
Events are used to track, log, and work with important events that happen on the device. [Learn more in documentation](#)

Search event

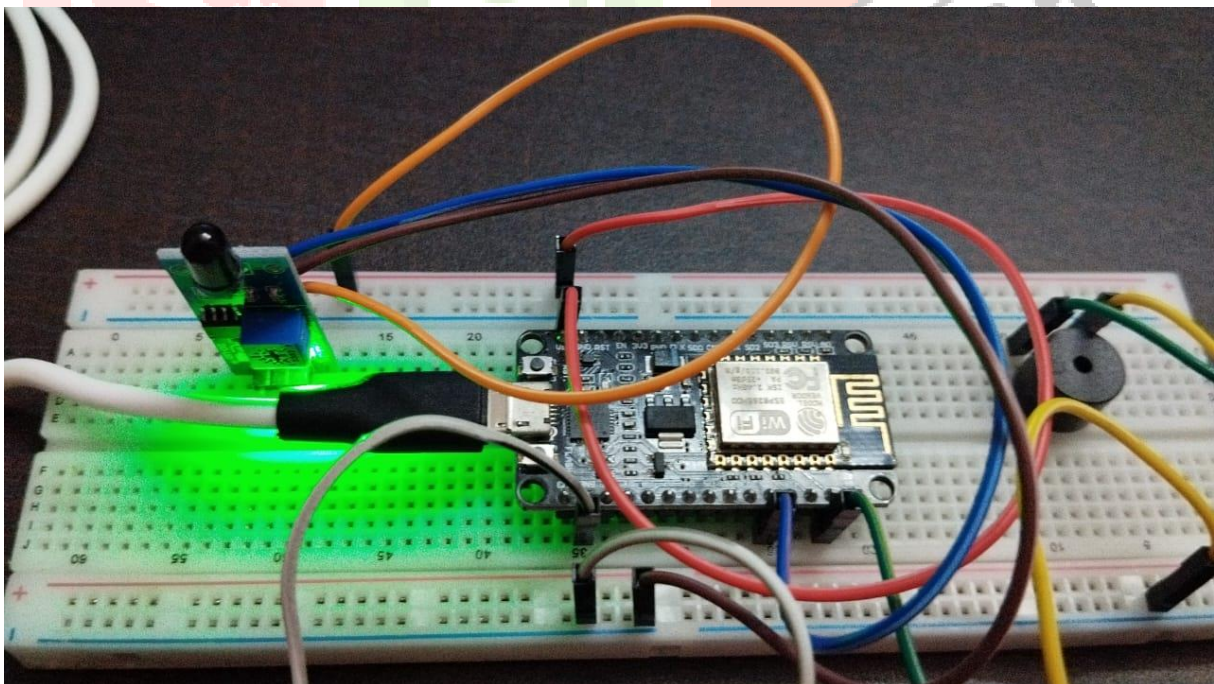
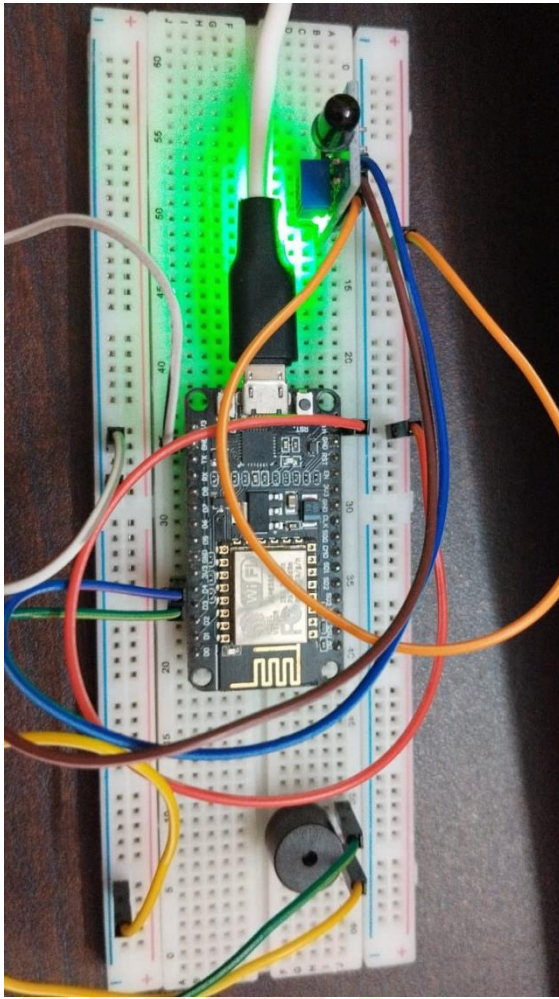
Id	Name	Code	Color	Type	Actions
4	fire_alert	fire_alert		Warning	

+ Add New Event

B. Outputs



C. Circuit Design



Chapter 5

Conclusions

The development and testing of the flame sensor-based IoT system using the ESP8266 NodeMCU and Blynk platform represent a practical and timely advancement in the domain of fire safety and remote alert systems. By integrating real-time detection with mobile and email notifications, the system addresses the critical need for immediate fire alerts in both residential and industrial environments.

