Mini Project Report on

Smart Weather Station

Submitted in partial fulfilment of the requirements of
the degree of Bachelor in Engineering

By

Ashok Prajapati TE9-52
Harsh Shah TE9-66
Atharv Vakhare TE9-69
Prathamesh Wadekar TE9-70

Under the guidance of **Vandana Soni**

DEPARTMENT OF COMPUTER ENGINEERING
SHAH AND ANCHOR KUTCHHI ENGINEERING COLLEGE
CHEMBUR, MUMBAI – 400088.

University of Mumbai

(AY 2024-25)

Mahavir Education Trust's
Shah & Anchor Kutchhi Engineering College
An Autonomous Institute Affiliated to University of Mumbai

UG Program in Computer Engineering
(Accredited by NBA for 3 years from AY 2022-23)

Certificate

This is to certify that the report of the Mini Project entitled

is a bonafide work of

Name of Student	Class	Roll No
Ashok Prajapati	TE9	52
Harsh Shah	TE9	66
Atharv Vakhare	TE9	69
Prathamesh Wadekar	TE9	70

submitted to the
UNIVERSITY OF MUMBAI
during Semester V
in

COMPUTER ENGINEERING

Guide

I/c Head of Department

Principal

Mini Project Approval

This Mini Project entitled **Smart Weather Station** by **Harsh Shah, Atharv Vakhare, Prathamesh Wadekar** is approved for the degree of **Bachelor of Engineering** in **Computer Engineering.**

Examiners

1……
(Internal Examiner Name & Sign)

2……
(External Examiner name & Sign)

Date:

Place:

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name of student	Class	Roll No.	Signature
Ashok Prajapati	TE9	52	
Harsh Shah	TE9	66	
Atharv Vakhare	TE9	69	
Prathamesh Wadekar	TE9	70	

Date:

Place: Mumbai

Contents

Abstract

The Smart Weather Station project is aimed at producing an efficient and user-friendly system for tracking real-time changes in the environmental conditions, particularly temperature and humidity. Using the ESP32 microcontroller and the DHT22 sensor, the system aggregates accurate weather data and sends it wirelessly via Wi-Fi connectivity to a Blynk mobile application. The Blynk app now becomes a really powerful interface that allows users to view data directly from their smartphones. It represents the integration of IoT components and, at the same time, gives an example of the possibilities of smart home applications and solutions for monitoring environmental changes. Being scalable, it allows adding many sensors and functionalities for comprehensive analysis of weather elements, thus fostering progress in smart technology and environmental awareness.

Acknowledgement

We extend our sincerest appreciation to Vandana Soni for her invaluable guidance and unwavering support throughout our project. Her mentorship has played a pivotal role in our achievements, and we are deeply grateful for the time and dedication she has invested in us. Thank you, ma’am, for your exceptional guidance and encouragement.

Thanking you,

Ashok Prajapati

Harsh Shah

Atharv Vakhare

Prathamesh Wadekar

List of Figures

1. Introduction

1.1 Introduction

Recently, with the rising demand for smart technology and IoT solutions, society has witnessed the increasingly growing requirement for higher levels of automation, monitoring, and data management in nearly all fields. Perhaps the most important impact of IoT was in environmental monitoring. Originally used merely as weather stations, these have become smart devices where one can retrieve real-time data from anywhere in the world.

The basis of this project will be on creating a Smart Weather Station using ESP32 microcontrollers and the DHT22 temperature and humidity sensor. ESP32 microcontrollers are famously known for their Wi-Fi and Bluetooth capabilities, which makes them an ideal choice for IoT applications. Conversely, the DHT22 sensor is renowned to be reliable and give very accurate readings so that the user will receive thorough information about his surroundings.

The Smart Weather Station will gather environmental data, and it will send this to the Blynk application, as it contains a user-friendly interface for data visualization and monitoring. Using Blynk, people may create dashboards that best serve their application, which ultimately provides an easy read and interpretation of weather information at hand in real-time. The project does not only demonstrate that creating an efficient weather monitoring system is possible, but also calls attention to how accessible environmental data may be for peoples and communities. Developing this Smart Weather Station will greatly help to increase the awareness on the situation of the weather and deeper understanding of the impacts of climate change and local weather phenomena.

