FIRE DETECTION IN EV VEHICLES USING IOT

A MINI PROJECT REPORT

Submitted by

(1) Saabika Roshni S (312422106136)
(2) Sahana S (312422106138)
(3) Samyuktha B (312422106141)
(4) Subashri M (312422106161)
(5) Sushmitha V (312422106165)

In partial fulfilment of co-curricular activities organized by

ST. JOSEPH'S INSTITUTE OF TECHNOLOGY
OMR, CHENNAI-600119

CONTENTS

1.Introduction:

Electric vehicles (EVs) have gained significant popularity in recent years due to their environmentally friendly nature and potential to reduce greenhouse gas emissions. However, the lithium-ion batteries used in EVs pose a risk of thermal runaway, which can lead to battery fires or explosions. These incidents not only pose a safety hazard to passengers but also raise concerns about the widespread adoption of EVs.

This project aims to design and implement an IoT-based fire detection system specifically tailored for EVs. The proposed system will incorporate a network of strategically placed sensors capable of monitoring critical parameters such as temperature, smoke within the battery compartment and other vital components of the EV.

2.IOT Design:

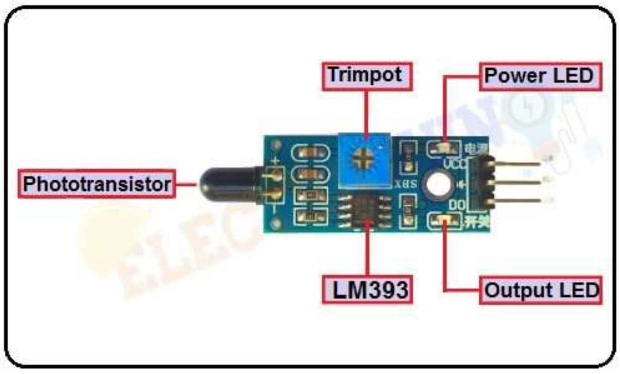
IoT systems combine physical and digital components that collect data from physical devices and deliver actionable, operational insights. These components include: physical devices, sensors, data extraction and secured communication, gateways, cloud servers, analytics, and dashboards.

There are two mainly components available namely hardware and software components in the Iot based fire detection System.

2.1. Hardware Components:

- ❖ Sensor - YG1006
- ❖ Module – NodeMCU
- ❖ Wi-Fi Module – ESP8266

2.1.1 Sensor – YG1006:



YG1006 sensor is a high speed and highly sensitive NPN silicon phototransistor. This Sensor can be used to detect fire source or other light sources of the wave length in the range of 760nm - 1100 nm. Due to its black epoxy, the sensor is sensitive to infrared radiation. It can be used as flame Sensor for Smart Car. Flame sensor is the most sensitive to ordinary light that is why its reaction is generally used as flame alarm purposes.

2.1.1.1. Specification:

➢ The range of operating voltage ranges from 3.3V to 5V.

➢ The operating current is 15 mA.

➢ The comparator chip used is LM393.

➢ The type of sensor is YG1006 Photo Transistor.

➢ Operating temperature ranges from -25°C to 85°C.

➢ The output type is Digital o/p or Digital & Analog output.

➢ Red LED is for power and green LED is for output.

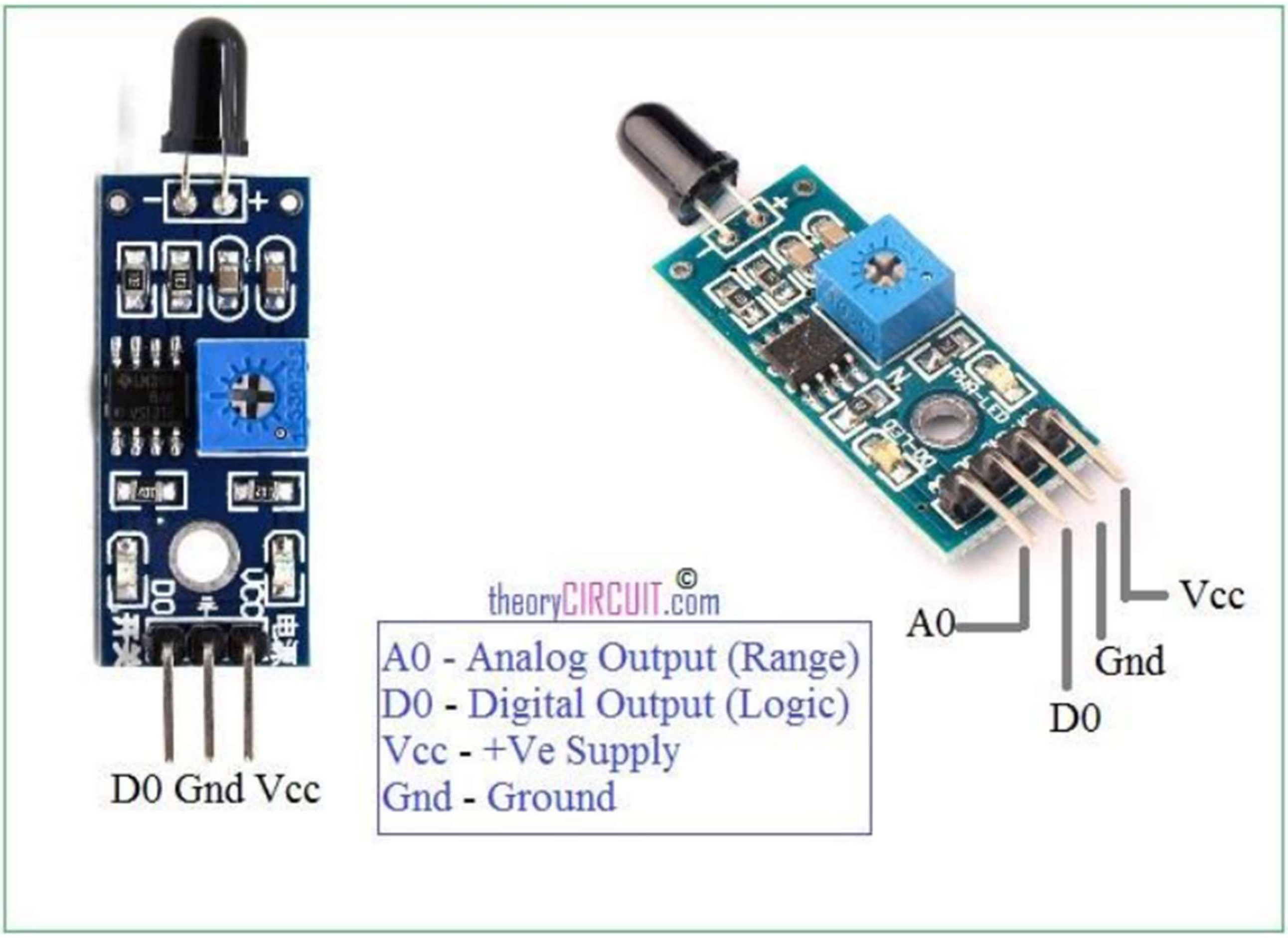
➢ The detection angle is from 0 to 60 degrees.

