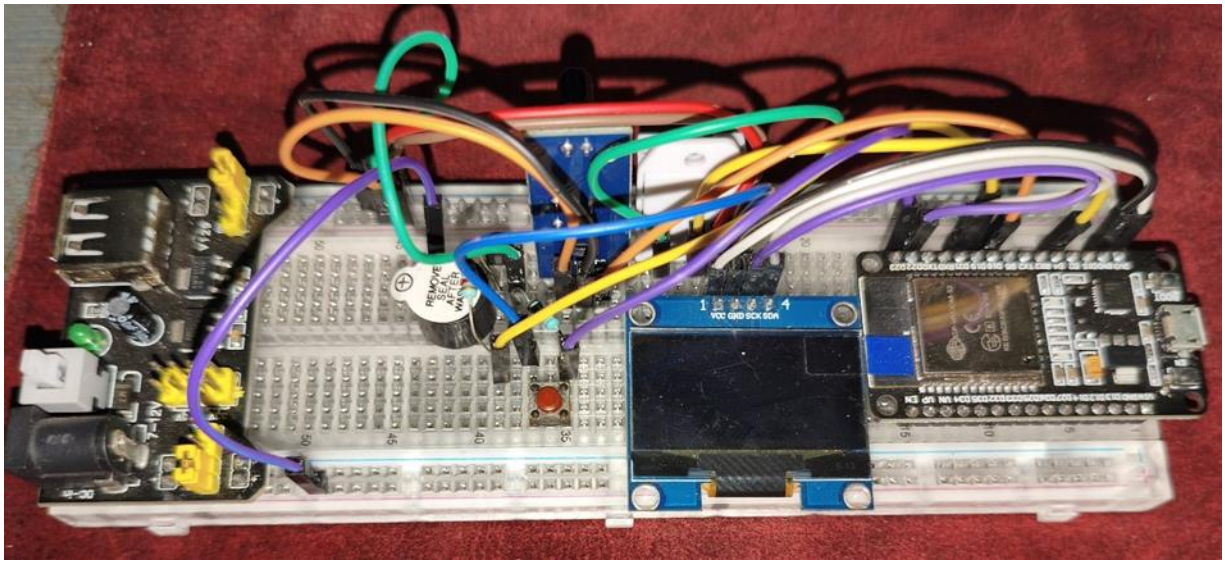


FIRE THERMOMIST

PREVENT DETECT SAVE

(A Real-time Fire and Environmental Monitoring System Using ESP32)

PROJECT BY:

Abhishek Vishwakarma

BACHELOR OF ENGINEERING

(Computer Science & Engineering)

SESSION: 2024-2025

PROJECT COMPLETION CERTIFICATE

This is to certify that the project titled **“Fire Thermomist”** has been personally developed and completed by **Abhishek Kumar Vishwakarma**, B.Tech student, as a self-initiated learning and development project.

The work presented is original and not submitted as part of any academic curriculum. All development, coding, and testing were carried out independently for skill enhancement and personal growth.

Date: 24/05/2025

Signature:

(Abhishek Kumar Vishwakarma)

ABSTRACT

Fire Thermomist is an innovative IoT-based fire and environmental monitoring system designed to detect early signs of fire hazards and monitor atmospheric conditions in sensitive environments. Built around the **ESP32 WROOM32 DevKit V1**, this self-initiated project integrates a **DHT22 sensor** for real-time temperature and humidity tracking, and a **flame sensor** for immediate fire detection. The system outputs visual feedback through a **1.3-inch OLED display**, complemented by a **buzzer alert** system to notify users during fire conditions.

The OLED screen showcases a three-stage animated boot sequence followed by real-time weather animations based on sensor readings. In the event of fire detection, the system triggers both an audible alert and a visual warning through the display. These animations are achieved by breaking GIF files into multiple frames, which are then rendered sequentially on the OLED—providing a more user-friendly and intuitive experience, even on resource-limited hardware.

The primary goal of this system is to enhance safety in locations such as **server rooms, laboratories, or storage facilities**, where even minor environmental changes can lead to major hazards.

