EXCEL ENGINEERING COLLEGE
(Autonomous)
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai
Accredited by NBA (AERO,CSE,ECE, MECH), NAAC with "A+" and Recognised by UGC (2f &12B)
KOMARAPALAYAM - 637303

EXPERIMENT – 1

MEASUREMENT OF TEMPERATURE AND HUMIDITY USING DHT11SENSOR WITH ARDUINO

AIM

The aim is to store and upload the measurement of temperature and humidity using dht11sensor with arduino.

APPARATUS REQUIRED

1.2 Hardware required

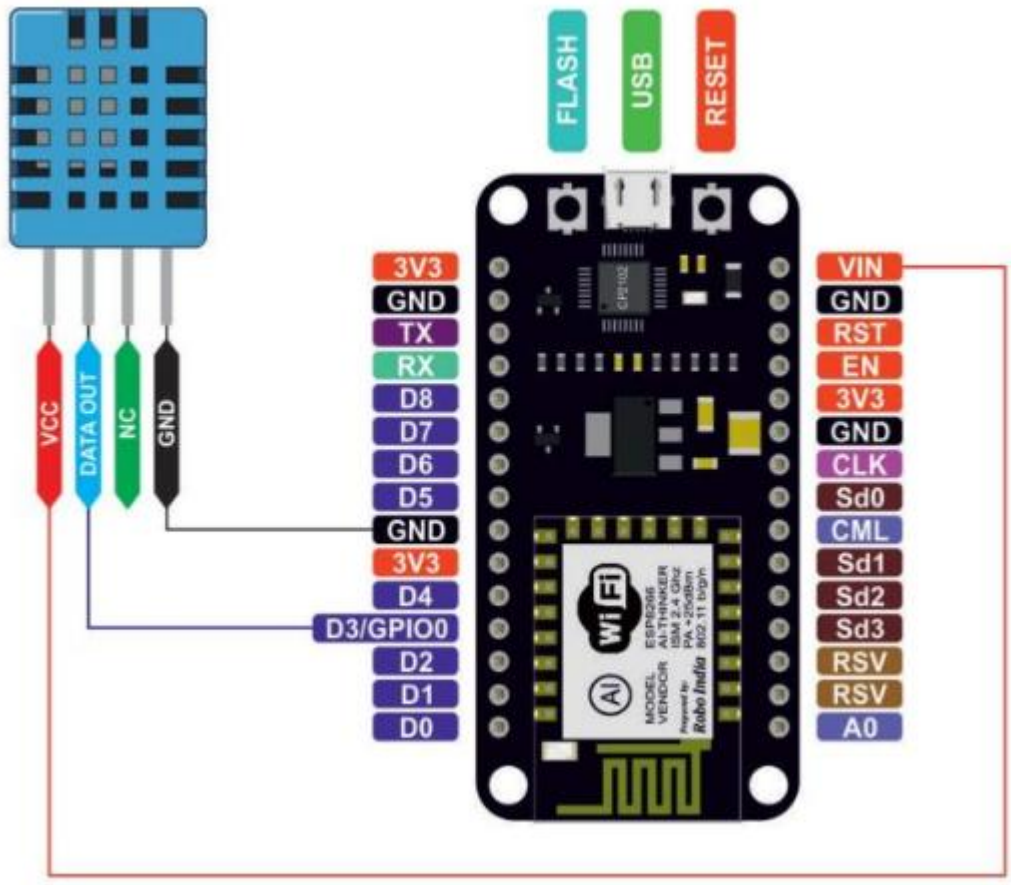
S.No.	Item	Quantity
1	NodeMCU	1
2	DHT11 Temperature and Humidity sensor	1
3	Jumper Male to female	3

2. Building Circuit

Make connection as mentioned ahead.