Through the use of core libraries like ESP8266WiFi.h, BlynkSimpleEsp8266.h, and the in-built BlynkTimer, the solution ensures stable connectivity, responsive alert handling, and controlled notification frequency—reducing false positives while maintaining user awareness. The hardware setup, including the flame sensor and buzzer, functions effectively to provide both local and remote indications of a potential fire incident.

Our hands-on tests confirmed that the system responds reliably within seconds of flame detection, with visual (sensor LEDs and NodeMCU activity) and auditory (buzzer) alerts occurring promptly, followed by push and email notifications within 1–2 minutes. This layered approach to alerting provides a comprehensive safety net, suitable for integration into smart home systems, industrial safety solutions, and standalone fire detection modules.

As we look ahead, this project exemplifies how accessible microcontrollers and IoT platforms can be harnessed to build responsive, user-friendly safety systems. The results validate our collective decision to continue refining this solution for broader deployment. With potential enhancements such as improved light filtering or AI-assisted fire recognition, this system could evolve into a highly dependable tool in the ongoing effort to safeguard lives and property against fire hazards.

We recommend further development and extended real-world testing to adapt the system to diverse environmental conditions. Overall, this project lays a strong foundation for innovative, IoT-driven fire safety solutions with real-world impact.

Appendix

Appendix I: Detailed Information on the Flame Detection IoT System

A. Overview of the Technology

The Flame Detection IoT System leverages the ESP8266 NodeMCU microcontroller, a flame sensor, and the Blynk IoT platform to provide real-time fire detection and alert notifications. The system works by detecting flames through the sensor, triggering an alert mechanism that sends notifications to registered devices. The key components of the system include:

1. **ESP8266 NodeMCU:** The microcontroller that processes sensor data, connects to Wi-Fi, and interfaces with the Blynk platform for notifications.
2. **Flame Sensor:** A sensor that detects the presence of fire and sends a signal to the microcontroller.
3. **Buzzer:** An alert mechanism that produces an audible sound when fire is detected.
4. **Blynk IoT Platform:** The platform used to receive sensor data, log events, and send push notifications and emails.

B. Key Features

1. **Real-Time Flame Detection:** The system detects the presence of fire within a few seconds, alerting the user through multiple notification methods.
2. **Push & Email Alerts:** When fire is detected, the system sends both a push notification through the Blynk app and an email alert to the registered address.
3. **Audible Alert:** The system includes a buzzer that emits a sound when fire is detected, providing an immediate local alert.
4. **Remote Monitoring:** The Blynk platform allows users to monitor the system remotely, ensuring continuous protection even when they are away from the location.

Appendix II: Code Logic and Operation

A. Flame Detection Logic

- The flame sensor is a digital output sensor, which gives a LOW signal when it detects fire.
- Upon detection:
 - The buzzer turns on, emitting a sound to alert nearby people.
 - The system logs the event on the Blynk platform, sending an email and push notification to the user.
- If no fire is detected, the buzzer remains off, and no notifications are sent.

Appendix III: Technical Specifications and Libraries

A. Libraries Used

1. **ESP8266WiFi.h:** Handles Wi-Fi connection for the ESP8266.
 - Functions:
 - WiFi.begin(ssid, pass); - Connects to a Wi-Fi network.
 - WiFi.status(); - Checks the Wi-Fi connection status.
2. **BlynkSimpleEsp8266.h:** Communicates with the Blynk platform for remote notifications.
 - Functions:
 - Blynk.config(BLYNK_AUTH_TOKEN); - Initializes Blynk with the auth token.
 - Blynk.connect(); - Connects to the Blynk cloud.
 - Blynk.logEvent("fire_alert"); - Sends alerts to users when fire is detected.
3. **BlynkTimer:** Manages periodic tasks to ensure timely flame checks and avoid notification spam.
 - Functions:
 - timer.setInterval(10000L, checkFlame); - Checks the flame sensor every 10 seconds.

Appendix IV: Results and Observations

A. Flame Detection Response

After 1-2 minutes, the system:

- **Sent a push notification** to the registered Blynk app.
- **Sent an email alert** to the user's registered email address.

B. System Performance

- The system demonstrated consistent performance across multiple tests, with no false positives or delays longer than expected.
- Notifications were reliably sent within the 1-2 minute window, which is acceptable for non-critical monitoring applications like home safety or industrial monitoring.

