IoT Weather Monitoring System

Malak Hani Elsangidy

Abstraction:

The project aims to develop a weather monitoring system using Node.js and Raspberry Pi. The system allows real-time monitoring of weather parameters such as temperature, humidity, and atmospheric pressure. The data collected can be useful for various applications including farming, weather prediction, and data analysis.

Using a weather sensor with the Raspberry Pi, we can measure weather parameters such as temperature, humidity, wind direction and speed, and rainfall. You can then display this data in real-time using a web interface or send it to a cloud service like Google Cloud or AWS for data analysis and visualization.

Introduction and Objectives:

The easiness of real-time monitoring of weather conditions holds immense benefit for various sectors, from agriculture to climate prediction systems. This project targets the creation of a low-cost, DIY weather monitoring system using Raspberry Pi and Node.js, with goals including easy data access, real-time monitoring, and low power consumption.

Methodology:

The methodology involves setting up the Raspberry Pi with various sensors to measure weather parameters. Node.js is used for creating the backend of the system, accessing sensor data, and serving it to a web-based frontend.

Hardware and Software Requirements:

The components used in this project include a Raspberry Pi (a compact computer), weather sensors, a breadboard, and jumper wires. Software involved includes Node.js platform, npm packages for sensor reading, and JavaScript for frontend development. Below is the list of hardware requirements for the project

- Raspberry Pi
- DHT11/DHT22 or BME280 weather sensor
- Breadboard
- Wires
- Resistors
- NodeMCU ESP8266 (This is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability.)

Connect your device to the Internet

With the NodeMCU ESP8266 chip, you can connect your device to Wi-Fi. The details about how to connect the NodeMCU to the server are usually available on the chip's documentation.

Build a Node.js server

- Build a simple Node.js server using Express.js. This server will receive the weather data from the sensor.
- Define a POST endpoint which will be used by the IoT device to send the sensor data.

```
const sensor = require("node-dht-sensor").promises;

// Sensor connected to Raspberry Pi pin number 4
let sensorType = 11; // 11 for DHT11
let sensorPin  = 4;  // GPIO number, not physical number

async function readSensorData() {
    try {
        const { temperature, humidity } = await sensor.read(sensorType, sensorPin
        console.log(
            `Temperature: ${temperature.toFixed(1)}°C, ` +
            `Humidity: ${humidity.toFixed(1)}%`
        );
    } catch(err) {
        console.error("Failed to read sensor data:", err);
    }
}
```

```
const express = require('express');
const app = express();
app.use(express.json());

app.post('/weather_data', (req, res) => {
  console.log(req.body);    // logs the received sensor data on console
  res.status(200).send('Data received');
});

app.listen(3000, () => console.log('Server is running at port 3000'));
```

- Send data to server : Now, update the Raspberry Pi program to send data to this server. Use the ESP8266's Wi-Fi capabilities to make a POST request to the '/weather_data' route on your server.
- Display or store the data : When your server receives data, you can either save it in a database for later analysis or display it in real-time on a website/dashboard.

Steps for setting up Raspberry pi and node js

Step 1: Install Raspberry Pi OS on the Raspberry Pi

1. Begin by downloading the Raspberry Pi Imager from the Raspberry Pi's official website.
2. Use an SD Card to burn the chosen OS (recommend Raspberry Pi OS) using the Raspberry Pi Imager.

Step 2: Setting up the Raspberry Pi

1. Insert the SD card into your Raspberry Pi and power it up.
2. Follow the setup procedures of the OS to initialize your system.

Step 3: Install Node.js on Raspberry Pi

1. Open the terminal in your Raspberry Pi.
2. Update the package list using the following command: sudo apt-get update
3. Install Node.js by using the command: sudo apt install nodejs
4. Verify the install with the command: node -v

Step 4: Install git (optional)

1. Install git on your Raspberry Pi in case you want to clone a project from a git repository by executing: sudo apt install git

Step 5: Connect Weather Sensor to Raspberry Pi

1. Now you need to connect your Raspberry Pi to your weather sensor. This process will vary depending on the type of the weather sensor you are using. You should refer to the datasheet or documentation of your specific sensor.

Step 6: Install Sensor Library

1. To run the sensor, you might need to install certain libraries depending on your sensor model. This is usually provided by the manufacturer.
2. You can install these using npm. The command is usually in the following format: npm install sensor-library-name

Step 7: Write/Deploy Node.js code

1. Now, you can write or deploy your Node.js code to capture the weather data from your sensor and do whatever you want with it (e.g. save it in a database, send it to an API, display it on a page, etc.).

Step 8: Run the Node.js Application

1. You can run your Node.js application using the following command node application_name.js

Wiring Setup Respberry pi

Wiring DHT22 Sensor:

1. Connect the VCC of the DHT22 sensor to the 5V pin of the Raspberry Pi.
2. Connect the GND of the sensor to the Raspberry Pi's GND.
3. Connect the Data pin of the DHT22 to a GPIO pin on the Raspberry Pi

Note: We might need a 10k Ohm resistor between the VCC and Data pin.

