# IOT BASED LPG CYLINDER LEVEL & LEAKAGE DETECTING SYSTEM

A PROJECT REPORT

Submitted by

**ABIR HALDER; AJOY BISWAS; AYAN MONDAL; PAPAI MONDAL; SANTANU BISWAS; SRIJA DASGUPTA; TREEDIP PAUL; TUSHAR MONDAL**

in partial fulfillment of the requirements for the award of the degree

of

**BACHELOR OF TECHNOLOGY**

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

BRAINWARE UNIVERSITY

Barasat, Kolkata – 700125

INDIA

July, 2024

# IOT BASED LPG CYLINDER LEVEL & LEAKAGE DETECTING SYSTEM

*Submitted by*

**ABIR HALDER; AJOY BISWAS; AYAN MONDAL; PAPAI MONDAL; SANTANU BISWAS; SRIJA DASGUPTA; TREEDIP PAUL; TUSHAR MONDAL**

*in partial fulfillment of the requirements for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**


**BRAINWARE UNIVERSITY**
*398, Ramkrishnapur Road, Barasat, North 24 Parganas, Kolkata - 700 125*

**BRAINWARE UNIVERSITY**
*398, Ramkrishnapur Road, Barasat, North 24 Parganas, Kolkata - 700 125*

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

## BONAFIDE CERTIFICATE

Certified that this project report **IOT BASED LPG CYLINDER LEVEL & LEAKAGE DETECTING SYSTEM** is the bonafide work of **ABIR HALDER; AJOY BISWAS; AYAN MONDAL; PAPAI MONDAL; SANTANU BISWAS; SRIJA DASGUPTA; TREEDIP PAUL; TUSHAR MONDAL** who carried out the project work under my supervision.

Signature of the Head of the Department

Dr. Angshuman Majumdar
**HEAD OF THE DEPARTMENT**
Department of Electronics &
Communication Engineering
BRAINWARE UNIVERSITY
Barasat, Kolkata - 700125

Signature of the Supervisor

Debasis Mukherjee
**SUPERVISOR**
Associate Professor
Department of Electronics &
Communication Engineering

Signature of the Supervisor

Pankaj Kumar Sanda
**SUPERVISOR**
Assistant Professor
Department of Electronics &
Communication Engineering

# TABLE OF CONTENTS

# ABSTRACT

Liquefied Petroleum Gas (LPG) is widely used in households, but the consumer is unaware of the daily rate of consumption and the time frame when he/she needs to book a refill. In this project, we present an Internet of Things (IoT) based system which monitors different aspects related to LPG cylinder, and thereby keeps the consumer updated via a mobile application. Additional features include displaying the percentage of LPG consumed; the real-time weight of the cylinder and the safety feature is inherent part of this automation system, as when gas leakage is detected it alerts the consumer via mobile application and inbuilt buzzer. The system is designed such that it can be used as an LPG cylinder stand. The weight of LPG is measured using the load sensor and the output of the sensor relates to ESP8266 microcontroller.

**Design and Implementation of Smart LPG Gas Level Monitoring and Safety System Using IoT:**

This system integrates gas sensors, IoT technology, and real-time data processing to continuously monitor LPG levels and promptly detect potential leaks1.

**Key components:**

Gas Sensors: These sensors detect gas leakage.

ESP32 Microcontroller: Equipped with integrated Wi-Fi capabilities, it ensures reliable data transmission and seamless connectivity with other smart devices.

Mobile Application: Users can access the system remotely via an intuitive mobile app to view LPG consumption patterns, gas leakage alerts, and real-time status updates.

Automatic Shut-off Mechanism: In case of a significant leak, the system automatically shuts off the cylinder valve to mitigate risks.

The system also addresses the challenge of untimely emptying of cylinders by providing real-time information on LPG levels.

Google Firebase is used for data gathering and storage.

**IoT-Based Gas Cylinder Monitoring, Alerting, and Control Circuit:**

This system consists of a network of sensors, microcontrollers, and communication modules for real-time monitoring, alerting, and control of gas cylinders.

It leverages IoT technology to enhance safety and efficiency.

While the details of specific components may vary, the overall approach aligns with your project's goals.

**Smart LPG Monitoring and Automatic Booking System Using IoT:**

This system focuses on measuring and displaying the gasoline content present in household LPG cylinders and automatically booking a new cylinder.

It predicts the working days of the gasoline content, providing convenience to users.

Design and Implementation of IoT and AI-Enabled Innovative LPG Cylinder Monitoring:

This prototype detects LPG leakage, continuously monitors the weight of remaining gas, and updates data through IoT to the cloud.

The system aims to prevent accidents caused by unnoticed gas leaks.

# INTRODUCTION

**Liquefied Petroleum Gas (LPG): A Versatile Fuel for Everyday Needs**

Ever fired up a summer barbecue or used a portable camping stove? The convenient fuel powering those flames might be liquefied petroleum gas, also known as LPG or LP gas. This versatile fuel source has a wide range of applications in our daily lives, from cooking and heating homes to powering vehicles and industrial processes.

LPG is a mixture of hydrocarbon gases, primarily propane and butane. It's a flammable gas that's easily compressed into a liquid state. This liquid form makes it portable and easy to store, allowing for convenient transportation and use in various settings.

In the next part, we can delve deeper into the different uses of LPG, its advantages, or how it compares to other fuel sources. Let me know what interests you the most!

**The Advantages of LPG:**

LPG, or Liquefied Petroleum Gas, offers a multitude of benefits that make it a popular fuel source for various applications. Here is a detailed breakdown of its advantages:

Clean Burning:

■ Reduced Emissions: Compared to other fossil fuels like coal or oil, LPG burns cleaner, releasing fewer harmful pollutants like sulfur oxides and nitrogen oxides. This translates to cleaner air and improved respiratory health, especially in urban areas.

■ Lower Greenhouse Gas Emissions: While still a fossil fuel, LPG produces less carbon dioxide than coal and gasoline when burned. This can contribute to a smaller carbon footprint and a reduced impact on climate change.

■ High Calorific Value: LPG packs a powerful punch. It has a high heating value, meaning it releases a significant amount of heat when burned. This translates to efficient fuel consumption and faster cooking times.

■ Easy Storage and Transportation: The ability to compress LPG into a liquid state makes it incredibly portable and easy to store. It can be transported in tanks of various sizes, making it readily available in areas lacking natural gas pipelines.

■ Instantaneous Control: Unlike some fuels that require time to heat up, LPG offers precise and immediate control. You can easily adjust the flame size with a simple knob, allowing for efficient cooking and heating.

■ Wide Range of Uses: LPG is a true all-rounder. It's a popular fuel for cooking in homes and restaurants, powering heating systems in buildings, and even fueling vehicles like forklifts and taxis.

■ Adaptability to Off-Grid Applications: For locations without access to natural gas lines, LPG becomes a reliable alternative. It allows for independent energy solutions in remote areas, powering households and businesses.