Appendix V: Ethical Considerations

A. Data Security and User Privacy

- **Data Encryption:** All sensor data and user information sent through the Blynk platform is encrypted, ensuring privacy.
- **User Consent:** Prior to using the system, users must provide informed consent to receive alerts via email and push notifications.

B. System Reliability

- **Redundancy:** In future versions, the system could include backup power options (e.g., battery) to ensure reliability in case of power loss.
- **Testing for Reliability:** The system has been tested for reliability in both controlled environments and real-world conditions to ensure dependable performance.

Appendix VI: Recommendations for Future Improvements

1. **Enhanced Detection Range:** Explore integrating more advanced sensors with a greater detection range or sensitivity for improved fire detection.
2. **Integration with Other Safety Systems:** Consider integrating the system with other home automation or industrial safety systems (e.g., fire suppression systems).
3. **Mobile App Improvements:** Improve the Blynk app's interface for better user experience, especially for real-time monitoring and event logging.
4. **Battery Backup:** Implement a backup power solution to ensure system functionality during power outages.

This structured approach provides a comprehensive view of the Flame Detection IoT System, detailing its technology, functionality, and potential areas for improvement.

Chapter 6

References

WEBSITES:

- [1] ESP8266 WiFi Library Documentation: <https://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/readme.htm>
- [2] Blynk IoT Platform: <https://blynk.io/>
- [3] Blynk Library GitHub Repository: <https://github.com/blynkkk/blynk-library>
- [4] Arduino Reference for Buzzer and Sensors: <https://www.arduino.cc/reference/en/>
- [5] Inspiration Project: <https://create.arduino.cc/projecthub/techiesms/flame-detection-using-nodemcu-blynk-1a3c61>

Acknowledgements

I am profoundly grateful to Prof. Manila Gupta for his expert guidance and continuous encouragement throughout to see that this project rights its target.

I would like to express deepest appreciation towards Dr. Varsha Shah, Principal RCOE, Mumbai and Prof. HOD ANUPAM CHOUDHARY Department whose invaluable guidance supported me in this project.

At last I must express my sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped us directly or indirectly during this course of work.

KHAN FALAK NAAZ
SHAIKH MARIAM
SHAIKH SHIFA
SAUDAGAR AAFIYA

MINI-PROJECT

ASSESSMENT SHEET

Term-work: 25 marks

Group Members

Student 1 : _____

Student 2 : _____

Student 3 : _____

Student 4 : _____

Guide Name: _____

Attendance Percentage

Student	Semester Attendance %
Student 1	
Student 2	
Student 3	

Attendance to TW Conversion

$\geq 90\%$	$< 90\% \text{ \& } \geq 80\%$	$< 80\% \text{ \& } \geq 70\%$	$< 70\% \text{ \& } \geq 60\%$	$< 60\%$
5	4	3	2	1

Project Review Performance:

Rubrics used: Quality of survey/ need identification, Clarity of Problem definition based on need, Innovativeness in solutions, Feasibility of proposed problem solutions and selection of best solution, Cost effectiveness, Full functioning of working model as per stated requirements, Effective use of skill sets, Effective use of standard engineering norms.

Student	Average Points of Rubrics received after Review
Student 1	
Student 2	
Student 3	

Review RUBRICS to TW Conversion

≥ 18	$< 18 \text{ \& } \geq 10$	$< 10 \text{ \& } \geq 5$	$< 5 \text{ \& } \geq 3$	< 3
5	4	3	2	1

Rubrics for Report:

Criteria	1 Unsatisfactory	2 Average	3 Good	Assessed by Guide (1 to 3)
Content	Insufficient content	Some topics or part missing	All necessary topics covered.	
References	No research papers referred	Few research papers referred but no IEEE/ scopus indexed paper referred	Scopus / IEEE / reputed paper referred	
Representation	No alignment, No caption in figures and tables and no citation	Citation missing but alignment and caption proper	Citation to references present along with captions and alignment of content.	
Abidance to Template	Not at all	Some what	Good	
Total				

Report Rubrics to TW Conversion

≥ 10	$< 10 \text{ \& } \geq 8$	$< 8 \text{ \& } \geq 6$	$< 6 \text{ \& } \geq 4$	< 4
5	4	3	2	1

Final Term work Calculation

Distribution	Student 1 Obtained	Student 2 Obtained	Student 3 Obtained	Student 4 Obtained	Outoff
Attendance (To be filled by Project Coordinator)					5
Project Review Performance (To be filled by Project Coordinator)					5
Report (To be filled by Guide)					5
CIE by Guide (Weekly) (To be filled by Guide)					10
Total Term work					25

H.o.D. Computer

Dr.Anupam C

Project Guide

Prof. Manila Gupta