EL-702

Module 6: IoT and Industrial Automation

Submitted by:
Naitik
202421023

Submitted to:
Dr. Tapas Maiti

DAU Gandhinagar, Gujarat
28 April 2025

Combined Report for IoT and Industrial Automation

This report provides a comprehensive overview of the IoT project, which involves an Arduino R4 WiFi, ThingSpeak for cloud and analytics, DHT11 sensors, and a website for data visualization and control. The project collects sensor data, uploads it to ThingSpeak, and allows remote control of an LED via a website. The report is divided into sections for clarity.

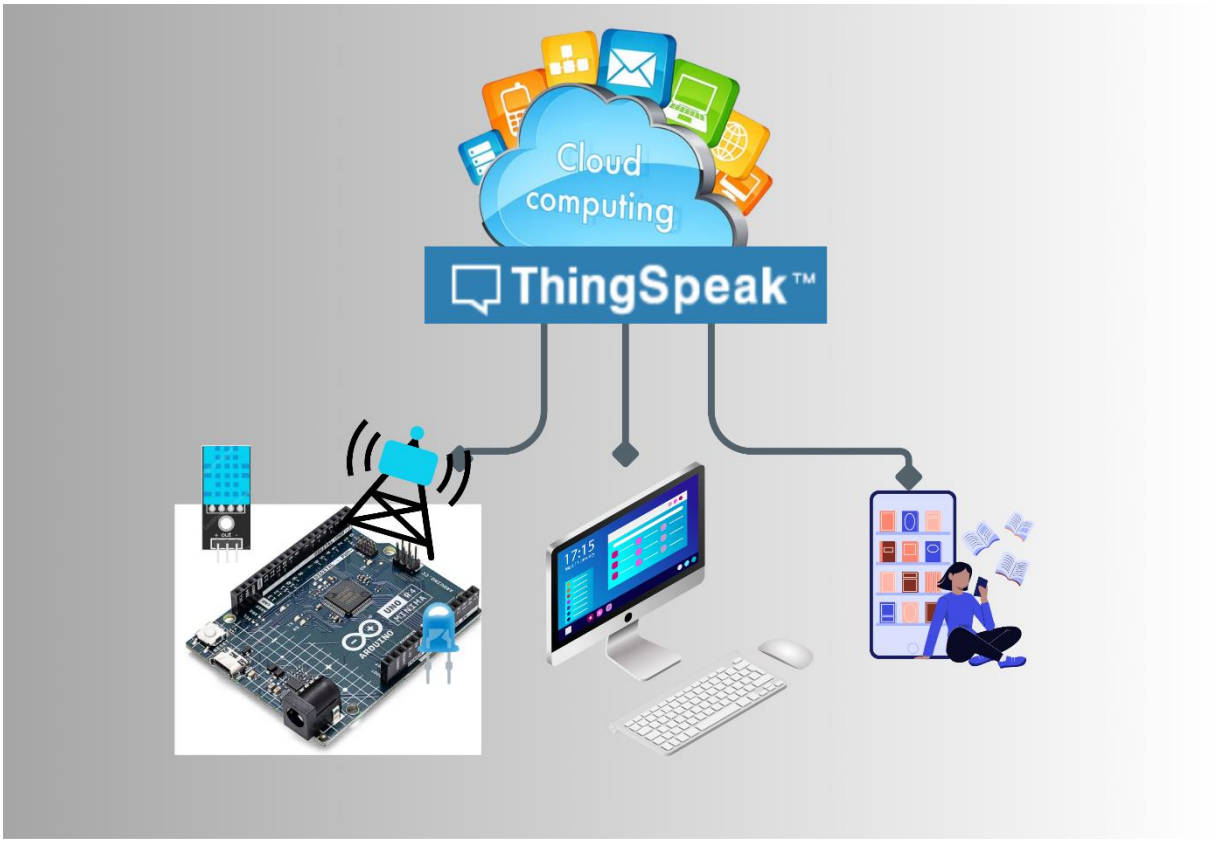


Fig. 1: *Birdseye view of the project.*

1. Arduino Code

The Arduino code is responsible for reading data from the DHT11 sensors, uploading it to ThingSpeak, and controlling an LED based on commands received from ThingSpeak. The code is modular and uses several libraries to interact with the sensors, WiFi, and ThingSpeak.