What sets this project apart is its complete independence from institutional backing. Designed, coded, and assembled entirely as a **personal learning initiative**, Fire Thermomist reflects the power of self-driven innovation. This project combines electronics, programming, and embedded system design to deliver a low-cost, effective safety tool—ready to be scaled or customized for various real-world applications.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to everyone who directly or indirectly supported me throughout the development of this project, **Fire Thermomist**. This project was entirely self-initiated, driven by my interest in electronics, embedded systems, and real-world safety applications.

First and foremost, I am thankful to the online open-source community and creators whose tutorials, libraries, and forums provided valuable guidance in understanding and implementing the ESP32 microcontroller, DHT22 sensor, OLED display, and flame detection modules. Their collective contributions played a crucial role in shaping my technical knowledge and debugging capabilities.

Lastly, this project has been a significant learning experience, and I acknowledge the countless hours of experimentation, failure, and success that went into building it. It has not only improved my technical skills but also taught me the value of patience, persistence, and independent learning.

This acknowledgment would be incomplete without appreciating the inspiration and curiosity that led me to start and finish this journey on my own terms.

Date: 24/05/2025 Abhishek Vishwakarma

Place: Lucknow 202310101150305

TABLE OF CONTENT

LIST OF ABBREVIATION

Abbreviation	Full Form
IoT	Internet of Things
ESP32	Espressif Systems Protocol 32-bit Microcontroller
MCU	Microcontroller Unit
OLED	Organic Light Emitting Diode
DHT22	Digital Humidity and Temperature Sensor (Model 22)
VCC	Voltage Common Collector (Power Supply Pin)
GND	Ground
GPIO	General Purpose Input/Output
IDE	Integrated Development Environment
GIF	Graphics Interchange Format
PWM	Pulse Width Modulation
API	Application Programming Interface (if used later)
°C	Degrees Celsius
RH	Relative Humidity
USB	Universal Serial Bus
LED	Light Emitting Diode

REQUIREMENTS

Hardware Requirements

S.No	Component	Quantity	Description
1.	ESP32 WROOM32 DevKit V1	1	Main microcontroller board with Wi-Fi and GPIO support
2.	DHT22 Sensor	1	For temperature and humidity monitoring
3.	Flame Sensor	1	For detecting fire or flames
4.	1.3" OLED Display (I2C)	1	For visual feedback and animations
5.	Buzzer	1	For sound alerts in case of fire detection
6.	Resistors (as needed)	–	For signal conditioning (if required)
7.	Breadboard / PCB	1	For connecting components
8.	Jumper Wires	As needed	For wiring and connectivity
9.	Micro USB Cable	1	For power and programming
10.	Power Supply (5V via USB)	1	To power the ESP32 board

Software Requirements

S.No	Tool / Software	Purpose
1.	Arduino IDE	Writing, compiling, and uploading code
2.	ESP32 Board Package	ESP32 support in Arduino IDE
3.	DHT Sensor Library	To read DHT22 sensor data
4.	Adafruit SSD1306 Library	For OLED display communication
5.	Adafruit GFX Library	For drawing on OLED
6.	GIF to Frame Converter Tool	For converting GIFs into displayable frames
7.	Serial Monitor (built-in)	For debugging and output monitoring

Chapter 1: Introduction

1.1 Objective of the Project

The objective of the Fire Thermomist project is to develop an IoT-based fire and environmental monitoring system capable of detecting early signs of fire hazards while continuously tracking ambient temperature and humidity levels. This system aims to provide real-time alerts using visual and audio outputs, making it suitable for sensitive environments such as server rooms and labs.

1.2 Motivation and Purpose

With the growing need for smart safety solutions in both commercial and personal spaces, the motivation behind this project was to build an affordable, efficient, and self-reliant system that can detect fires and hazardous environmental changes. The purpose is also educational—to enhance practical skills in embedded systems, IoT, and microcontroller programming.

1.3 Scope of the Project

The scope of Fire Thermomist includes monitoring temperature and humidity, detecting fire or flame presence, displaying sensor data and alerts on an OLED screen, and producing sound alerts through a buzzer. While currently designed as a prototype, it can be further scaled and integrated with cloud services for remote monitoring.