S.NO.	NodeMCU	DHT11
1.	Vin	VCC
2.	GND	GND
3.	D3	Data Out

ELECTRICAL CIRCUIT DIAGRAM



GETTING API KEY

1. Go to https://thingspeak.com/ and create an account if you do not have one. Login to your account.
2. Create a new channel by clicking on the button. Enter basic details of the channel. Than Scroll down and save the channel.
3. Channel Id is the identity of your channel. Note down this. Than go to API keys copy and paste this key to a separate notepad file will need it later.

CODE

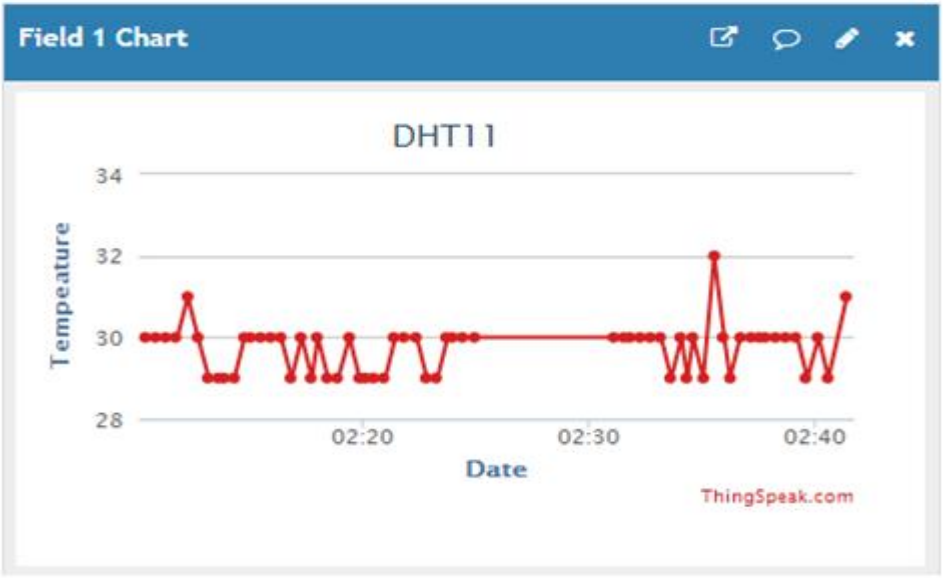
```
#include <DHT.h
#include <ESP8266WiFi.h>
```

```
String apiKey = "Your API of thingsspeak
const char *ssid =  "Your wifi Network name";
const char *pass =  "Network password";
const char* server = "api.thingspeak.com";

#define DHTPIN 0
DHT dht(DHTPIN, DHT11);

WiFiClient client;

void setup()
{
      Serial.begin(115200);
      delay(10);
      dht.begin();

      Serial.println("Connecting to ");
      Serial.println(ssid);


      WiFi.begin(ssid, pass);

     while (WiFi.status() != WL_CONNECTED)
     {
          delay(500);
          Serial.print(".");
     }
     Serial.println("");
     Serial.println("WiFi connected");

}

void loop()
{

     float h = dht.readHumidity();
     float t = dht.readTemperature();

           if (isnan(h) || isnan(t))
             {
                Serial.println("Failed to read from DHT sensor!");
                 return;
             }
```

```
                if (client.connect(server,80))
               {

                  String postStr = apiKey;
                  postStr +="&field1=";
                  postStr += String(t);
                  postStr +="&field2=";
                 postStr += String(h);
                  postStr += "\r\n\r\n";
                 client.print("POST /update HTTP/1.1\n");
                  client.print("Host: api.thingspeak.com\n");
                  client.print("Connection: close\n");
                  client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
                  client.print("Content-Type: application/x-www-form-urlencoded\n");
                  client.print("Content-Length: ");
                  client.print(postStr.length());
                  client.print("\n\n");
                  client.print(postStr);
                 Serial.print("Temperature: ");
                  Serial.print(t);
                  Serial.print(" degrees Celcius, Humidity: ");
                  Serial.print(h);
                  Serial.println("%. Send to Thingspeak.");
                 }
        client.stop();
       Serial.println("Waiting...");

  // thingspeak needs minimum 15 sec delay between updates, I've set it to 30 seconds
  delay(10000);
}
```

OUTPUT

Temperature:



Humidity:

RESULT

EXPERIMENT – 2

IMPLEMENTATION OF MOTION DETECTION SYSTEM USING PIR SENSOR

AIM

To design and implement a basic motion detection system using a PIR (Passive Infrared) sensor and Arduino, which detects motion in its range and triggers an output action such as lighting an LED or sounding a buzzer.