2.1.1.2Working Principle:

The sensors in the flame detector will detect the radiation that is sent by the flame the photoelectric converts the radiant intensity signal of the flame to a relevant voltage signal and this signal would be processed in a single chip microcomputer and converted into a desired output. Flame Sensors use UV (Ultra Violet) or IR(Infrared) or UV-IR Technology io identify flames below a second.

These sensors react to a detected flame based on the installation, although it includes sounding an alarm, disabling a fuel line & activating a fire control system.The flame sensor with UV technology works by simply sensing the UV radiation. Generally, all fires generate UV radiation at the ignition point so, in case of a fire, the sensor would become alert of it & generate a series of pulses that are changed by detector electronics and gives an alarm output.

IR sensors like all other photosensor work on the principle that a photon of sufficient energy can knock out electrons so that the resistance of the circuit is changed. The Flame sensor with IR technology works by checking the IR Spectrum band for particular ornamentation that hot gases emit. But, this kind of device needs a flickering movement of the flame.



2.1.1.3. Advantages:

- These sensors have High-speed response.

- These sensors are resistant to fake alarms.
- This sensor frequently & more accurately responds faster as compared to heat or smoke sensor because of the mechanisms it utilizes for detecting the flame.
- It has a long detection distance; environmental adaptability is good & high reliability.

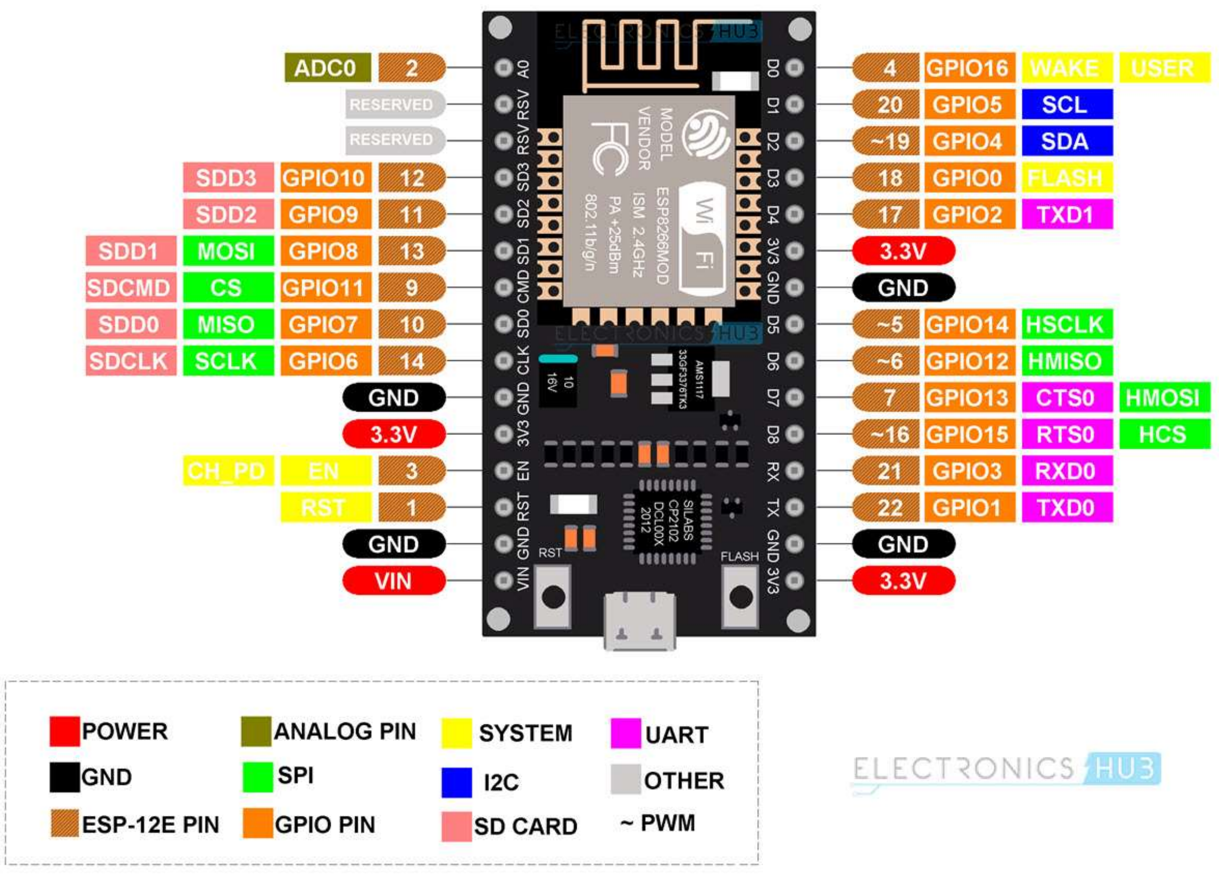
2.1.1.4. Applications:

- A flame-sensor is mainly used to detect & react to the occurrence of a flame/fire.
- Flame sensors are used in fire alarms, fire detection, drying systems, firefighting robot, industrial heating, hydrogen stations, domestic heating systems, industrial gas turbines, gas-powered cooking devices, etc.
- These are used in MDF factories, pharmaceuticals, fume cupboards, coal handling, spray booths, nuclear industry, fabrication of metal, clothing dryers, aircraft hangars, gas fuelled cookers, domestic heating systems, heating & drying systems in industries, generators and storage tanks.