1.1. File: `dht_module.h`

This file defines a class `DHTSensor` that encapsulates the functionality of the DHT11 sensor. It provides methods to initialize the sensor, read humidity and temperature, and validate the data.

Key Components:

- **Class `DHTSensor`:**
  - **Private Members:**

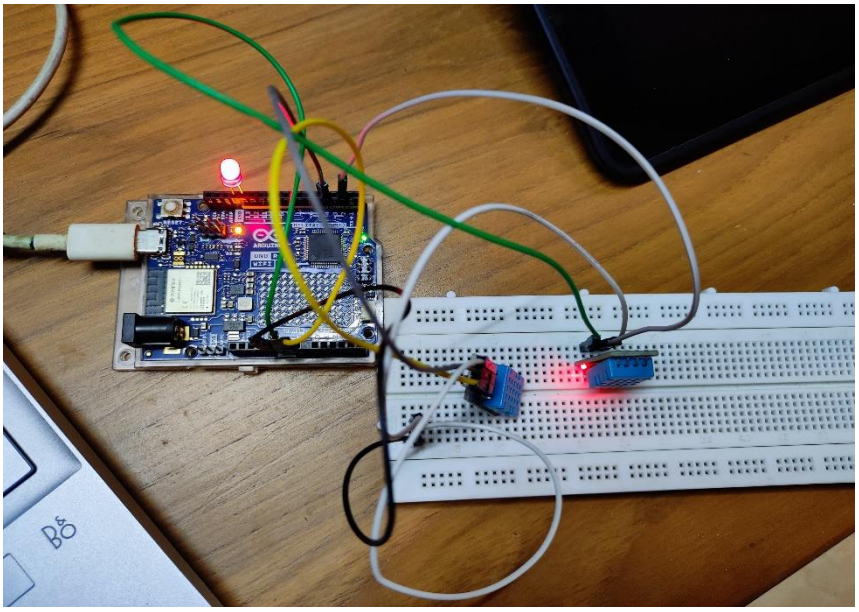
- `DHT dht`: An instance of the DHT library to interact with the sensor.
- `uint8_t pin`: The pin to which the sensor is connected.

0 **Public Methods:**

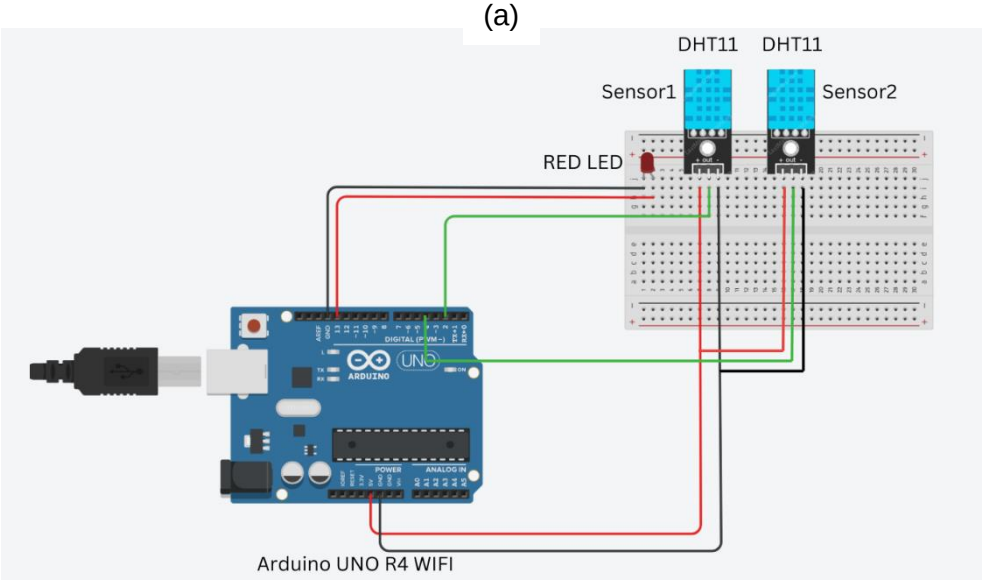
- `DHTSensor(uint8_t pin, uint8_t type)`: Constructor to initialize the sensor with a specific pin and type.
- `void begin()`: Initializes the sensor.
- `float getHumidity()`: Reads and returns the humidity value.
- `float getTemperature()`: Reads and returns the temperature value.
- `bool isDataValid(float humidity, float temperature)`: Validates the sensor data.