APPARATUS REQUIRED

- **PIR Sensor Module (e.g., HC-SR501)**

  Core sensor to detect motion based on infrared radiation.

- **Microcontroller Board**

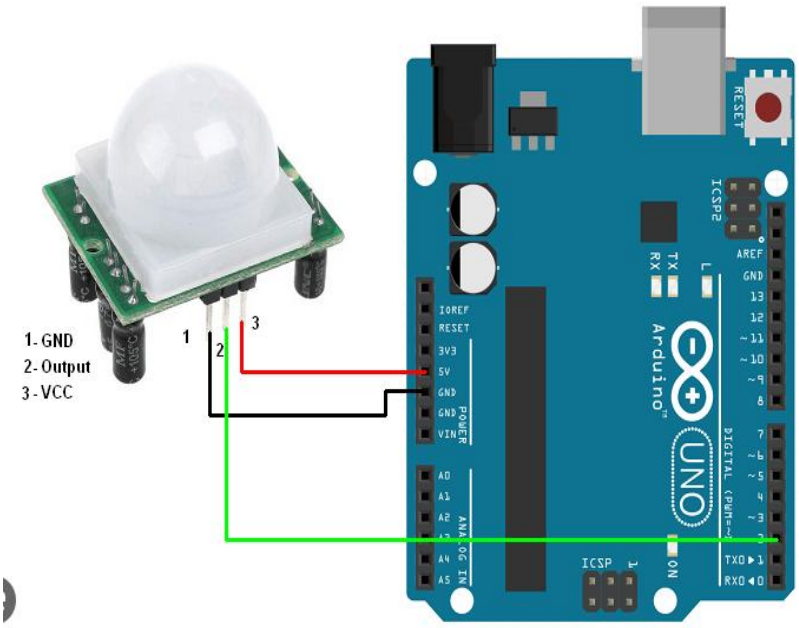
  Required to read sensor output and trigger responses.

  Examples:

  Arduino / Nano

  Raspberry Pi (for advanced projects)

ELECTRICAL CIRCUIT DIAGRAM



GETTING API KEY

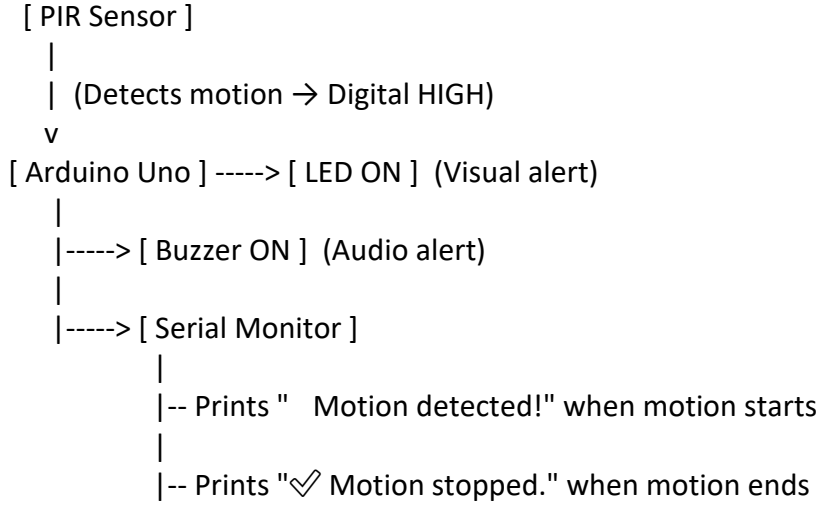
1. Go to https://ifttt.com, sign up.
2. Use Webhooks service → You'll get a unique API key (Webhook URL).
3. Format: https://maker.ifttt.com/trigger/{event_name}/with/key/{your_IFTTT_key}