1.4 Applications

- Server rooms
- Laboratories
- Data centers
- Warehouses
- Homes and apartments
- Electrical equipment enclosures

Chapter 2: Literature Review

2.1 Existing Systems and Limitations

Several fire detection and environmental monitoring systems are commercially available today. Most traditional fire alarms rely on smoke detection or heat sensors. While effective, these systems often lack real-time environmental data tracking and remote monitoring features. Additionally, they can be expensive and difficult to install in compact or temporary setups.

More advanced systems include IoT-enabled detectors that send alerts via the internet or mobile networks. However, such systems may still have limitations such as high cost, proprietary platforms, and dependence on external cloud infrastructure for operation.

2.2 Comparison with Fire Thermomist

The Fire Thermomist project offers a compact, affordable, and self-contained alternative. Unlike many existing solutions, it uses an ESP32 microcontroller that supports both sensor interfacing and wireless communication. The inclusion of temperature and humidity monitoring (via DHT22), flame detection, OLED display for visual feedback, and buzzer alerts provides a more comprehensive safety solution.

Moreover, the animations and real-time visual alerts enhance user experience and usability. Being open-source and customizable, Fire Thermomist can be adapted for various applications and future extensions such as mobile integration or cloud logging, setting it apart from closed-system commercial devices.

Chapter 3: System Overview

3.1 Block Diagram

The Fire Thermomist system consists of a central ESP32 microcontroller unit (MCU) that interfaces with multiple sensors including the DHT22 for temperature and humidity, and a flame sensor for fire detection. An OLED display module is connected for visual output of real-time data and alert animations, and a buzzer provides audible alerts upon fire detection.

3.2 System Architecture

The architecture follows a modular design where sensor inputs are continuously monitored by the ESP32 MCU. The MCU processes sensor data to determine environmental status and potential fire presence. Based on sensor readings, it updates the OLED display with current environmental conditions or alert animations and triggers the buzzer if a fire is detected.

3.3 Hardware and Software Summary

Hardware components include the ESP32 DevKit V1 board, DHT22 temperature and humidity sensor, flame sensor, OLED display, and buzzer. Software development was carried out using the Arduino IDE with libraries for sensor interfacing and OLED graphics. The firmware integrates sensor data processing, animation control, and alert mechanisms to form a cohesive fire monitoring system.

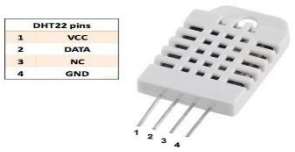
Chapter 4: Components Used

4.1 Hardware Components

1. ESP32 WROOM32 DevKit V1

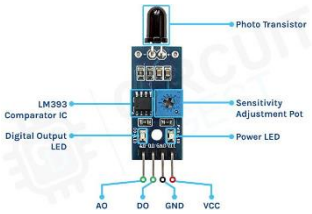
- Main microcontroller board based on ESP32 chip.
- Features Wi-Fi, Bluetooth, and multiple GPIO pins for sensor interfacing.
- Responsible for processing sensor data, controlling display, and generating alerts.

2. DHT22 Sensor



- Digital temperature and humidity sensor.
- Provides accurate readings of ambient environmental conditions.
- Communicates with ESP32 via a single data pin.

3. Flame Sensor



- Detects presence of fire or flame based on infrared light intensity.
- Outputs digital signal to ESP32 indicating fire detection.

4. OLED Display (1.3 inch, I2C interface)

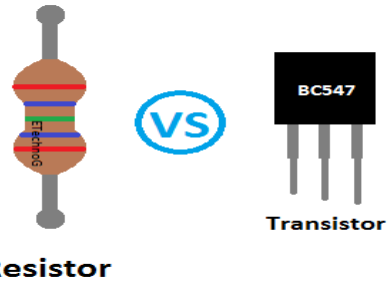
- Small organic LED display for showing real-time temperature, humidity, and alert animations.
- Communicates with ESP32 via I2C protocol.
- Supports custom animations by displaying frames converted from GIFs.

