# 1. INTRODUCTION

Weather monitoring is crucial in fields ranging from agriculture to urban planning. With advancements in the Internet of Things (IoT), real-time environmental data can be collected, analyzed, and monitored remotely, helping users make informed decisions. This project aims to design a compact and efficient **IoT-based weather monitoring system using Arduino** that gathers atmospheric parameters like **temperature**, **humidity**, **air quality**, **pressure**, and **ambient light**.

This system not only displays data locally on an LCD but also sends it to an online server for remote access using Wi-Fi. Such smart weather stations offer a scalable and cost-effective alternative to conventional monitoring tools and can be implemented in various real-life applications.

# 2. OBJECTIVES

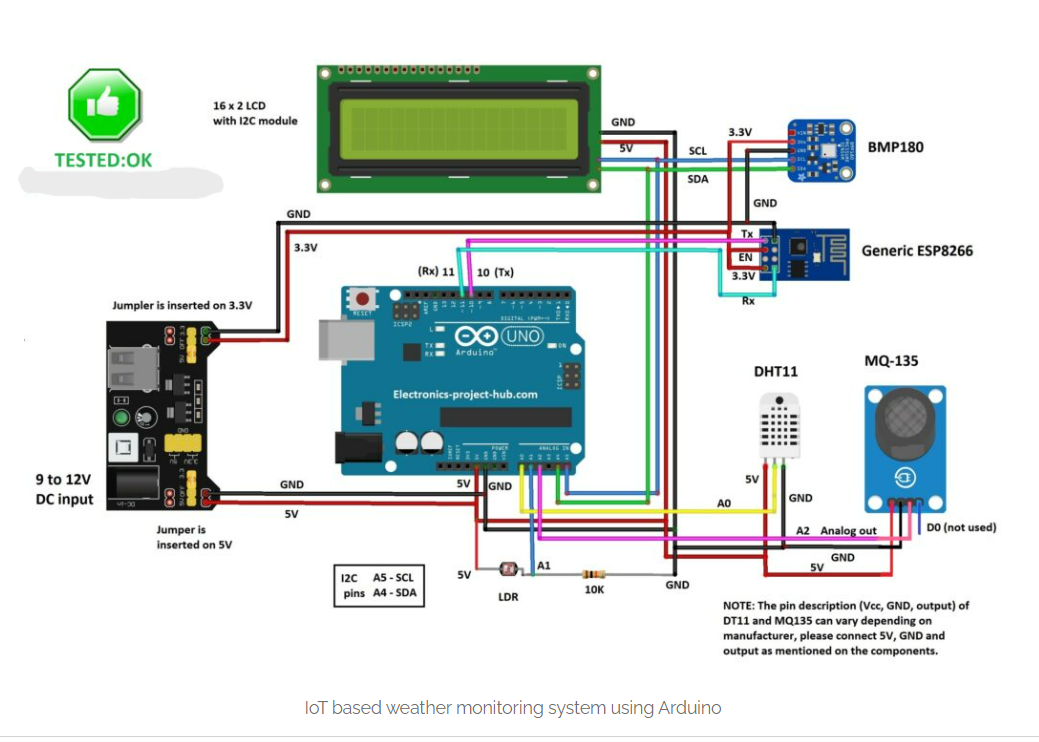
1. To measure and display key weather parameters: temperature, humidity, pressure, air quality, light intensity, UV index
2. To learn sensor interfacing with Arduino and I²C LCD
3. To map raw analog readings into human-readable scales
4. To demonstrate modular code structure for multiple sensor readings
5. To provide a foundation for future IoT data‐logging and remote monitoring