CODE

```
 // Motion Detection System Using PIR Sensor
const int pirPin = 2;
const int ledPin = 13;
const int buzzerPin = 12;
int motionState = LOW;
int val = 0;

void setup() {
 pinMode(pirPin, INPUT);
 pinMode(ledPin, OUTPUT);
 pinMode(buzzerPin, OUTPUT);

 Serial.begin(9600);
 Serial.println("Motion Detection System Ready");
}

void loop() {
 val = digitalRead(pirPin);
 if (val == HIGH) {
  digitalWrite(ledPin, HIGH);
```

```
    digitalWrite(buzzerPin, HIGH);
    if (motionState == LOW) {
      Serial.println("   Motion detected!");
      motionState = HIGH;
    }
  } else {
    digitalWrite(ledPin, LOW);
    digitalWrite(buzzerPin, LOW);
    if (motionState == HIGH) {
      Serial.println("✅ Motion stopped.");
      motionState = LOW;
    }
  }

  delay(100); // Small delay to reduce false triggering
}
```

OUTPUT



RESULT

EXPERIMENT – 3

DETECTION OF GAS LEAKAGE USING MQ-2 SENSOR AND ALERT MECHANISM

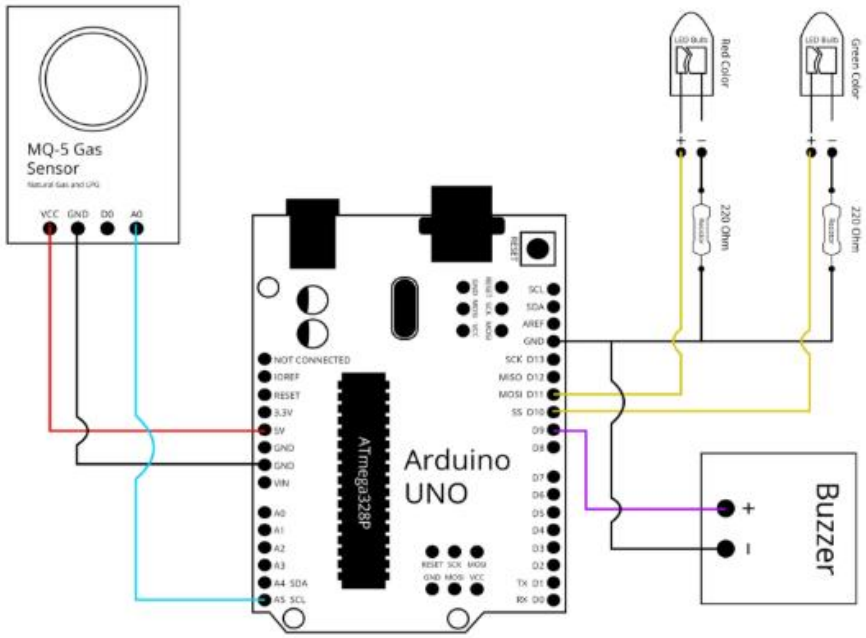
AIM

To design and implement a gas leakage detection system using the MQ-2 sensor that can accurately detect the presence of hazardous gases and trigger an alert mechanism to warn users of potential gas leaks, thereby enhancing safety and preventing accidents .

APPARATUS REQUIRED

1. MQ-2 Gas Sensor – To detect the presence of gases like LPG, methane, propane, hydrogen, smoke, etc.
2. Microcontroller (e.g., Arduino Uno / Arduino Nano / ESP8266) – To read sensor data and control the alert system.
3. Buzzer – To provide an audible alert when gas leakage is detected.
4. LED (optional) – For visual indication of gas leakage.

ELECTRICAL CIRCUIT DIAGRAM



GETTING API KEY

1. Go to thingspeak.com.
2. Register for a free account.
3. Create a new channel for your gas sensor data.
4. Under channel settings, you will see your **Write API Key**.
5. Use this key in your Arduino sketch to send gas sensor data.