1.2 Motivation

The motivation behind building a smart weather station lies in the increasing demand for localized and real-time weather data. Traditional weather stations are costly and often not accessible to individuals or small communities. This project aims to provide a low-cost, easy-to-deploy solution for weather monitoring in homes, farms, or educational settings. Additionally, integrating the system with the Blynk app provides a user-friendly interface for data visualization and remote access.

1.3 Organization of the Report

The report is broadly divided into four sections. In the first section, the project is introduced along with motivation and structure. The second section is a literature survey that covers existing solutions, identifies limitations in them, describing the problem statement and objectives. The third section deals with the proposed system: architecture, algorithm, hardware and software details, and results of experiments. In the fourth section, the final conclusion of the report is given, giving an overview of further improvements.

2. Literature Survey

2.1 Survey of Existing System

There are many commercially available weather stations to monitor temperature, but these cost a lot and, in many cases, are hard to customize so that they could be used by any individual. Some DIY solutions exist, based on Arduino and Raspberry Pi, which are highly flexible in nature but do not have integration with the mobile platforms like Blynk. The ESP32, together with the DHT22 sensor, can realize low-cost weather data capture and display in real time by combining its inbuilt Wi-Fi and Bluetooth functionalities.

2.2 Limitations of Existing Systems

More than that, available systems often will carry some kind of limitation - be it costs, availability of connectivity options, or the general absence of mobile application that allows access from a remote environment. Most traditional weather stations are also typically large in size, hence lacking the suitability for small scale usage in the urban setup, in houses, or schools. Under this aspect, still, customizable and scalable modern IoT-platform-based weather stations remain underdeveloped.

2.3 Problem Statements & Objectives

The primary problem addressed by this project is the need for an affordable, user-friendly, and real-time weather monitoring system that individuals or small communities can easily deploy. The key objectives are:

- To design a system using ESP32 and DHT22 sensor to monitor temperature and humidity.
- To develop a mobile interface using the Blynk app for real-time data visualization.
- To ensure the system is low-cost, easy to set up, and scalable for future sensor integration.

2.4 Scope

The scope of this project includes developing a prototype weather station that collects temperature and humidity data and displays it on a mobile platform. Future enhancements could integrate more sensors for wind speed, atmospheric pressure, and rain detection.

3. Proposed System

The proposed solution for the Smart Weather Station involves designing and implementing a comprehensive system that integrates hardware and software components to facilitate real-time environmental monitoring. The solution focuses on leveraging the ESP32 microcontroller, DHT22 temperature and humidity sensor, and the Blynk mobile application to create an accessible and efficient monitoring system.

3.1 Architecture/Framework

The proposed system consists of three main components:

- ESP32 Microcontroller: The central processing unit for collecting sensor data.
- DHT22 Sensor: Provides temperature and humidity data.
- Blynk App: A mobile application to display real-time data on a smartphone.

The architecture includes the ESP32, connected to the DHT22 sensor, gathering environmental data. This data is then transmitted via Wi-Fi to the Blynk cloud, from where it is accessed by the user's smartphone app.

3.2 Algorithm and Process Design

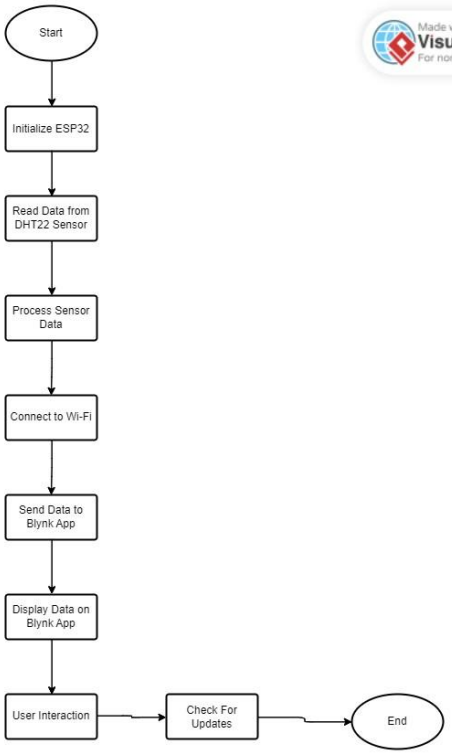

Fig 3.2.1:Process Flow Diagram

3.3 Details of Hardware & Software

Hardware:

3.3.1 ESP32 Microcontroller:

ESP32 is an extremely capable microcontroller integrating Wi-Fi and Bluetooth on the chip. It's the perfect module to get your IoT project started - for example, the smart weather station. It integrates two cores and has a clock speed of up to 240 MHz with a wide variety of peripheral interfaces, including GPIOs, UART, I2C, SPI, ADC - very low power consumption, which makes it suitable for battery-powered systems.