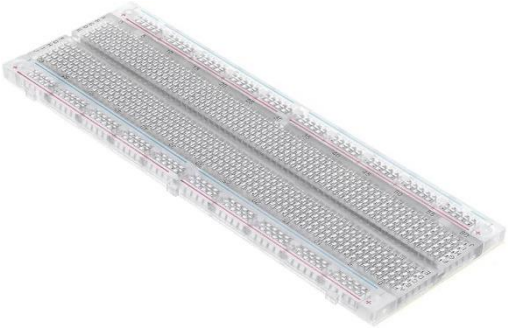
5. Buzzer

- Provides audible alerts when fire is detected.
- Controlled by ESP32's digital output pins.

6. Supporting Components

- Jumper wires for connections.
- Breadboard or PCB for mounting components.
- Resistors as needed for signal conditioning.
- Power supply via USB cable.





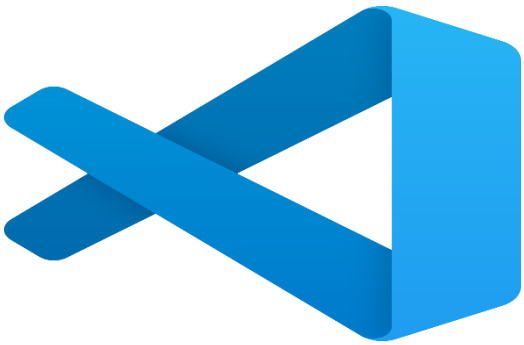
4.2 Software Components

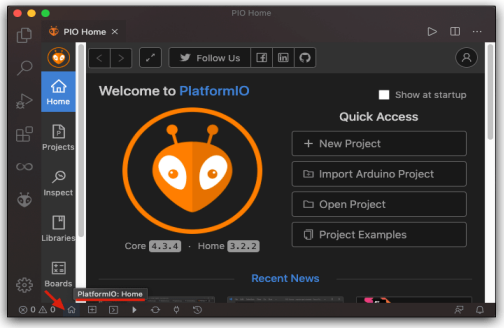
1. Arduino IDE

- Development environment used for coding, compiling, and uploading firmware to ESP32.

2. Libraries Used

- **DHT Sensor Library:** For interfacing and reading data from DHT22 sensor.
- **Adafruit SSD1306 Library:** For controlling OLED display.
- **Adafruit GFX Library:** Provides graphics primitives for OLED display.
- Additional utility code for GIF frame handling and animation.