CODE

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
int echo = 3;
int trigger =  2;
int buzzer=7;
long duration=0;
int distance=0;

void setup()
{
       lcd.begin();
       lcd.backlight();
       pinMode(trigger, OUTPUT);
       pinMode(echo, INPUT);
       pinMode(buzzer, OUTPUT);

}
void loop()
{
duration=0;
distance=0;

  digitalWrite(trigger,LOW);
  delayMicroseconds(2);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);
  duration = pulseIn(echo, HIGH);
  distance = duration * 340 / (2 * 10000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("DIS: ");lcd.print(distance); lcd.print(" cm");
  lcd.setCursor(0,1);
 if(distance<=30)
{

   digitalWrite(buzzer,HIGH);
   lcd.print("OBJECT DETECTED");

}
else
{
   digitalWrite(buzzer,LOW);
   lcd.print("NO OBJECT");
}
delay(1500);
                                                    }
```

OUTPUT

When distance > 30 cm:

DIS: 55 cm
NO OBJECT
(Buzzer is OFF)

When distance ⩽ 30 cm:

DIS: 25 cm
OBJECT DETECTED
(Buzzer is ON)

RESULT

EXPERIMENT – 4

DESIGN OF AUTOMATIC LIGHT CONTROL SYSTEM USING LDR SENSOR

AIM

To design and implement an automatic light control system using an LDR (Light Dependent Resistor) sensor that can detect ambient light levels and automatically switch electrical lights ON or OFF based on the surrounding brightness.

APPARATUS REQUIRED

1. **LDR (Light Dependent Resistor)**

   To detect the intensity of ambient light.

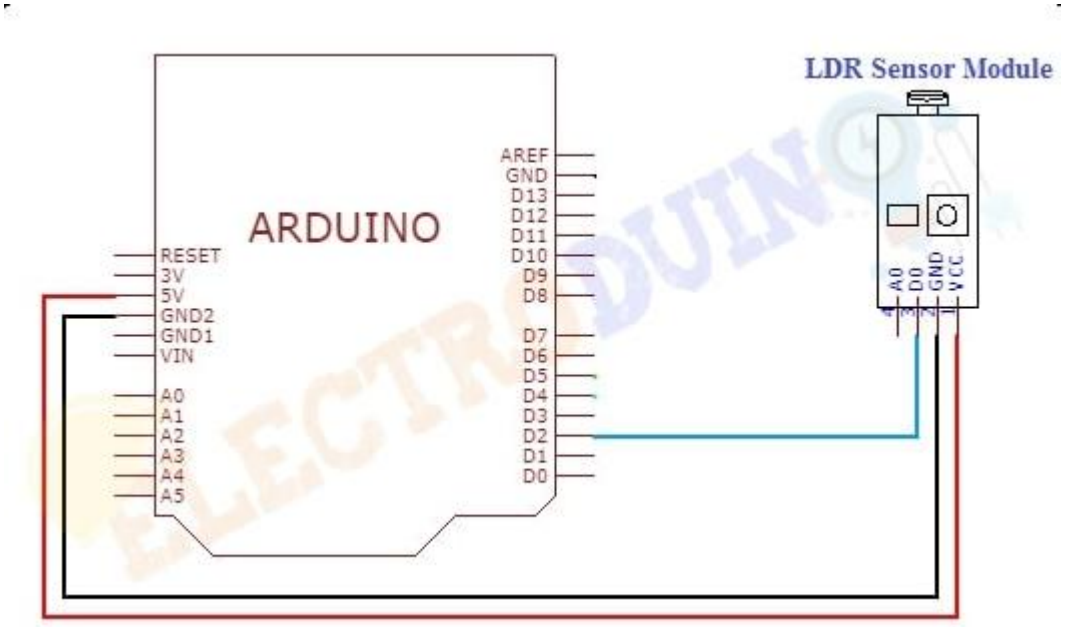
2. **Arduino Uno**

   To process the LDR input and control the light (optional for analog automation).

3. **LED**

   Represents the light to be controlled.

ELECTRICAL CIRCUIT DIAGRAM



GETTING API KEY

1. Go to: https://thingspeak.com
2. Sign up / Log in.
3. Click on **“Channels” → “New Channel”**.
4. Name your channel (e.g., "LDR Sensor Data").
5. Add a field (e.g., “Light Intensity”).
6. Save the channel.
7. Go to the **API Keys** tab.
8. Copy the **Write API Key** — use it in your Arduino/ESP code to send data.

CODE