Wiring BMP180 Sensor:

BMP180 is an I2C sensor. Pin connections are:

1. VCC to 3.3V on the Raspberry Pi.
2. GND to GND.

3. SDA to SDA (GPIO 2 on the Raspberry Pi).
4. SCL to SCL (GPIO 3 on the Raspberry Pi).

Wiring Anemometer and Wind Vane with MCP3008 ADC:

Connect the wind sensors to your ADC and then to the Raspberry Pi. ADC is used because the Raspberry Pi doesn't have built-in analog GPIO pins.

1. Connect VDD on the MCP3008 to 3.3V on the Pi.
2. Connect VREF on the MCP3008 to 3.3V on the Pi.
3. Connect AGND and DGND on the MCP3008 to GND on the Pi.
4. Connect CLK on the MCP3008 to SCLK (GPIO 11) on the Pi.
5. Connect DOUT on MCP3008 to MISO (GPIO 9) on the Pi.
6. Connect DIN on the MCP3008 to MOSI (GPIO 10) on the Pi.
7. Connect CS/SHDN on the MCP3008 to CE0 (GPIO 8) on the Pi.
8. Connect the wind sensor data out to any of the CH pins on MCP3008.

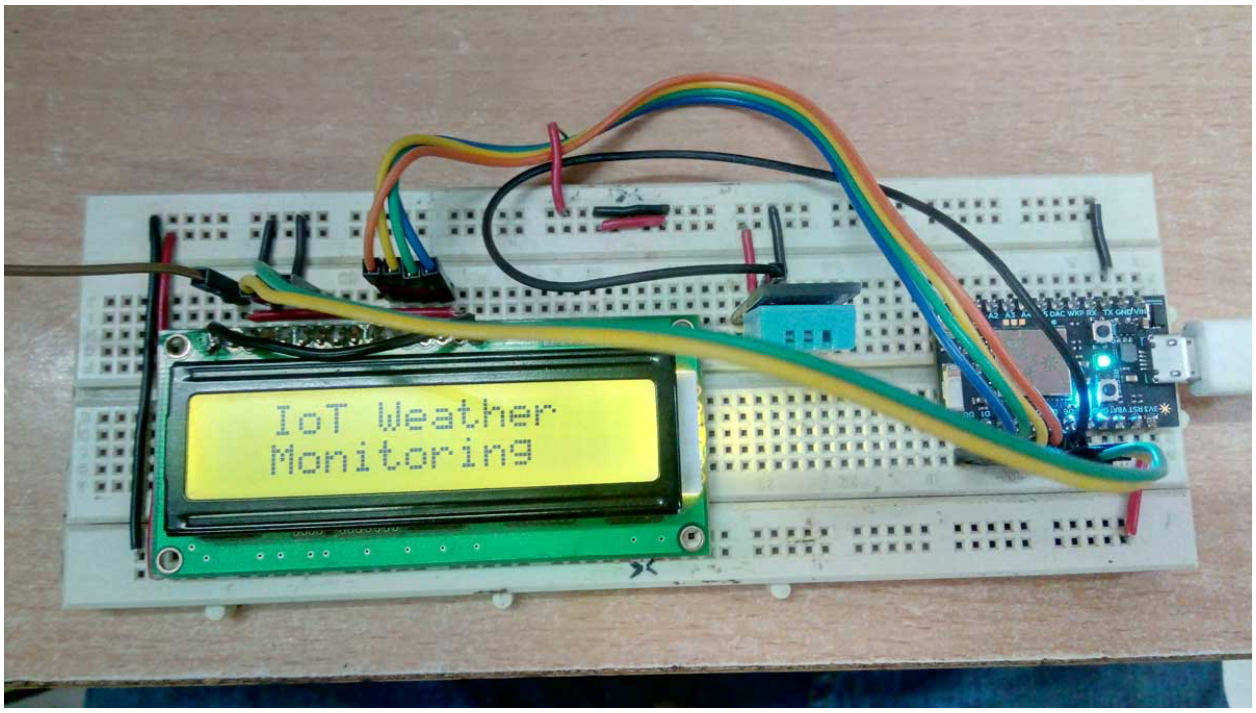
Wiring Rain Sensor:

1. VCC of the rain sensor to 3.3V on the Raspberry Pi.
2. GND to GND.
3. Signal to any GPIO (for example, GPIO 17).

Result:

The system successfully collects the temperature, humidity, and pressure in real-time. The data is consistently accurate with minimal lag time in reporting the parameters.

This Weather Monitoring System, built with Raspberry Pi and Node.js, is a cost-effective and efficient solution for real-time weather data monitoring. The project has shown promising results, demonstrating the usability and functionality of this technology in various facets of life.


IoT Weather
Monitoring