■ Industrial Applications: Beyond domestic uses, LPG plays a role in various industrial processes. It can be used for drying crops, powering generators, and in some manufacturing applications.

**Precautions with LPG:**

■ Location: Store LPG cylinders in a cool, well-ventilated area, preferably outdoors or in a designated storage space with proper ventilation. Avoid storing them in basements, kitchens, or near heat sources like stoves or ovens.

■ Position: Always keep the cylinder upright and secure it in place to prevent tipping or falling.

■ Inspection: Regularly inspect the cylinder for any damage, rust, or leaks. Look for cracks, dents, or loose connections. If you find any damage, replace the cylinder immediately.

■ Leak Detection: Use a soapy water solution to check for leaks around the valve and hose connections. Bubbles forming indicate a leak, requiring immediate attention. Turn off the valve and contact your gas supplier or a qualified technician for repair.

# THEORY

| Pin No | Function | Name |
|---|---|---|
| 1 | Ground (0V) | Ground |
| 2 | Supply voltage; 5V (4.7V – 5.3V) | Vcc |
| 3 | Contrast adjustment; through a variable resistor | $V_{EE}$ |
| 4 | Selects command register when low; and data register when high | Register Select |
| 5 | Low to write to the register; High to read from the register | Read/write |
| 6 | Sends data to data pins when a high to low pulse is given | Enable |
| 7 | 8-bit data pins | DB0 |
| 8 | | DB1 |
| 9 | | DB2 |
| 10 | | DB3 |
| 11 | | DB4 |
| 12 | | DB5 |
| 13 | | DB6 |
| 14 | | DB7 |
| 15 | Backlight $V_{CC}$ (5V) | Led+ |
| 16 | Backlight Ground (0V) | Led- |

# METHODOLOGY

The project you described sounds fascinating. Leveraging IoT technology, it aims to address the limitations of existing gas cylinder monitoring systems. Let me provide you with some insights based on existing research and implementations:

Design and Implementation of Smart LPG Gas Level Monitoring and Safety System Using IoT:

This system integrates gas sensors, IoT technology, and real-time data processing to continuously monitor LPG levels and promptly detect potential leaks.

Key components:

Gas Sensors: These sensors detect gas leakage.

ESP32 Microcontroller: Equipped with integrated Wi-Fi capabilities, it ensures reliable data transmission and seamless connectivity with other smart devices.

Mobile Application: Users can access the system remotely via an intuitive mobile app to view LPG consumption patterns, gas leakage alerts, and real-time status updates.

Automatic Shut-off Mechanism: In case of a significant leak, the system automatically shuts off the cylinder valve to mitigate risks.

The system also addresses the challenge of untimely emptying of cylinders by providing real-time information on LPG levels.

Google Firebase is used for data gathering and storage.

IoT-Based Gas Cylinder Monitoring, Alerting, and Control Circuit:

This system consists of a network of sensors, microcontrollers, and communication modules for real-time monitoring, alerting, and control of gas cylinders.

It leverages IoT technology to enhance safety and efficiency.

While the details of specific components may vary, the overall approach aligns with your project's goals.

Smart LPG Monitoring and Automatic Booking System Using IoT:

This system focuses on measuring and displaying the gasoline content present in household LPG cylinders and automatically booking a new cylinder3.

It predicts the working days of the gasoline content, providing convenience to users.

Design and Implementation of IoT and AI-Enabled Innovative LPG Cylinder Monitoring:

This prototype detects LPG leakage, continuously monitors the weight of remaining gas, and updates data through IoT to the cloud4.

The system aims to prevent accidents caused by unnoticed gas leaks.

# MATERIALS

**Insight into ESP8266 NodeMCU Features & Using It with Arduino IDE:**

With the advancement in technology, intelligent systems are introduced every day. Everything is getting more sophisticated and intelligible. There is an increase in the demand of cuttingedge technology and smart electronic systems. Microcontrollers play a very important role in the development of the smart systems as brain is given to the system. Microcontrollers have become the heart of the new technologies that are being introduced daily. A microcontroller is mainly a single chip microprocessor suited for control and automation of machines and processes. Today, microcontrollers are used in many disciplines of life for carrying out automated tasks in a more accurate manner. Almost every modern day device including air conditioners, power tools, toys, office machines employ microcontrollers for their operation. Microcontroller essentially consists of Central Processing Unit (CPU), timers and counters, interrupts, memory, input/output ports, analogue to digital converters (ADC) on a single chip. With this single chip integrated circuit design of the microcontroller the size of control board is reduced and power consumption is low.

The Internet of Things (IoT) has been a trending field in the world of technology. It has changed the way we work. Physical objects and the digital world are connected now more than ever.

**ESP12E Module:**

The development board equips the ESP-12E module containing ESP8266 chip having

Tensilica Xtensa® 32-bit LX106 RISC microprocessor which operates at 80 to 160


**Fig: 1**

MHz adjustable clock frequency and supports RTOS.

There's also **128 KB RAM and 4MB of Flash memory** (for program and data storage) just enough to cope with the large strings that make up web pages, JSON/XML data, and everything we throw at IoT devices nowadays.

The ESP8266 Integrates **802.11b/g/n HT40 Wi-Fi transceiver**, so it can not only connect to a Wi-Fi network and interact with the Internet, but it can also set up a network of its own, allowing other devices to connect directly to it. This makes the ESP8266 NodeMCU even more versatile.

**Power Requirement:**

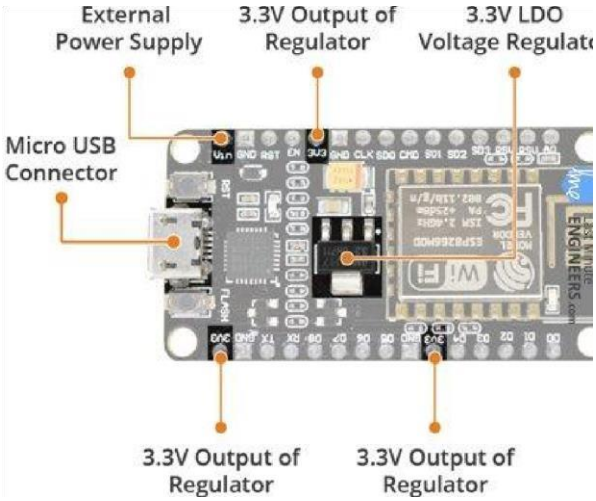



**Fig: 2**

As the operating voltage range of ESP8266 is **3V to 3.6V**, the board comes with a LDO voltage regulator to keep the voltage steady at 3.3V. It can reliably supply up to 600mA, which should be more than enough when ESP8266 pulls as much as **80mA during RF transmissions**. The output of the regulator is also broken out to one of the sides of the board and labelled as 3V3.

This pin can be used to supply power to external components.

**Power to the ESP8266 NodeMCU** is supplied via the **on-board MicroB USB connector**. Alternatively, if you have a regulated 5V voltage source, the **VIN pin** can be used to directly supply the ESP8266 and its peripherals.