```
        #include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
int lightsensor=A2;
int led = 2;
int lightvalue=0;
void setup()
{
pinMode(led , OUTPUT);
lcd.begin();
lcd.backlight();
}
void loop()
{
lightvalue= analogRead(lightsensor);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("LIGHT: "); lcd.print(lightvalue);
if (lightvalue < 950)
{
digitalWrite(led, HIGH);
lcd.setCursor(0,1);
lcd.print("LAMP: ON");
```

```
}
else
{
digitalWrite(led, LOW);
lcd.setCursor(0,1);
lcd.print("LAMP: OFF");
}
delay(3000);
}
```

OUTPUT

```
right reading = 3.60
left reading = 3.52
right reading = 3.52
left reading = 3.62
right reading = 3.62
left reading = 3.57
```

RESULT

EXPERIMENT – 5

DEVELOPMENT OF SMART IRRIGATION SYSTEM USING SOIL MOISTURE SENSOR

AIM

To design and develop a smart irrigation system that optimizes water usage by automatically controlling irrigation based on real-time soil moisture levels, thereby promoting efficient water management in agriculture.

APPARATUS REQUIRED

1. **Soil Moisture Sensor**

   Measures the water content in soil.

2. **Microcontroller (e.g., Arduino UNO / NodeMCU / ESP32**

   Processes sensor data and controls outputs.

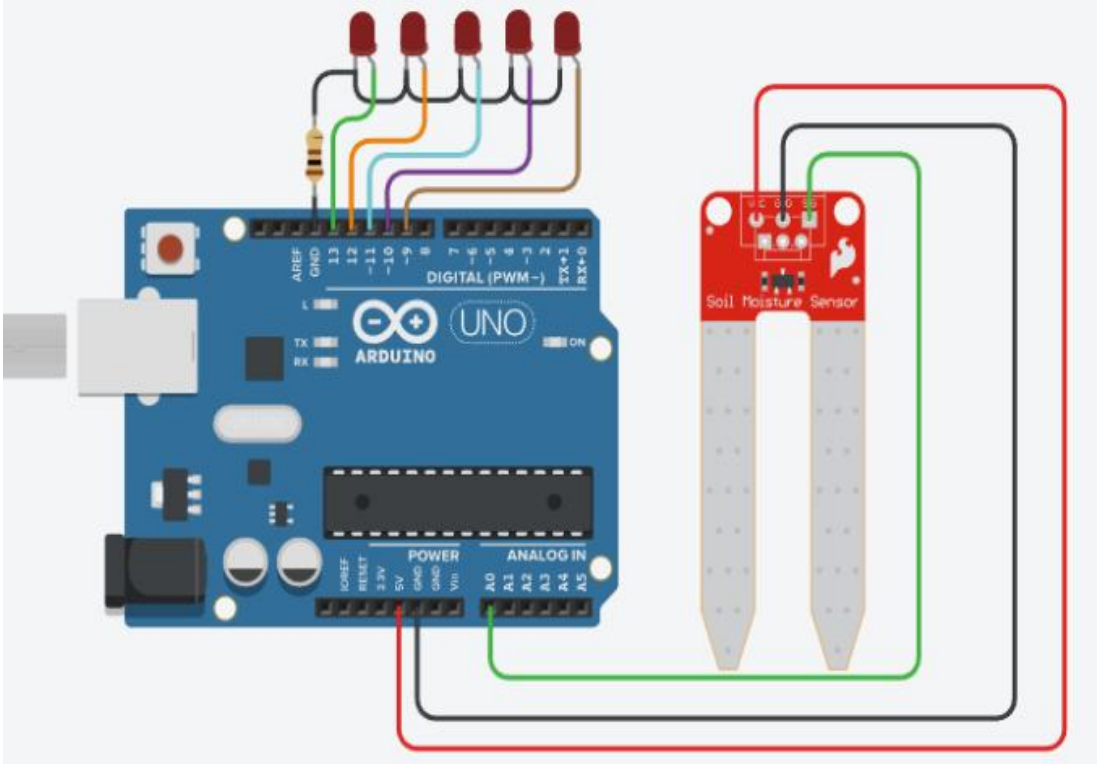
3. **Breadboard / PCB (Optional)**

   For prototyping the circuit.

4. **LCD Display / Serial Monitor (optional)**

   To display sensor readings.

ELECTRICAL CIRCUIT DIAGRAM



GETTING API KEY

1. Go to: https://thingspeak.com
2. Create an account or log in.
3. Create a new **channel**.
4. Go to **API Keys** tab.
5. Copy the **Write API Key** – this is used in your Arduino/ESP code to send data.

CODE