Usage:

This class is used in the main Arduino code to manage the DHT11 sensors and read data from them.



(a)



(b)

Fig. 2: *Circuit connections with two DHT11 sensors and an LED.*

1.2. File: `DHT11_LED.ino`

This is the main Arduino sketch that handles sensor data collection, WiFi communication, and ThingSpeak integration. It also controls an LED based on commands received from ThingSpeak.

Key Components:

- **Pin Configuration:**
  - `DHT11_PIN1` and `DHT11_PIN2`: Pins connected to the two DHT11 sensors.
  - `ledPin`: Pin connected to the LED.
- **Timing Intervals:**
  - `READ_INTERVAL`: Interval between sensor readings (15 seconds).
  - `THINGSPEAK_INTERVAL`: Interval between ThingSpeak updates (15 seconds).
- **WiFi and ThingSpeak Setup:**
  - `WiFiClient client`: Manages the WiFi connection.
  - `ThingSpeak`: Manages communication with ThingSpeak.
  - `myChannelNumber` and `ledChannelNumber`: ThingSpeak channel IDs for sensor data and LED control.
  - `myWriteAPIKey` and `myReadAPIKey`: API keys for writing and reading data from ThingSpeak.
- **DHT Sensor Objects:**
  - `dht1` and `dht2`: Instances of the `DHTSensor` class for the two sensors.
- **Time Tracking:**
  - `lastReadTime`: Tracks the last time sensor data was read and uploaded.

Functions:

- **`void printNetworkInfo()`:** Prints WiFi network information (status, SSID, IP address).
- **`void connectToWiFi()`:** Connects to the WiFi network using the credentials from `secrets.h`.
- **`void reconnectWiFi()`:** Reconnects to WiFi if the connection is lost.
- **`void updateThingSpeak(float humidity1, float temperature1, float humidity2, float temperature2)`:** Uploads sensor data to ThingSpeak.
- **`void processSensorData(DHTSensor& sensor, float& humidity, float& temperature)`:** Reads and processes data from a single sensor.
- **`void controlLED()`:** Reads the LED control value from ThingSpeak and controls the LED accordingly.

Setup and Loop:

- **`void setup()`:** Initializes serial communication, WiFi, ThingSpeak, and the DHT sensors. It also sets up the LED pin as an output.
- **`void loop()`:** Continuously checks the WiFi connection, reads sensor data, updates ThingSpeak, and controls the LED based on the latest value from ThingSpeak.

1.3. File: `secrets.h`

This file contains sensitive information such as WiFi credentials and ThingSpeak API keys. It is included in the main Arduino sketch to provide these values without hardcoding them.

Key Components:

- **WiFi Credentials:**
  - `SECRET_SSID`: The SSID of the WiFi network.
  - `SECRET_PASS`: The password for the WiFi network.
- **ThingSpeak Channel IDs:**
  - `SECRET_CH_ID_1`: Channel ID for sensor data.
  - `SECRET_CH_ID_2`: Channel ID for LED control.
- **ThingSpeak API Keys:**
  - `SECRET_WRITE_APIKey`: API key for writing data to ThingSpeak.
  - `SECRET_READ_APIKey`: API key for reading data from ThingSpeak.

---

2. ThingSpeak Channels

ThingSpeak is used as the cloud platform for storing sensor data and controlling the LED. Two channels are used in this project:

2.1. Sensor Data Channel (Channel ID: 2717430)

- **Purpose:** Stores humidity and temperature data from the two DHT11 sensors.
- **Fields:**
  - Field 1: Humidity from Sensor 1.
  - Field 2: Temperature from Sensor 1.
  - Field 3: Humidity from Sensor 2.
  - Field 4: Temperature from Sensor 2.
- **API Key:** `5KSKJ52T5SAV89DU` (Write API Key).

2.2. LED Control Channel (Channel ID: 2753822)

- **Purpose:** Stores the control value for the LED (1 for ON, 0 for OFF).
- **Fields:**
  - Field 1: LED control value.
- **API Key:** `TWYWVWMKXN6KKVYB` (Read API Key).