2.1.2 Module - NodeMCU:

NodeMCU (**Node Microcontroller Unit**) is an open-source firmware and development kit designed for the Internet of Things (IoT) applications. It is based on the ESP8266 Wi-Fi module, which integrates a microcontroller unit (MCU) and Wi-Fi connectivity capabilities. NodeMCU provides a platform for building IoT projects and prototypes quickly and easily.

Through its pins we can read inputs - light on a sensor, a finger on a button, or a Twitter message -and turn them into an output - activating a motor, turning on an LED, publishing something online. It has also WiFi capabilities, so we can control it wirelessly and make it work on a remote installation easily! We can tell our board what to do by sending a set of instructions to the microcontroller on the board. To do so we can use the Arduino Software IDE.



2.1.2.1 Specification:

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB

- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna
- Small Sized module to fit smartly inside your IoT projects.

2.1.2.2 Working Principle:

NodeMCU, like other microcontroller-based development boards, follows a typical workflow for development and operation. Here's a general overview of how NodeMCU works:

1. **Setup**: To begin working with NodeMCU, you first need to set up your development environment. This usually involves installing the necessary drivers for the NodeMCU board and choosing a programming language or environment. As mentioned earlier, NodeMCU supports programming in Lua or using the Arduino IDE.

2. **Programming**: Once your development environment is set up, you can start writing code for your NodeMCU project. Depending on your chosen programming language, you'll write scripts or sketches to define the behavior of your project. This could involve tasks such as connecting to a Wi-Fi network, reading sensor data, controlling actuators, or communicating with other devices over the internet.

3. **Flashing Firmware**: If you're using Lua, you'll typically upload your Lua scripts directly to the NodeMCU board. If you're using the Arduino IDE, you'll write your code in the IDE, compile it, and then upload the compiled binary to the NodeMCU board via USB.

4. **Execution**: Once the firmware is uploaded to the NodeMCU board, it will begin executing the code you've written. This could involve tasks such as connecting to a Wi-Fi network, periodically reading sensor data, responding to incoming requests, or controlling connected devices.

5. **Interfacing with Hardware**: NodeMCU provides GPIO pins that allow you to interface with various hardware components such as sensors, actuators, displays, and more. You can use these GPIO pins to read sensor data, control LEDs or motors, or communicate with other devices using protocols like I2C, SPI, or UART.

6. **Networking**: One of the key features of NodeMCU is its built-in Wi-Fi connectivity. This allows your NodeMCU board to connect to local Wi-Fi networks, as well as communicate with other devices or servers over the internet. You can use this capability to send data to cloud services, control devices remotely, or receive updates and commands from other devices.

7. **Feedback and Iteration**: As your project progresses, you'll likely need to test and iterate on your code to fine-tune its behavior and fix any issues that arise. NodeMCU supports serial communication, which allows you to send debugging messages and receive feedback from the board via the USB interface.

2.1.2.3 APPLICATIONS:

The NodeMCU board is widely used for various IoT applications due to features and simplicity. Here are some common uses of NodeMCU:

- **IoT Prototyping**: NodeMCU is popular for rapid prototyping of IoT projects. Its built-in Wi-Fi connectivity allows developers to quickly connect the board to the internet and communicate with cloud services, sensors, actuators, and other IoT devices. It simplifies the development process by providing a convenient platform for experimenting and testing ideas.
- **Sensor Networks**: NodeMCU can serve as a hub or node in a wireless sensor network. It can collect data from various sensors such as temperature, humidity, motion, light, and send it to a centralized server or cloud platform for analysis and monitoring. This enables applications like environmental monitoring, agriculture automation, and industrial sensor networks.
- **Internet Connectivity for Devices**: NodeMCU can provide internet connectivity to devices that lack built-in Wi-Fi capabilities. By interfacing the NodeMCU board with a microcontroller or other devices, you can enable them to connect to the internet, access web services, and communicate with other devices over Wi-Fi.
- **Educational Purposes:** NodeMCU is a beginner-friendly platform for learning about IoT, programming, and electronics. Its simplicity, extensive

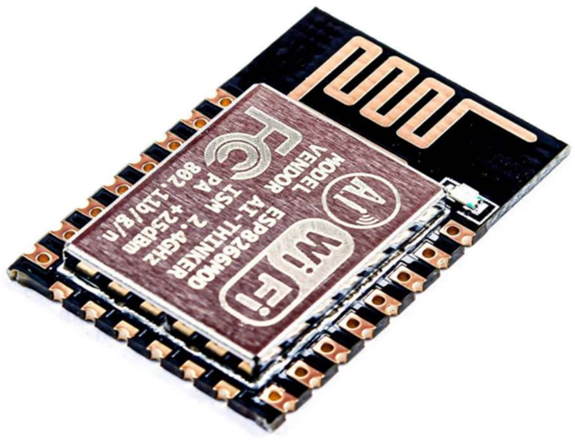
documentation, and large community support make it suitable for educational institutions and individuals who want to understand and experiment with IoT concepts.

- **DIY Projects**: NodeMCU is often used in do-it-yourself (DIY) projects that involve IoT functionality. Whether you want to build a weather station, a smart pet feeder, a remote-controlled robot, or a personal assistant, NodeMCU provides a flexible and affordable platform to bring your ideas to life.
- **IOT Gateway**: NodeMCU can act as an IoT gateway, serving as a bridge between IoT devices and the cloud. It can aggregate data from multiple devices, process it, and transmit it to cloud platforms for storage and analysis. This enables centralized control, monitoring, and management of IoT deployments.

