



## **LAB REPORT**

**Course Title:** Embedded Systems and IoT Lab

**Course Code:** CSE-234

**Topic Name:** Temperature & Humidity Sensors  
interfacing with Arduino.

**Submitted To:**

Avizit Nandi

Lecturer

Department of CSE

Daffodil International University

**Submitted By:**

Name: Raihanul Islam Rahat

ID: 221-15-5150

Section: 61-T2

Department of CSE

Daffodil International University

<b>Date of Submission: 07-08-2025</b>
---------------------------------------

## Lab Report No: 07

### Objective:

1. To learn how to interface a temperature and humidity sensor (e.g., DHT11/DHT22) with an Arduino Uno.
2. To understand how to read environmental data (temperature in °C/°F and humidity in %) from the sensor.
3. To visualize sensor data on an LCD display for real-time monitoring.
4. To apply conditional logic to control an LED based on sensor readings (e.g., temperature threshold).
5. To simulate and verify the sensor circuit and code using Tinkercad before physical implementation.
- 6.

### Introduction:

In the modern era of embedded systems and the Internet of Things (IoT), environmental monitoring plays a crucial role in a wide range of applications, including smart homes, agriculture, weather forecasting, and industrial automation. Temperature and humidity are two fundamental environmental parameters that must be accurately measured and monitored for ensuring system stability and safety.

This experiment focuses on interfacing a digital temperature and humidity sensor (such as the DHT11 or DHT22) with an Arduino Uno microcontroller. The sensor collects data from the surrounding environment, which is then processed and displayed on an LCD screen. Additionally, an LED is used to provide a simple visual indication when specific environmental thresholds are exceeded, such as high temperature.

### Theory:

Temperature and humidity sensors are electronic devices that detect and measure environmental temperature and relative humidity. In this experiment, a common sensor like **DHT11** or **DHT22** is typically used, which combines both temperature and humidity sensing in a single module.

#### 1. DHT11

- These sensors use a **capacitive humidity sensor** and a **thermistor** to measure the surrounding air.
- The sensor provides **digital output**, meaning no analog-to-digital conversion is needed.
- DHT11 is cheaper with lower accuracy and a narrower range, while DHT22 offers higher precision and a wider operating range.

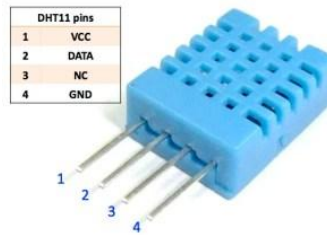


Figure 1: DHT11 sensor

## 2. Arduino Uno:

- A widely used **microcontroller board** based on the ATmega328P.
- It reads data from the sensor using digital pins and processes it using prewritten.
- It can control output devices like LEDs and displays based on sensor input.

## 3. LCD Display (16x2):

- Used to display real-time readings of temperature (in °C and °F) and humidity (%).
- Connected to the Arduino via multiple digital pins or using I2C protocol.



Figure 2: LCD Display

## 4. LED Indicator:

- An **LED** is used to give visual feedback based on a predefined condition (e.g., turns on when temperature > 30°C).
- Controlled by a digital output pin of the Arduino.

## 5. Working Principle:

- The sensor continuously measures temperature and humidity.
- The Arduino reads this data, processes it, and then:
  - Displays the result on the LCD screen.
  - Activates or deactivates the LED depending on threshold condition

## Equipment:

- Arduino Uno
- USB Cable
- LED
- Resistor
- Breadboard
- Jumper Wires
- Temperature & Humidity sensor
- LCD Display

## Circuit Diagram:

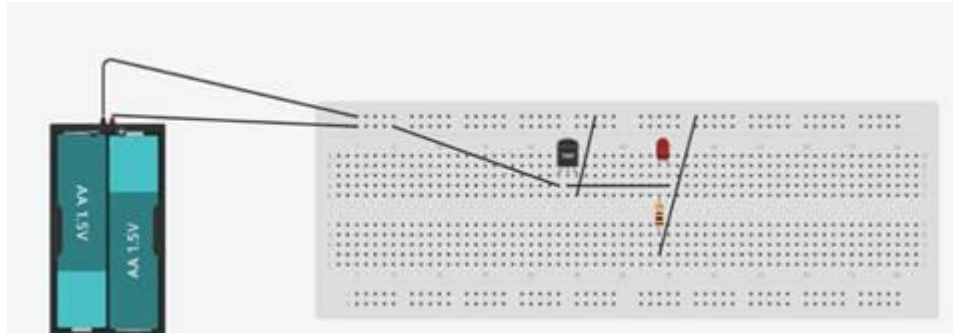


Figure-3: Simulated Output from Tinkercad (using LED, Register, Arduino & Sensor)