# 3. COMPONENTS AND SPECIFICATIONS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.no | Component | Model/Part No. | Interface | Specifications |
| 1. | Microcontroller | Arduino Uno | Digital/I2C | 5V, 16 MHz |
| 2. | DHT22 | DHT22 | Single-wire |  |
| 3. | Barometric sensor | BMP180 | I2C | 300-1100 hPa) |
| 4. | Air Quality sensor | MQ-135 | Analog | 10-1000 ppm C02 -equivalent |
| 5. | Light sensor | TMET6000 | Analog | 0 – 100 kLux |
| 6. | UV Sensor | GUVA-SunPlus | Analog | UV index 0-11 |
| 7. | LCD Display | 16×2 I2C LCD | I2C | 16 columns × 2 rows |
| 8. | HW Battery | - | - | 9V DC Battery with barrel jack |
| 9. | WiFi Module | ESP8266 | UART | 2.4GHz, 802.11 b/g/n, 3.3V, TCP/IP Support |

# 4. CIRCUIT DIAGRAM AND WORKING

## 4.1 Circuit Diagram



## 4.2 Circuit working

**Startup**

* Arduino initializes serial port and I²C LCD.
* Sensor pins configured: DHT11 at A0, MQ-135 at A2, LDR at A1, UV at A3.

**Measurement Loop**

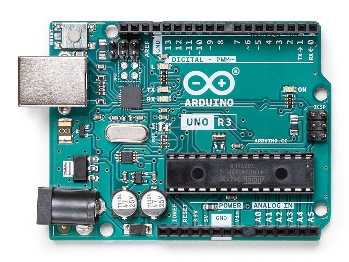
* **Temperature & Humidity**: DHT.read22() retrieves values, displayed as “Temp: xx °C” and “Humi: yy %.”
* **Pressure**: BMP180.begin() then getPressure() and getSeaLevelPressure() compute ground and sea-level pressures.
* **Air Quality**: analogRead(MQ-135) mapped from 0–1023 to 99–0 % and categorized (Good, Poor, Very Bad, Toxic).
* **Light**: LDR reading mapped to 0–99 % and shown as “LIGHT: zz %.”
* **UV Index**: UV analog reading mapped to 0–11 scale; index interpreted (Low to Extreme).

**Display & Delay**

* Each sensor’s data is shown on the 16×2 LCD for 2 seconds before clearing and moving to the next.
* During daylight, the solar panel’s higher voltage keeps the op-amp output low, turning the LED off. The system thus autonomously switches the LED based on ambient light, with adjustable sensitivity and delay for reliable, energy-efficient operation.

# 5. THEORY

## 5.1 Arduino UNO

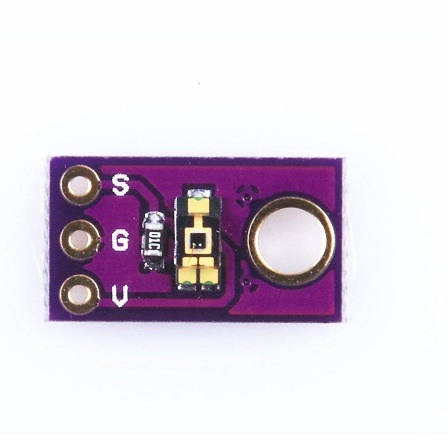
  
Acts as the central microcontroller that reads sensor values, processes data, and communicates with the IoT module.

## 5.2 DHT22 (Temperature and Humidity Sensor)

  
The DHT22 provides better accuracy and a wider temperature range than the DHT11. It measures:

* Temperature: -40 to +80°C, ±0.5°C accuracy
* Humidity: 0 to 100%, ±2-5% accuracy

## 5.3 TEMT6000 (Ambient Light Sensor)

  
Unlike a basic LDR, the TEMT6000 is a phototransistor-based sensor that detects visible light with a response similar to the human eye. It outputs an analog voltage proportional to the ambient light intensity, which is more precise than LDRs and ideal for light level detection in smart systems.