**Peripherals and I/O:**

The ESP8266 NodeMCU has total **17 GPIO** pins broken out to the pin headers on both sides of the development board. These pins can be assigned to all sorts of peripheral duties, including:

■ ADC channel – A 10-bit ADC channel.

■ UART interface – UART interface is used to load code serially.

■ PWM outputs – PWM pins for dimming LEDs or controlling motors.

■ SPI, I2C & I2S interface – SPI and I2C interface to hook up all sorts of sensors and peripherals.

■ I2S interface – I2S interface if you want to add sound to your project.

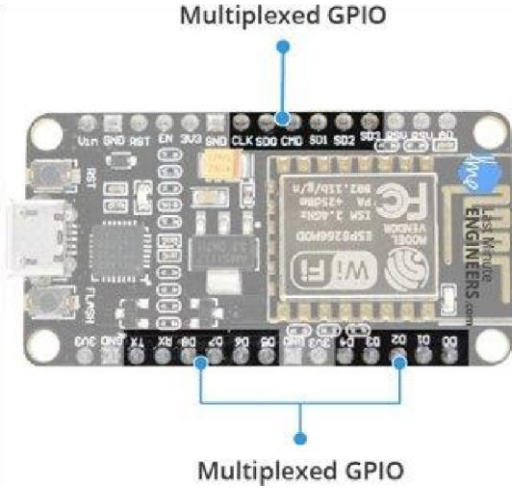



**Fig: 3**

**On-board Switches & LED Indicator:**

The ESP8266 NodeMCU features two buttons. One marked as **RST** located on the top left corner is the Reset button, used of course to reset the ESP8266 chip. The other **FLASH** button on the bottom left corner is the download button used while upgrading firmware.

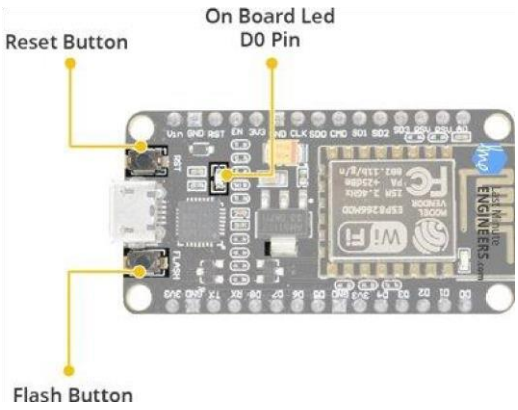
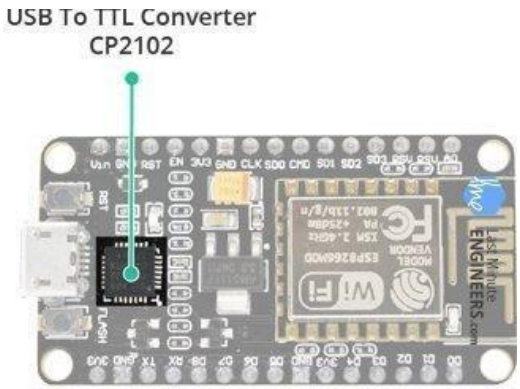



**Fig: 4**

The board also has a **LED indicator** which is user programmable and is connected to the D0 pin of the board.

**Serial Communication:**

The board includes CP2102 USB-to-UART Bridge Controller from Silicon Labs, which converts USB signal to serial and allows your computer to program and communicate with the ESP8266 chip.



**ESP8266 NodeMCU Pinout**

The ESP8266 NodeMCU has total 30 pins that interface it to the outside world. The connections are as follows:

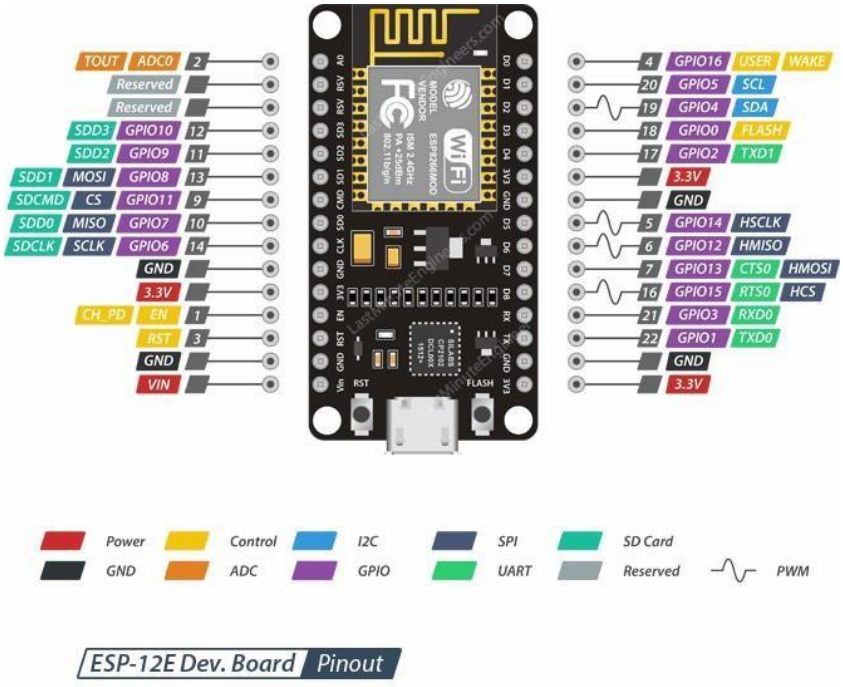



**Fig:6**

**Power Pins:** There are four power pins viz. one VIN pin & three 3.3V pins. The VIN pin can be used to directly supply the ESP8266 and its peripherals, if you have a regulated 5V voltage source. The 3.3V pins are the output of an on-board voltage regulator. These pins can be used to supply power to external components.

**GND** is a ground pin of ESP8266 NodeMCU development board.

**I2C** Pins are used to hook up all sorts of I2C sensors and peripherals in your project. Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum. It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

**GPIO Pins** ESP8266 NodeMCU has 17 GPIO pins which can be assigned to various functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically. Each digital enabled GPIO can be configured to internal pull-up or pulldown, or set to high impedance. When configured as an input, it can also be set to edgetrigger or level-trigger to generate CPU interrupts.

**ADC** Channel the NodeMCU is embedded with a 10-bit precision SAR ADC. The two functions can be implemented using ADC viz. Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin. However, they cannot be implemented at the same time.

**UART** Pins ESP8266 NodeMCU has 2 UART interfaces, i.e. UART0 and UART1, which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. It supports fluid control. However, UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

**SPI** Pins ESP8266 features two SPIs (SPI and HSPI) in slave and master modes. These SPIs also support the following general-purpose SPI features:

■ 4 timing modes of the SPI format transfer

■ Up to 80 MHz and the divided clocks of 80 MHz

■ Up to 64-Byte FIF

**SDIO** Pins ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards. 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

**PWM** Pins the board has 4 channels of Pulse Width Modulation (PWM). The PWM output can be implemented programmatically and used for driving digital motors and LEDs. PWM frequency range is adjustable from 1000 µs to 10000 µs, i.e., between 100 Hz and 1 kHz.