3.3.2 DHT22 Temperature and Humidity Sensor:

DHT22 is a widely known sensor for measuring temperature and humidity. It reads within an accuracy of 0–100% relative humidity as well as a temperature range of -40°C to 80°C. It needs just one digital pin of the ESP32 for communication. Hence, it's straightforward to integrate. The precision is in the order of ±0.5°C for temperature as well as ±2–5% for humidity that meets most needs in weather monitoring.

3.3.3 Connecting Wires:

Jumper wires are used to connect the ESP32 to the DHT22 sensor.
Typically, a simple connection includes:

- One wire to supply power (3.3V or 5V) from the ESP32 to the sensor.
- A ground wire from the sensor to the ESP32.
- A data wire to transfer the sensor readings to one of the ESP32's GPIO pins.

3.3.4 Power Supply (5V):

The ESP32 can be powered by a 5V power supply via a micro-USB cable, or a dedicated 5V supply connected to the VIN pin. In mobile setups or field deployments, the system can be powered by batteries (e.g., Li-ion) with a step-down module to ensure consistent voltage to the ESP32.

Software:

3.3.5 Arduino IDE for ESP32 Programming:

The Arduino IDE (Integrated Development Environment) is a userfriendly platform for writing and uploading code to the ESP32. To program the ESP32, you must first install the ESP32 board package in the Arduino IDE. This enables compatibility between the IDE and the ESP32. Key components include:

- ESP32 Board Definition: Installed through the "Boards Manager" in the Arduino IDE to allow direct compilation and upload of code to the ESP32.
- DHT Library: Used for communicating with the DHT22 sensor. You can install this library via the Library Manager in the Arduino IDE.

After setting up the environment, the code is written in the Arduino language (C/C++), where you will define the ESP32's pin configurations, DHT22 reading functions, and Wi-Fi connection for Blynk integration.

3.3.6 Blynk App for Data Visualization:

The Blynk app allows real-time monitoring of the weather station's data on a smartphone. It connects to the ESP32 via the cloud and displays sensor readings on customizable widgets such as gauges, graphs, or labels. The app is available on Android and iOS, and setting up the project involves:

- Creating a Blynk Project: Upon registering, you create a new project, select ESP32 as the device type, and receive a unique authentication token that will be used to link the microcontroller to the app.
- Setting Up Widgets: You can add various widgets (e.g., for displaying temperature and humidity) and configure their properties (pin assignment, refresh rate) within the app's user interface.
- Wi-Fi and Cloud Communication: Blynk uses the Blynk Cloud to relay data between the ESP32 and the app. This enables remote access to the weather data from anywhere with internet connectivity.

3.3.7 Blynk Library and ESP32 Board Setup in Arduino IDE:

- Blynk Library: The Blynk library allows the ESP32 to communicate with the Blynk Cloud. The library handles all the backend communication, enabling the ESP32 to send data to the app or receive commands from it. It also supports various communication protocols, such as Wi-Fi, Bluetooth, and Ethernet. To install it:
- Go to the Arduino IDE's Library Manager, search for "Blynk," and install the library.
- Include the library in your code using #include <BlynkSimpleEsp32.h>.
- ESP32 Board Setup: You'll need to set up the ESP32 in the Arduino IDE by installing the **ESP32 Board Manager**:
- n the Arduino IDE, go to **File -> Preferences**, and paste the following URL in the "Additional Boards Manager URLs" section: https://dl.espressif.com/dl/package_esp32_index.json
- Then, go to **Tools -> Board -> Boards Manager** and search for "ESP32." Install the package for the ESP32 board.