Overall, NodeMCU is a versatile development board that simplifies IoT prototyping and enables a wide range of IoT applications. Its built-in Wi-Fi capability, programmable microcontroller, and compatibility with theArduino ecosystem make it a popular choice among hobbyists, students, and professionals working on IoT projects.

2.1.3 Wi-Fi Module – ESP8266

The ESP8266 Wi-Fi Module is a self-contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your Wi-Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor.



2.1.3.1ESP8266 FUNCTIONS:

- ESP8266 has many applications when it comes to the IoT. Here are just some of the functions the chip is used for:

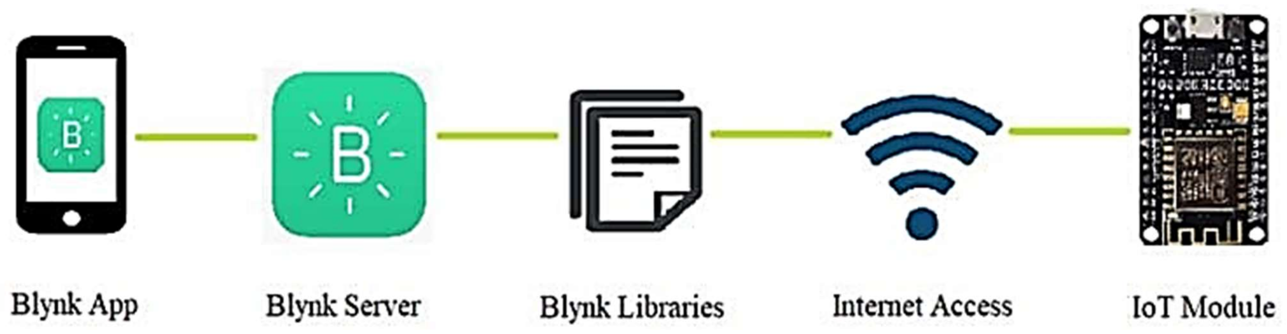
- **Networking:** The module's Wi-Fi antenna enables embedded devices to connect to routers and transmit data
- **Data Processing:** Includes processing basic inputs from analog and digital sensors for far more complex calculations with an RTOS or Non-OS SDK
- **P2P Connectivity:** Create direct communication between ESPs and other devices using IoT P2P connectivity.
- **Web Server:** Access pages written in HTML or development languages.

2.1.3.2. ESP8266 APPLICATIONS:

The ESP8266 modules are commonly found in the following IoT devices:

- Smart Security devices, including surveillance cameras and smart locks
- Smart energy devices, including HVACs and thermostats
- Smart industrial devices, including Programmable Logic Controllers (PLCs)
- Smart Medical devices, including wearable health monitor

2.2. Software Components:



- ❖ Blynk IoT Platform
- ❖ Arduino IDE
- ❖ Blynk IoT mobile App

2.2.1 Blynk IoT Platform:

The Project involves setting up a system to control ESP8266 microcontroller using Blynk IoT platform and Mobile App.

Blynk is a popular Internet of Things (IoT) platform. Blynk empowers users to connect their hardware to the cloud and create iOS, Android, and web applications, analyse real-time and historical data from devices, remotely control them from anywhere, receive important notifications, and much more.

2.2.2 Arduino IDE:

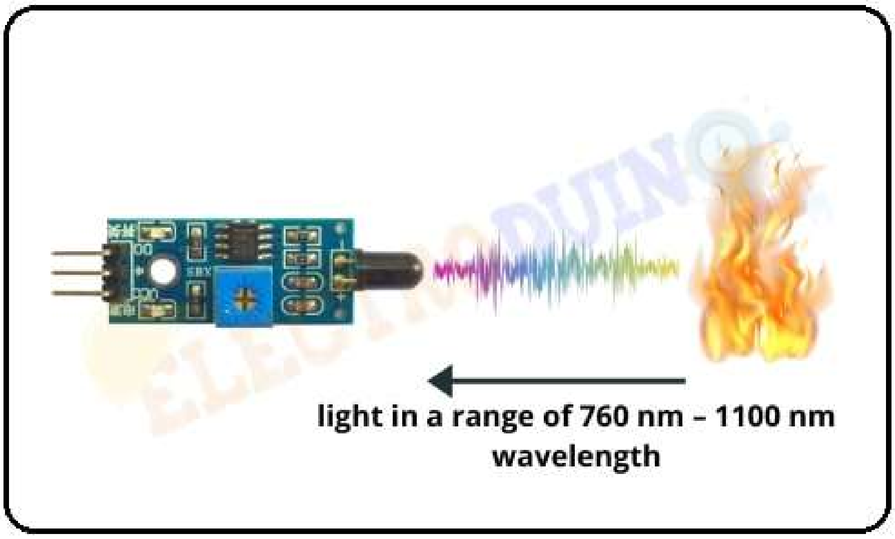
The Arduino IDE is a software application used for writing, compiling, and uploading code to Arduino boards and compatible microcontrollers like the ESP8266.

2.2.3 Blynk IoT Mobile App or web Dashboard:

Mobile App is installed and connected to ESP8266.This App will create a communication channel between the microcontroller and Blynk platform.

3.Working:

3.1 Flame Detection by Yg1006:



The YG1006 sensor detects the presence of a flame or heat or fire by sensing the infrared (IR) radiation emitted by flames. The infra-red flame detector works by checking the infrared spectral band for certain ornamentation that hot gases release when temperature is raised i.e. Battery is over heated.

The YG1006 flame sensor is connected to one of the digital input pins of the NodeMCU board. The NodeMCU firmware is programmed to continuously monitor the state of this input pin. When the sensor detects a flame and sends a HIGH (or LOW, depending on the configuration) signal, the NodeMCU interprets this as the presence of a flame.