**Control Pins** are used to control ESP8266. These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

■ EN pin – The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.

■ RST pin – RST pin is used to reset the ESP8266 chip.

■ WAKE pin – Wake pin is used to wake the chip from deep-sleep.

**Installing the ESP8266 Core on Windows OS:**

Let's proceed with installing ESP8266 Arduino core. The first thing is having latest Arduino IDE (Arduino 1.6.4 or higher) installed on our PC.

To begin, we'll need to update the board manager with a custom URL. Open up Arduino IDE and go to **File** > **Preferences**. Then, copy below URL into the **Additional Board Manager URLs** text box situated on the bottom of the window: http://arduino.esp8266.com/stable/package_esp8266com_index.json

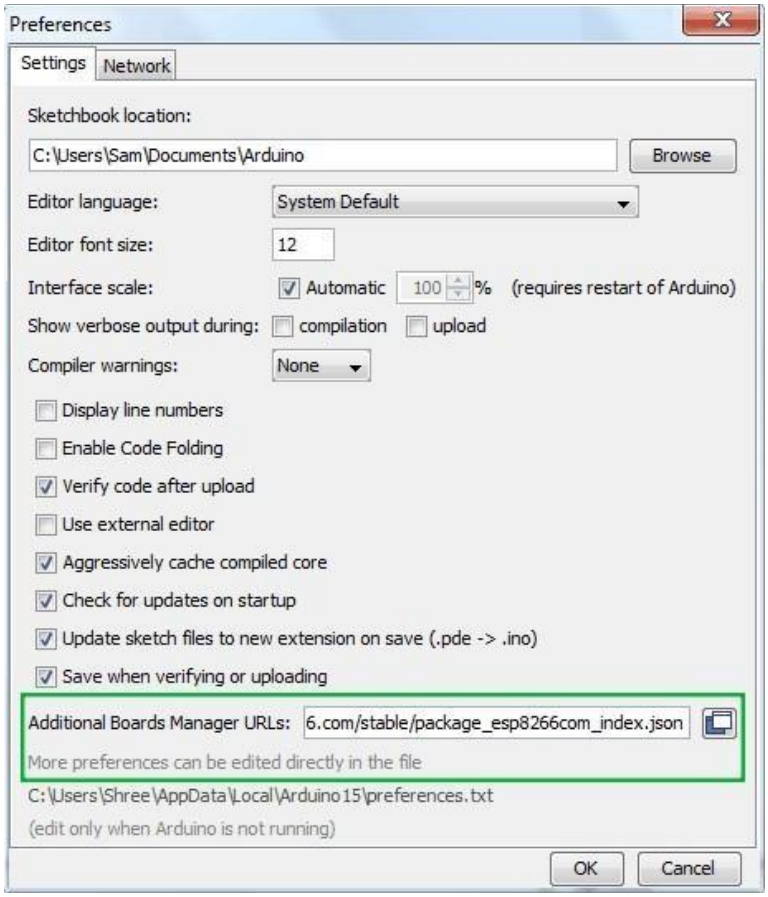
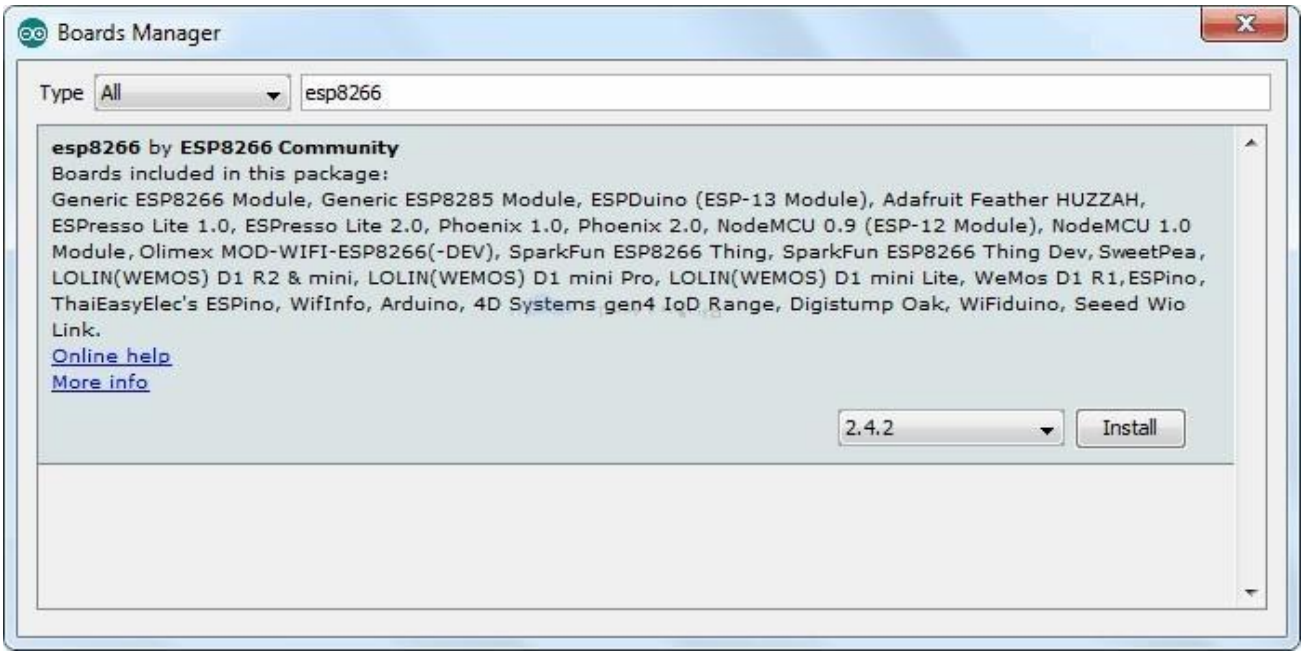