Here’s a basic Arduino code to demonstrate the integration of ESP32 with DHT22 and Blynk:

```
#define BLYNK_TEMPLATE_ID "TMPL3eoyQS5kD"
#define BLYNK_TEMPLATE_NAME "Weather"
#define BLYNK_AUTH_TOKEN "kjJYg5rhqDxOeSu6rAJVtWk2evrGeX4U"

#include <WiFi.h>
#include <DHT.h>
#include <BlynkSimpleEsp32.h> // Include the Blynk library

// WiFi Credentials
char ssid[] = "Atharv Vakhare";
char pass[] = "morya123";

// DHT22 Configuration
#define DHTPIN 5     // DHT22 data pin
#define DHTTYPE DHT11 // DHT 22 (AM2302)
DHT dht(DHTPIN, DHTTYPE);

BlynkTimer timer;

int detectPrecipitation(float temperature, float humidity) {
  int precipitationPercent = 0;

  // Simple estimation logic: Higher humidity and moderate temperature indicate higher precipitation chances
  if (humidity >= 80 && (temperature >= 0 && temperature <= 25)) {
    precipitationPercent = 90;  // High probability of precipitation
  } else if (humidity >= 60 && (temperature >= 0 && temperature <= 25)) {
    precipitationPercent = 60;  // Moderate probability
  } else if (humidity >= 40 && temperature <= 30) {
    precipitationPercent = 30;  // Lower probability
  } else {
    precipitationPercent = 10;  // Very low probability
  }

  return precipitationPercent;
}
```

```
// Function to read and send sensor data to Blynk
void sendSensorData() {
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

  // Check if readings are valid
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // Print data to Serial Monitor
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.print(" °C, Humidity: ");
  Serial.print(humidity);
  Serial.println(" %");

  // Determine temperature category
  String temperatureCategory;
  if (temperature >= 1 && temperature <= 18) {
    temperatureCategory = "Cold";
  } else if (temperature >= 19 && temperature <= 28) {
    temperatureCategory = "Quite Cold";
  } else if (temperature >= 29 && temperature <= 40) {
    temperatureCategory = "Hot";
  } else {
    temperatureCategory = "Out of range"; // Handle
temperatures outside the expected range
  }

  // Print temperature category to Serial Monitor
  Serial.print("Temperature Category: ");
  Serial.println(temperatureCategory);

  int precipitationPercent = detectPrecipitation(temperature,
humidity);
  Serial.print("Precipitation: ");
  Serial.print(precipitationPercent);
  Serial.println(" %");


  // Send data to Blynk
```

```
  Blynk.virtualWrite(V0, temperature); // Send temperature
to V0
  Blynk.virtualWrite(V1, humidity);  // Send humidity to V1
  Blynk.virtualWrite(V2, temperatureCategory); // Send
temperature category to V2
  Blynk.virtualWrite(V3, precipitationPercent);
}

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

  // Connect to Blynk
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

  // Start a timer to send sensor data every minute
  timer.setInterval(1000L, sendSensorData); // 1000 ms = 1
second
}

void loop() {
  Blynk.run();
  timer.run(); // Run the timer to trigger the sensor reading
}
```

3.4 Experiment and Results

The system was tested in different environments to validate the accuracy of the DHT22 sensor readings. Temperature and humidity readings were collected over 24 hours, with data being displayed in real-time on the Blynk app. The system demonstrated stable performance, with the sensor data accurately reflecting environmental conditions.

3.4.1 Connection:

The following image shows the wiring of the DHT22 temperature and humidity sensor to the ESP32 microcontroller. The connections are as follows:

- **VCC** (power) of the DHT22 is connected to the 3.3V pin on the ESP32.
- **GND** (ground) of the DHT22 is connected to a ground (GND) pin on the ESP32.
- **DATA** pin of the DHT22 is connected to GPIO pin 4 of the ESP32, with a 10kΩ pull-up resistor between VCC and DATA.