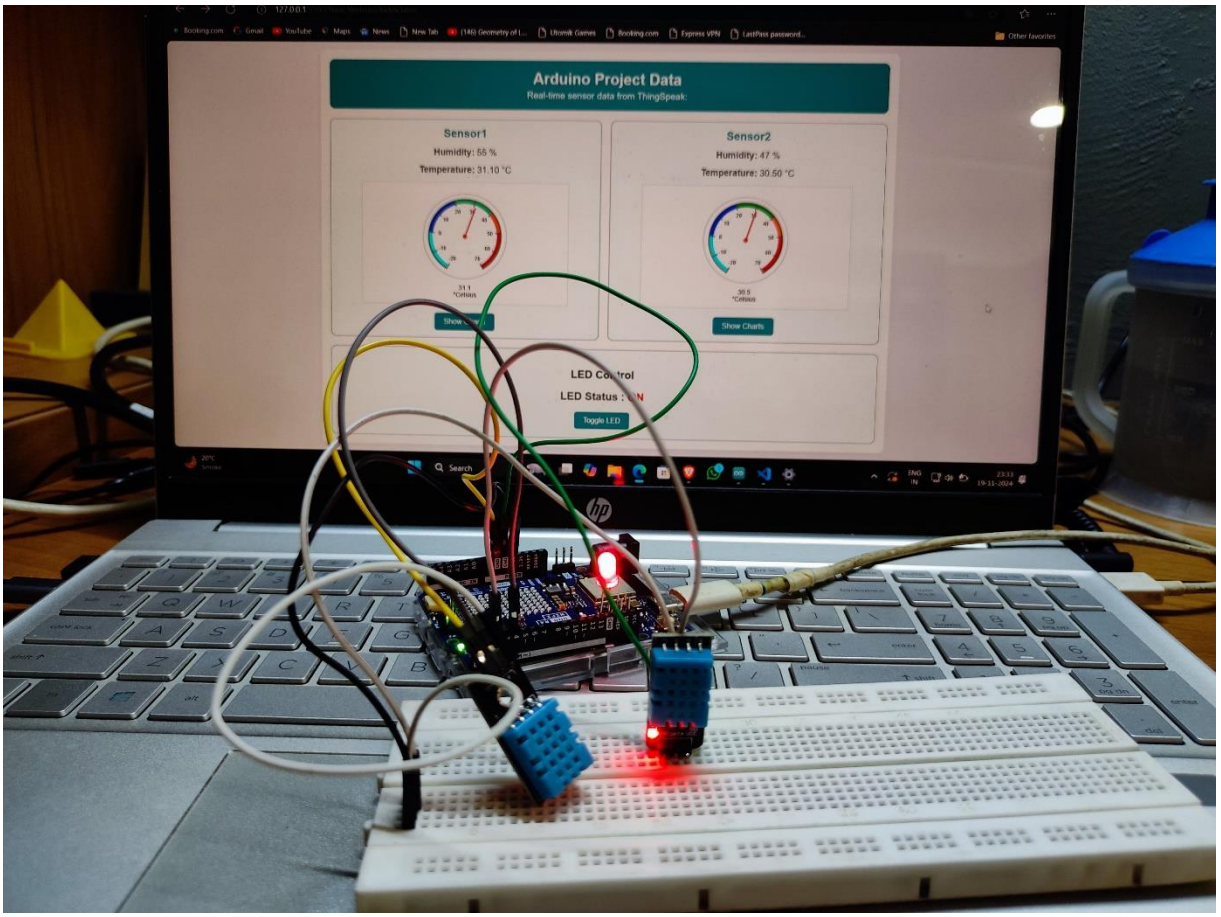


Fig. 3: Working project of *Website interface displaying real-time sensor data and LED control.*

3. Website Code

The website is designed to display real-time sensor data from ThingSpeak and provide an interface to control the LED. The website uses the ThingSpeak REST API to fetch data and send control commands.

3.1. File: `index.html`

This file defines the structure of the website. It includes sections for displaying sensor data and controlling the LED.

Key Components:

- **HTML Structure:**
  - **Sensor Data Display:**
    - Two sections for Sensor 1 and Sensor 2, each displaying humidity and temperature.
    - Embedded ThingSpeak widgets for visualizing data.
    - Buttons to navigate to detailed analytics charts.
  - **LED Control:**
    - Displays the current status of the LED (ON/OFF).
    - A button to toggle the LED state.