**Fig: 7**

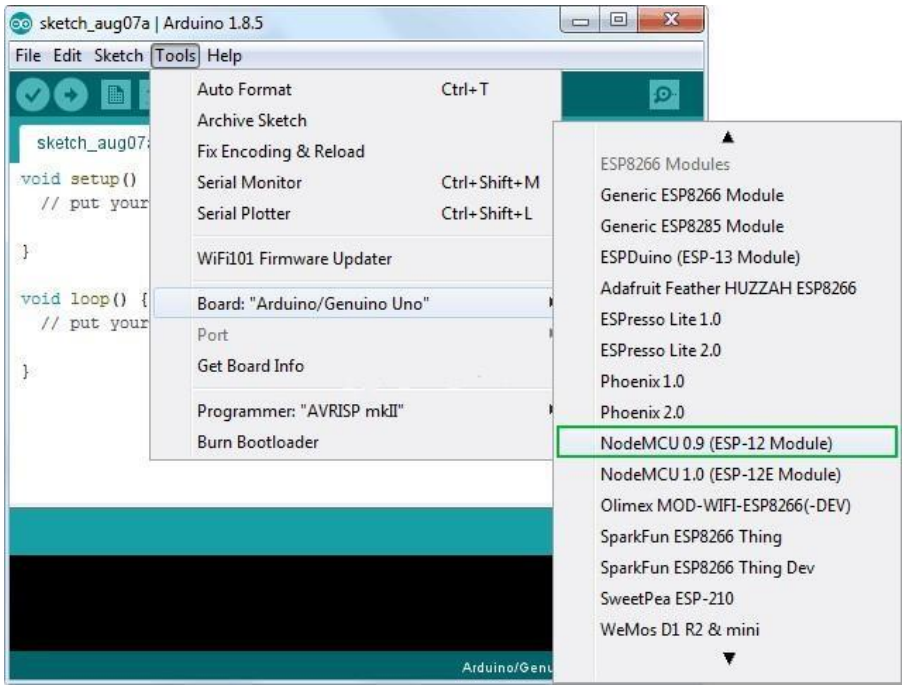
Hit OK. Then navigate to the Board Manager by going to **Tools** > **Boards** > **Boards Manager**. There should be a couple new entries in addition to the standard Arduino boards. Filter our search by typing **esp8266**. Click on that entry and select Install.




The board definitions and tools for the ESP8266 include a whole new set of gcc, g++, and other reasonably large, compiled binaries, so it may take a few minutes to download and install (the archived file is ~110MB). Once the installation has completed, a small

***INSTALLED*** text will appear next to the entry. Now close the Board Manager.

Before we get to uploading sketch & playing with LED, we need to make sure that the board is selected properly in Arduino IDE. Open Arduino IDE and select **NodeMCU 0.9 (ESP-12 Module)** option under your **Arduino IDE** > **Tools** > **Board** menu.




Now, plug ESP8266 NodeMCU into computer via micro-B USB cable. Once the board is plugged in, it should be assigned a unique COM port. On Windows machines, this will be something like **COM#,** and on Mac/Linux computers it will come in the form of **/dev/tty.usbserial-XXXXXX**. Select this serial port under the **Arduino IDE** > **Tools** > **Port** menu. Also select the **Upload Speed: 115200**

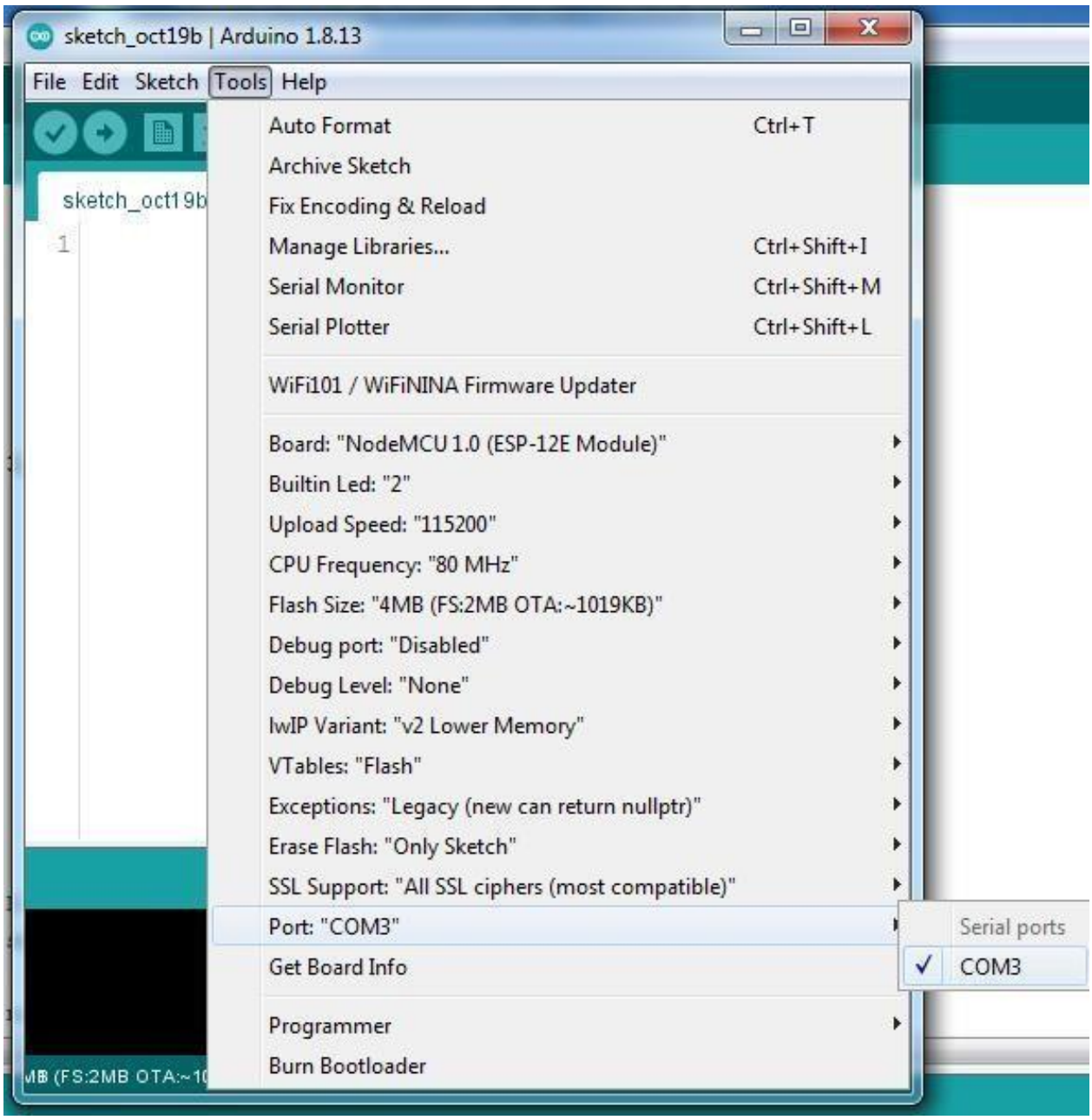



**Fig: 10**

More attention needs to be given to selecting board, choosing COM port and selecting Upload speed.

**Load Cell:**

A **load cell** is a transducer that is used to create an electrical signal whose magnitude is directly proportional to the force being measured. It is basically a device that measures strain and then

**Fig: 11**

converts force into electric energy which serves as a measurement for scientists and workers. The strain measurement by load cells helps in maintaining the integrity of the unit under pressure and protects people and equipment nearby. Load cell comes in various ranges like 5kg, 10kg, and 100kg and more, here we have used Load cell, which can weigh up to 5kg.