```
      #include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
int moisturesensor=A3;
int relay=3;
int moistvalue=0;

void setup()
{
  lcd.begin();
  lcd.backlight();
  Serial.begin(9600);
  pinMode(moisturesensor,INPUT);
  pinMode(relay,OUTPUT);
  }
void loop()
{
   moistvalue=analogRead(moistvalue);
// Serial.println(irvalue);
if (moistvalue>=800)
{
  digitalWrite(relay,HIGH);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("MOIST VALUE:");lcd.print(moistvalue);
```

```
  lcd.setCursor(0,1);
  lcd.print("TURN ON FAN");



}

else
{

  digitalWrite(relay,LOW);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("MOIST VALUE:");lcd.print(moistvalue);
  lcd.setCursor(0,1);
  lcd.print("TURN OFF FAN");

}
delay(2000);
}
```

OUTPUT

```
Reading From the Sensor
Mositure : 59%
Mositure : 72%
Mositure : 76%
Mositure : 78%
```

RESULT

EXPERIMENT – 6

MONITORING OF WATER LEVEL USING ULTRASONIC SENSOR WITH THRESHOLD ALERT

AIM

To design and implement a water level monitoring system using an ultrasonic sensor that continuously measures the water level in a tank or reservoir and generates alerts when the water level crosses predefined threshold limits, ensuring efficient water management and preventing overflow or shortage.

APPARATUS REQUIRED

1. **Ultrasonic Sensor (e.g., HC-SR04)**

   Measures distance from the water surface to the sensor.

2. **Microcontroller (e.g., Arduino UNO / NodeMCU / ESP32)**

   Processes sensor data and triggers alerts.

3. **Power Supply (USB Cable, Battery Pack, or Adapter)**

   Powers the microcontroller and connected components.

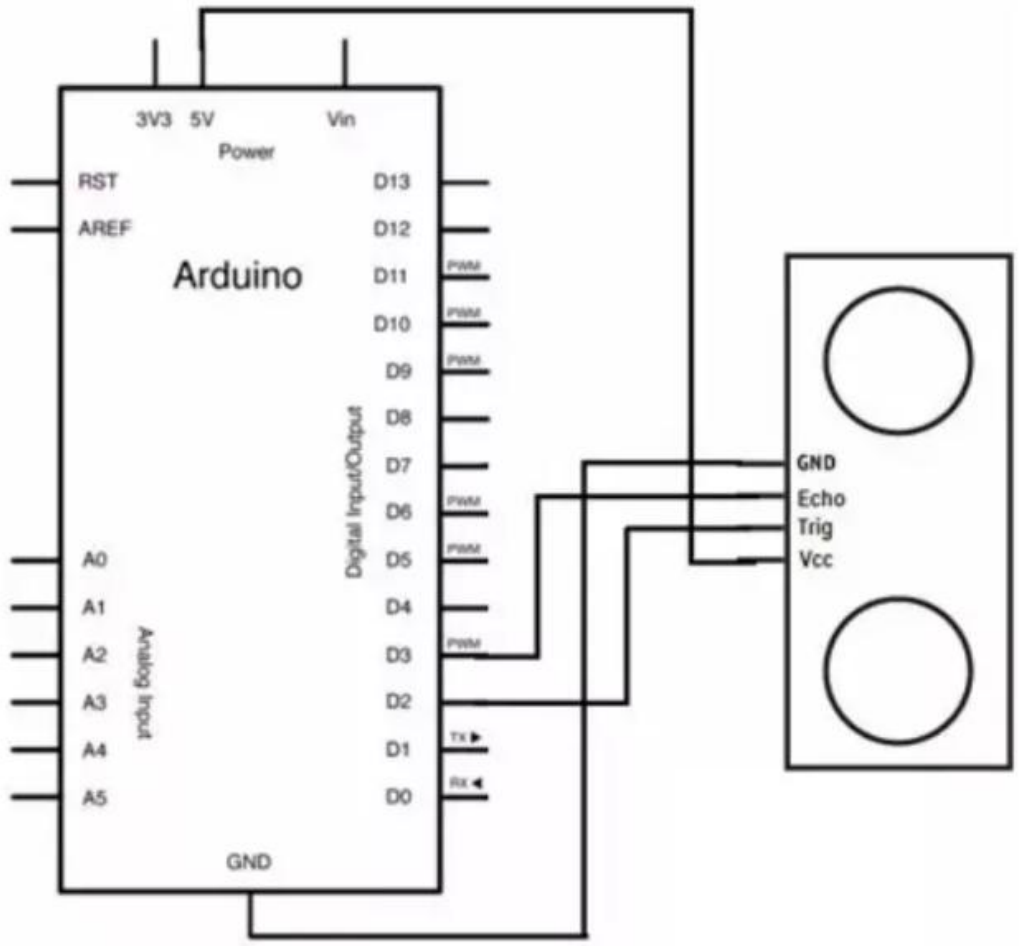