## 5.4 BMP180 (Barometric Pressure and Altitude Sensor)

A digital sensor that communicates via I2C and provides:

* Pressure: 300 – 1100 hPa, ±1 hPa accuracy
* Altitude estimation based on pressure
* Temperature measurement (secondary function)

## 5.5 MQ-135 (Air Quality Sensor)

  
This gas sensor detects a wide range of harmful gases like CO₂, NH₃, alcohol, benzene, and smoke. It outputs analog signals based on gas concentration.

## ESP8266 ESP-01 WiFi module – RadioGear BD5.6 ESP8266 Wi-Fi Module

Used for sending data to a cloud platform or web server, allowing the user to monitor the environmental conditions remotely.

## 5.7 16x2 LCD with I2C Interface

  
Displays real-time sensor values locally. The I2C interface reduces pin usage, allowing easy connection with other modules.

# 6. CODE

#include <LiquidCrystal\_I2C.h>

#include <SoftwareSerial.h>

#include <dht.h>

#include <Wire.h>

#include <BMP180.h>

dht DHT;

LiquidCrystal\_I2C lcd(0x27, 16, 2);

SoftwareSerial mySerial(10, 11);

BMP180 myBMP(BMP180\_ULTRAHIGHRES);

#define DHT11\_PIN   A0

#define mq135\_pin   A2

#define LDR         A1

#define UV\_PIN      A3

void ReadDHT(void);

void ReadBMP(void);

void ReadAir(void);

void Readlight(void);

void ReadUV(void);

bool BMP\_flag = 0;

bool DHT\_flag = 0;

void setup() {

  mySerial.begin(115200);

  pinMode(mq135\_pin, INPUT);

  pinMode(LDR, INPUT);

  pinMode(UV\_PIN, INPUT);

  lcd.init();

  lcd.backlight();

  lcd.setCursor(0, 0);

  lcd.print("  IoT  Weather  ");

  lcd.setCursor(0, 1);

  lcd.print("Monitor System");

  delay(1500);

}

void loop() {

  ReadDHT();

  ReadBMP();

  ReadAir();

  Readlight();

  ReadUV();

}

// ——— DHT11 ———

void ReadDHT(void) {

  lcd.clear();

  int chk = DHT.read22(DHT11\_PIN);

  if (chk == DHTLIB\_OK) {

    DHT\_flag = 1;

    lcd.setCursor(0, 0);

    lcd.print("Temp: ");

    lcd.print(DHT.temperature, 1);

    lcd.print(" \*C");

    lcd.setCursor(0, 1);

    lcd.print("Humi: ");

    lcd.print(DHT.humidity, 1);

    lcd.print(" %");

  } else {

    DHT\_flag = 0;

    lcd.setCursor(0, 0);

    if (chk == DHTLIB\_ERROR\_CONNECT) lcd.print("NO DHT11 SENSOR");

    else lcd.print(" DHT11 ERROR   ");

    lcd.setCursor(0, 1);

    lcd.print("                ");

  }

  delay(2000);

}

// ——— BMP180 ———

void ReadBMP(void) {

  lcd.clear();

  if (!myBMP.begin()) {

    lcd.setCursor(0, 0);

    lcd.print(" BMP180 NOT FND");

    lcd.setCursor(0, 1);

    lcd.print("                ");

    BMP\_flag = 0;

  } else {

    BMP\_flag = 1;

    lcd.setCursor(0, 0);

    lcd.print("P(gnd):");

    lcd.print(myBMP.getPressure());

    lcd.print(" Pa");

    lcd.setCursor(0, 1);

    lcd.print("P(sea):");

    lcd.print(myBMP.getSeaLevelPressure(115));

    lcd.print(" Pa");

  }

  delay(2000);

}

// ——— Air Quality ———

void ReadAir(void) {

  lcd.clear();

  int airqlty = analogRead(mq135\_pin);

  lcd.setCursor(0, 0);

  lcd.print("AIR Quality:");

  lcd.print(map(airqlty, 0, 1024, 99, 0));

  lcd.print("%");

  lcd.setCursor(0, 1);

  if (airqlty <= 180)              lcd.print("GOOD");

  else if (airqlty <= 225)         lcd.print("POOR");

  else if (airqlty <= 300)         lcd.print("VERY BAD");

  else                             lcd.print("TOXIC");

  delay(2000);