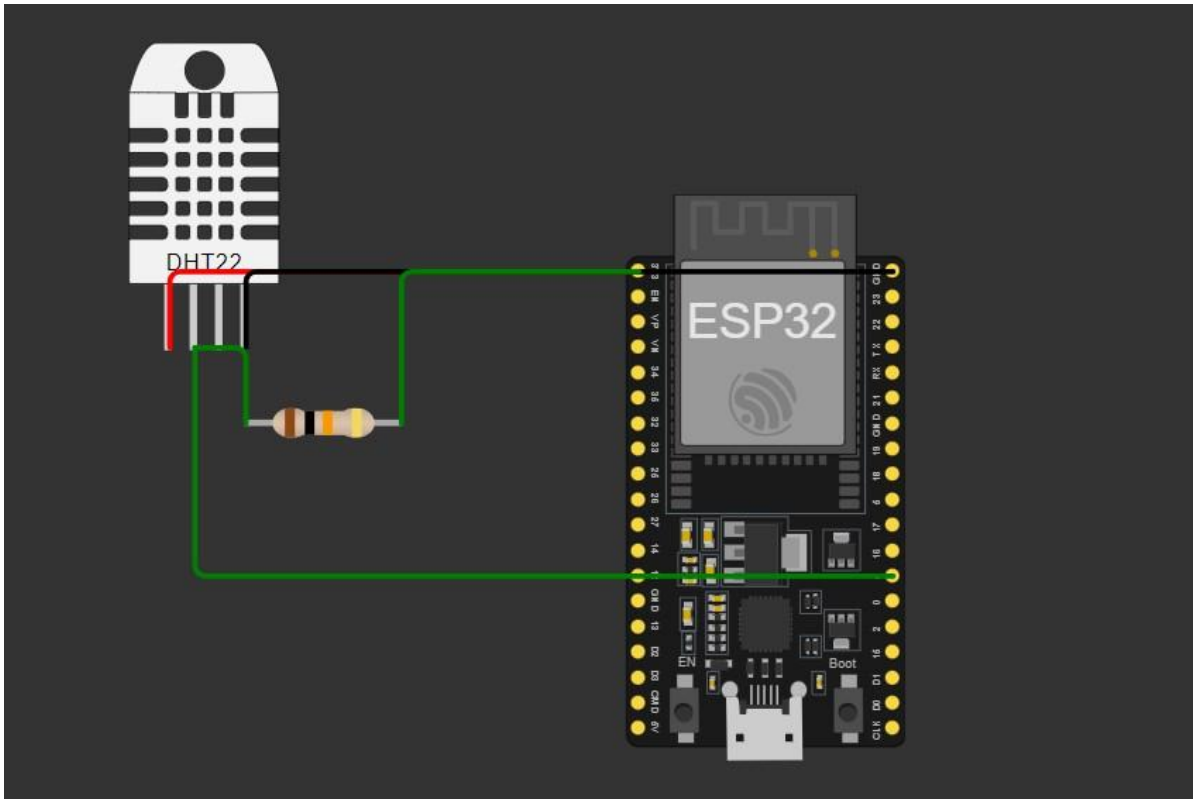

Fig 3.4.1.1: Circuit Design

3.4.2 Blynk App Output:

After wiring the hardware components and uploading the program, the system was tested for reading temperature and humidity values. The results were displayed in real-time on the Blynk app, which is illustrated in the following image:

The image below shows the real-time temperature (in Celsius) and humidity (in Fahrenheit) displayed in the Blynk app. The app is configured to display the readings using gauges for easy visualization.