4. **Buzzer / Alarm Module**

   Produces sound when water level crosses threshold.

5. **LED**

   Visual indicators for low, normal, or high water levels

ELECTRICAL CIRCUIT DIAGRAM



GETTING API KEY

1. Go to https://thingspeak.com
2. Click **Sign Up** and create a free account (you'll need a MathWorks account).
3. Confirm your email and log in.
4. Once logged in, click on "Channels" > "New Channel"
5. Click "Save Channel"
6. Go to your channel dashboard
7. Click the "API Keys" tab
8. You'll see two keys:

- Write API Key: for sending (uploading) data
- **Read API Key**: for reading (downloading) data
- Copy the **Write API Key**

CODE

```
    #include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
int echo = 3;
int trigger =  2;
int buzzer=7;
long duration=0;
int distance=0;

void setup()
{
       lcd.begin();
       lcd.backlight();
       pinMode(trigger, OUTPUT);
       pinMode(echo, INPUT);
       pinMode(buzzer, OUTPUT);

}
void loop()
{
duration=0;
distance=0;

  digitalWrite(trigger,LOW);
  delayMicroseconds(2);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);
  duration = pulseIn(echo, HIGH);
  distance = duration * 340 / (2 * 10000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("DIS: ");lcd.print(distance); lcd.print(" cm");
  lcd.setCursor(0,1);
 if(distance<=30)
{

   digitalWrite(buzzer,HIGH);
   lcd.print("OBJECT DETECTED");

}
else
{
   digitalWrite(buzzer,LOW);
   lcd.print("NO OBJECT");
}
delay(1500);
}
```

OUTPUT

```
Distance = 10.00 cm
Distance = 10.00 cm
Distance = 8.00 cm
Distance = 8.00 cm
Distance = 7.00 cm
Distance = 7.00 cm
```

RESULT