Usage:

The HTML structure is styled using `style.css` and interacts with the ThingSpeak API via `script.js`.

3.2. File: `script.js`

This file handles the interaction with the ThingSpeak API, updates the website with real-time data, and controls the LED.

Key Components:

- **Fetching Sensor Data:**
  - The `fetchData` function fetches the latest sensor data from ThingSpeak and updates the HTML elements.
  - Data is fetched every 15 seconds using `setInterval`.
- **LED Control:**
  - The `writeData` function sends a command to ThingSpeak to toggle the LED state.
  - The LED status is updated based on the response from ThingSpeak.
- **Event Handling:**
  - The LED toggle button triggers the `writeData` function when clicked.

Usage:

The script ensures that the website displays real-time data and allows users to control the LED remotely.

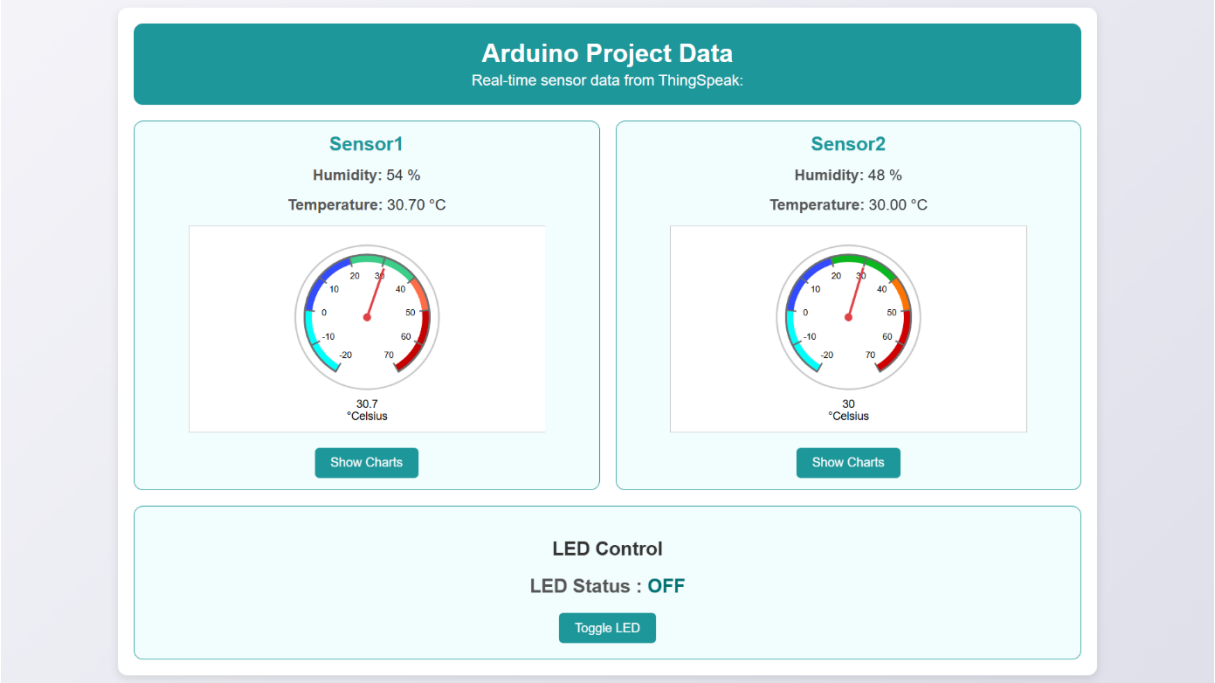


Fig. 4: *Website interface displaying real-time sensor data and LED control.*

Summary

The Arduino code collects data from two DHT11 sensors, uploads it to ThingSpeak, and controls an LED based on commands received from ThingSpeak. The ThingSpeak channels store the sensor data and the LED control value. The website interacts with ThingSpeak to display real-time data and provide an interface for controlling the LED.

This modular approach ensures that the project is scalable and easy to understand. The integration of Arduino, ThingSpeak, and the website provides a complete IoT solution for data collection, visualization, and remote control.It is the best platform for our application as it simplifies the management of Arduino variables and their states, eliminating the complexity for users. Its user-friendly interface makes it easy for beginners to set up and control devices, making it an ideal choice for smart home applications.