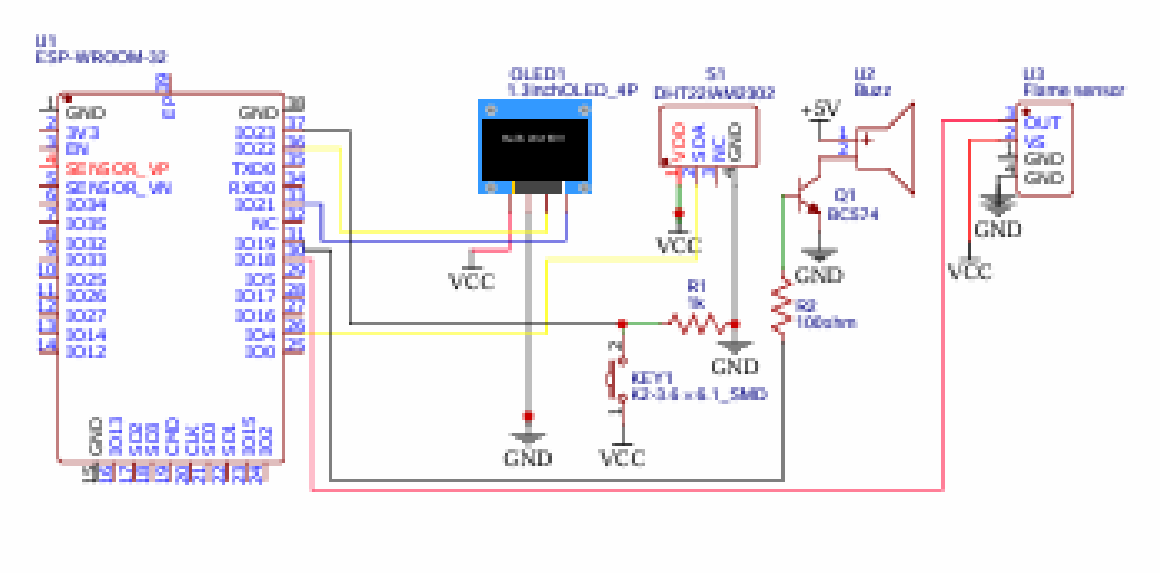




Chapter 5: Circuit Design and Implementation

5.1 Circuit Diagram

The overall circuit diagram for the Fire Thermomist project is shown below. It illustrates the connections between the ESP32 microcontroller, sensors, OLED display, and buzzer. Pin numbers are marked as XX for later editing.



Schematic_Fire-Thermomist_2025-05-24pdf.pdf

5.2 Pin Configuration

Component	ESP32 Pin (GPIO No.)	Notes
DHT22 Sensor	GPIO 04	Data pin connected to GPIO 04
Flame Sensor	GPIO 18	Digital output to GPIO 18
OLED Display	GPIO 21 (SDA), GPIO 22 (SCL)	I2C interface pins
Button	GPIO 23	Button sense pin
Buzzer	GPIO 19	Digital output to GPIO 19

5.3 Hardware Connections

- The DHT22 sensor is connected to the ESP32 digital GPIO pin with power and ground lines properly connected.
- The flame sensor outputs a digital signal to the ESP32 to indicate flame presence.
- The OLED display is connected through I2C pins (SDA and SCL) for communication and powered with 3.3V and ground.
- The buzzer is connected to a GPIO pin configured as digital output to generate sound alerts.

Chapter 6: Software and Coding

6.1 Tools and IDE Used

The firmware for the Fire Thermomist system was developed using the Arduino IDE. This environment provides easy code compilation and uploading support for the ESP32 microcontroller.

6.2 Code Structure

The program is structured into modular functions for sensor reading, display management, fire detection logic, and alert handling. The main loop continuously reads sensor data and updates outputs accordingly.

6.3 Animation Implementation

Animations for boot sequence, weather display, and fire alerts were created by converting GIF frames into bitmap arrays. These frames are displayed sequentially on the OLED screen to provide smooth visual feedback.

6.4 Fire Detection Logic

The flame sensor output is continuously monitored. When a flame is detected, the system triggers an immediate alert by activating the buzzer and switching the OLED display to fire alert animations. Temperature and humidity readings are also used to ensure environmental conditions are within safe limits.

6.5 Challenges Faced in Coding

- Implementing smooth animation on a limited OLED display memory required frame optimization.
- Ensuring real-time responsiveness while managing multiple sensor inputs and display updates was challenging.
- Debugging sensor reading inconsistencies due to environmental noise.
- Integrating buzzer alerts without blocking the main program flow.

Chapter 7: Working and Output

7.1 Boot and Start up Process

On powering up, the system runs a boot animation on the OLED display. This sequence includes a 3-stage animation showing system initialization, followed by a weather animation that displays current temperature and humidity readings.

7.2 Real-time Monitoring

Once initialized, the system continuously reads data from the DHT22 sensor and flame sensor. Temperature and humidity values are updated on the OLED display in real time, along with appropriate weather animations.

7.3 Fire Alert Scenario

If the flame sensor detects a fire, the system immediately switches to fire alert mode. The buzzer sounds continuously, and the OLED display shows fire detection animations to alert nearby users.

7.4 Sample Outputs

- Real-time temperature and humidity readings displayed with animated icons.
- Fire detection animation showing flames and alert symbols on OLED.
- Buzzer alert audible during fire detection.

Chapter 8: Conclusion and Future Scope

8.1 Summary of the Project

The Fire Thermomist project successfully created an IoT-based fire detection and environmental monitoring system. Using ESP32, DHT22 sensor, flame sensor, OLED display, and buzzer, it provides real-time monitoring of temperature, humidity, and flame detection. The system features animated displays and audible alerts, making it user-friendly and effective for sensitive areas such as server rooms and labs.