Now the electrical signals generated by Load cell is in few millivolts, so they need to be further amplified by some amplifier and hence HX711 Weighing Sensor comes into picture.

**HX711 Load cell Amplifier Module:**

This Load Cell Amplifier is a small breakout board for the HX711 IC that allows easily read load cells to measure weight. By connecting the amplifier to microcontroller will be able to read the changes in the resistance of the load cell and with some calibration to get very accurate weight measurements. This can be handy for creating industrial scale, process control, or simple presence detection. The HX711 uses a two-wire interface (Clock and Data) for communication. Any microcontroller's GPIO pins should work and numerous libraries have been written making it easy

to read data from the HX711. Load cells use a four wire to connect to the HX711. These are commonly colored Red, Black, White, and Green. Each color corresponds to the conventional color coding of load cells.

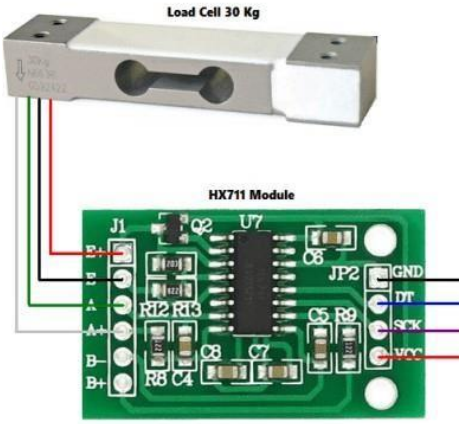



**Fig: 12**

**Flame Sensor:** Flame sensor is an electronic device which is capable of sensing/detection of fire or a high temperature zone. It gives an indication through an LED attached at its top, just after sensing the fire. These types of sensors are usually used for short ranges. They are able to detect the fire up to 3 feet. Flame sensors are the most common device available in the market these days due to its good results and cost efficiency. A flame sensor module consists of a flame sensor, resistor, capacitor, potentiometer, and comparator LM393 in an integrated circuit. It can detect infrared light with a wavelength ranging from 700nm to 1000nm.The far-infrared flame probe converts the strength changes of the external infrared light into current changes.

**Where and why is flame sensors used?**

Flame sensors are utilized in a few hazardous environments, such as hydrogen stations, industrial heating and drying systems, industrial gas turbines, domestic heating systems and gas-powered cooking devices. Their primary purpose is to minimize the risks associated with combustion. Often, a flame sensor responds more swiftly than a heat or smoke detector

**Flame Sensor Module:**

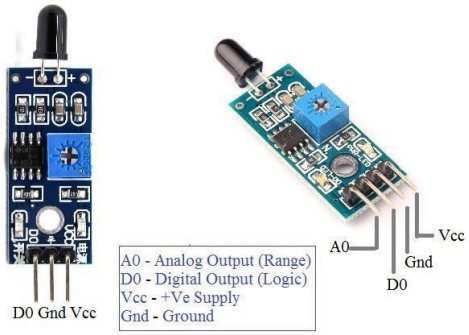
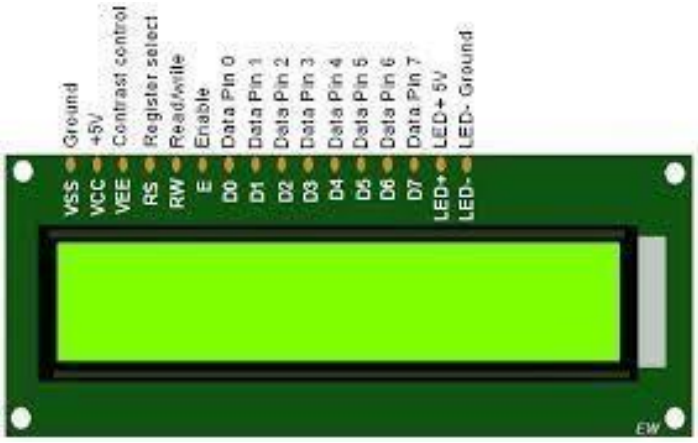



**Fig: 13**

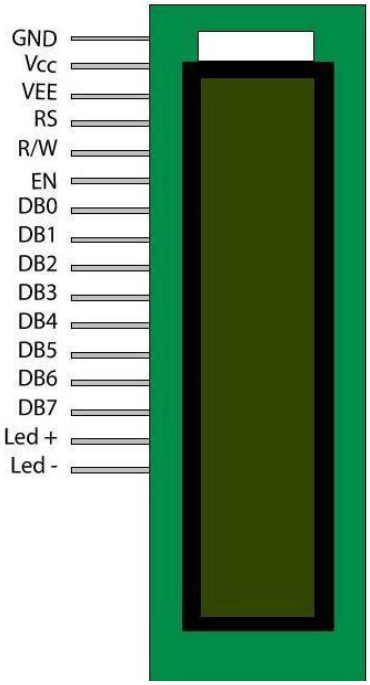
**Liquid crystal displays** (LCDs) have materials which combine the properties of both liquids and crystals. An LCD consists of two glass panels, with the liquid crystal material sand witched in between them. The inner surface of the glass plates are coated with transparent electrodes which define the character, symbols or patterns to be displayed polymeric layers are present in between the electrodes and the liquid crystal, which makes the liquid crystal molecules to maintain a defined orientation angle.




The LCD is a dot matrix liquid crystal display that displays alphanumeric characters and symbols. Liquid Crystal Display screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on. A 16x2 LCD means it can display