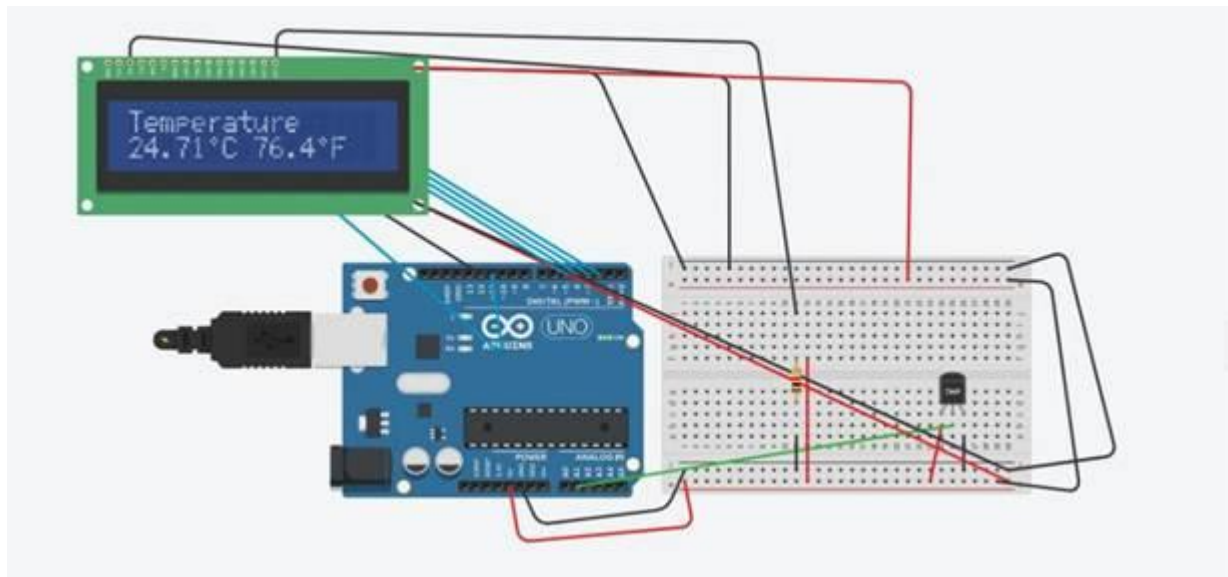
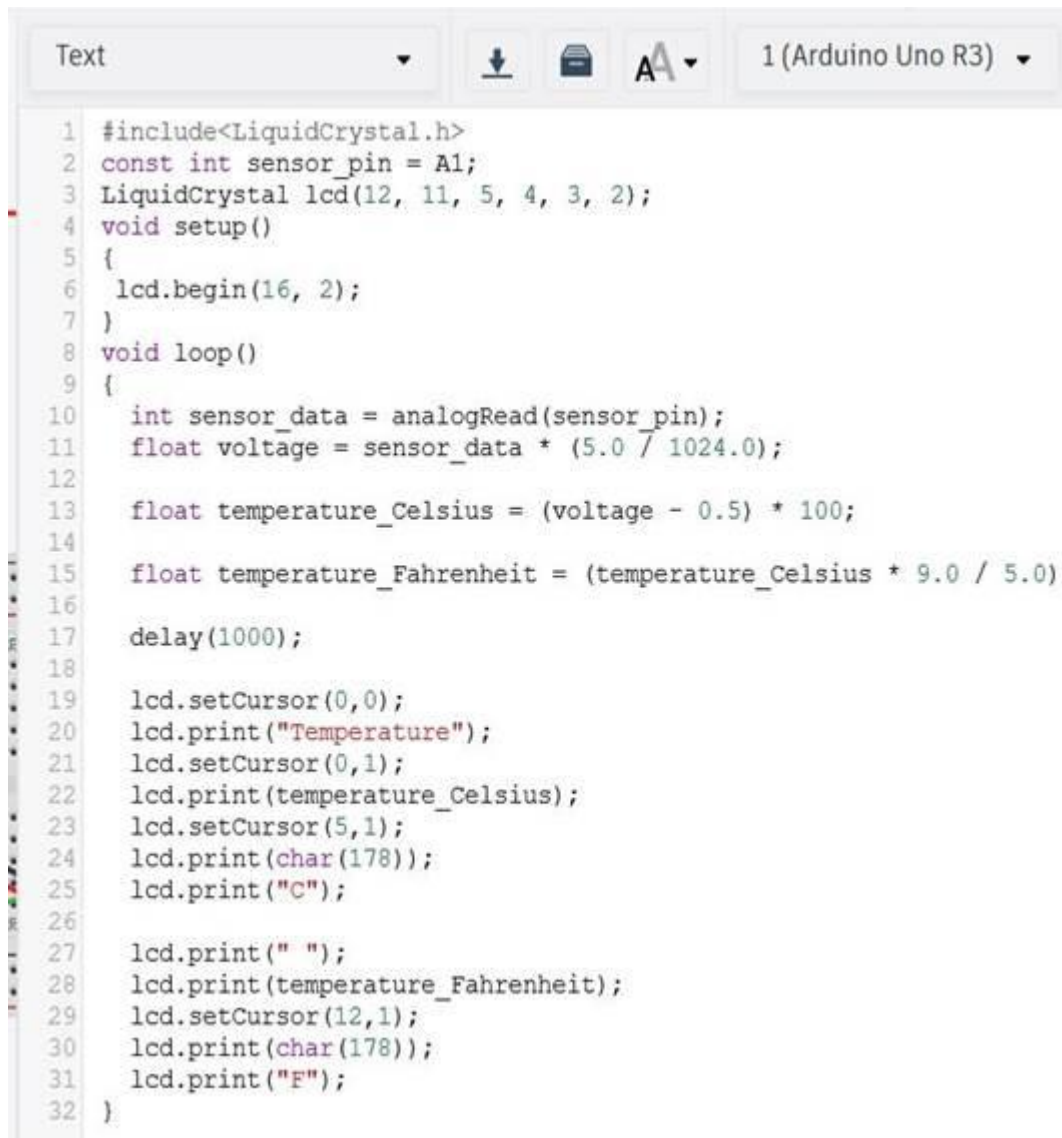