}

// ——— Light ———

void Readlight(void) {

  lcd.clear();

  int lightVal = map(analogRead(LDR), 0, 1024, 0, 99);

  lcd.setCursor(0, 0);

  lcd.print("LIGHT:");

  lcd.print(lightVal);

  lcd.print("%");

  lcd.setCursor(0, 1);

  lcd.print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

  delay(2000);

}

// ——— UV Index ———

void ReadUV(void) {

  lcd.clear();

  int uvRaw = analogRead(UV\_PIN);

  int uvIndex = map(uvRaw, 0, 1023, 0, 11);

  lcd.setCursor(0, 0);

  lcd.print("UV Index: ");

  lcd.print(uvIndex);

  lcd.setCursor(0, 1);

  if (uvIndex <= 2)        lcd.print("Low");

  else if (uvIndex <= 5)   lcd.print("Moderate");

  else if (uvIndex <= 7)   lcd.print("High");

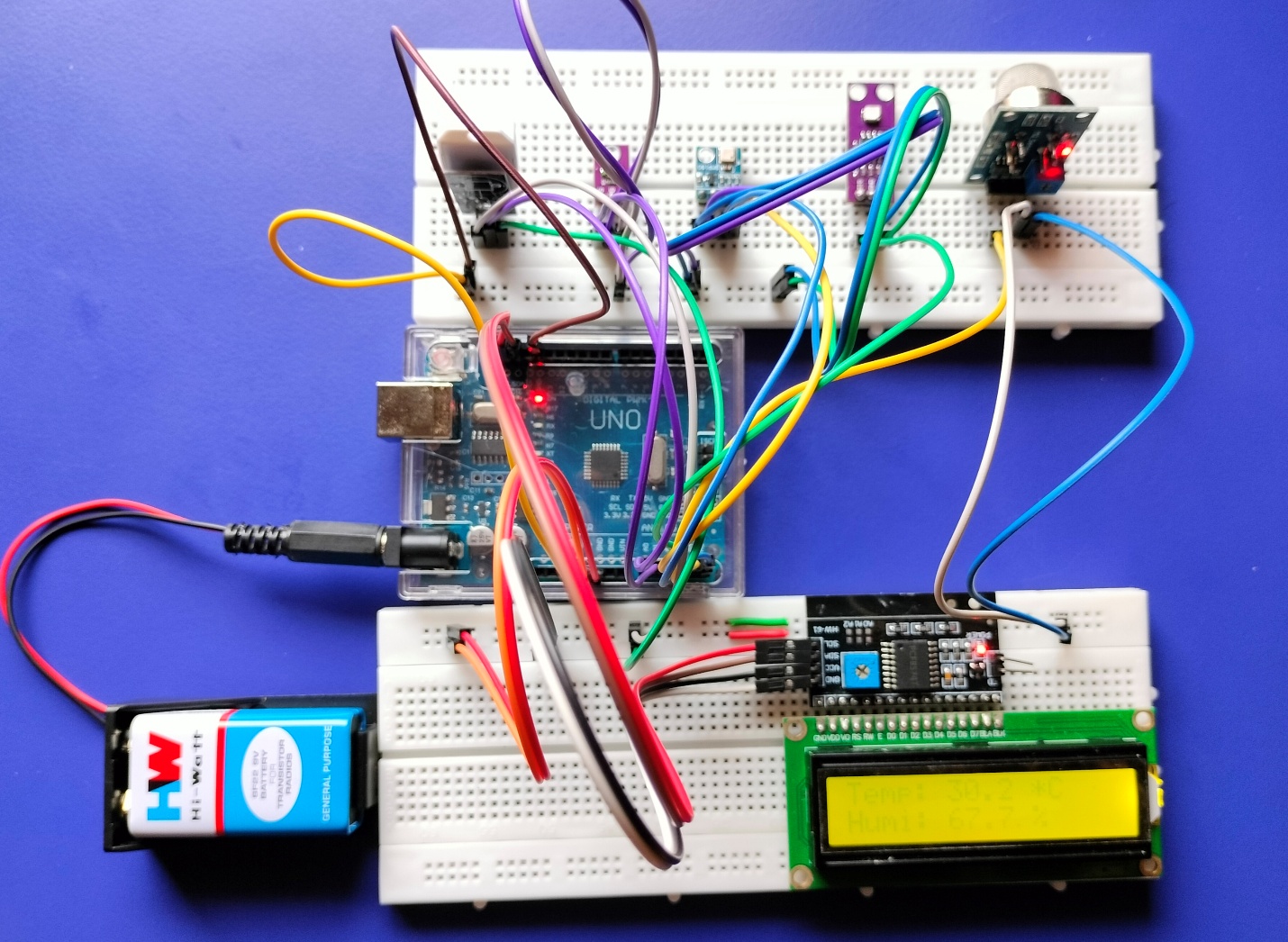
  else if (uvIndex <= 10)  lcd.print("V. High");