8.2 Limitations

- Flame detection range is limited to close proximity.
- No remote monitoring or mobile alerts implemented.
- Dependent on continuous power supply via USB; no battery backup.
- OLED display size limits the amount of information and animation complexity.

8.3 Future Enhancements

- Integration of Wi-Fi/Bluetooth for remote alerts and data logging.
- Adding a backup battery for uninterrupted operation.
- Incorporation of additional sensors like smoke and gas detectors.
- Development of a mobile app or cloud dashboard for real-time monitoring.

Chapter 9: References

9.1 Web Links

- Official ESP32 Documentation: https://www.espressif.com/en/products/socs/esp32
- DHT22 Sensor Datasheet and Usage: https://learn.adafruit.com/dht
- Flame Sensor Overview: https://lastminuteengineers.com/flame-sensor-arduino-tutorial/
- OLED Display with ESP32: https://www.arduino.cc/en/Tutorial/OLEDDisplay

9.2 Libraries Used

- DHT Sensor Library by Adafruit
- Adafruit SSD1306 OLED Library
- Arduino Core for ESP32

9.3 Tutorials and Communities

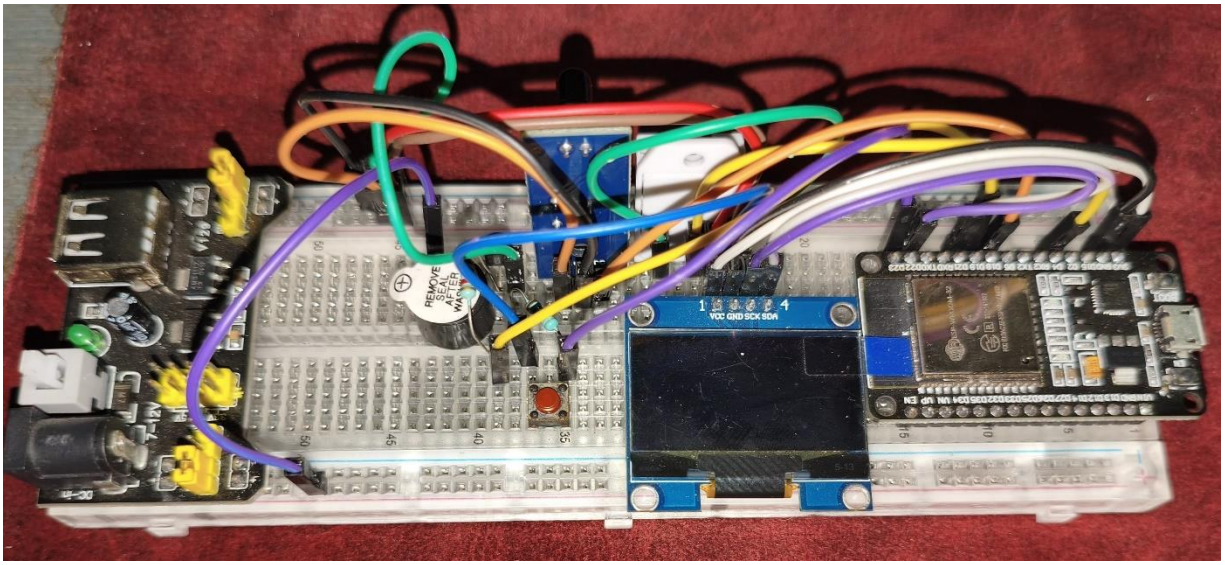
- Arduino Forum: https://forum.arduino.cc/
- ESP32 Community on GitHub: https://github.com/espressif/arduino-esp32
- Stack Overflow (Embedded Systems): https://stackoverflow.com/questions/tagged/esp32

Chapter 10: Appendix

A.1 Complete Source Code

You will find on my GitHub: https://github.com/AbhiVish6386

A.2 Additional Images



THANK YOU !

By: Abhishek Vishwakarma

B.tech CSE (DS+AI) 2nd Year.