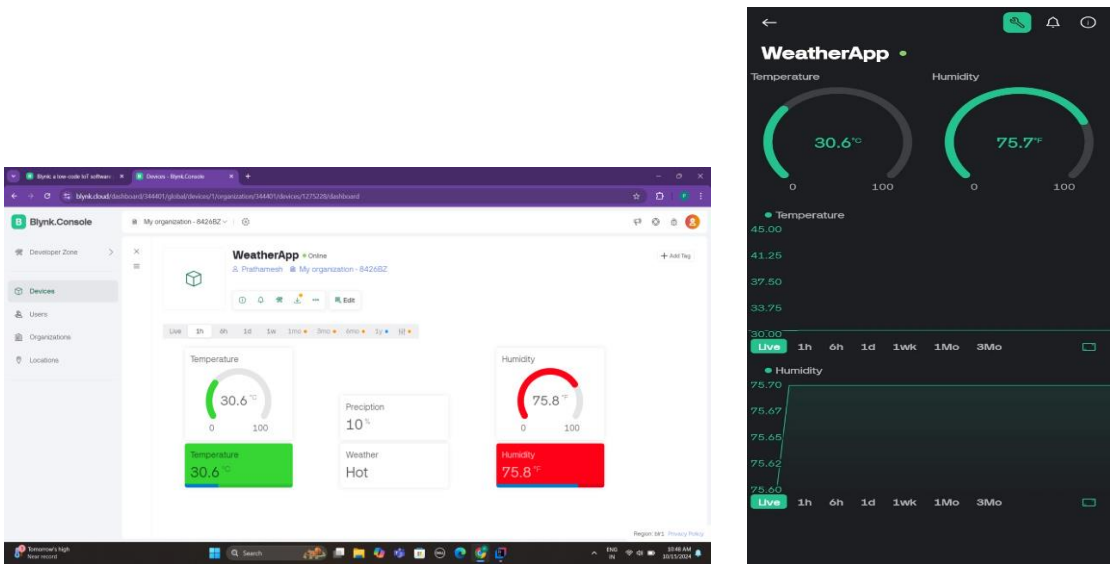

Fig3.4.2.1:Output

4. Conclusion and Future work

The Smart Weather Station project has been thoroughly proven for functional application using modern microcontroller technology, precise sensing capabilities, and user-friendly mobile applications for real-time environmental monitoring. Through the use of the ESP32 microcontroller along with the DHT22 sensor, it can efficiently collect temperature and humidity data and send it to the Blynk app and make such data available online. It emphasizes the importance of available data for current weather conditions for everyday application in making decision bases on current environmental conditions. The Smart Weather Station, after thorough test and calibration, has been proven to serve as a trusted tool in monitoring local conditions, increasing awareness regarding climatic conditions as well as potential impact. The intuitive design of the Blynk application makes user interaction easy and helps visualize data while alerting the user to any significant changes in the weather patterns.

The current implementation provides the functionality, while there are many enhancements still to be made in the future. Future versions would be possible that would integrate with more environmental sensors, such as air quality sensors recording particulate matter, CO2 levels, rainfall, and wind speed, which will effectively evaluate in finer detail the environmental condition. There would be installation of a cloud-based database holding historical weather data that could enable the user to analyze over time, insights into weather patterns, and essentially long-term climate impacts. Further, having a web interface along with the Blynk app will offer alternative means of access and visualization of data, such as in-depth reports and historical comparisons. It can allow integration with home automation and open more ways for automatic responses to environmental changes, and thereby lead to increased energy efficiency. Further development of such an application might include not only sending notifications for extreme weather conditions, or customizable alerts, but it could further bring more control over what data visualization is brought into the scope of the user. In aiming to address these enhancements, the Smart Weather Station can grow into a more all-encompassing monitoring apparatus for the environment in order to inform and arm users with important data that brings them closer to some level of understanding and awareness regarding the weather and how it comes to mean something important for humanity.

5. References

[1] Hussein, Zaid Khudhur, et al. "Low-cost smart weather station using Arduino and ZigBee." *Telkomnika (Telecommunication Computing Electronics and Control)* 18.1 (2020): 282-288.

[2] Djordjevic, Milos, and Danijel Dankovic. "A smart weather station based on sensor technology." *Facta Universitatis, Series: Electronics and Energetics* 32.2 (2019): 195210.

[3] Tenzin, Sonam, et al. "Low-cost weather station for climate-smart agriculture." *2017 9th international conference on knowledge and smart technology (KST)*. IEEE, 2017.

[4] Satyanarayana, K. N. V., et al. "Mobile app & iot based smart weather station." *International Journal of Electronics, Communication and Instrumentation Engineering Research and Development (IJECIERD)* 7.4 (2017): 1-8.

[5] Bogdan, Mihai. "About The Smart Weather Station." *Acta Universitatis Cibiniensis. Technical Series* 68.1 (2016): 26-29.

[6] bin Sadli, Muhamad Dan Darrawi. "An IoT-based Smart Garden with Weather Station System." *2019 IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*. IEEE, 2019.

[7] Mestre, Gonçalo, et al. "An intelligent weather station." *Sensors* 15.12 (2015): 31005-31022.