3.2 Wi-Fi Connection and Blynk Authentication:

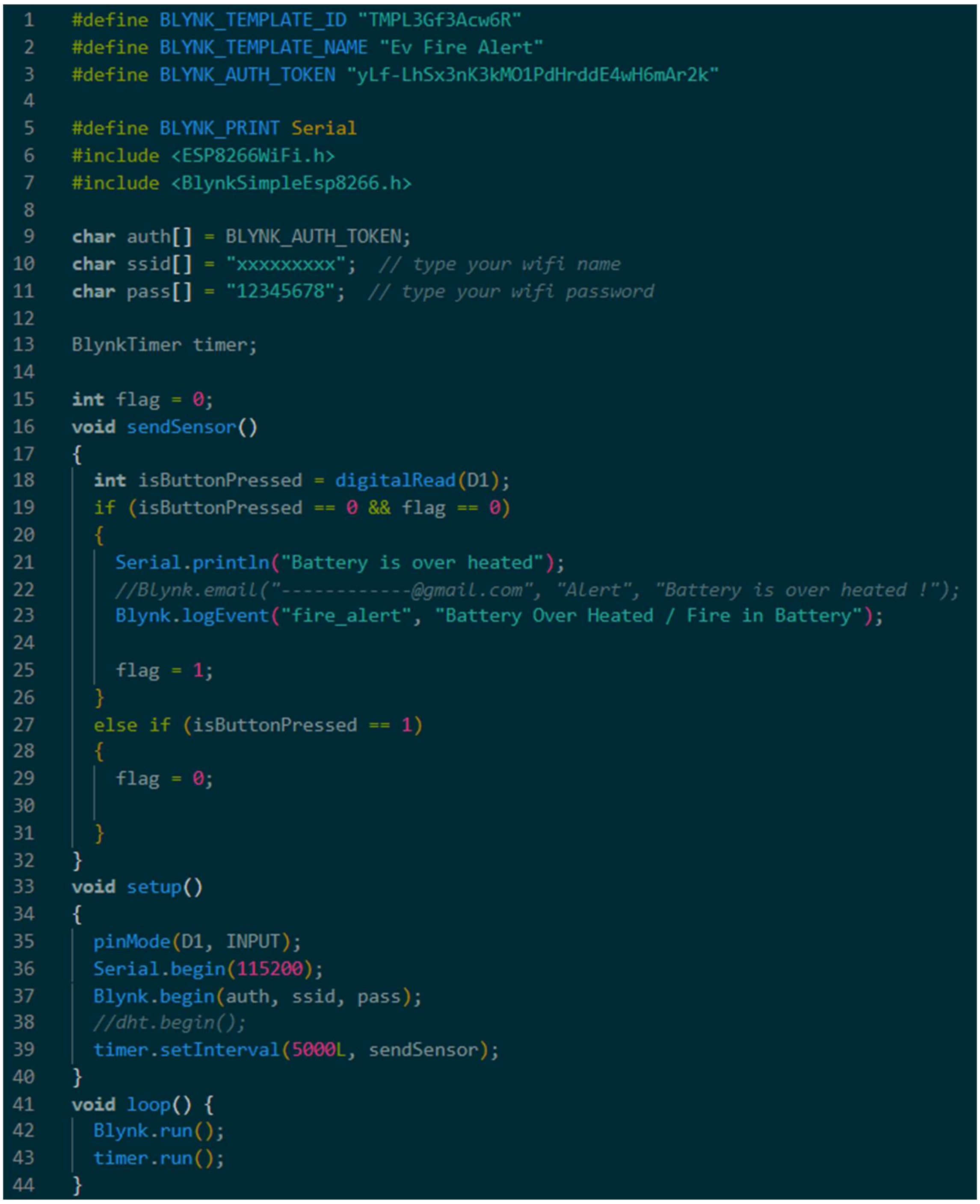
The NodeMCU board has built-in Wi-Fi capabilities. In the firmware, you need to configure the NodeMCU to connect to your local Wi-Fi network by providing the SSID (network name) and password.

Additionally, you need to authenticate with the Blynk platform by providing an authentication token. This token is obtained when you create a new project in the Blynk app or web dashboard.

3.3 Blynk Library Integration:

The Blynk library is installed in the Arduino IDE (Integrated Development Environment). It provides functions and methods to connect to the Blynk cloud, send data, and receive commands. The following source code connect the Blynk cloud to the NodeMCU through ESP8266 Wi-fi module.

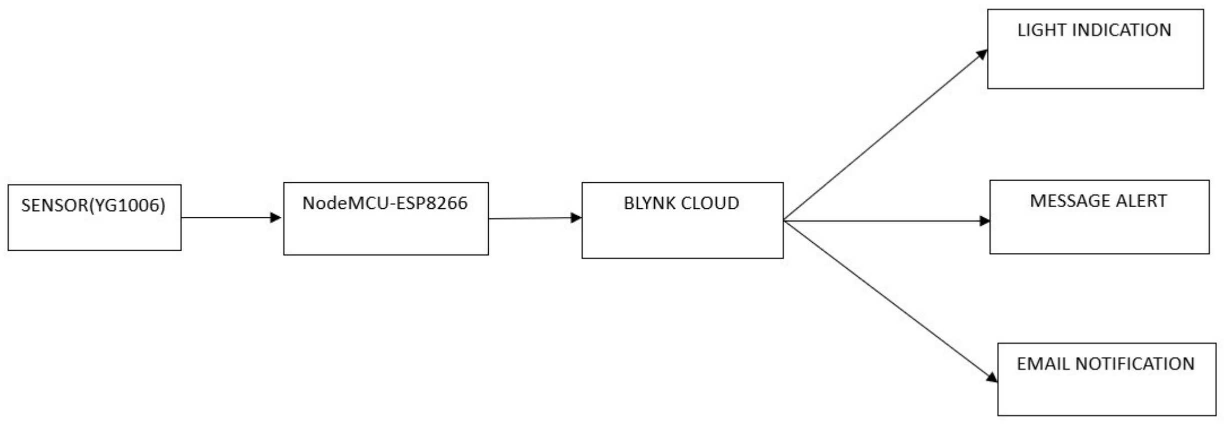
3.4.1 Source Code:



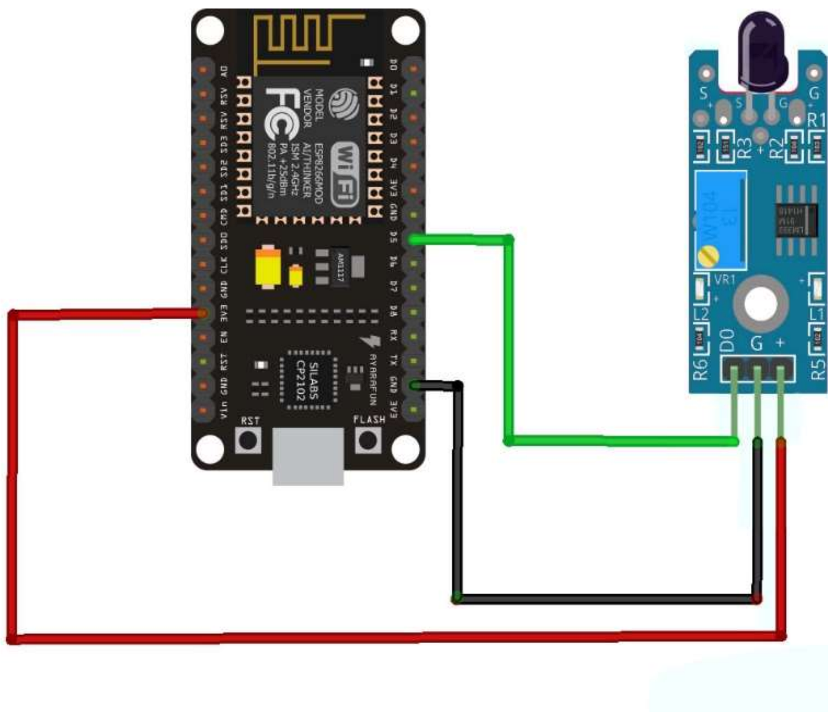
```
#define BLYNK_TEMPLATE_ID "TMPL3Gf3Acw6R"
#define BLYNK_TEMPLATE_NAME "Ev Fire Alert"
#define BLYNK_AUTH_TOKEN "yLf-LhSx3nK3kMO1PdHrddE4wH6mAr2k"

#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "xxxxxxxxx";  // type your wifi name
char pass[] = "12345678";  // type your wifi password

BlynkTimer timer;

int flag = 0;
void sendSensor()
{
  int isButtonPressed = digitalRead(D1);
  if (isButtonPressed == 0 && flag == 0)
  {
    Serial.println("Battery is over heated");
    //Blynk.email("-----------@gmail.com", "Alert", "Battery is over heated !");
    Blynk.logEvent("fire_alert", "Battery Over Heated / Fire in Battery");

    flag = 1;
  }
  else if (isButtonPressed == 1)
  {
    flag = 0;

  }
}
void setup()
{
  pinMode(D1, INPUT);
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
  //dht.begin();
  timer.setInterval(5000L, sendSensor);
}
void loop() {
  Blynk.run();
  timer.run();
}
```

3.5 Sending Flame Detection Data to Blynk: Once the NodeMCU detects the presence of a flame based on the signal from the YG1006 sensor, it can use the Blynk library to send this information to the Blynk platform. The Blynk library provides various widgets and virtual pins that can be used to represent and display data in the Blynk app or web dashboard. The Blynk app or web dashboard will then display the flame detection status based on the received data. The Blynk App send push notifications or email alerts when a flame is detected.

4.Block Diagram:



5.Circuit Diagram:

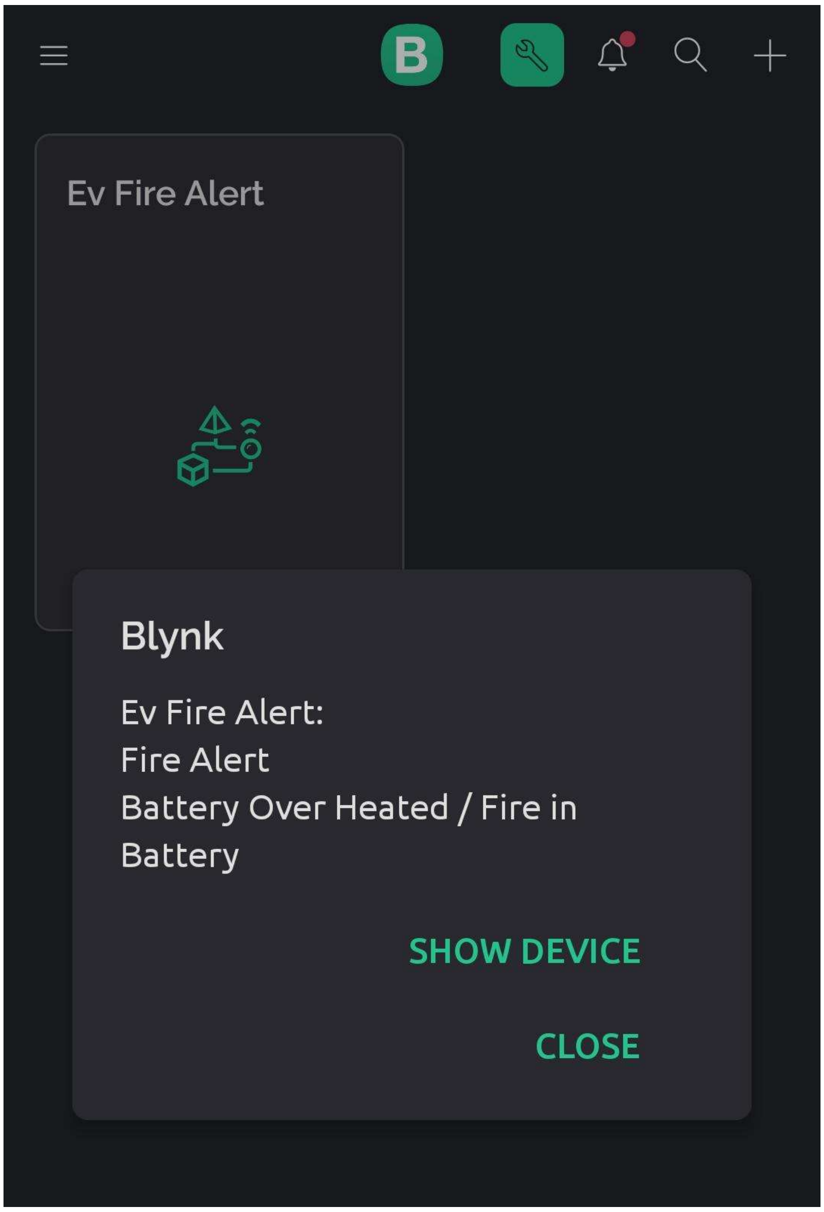


6.Output:

The system is designed to detect fires in electric vehicles (EVs) and provide real-time notifications to the vehicle owner and emergency services. The key outputs of this system are:

Mobile App Notification (Blynk):

When the system detects a fire in the EV, it sends a notification to the vehicle owner's mobile device through the Blynk app. This allows the owner to be immediately aware of the emergency and take appropriate actions.

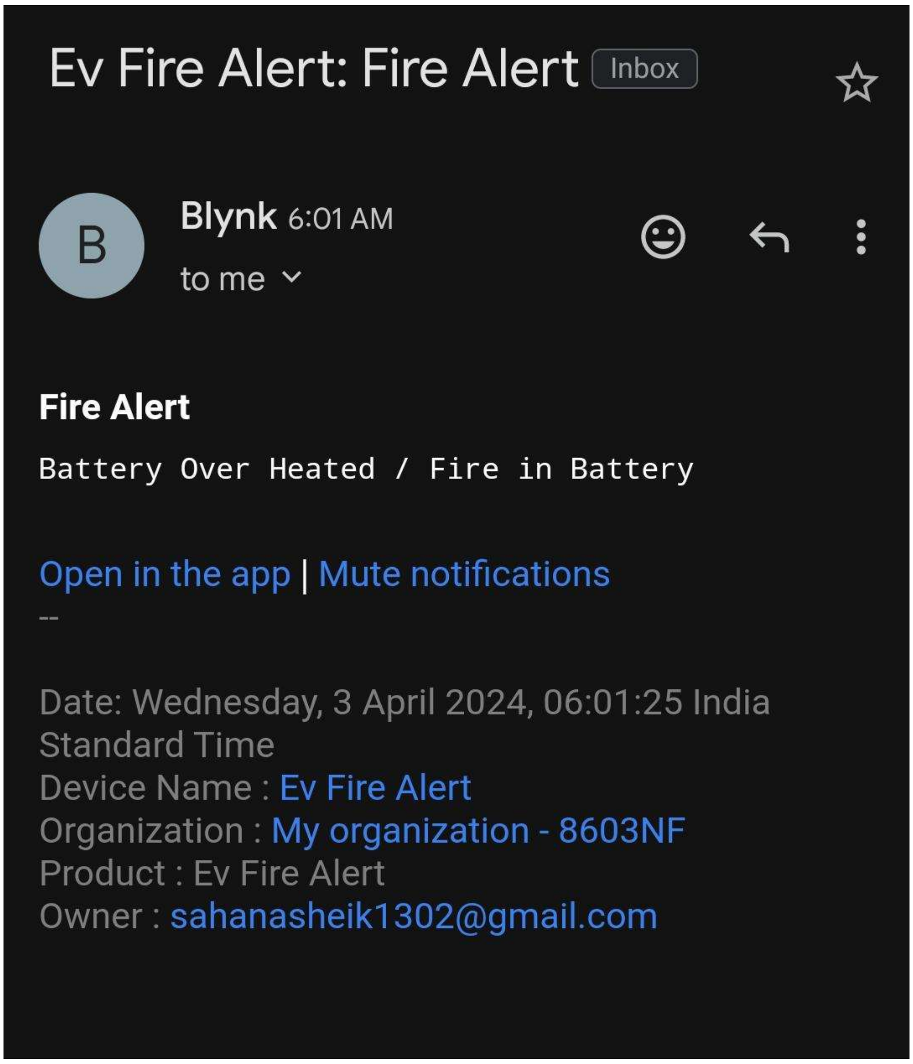


Blynk

Ev Fire Alert:
Fire Alert
Battery Over Heated / Fire in Battery

Email Notification:

In addition to the mobile app notification, the system also sends an email to the vehicle owner and/or emergency contacts when a fire is detected. The email includes details about the incident, such as the location of the vehicle and the timestamp of the fire detection.