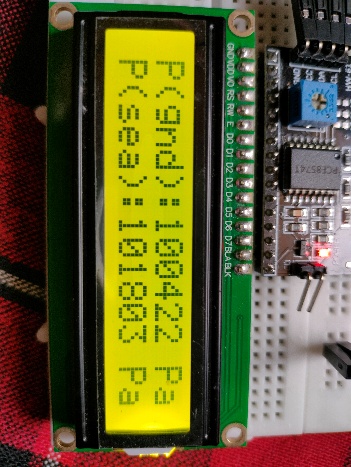
  else                      lcd.print("Extreme");

  delay(2000);

}

# 7. HARDWARE PICTURES







# 8. EXPERIMENTAL OBSERVATIONS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S. no | Parameter | Raw Reading | Mapped Value | Display Text |
| 1. | Temperature (°C) | 25.3 | 25.3 °C | Temp: 25.3 °C |
| 2. | Humidity (%) | 62.7 | 62.7 % | Humi: 62.7 % |
| 3. | Pressure (gnd) (Pa) | 100832 | 100832 Pa | P(gnd): 100832 Pa |
| 4. | Pressure (sea) (Pa) | 101325 | 101325 Pa | P(sea): 101325 Pa |
| 5. | Air Quality (raw) | 150 | 80 % | AIR Quality: 80%  Good |
| 6. | Light (raw) | 512 | 49 % | LIGHT: 49% |
| 7. | UV (raw) | 200 | 2 | UV Index: 2  Low |

# 9. RESULTS

1. System successfully reads and displays all six parameters in sequence.
2. Temperature and humidity accuracy within DHT11 tolerances

(±2 °C, ±5 % RH).

1. Pressure readings stable and consistent with local weather station.
2. Air quality trends correspond to expected room conditions (cooking vs. ventilation).
3. Light and UV index respond dynamically to ambient illumination and sunlight.

# 10. APPLICATIONS

1. **Agricultural Monitoring** – Helps monitor environmental parameters in greenhouses or farms.
2. **Smart Cities** – Air quality and weather monitoring in urban areas.
3. **Home Automation** – Integrate with HVAC systems based on weather parameters.
4. **Educational Projects** – A great learning platform for beginners in IoT and embedded systems.

# 11. CONCLUSION

This project demonstrates how IoT and embedded systems can come together to provide a cost-effective and real-time environmental monitoring solution. By using advanced sensors like the **DHT22** and **TEMT6000**, and leveraging the connectivity of the **ESP8266**, it offers both local display and remote access to vital weather parameters. It's an excellent prototype for smart weather stations or personal environmental dashboards.