Figure-4: Arduino Code for Temperature & humidity sensor (using LED, Register, Arduino & Sensor)

Code:



The image shows a screenshot of an Arduino IDE code editor. At the top, there is a toolbar with a 'Text' dropdown menu, a download icon, a save icon, a font size selector set to 'A', and a board selector set to '1 (Arduino Uno R3)'. The code is written in C++ and is as follows:

```
1 #include<LiquidCrystal.h>
2 const int sensor_pin = A1;
3 LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
4 void setup()
5 {
6   lcd.begin(16, 2);
7 }
8 void loop()
9 {
10   int sensor_data = analogRead(sensor_pin);
11   float voltage = sensor_data * (5.0 / 1024.0);
12
13   float temperature_Celsius = (voltage - 0.5) * 100;
14
15   float temperature_Fahrenheit = (temperature_Celsius * 9.0 / 5.0)
16
17   delay(1000);
18
19   lcd.setCursor(0,0);
20   lcd.print("Temperature");
21   lcd.setCursor(0,1);
22   lcd.print(temperature_Celsius);
23   lcd.setCursor(5,1);
24   lcd.print(char(178));
25   lcd.print("C");
26
27   lcd.print(" ");
28   lcd.print(temperature_Fahrenheit);
29   lcd.setCursor(12,1);
30   lcd.print(char(178));
31   lcd.print("F");
32 }
```

**Figure-5: Arduino Code for Temperature & humidity sensor (using LCD Display, Register, Sensor & Arduino)**

## Discussion:

### 1. **Successful Sensor Interfacing and Data Acquisition**

The DHT11/DHT22 sensor was successfully interfaced with the Arduino Uno using a digital pin. The sensor consistently provided real-time temperature and humidity values, which confirms correct wiring and proper communication between the sensor and the microcontroller.

### 2. **Real-Time Display on LCD**

The 16x2 LCD display correctly showed the temperature in both Celsius and Fahrenheit, along with humidity percentage. This verified that data conversion and formatting were handled properly in the Arduino code and that the LCD connections and initialization were accurate.

### 3. **Conditional LED Control**

The LED responded to specific temperature conditions as programmed (e.g., turned ON when temperature exceeded 30°C). This indicates that the conditional logic using if statements in the code was implemented correctly and that digital output control from Arduino was functioning as expected.

### 4. **Simulation and Testing with Tinkercad**

The circuit was first simulated in Tinkercad to avoid physical errors and confirm logic flow. This step helped validate the design before actual hardware implementation and is particularly useful for beginners to visualize connections and troubleshoot virtually.

### 5. **Understanding Practical IoT Applications**

This experiment demonstrates a basic model of how environmental monitoring systems work in real-life IoT projects. The combination of sensor data acquisition, real-time processing, and output control (LED/LCD) lays the foundation for designing more complex automated systems like smart homes, greenhouses, or industrial climate monitoring.

## Conclusion:

1. Successfully interfaced a temperature & humidity sensor with Arduino.
2. Real-time data was displayed on an LCD screen correctly.
3. LED responded accurately to temperature conditions.
4. Simulation in Tinkercad helped verify the circuit before hardware testing.
5. Gained hands-on experience with sensor-based environmental monitoring.

Overall, the experiment strengthened foundational skills essential for environmental monitoring and IoT applications, paving the way for more complex sensor-based projects.