16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

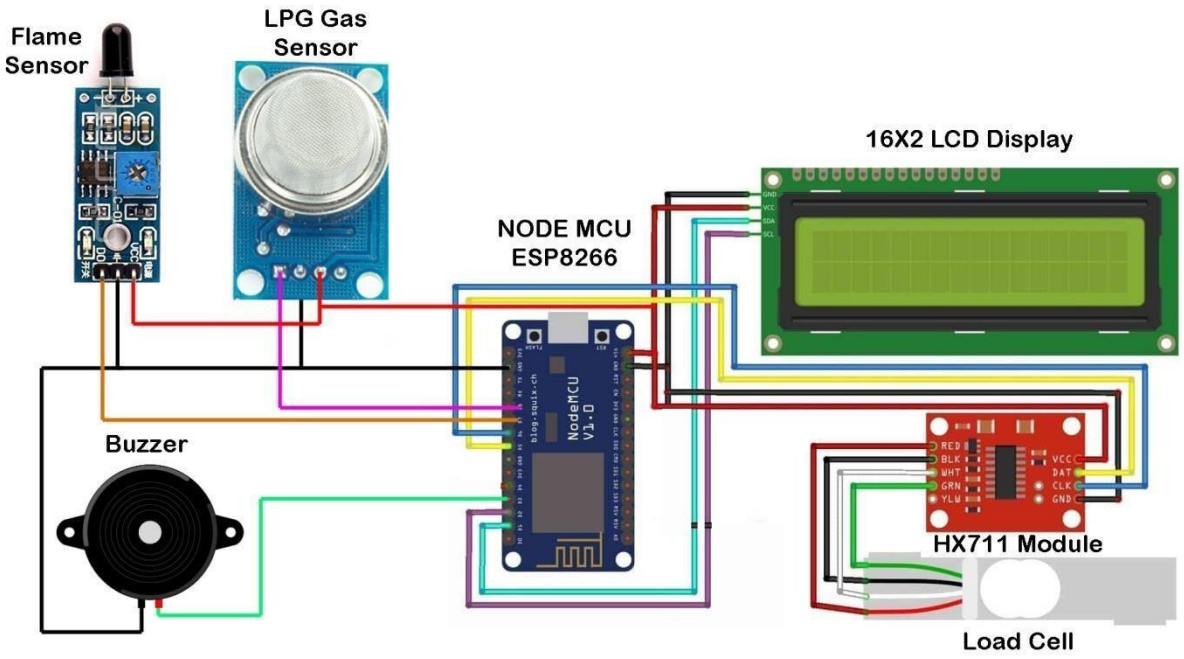



# RESULTS



LPG GAS 10%
LPG GAS 20%

# ANALYSIS


Flame Sensor
LPG Gas Sensor
16X2 LCD Display
NODE MCU ESP8266
Buzzer
HX711 Module
Load Cell

# APPENDICES

Source code

```
// MAX30100 ESP8266 WebServer

#include <ESP8266WebServer.h>

#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <Arduino.h>

#include “HX711.h”


// HX711 circuit wiring const int

LOADCELL_DOUT_PIN = 12; // D6 const int LOADCELL_SCK_PIN = 13;  // D7 const int

gasSensor = 14;         // D5 const int flameSensor = 2;       //

D4 const int readyLed

= 15;         // D8


LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE); // SET I2C Address


HX711 scale;


/*Put your SSID & Password*/  const char *ssid = “ABCD1234”;   // Enter

SSID here const char *password = “abcd1234”; // Enter Password here

float currWeight = 0;
```

```
String weightMsg = “”;

long sample = 0; float val = 0; long
count = 0; float calibValue = -1082.22;

int limitingWeight = 100; // Change Here

String gasMsg = “No”;
String flameMsg = “No”;

ESP8266WebServer server(80);

void setup()
{
    Serial.begin(115200);
    // Pull up for setpoint keys
    //  set up the LCD’s number of columns and rows:
    pinMode(gasSensor, INPUT);
pinMode(flameSensor, INPUT);      pinMode(A0,
OUTPUT); pinMode(readyLed, OUTPUT);
digitalWrite(readyLed, LOW); lcd.begin(16, 2);
lcd.backlight(); // makes Backligh ON.

      Lcd.clear();    // Clears LCD      lcd.setCursor(0, 0);
    // Print a message to the LCD.      Lcd.print(“  Smart LPG”);
lcd.setCursor(0, 1); // Move coursor to second Line      lcd.print(“
```

```
Cylinder Monitor");       delay(3000);       lcd.clear();
    Serial.println("Connecting to ");
    Serial.println(ssid);

    // connect to your local wi-fi network
    WiFi.begin(ssid, password);

    // check wi-fi is connected to wi-fi network       while (WiFi.status()
!= WL_CONNECTED)
    {            delay(1000);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected..!");
    Serial.print("Got IP: ");
    Serial.println(WiFi.localIP());

    lcd.setCursor(0, 0); lcd.print("Got
      IP: "); lcd.setCursor(0, 1);
      lcd.print(WiFi.localIP());

    delay(10000);

    server.on("/", handle_OnConnect);       server.onNotFound(handle_NotFound);
```

```
    server.begin();
    Serial.println("HTTP server started");
    // lcd.clear();       delay(1000);
initializeLoadCell();       for(int I = 0 ; I <
3; i++)
    {             digitalWrite(readyLed, HIGH);
delay(500);             digitalWrite(readyLed,
LOW);             delay(500);
    }
}

void loop()
{ bool gasRead = digitalRead(gasSensor); bool flameRead =
      digitalRead(flameSensor); if
      (gasRead)
      {             gasMsg = "No";
}      else
    {
        gasMsg = "Yes";
    }

    if (flameRead)
    {             flameMsg = "No";
```

```
digitalWrite(readyLed, LOW);
    }       else
    {               flameMsg = “Yes”;
digitalWrite(readyLed, HIGH);
    }

    currWeight = scale.get_units();       if
(currWeight < 0)              currWeight = 0;       if
(currWeight
< 50)              weightMsg = “Cylinder almost empty. Please book.”;
else              weightMsg = “Cylinder weight ok.”;
server.handleClient(); scale.power_down(); // put

        the ADC in sleep mode delay(2000);        scale.power_up();

}

void handle_OnConnect() { server.send(200, “text/html”, SendHTML(currWeight,
weightMsg, gasMsg, flameMsg)); }

void handle_NotFound() { server.send(404, “text/plain”, “Not found”); }
```

```
String SendHTML(float currWeight, String weightMsg, String gasMsg, String flameMsg) {       String ptr =
"<!DOCTYPE html>";      ptr += "<html>";      ptr += "<head>";      ptr += "<title>ESP8266 WebServer</title>";
ptr +=
        "<meta name='viewport' content='width=device-width, initial-scale=1.0'>";      ptr += "<link rel='stylesheet'
"
          "href='https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.7.2/css/"
          "all.min.css'>";       ptr += "<link rel='stylesheet' type='text/css'
href='styles.css'>";
ptr += "<style>";       ptr += "body { background-color: #fff; font-family: sans-serif; color: "
          "#333333;font: 14px Helvetica, sans-serif box-sizing: border-box;}"; ptr += "#page { margin:
       20px; background-color: #fff;}"; ptr +=
       ".container { height: inherit; padding-bottom: 20px;}"; ptr += ".header { padding: 20px;}";
       ptr += ".header h1 { padding-bottom: 0.3em; color: #008080; font-size: 45px; "          "font-weight: bold; font-
       family: Garmond, 'sans-serif'; text-align: "
          "center;}";
     ptr += "h2 { padding-bottom: 0.2em; border-bottom: 1px solid #eee; margin: "
          "2px;text-align: left;}";       ptr += ".header h3 { font-weight: bold; font-family: Arial,
'sans-serif'; "
          "font-size: 17px; color: #b6b6b6; text-align: center;}";       ptr += ".box-full { padding: 20px;
border 1px solid #ddd; border-radius: 1em "
          "1em 1em 1em; box-shadow: 1px 7px 7px 1px rgba(0,0,0,0.4); "
          "background: #fff; margin: 20px; width: 300px;}";       ptr += "@media (max-width: 494px) {
```

```
#page { width: inherit; margin: 5px “
        “auto; }  #content { padding: 1px;} .box-full { margin: 8px 8px 12px “
        “8px; padding: 10px; width: inherit;; float: none; } }”;       ptr += “@media (min-width: 494px) and
(max-width: 980px) { #page { width: “
        “465px;margin 0 auto; } .box-full { width: 380px; } }”;       ptr += “@media (min-width: 980px) {
#page { width: 930px; margin: auto; } }”;       ptr += “.sensor { margin: 12px 0px; font-size: 2.5rem;}”;
ptr += “.sensor-labels
{ font-size: 1rem; vertical-align: middle; “
“padding-bottom:15px;}”;       ptr += “.units { font-
size: 1.2rem;}”;       ptr +=
       “hr { height: 1px; color: #eee; background-color: #eee; border: none;}”;       ptr += “</style>”;

    // Ajax Code Start ptr += “<script>\n”;  ptr +=
      “setInterval(loadDoc,1000);\n”; ptr
      += “function loadDoc() {\n”;  ptr += “var xhttp = new XMLHttpRequest();\n”;
ptr += “xhttp.onreadystatechange = function() {\n”;       ptr += “if (this.readyState ==
4 && this.status == 200) {\n”;       ptr += “document.body.innerHTML
=this.responseText}\n”;       ptr
+= “};\n”;       ptr += “xhttp.open(\”GET\”, \”/\”, true);\n”;
ptr += “xhttp.send();\n”;       ptr += “}\n”;       ptr +=
“</script>\n”;
    // Ajax Code END

    ptr += “</head>”;       ptr += “<body>”;       ptr += “<div id=’page’>”;       ptr += “<div class=’header’>”;
ptr += “<h1>IoT Based LPG Cylinder Level & Leakage Detection System</h1>”;
ptr += “</div>”;       ptr += “<div id=’content’
align=’center’>”;       ptr += “<div class=’box-full’
```

```
align='left'>";       ptr += "<h2>Sensor Readings</h2>";
ptr
+= "<div class='sensors-container'>";

    // For Gas weight       ptr += "<p class='sensor'>";       ptr += "<I class='fas fa-gas-
pump' style='color:#f7347a'></i>"; ptr += "<span class='sensor-labels'> Weight:
</span>"; ptr += currWeight; ptr += " gm</p>";

    // For Booking       ptr += "<p class='sensor'>";       ptr += "<span
class='sensor-labels'> LPG Status: </span>";
ptr += weightMsg;       ptr += "</p>";

    // For Gas leakage       ptr += "<p class='sensor'>";       ptr += "<I class='fas fa-
oil-can' style='color:#f7347a'></i>";       ptr += "<span class='sensor-labels'> Is LPG
Leaking: </span>";       ptr += gasMsg;       ptr += "</p>";

    // For Flame       ptr += "<p class='sensor'>";       ptr += "<I class='fas fa-
burn' style='color:#f7347a'></i>";       ptr += "<span class='sensor-labels'> Is
Flame: </span>";       ptr += flameMsg;       ptr += "</p>";

    ptr += "</div>";       ptr +=
"</div>";       ptr
+= "</div>";       ptr +=
"</div>";       ptr +=
"</div>";       ptr +=
"</body>";       ptr +=
"</html>";       return ptr;

}
```

```
  void initializeLoadCell()
{
    scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);

    Serial.println("Before setting up the scale:");
    Serial.print("read: \t\t");
    Serial.println(scale.read()); // print a raw reading from the ADC

    Serial.print("read average: \t\t");
    Serial.println(scale.read_average(20)); // print the average of 20 readings from the ADC

    Serial.print("get value: \t\t");
    Serial.println(scale.get_value(5)); // print the average of 5 readings from the ADC minus the tare weight (not set yet)

    Serial.print("get units: \t\t");
    Serial.println(scale.get_units(5), 1); // print the average of 5 readings from the ADC minus tare weight (not set) divided
                                           // by the SCALE parameter (not set yet)

    scale.set_scale(calibValue);
    // scale.set_scale(-471.497);                  // this value is obtained by calibrating the scale  with known weights; see the README for details
    scale.tare(); // reset the scale to 0
```

```
Serial.println("After setting up the scale:");

    Serial.print("read: \t\t");

Serial.println(scale.read()); // print a raw reading from the ADC

    Serial.print("read average: \t\t");
    Serial.println(scale.read_average(20)); // print the average of 20 readings from the ADC

    Serial.print("get value: \t\t");
    Serial.println(scale.get_value(5)); // print the average of 5 readings from the ADC minus the tare weight, set with tare()

   Serial.print("get units: \t\t");
    Serial.println(scale.get_units(5), 1); // print the average of 5 readings from the ADC minus tare weight, divided
                                           // by the SCALE parameter set with set_scale

    Serial.println("Readings:");  }
```