Ev Fire Alert: Fire Alert

Blynk 6:01 AM

to me

Fire Alert

```
Battery Over Heated / Fire in Battery
```

Open in the app | Mute notifications

--

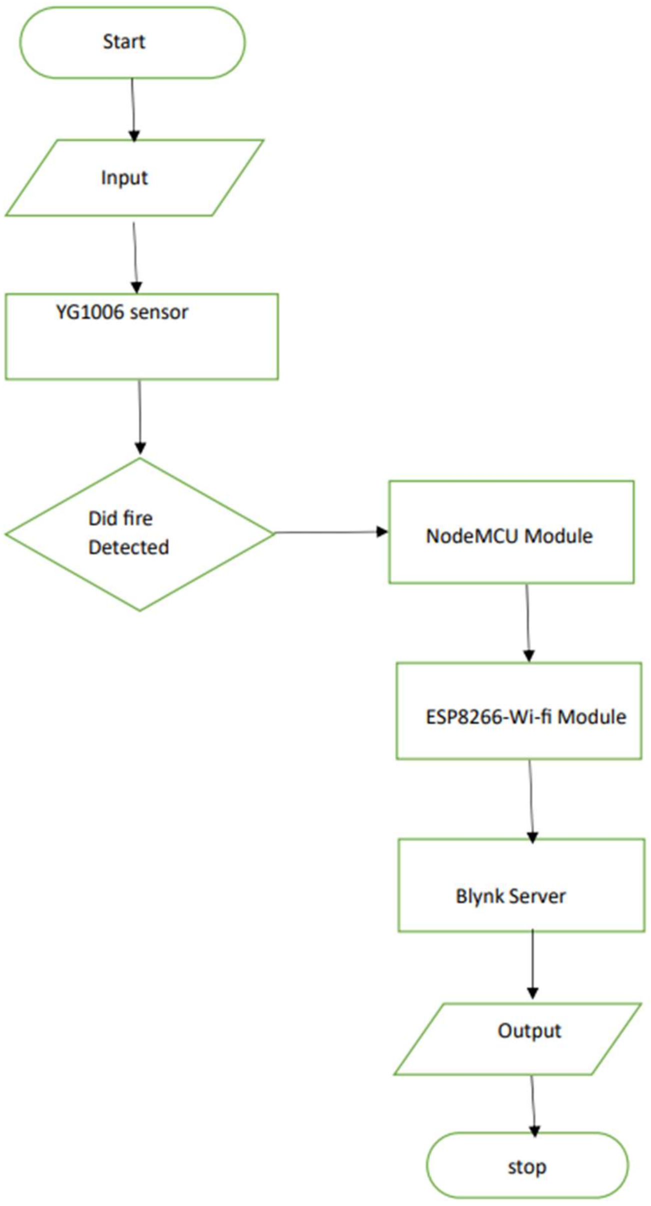
Date: Wednesday, 3 April 2024, 06:01:25 India Standard Time
Device Name : Ev Fire Alert
Organization : My organization - 8603NF
Product : Ev Fire Alert
Owner : sahanasheik1302@gmail.com

Sensor Data Logging:

The system continuously monitors the EV's environment and logs relevant sensor data, such as temperature, smoke, and flame levels. This data can be accessed and analyzed through the Blynk app or a web-based dashboard, allowing for post-incident investigation and system performance evaluation.

Overall, the "Fire Detection in EV Vehicles using IoT" system aims to enhance the safety of electric vehicles by providing an early warning system for fires, enabling timely response and potential mitigation of damage or injury.

7.Results and Discussion



Accuracy of Detection: Evaluate the accuracy of the IoT-based fire detection system compared to traditional methods. This could include metrics such as detection time, false positives/negatives, and overall reliability. Highlight any instances where the IoT system outperformed or underperformed compared to conventional detection methods.

Early Warning Capability: Discuss the system's ability to provide early warnings of potential fire hazards. Early detection is crucial in preventing catastrophic events in EVs, so any improvements in this area are significant.

Real-time Monitoring: Highlight the advantages of real-time monitoring enabled by IoT devices. This allows for continuous surveillance of the vehicle's components, enabling prompt response to any anomalies or potential fire risks.

Data Analysis and Insights: Discuss how the IoT system gathers and analyzes data related to temperature, battery health, electrical currents, and other relevant parameters. This data can provide valuable insights into the conditions that lead to fire hazards in EVs, informing future preventive measures and design improvements.

Remote Notifications and Alerts: Emphasize the importance of remote notifications and alerts provided by the IoT system. This feature enables vehicle owners, manufacturers, and emergency responders to be promptly informed of any detected fire risks, allowing for swift action to mitigate the threat.

Cost-effectiveness: Assess the cost-effectiveness of implementing an IoT-based fire detection system compared to traditional fire suppression systems or post-fire damage control measures. Consider factors such as installation costs, maintenance requirements, and potential savings from preventing fire-related incidents.

Scalability and Adaptability: Discuss the scalability and adaptability of the IoT system to different types of EVs and varying operating conditions. A flexible system that can be easily integrated into different vehicle models and environments enhances its practical utility and market potential.

Integration with Vehicle Systems: Explore how the IoT fire detection system integrates with other vehicle systems, such as onboard diagnostics and safety protocols. Seamless integration ensures that the detection system can effectively communicate with the vehicle's onboard systems to implement preventive measures or initiate emergency responses when necessary.

Regulatory Compliance: Consider the implications of IoT-based fire detection systems for regulatory compliance in the automotive industry. Highlight any standards or regulations that govern fire safety in EVs and discuss how the IoT system helps meet or exceed these requirements.

Future Developments and Research Directions: Lastly, suggest potential areas for further research and development to enhance the capabilities of IoT-based fire detection systems in EVs. This could include improving sensor technologies, refining algorithms for data analysis, or exploring novel approaches to fire prevention and mitigation.

8.Conclusion

In conclusion, the integration of IoT technology for fire detection in electric vehicles (EVs) represents a crucial advancement in automotive safety. By harnessing IoT sensors and connectivity, EVs can continuously monitor critical parameters such as temperature and smoke levels, enabling early detection of fire incidents. This proactive approach enhances passenger safety and mitigates the risk of catastrophic accidents.

Additionally, the real-time data gathered by IoT-enabled EVs facilitates rapid response and intervention, minimizing potential damage. As the automotive industry continues to embrace IoT innovation, further research and development in this area will undoubtedly lead to even more robust and reliable fire detection systems, ensuring the continued safety and security of EV passengers and vehicles alike.

Throughout this project, we have explored the feasibility and benefits of utilizing EVs as mobile fire detection platforms. We have demonstrated that EVs equipped with temperature sensors, smoke detectors, and other environmental monitoring devices can autonomously patrol designated areas, continuously scanning for signs of fire outbreaks. This proactive approach not only reduces the response time but also minimizes the risk of catastrophic damage by enabling early intervention.