# CONCLUSION

This project has presented a unique vision of the concepts which are used in this field. It aims to promote technology innovation to achieve a reliable and efficient outcome from the various instruments. Experimental work has been carried out carefully. The result shows that higher efficiency is indeed achieved using the embedded system. With a common digitalized platform, these latest instruments will enable increased flexibility in control, operation, and expansion; allow for embedded intelligence, essentially foster the resilience of the instruments; and eventually benefit the customers with improved services, reliability, and increased convenience. This project presents the major features and functions of the various concepts that could be used in this field in detail through various categories. Since this initial work cannot address everything within the proposed framework and vision, more research and development efforts are needed to fully implement the proposed framework through a joint effort of various entities.

Let us summarize the key points and conclusions regarding IoT-based LPG cylinder level and leakage detection systems:

Automatic LPG Gas Leakage Detection and Cut-off System:

This system represents a pivotal advancement in household safety measures.

Traditional approaches involve basic toxic gas sensors coupled with alarm systems.

However, with IoT integration, this system revolutionizes safety standards.

Key features:

Gas Sensors: Detect gas leaks.

Cut-off Mechanism: Swiftly activates to prevent escalation.

Real-time Communication: Instant alerts sent to connected mobile devices.

Overall, this solution sets a new standard for proactive gas leakage detection and emergency response systems1.

IOT and ARDUINO-Based Gas Level and Leakage Detection System:

By detecting gas leakage, this system prevents LPG gas burst accidents at home.

Users receive alerts on their mobile devices when the LPG level is critically low (below 20%) or when there is a gas leakage.

Smart LPG Gas Level Monitoring and Leakage Detection System Using IoT:

This system triggers an alarm if gas leakage is detected.

It offers accurate output measured from LPG sensors, low maintenance, and low operating costs.

Smart LPG Gas Level Detection and Safety System Using IoT:

Monitors different aspects related to LPG cylinders.

Keeps consumers updated via a mobile application.

In conclusion, IoT-based LPG cylinder monitoring systems enhance safety, provide real-time awareness, and empower users to respond promptly to gas-related incidents. By integrating advanced technologies, we are making significant strides in household safety.

# REFERENCES

1. https://www.sciencedirect.com
2. https://www.researchgate.net
3. https://www.jetir.org
4. https://iotdesignpro
5. https://link.springer.com
6. https://ieeexplore.ieee.org