VIETNAM NATIONAL UNIVERSITY, HANOI

**INTERNATIONAL SCHOOL**



***INS3188 – COMPUTERS AND PERIPHERALS***

PROJECT REPORT

ESP8266 Module, DHT11 Temperature Sensor, LCD, RGB Led and Buzzer

Information

Lecturer: Nguyễn Thanh Tùng

Group Member:

Đặng Phương Thảo – 19071628

Class: ICE2019A

Hanoi – 2024

Table of Contents

[**I. Introduction** 4](#_Toc162960193)

[1. Project 4](#_Toc162960194)

[2. ESP8266 4](#_Toc162960195)

[3. Temperature Sensor 4](#_Toc162960196)

[4. LCD 5](#_Toc162960197)

[5. Led RGB 5](#_Toc162960198)

[6. Buzzer 5](#_Toc162960199)

[**II.**  **Equipment specifics** 5](#_Toc162960200)

[1. Hardware 5](#_Toc162960201)

[1.1. Temperature and humidity sensor DHT11 5](#_Toc162960202)

[1.2. Breadboard 6](#_Toc162960203)

[1.3. Led and Resistors 6](#_Toc162960204)

[1.4. Breadboard Plug Wire 6](#_Toc162960205)

[2. Software 6](#_Toc162960206)

[**III.**  **Practical Implementation** 7](#_Toc162960207)

[1. Set up on Blynk and get the API Key 7](#_Toc162960208)

[1.1. Create a Blynk account 7](#_Toc162960209)

[1.2. Create a new Template 7](#_Toc162960210)

[1.3. Setup for this template 8](#_Toc162960211)

[1.4. Setup Web Dashboard 9](#_Toc162960212)

[2. Connecting IoT Devices 10](#_Toc162960213)

[2.1. Module ESP8266 and DHT11 10](#_Toc162960214)

[2.2. Module ESP8266 and led RGB 11](#_Toc162960215)

[2.3 Module ESP8266 and Buzzer 11](#_Toc162960216)

[2.4 Module ESP8266 and LCD(through I2C) 11](#_Toc162960217)

[2.5 Complete 12](#_Toc162960218)

[3. Upload the Code and Software to the Arduino board 12](#_Toc162960219)

[3.1. Arduino board 12](#_Toc162960220)

[3.2. Uploading the Code to Arduino IDE for the Project 13](#_Toc162960221)

[**IV.**  **Results** 14](#_Toc162960222)

[**V.**  **Conclusion** 15](#_Toc162960223)

[**VI.**  **Appendix** 16](#_Toc162960224)

List of Figures and Tables

Figure 1 ESP8266............................................................................................... 5

Figure 2 Temperature Sensor ............................................................................. 5

Figure 3 DHT11 ................................................................................................. 7

Figure 4 LCD (I2C)............................................................................................ 7

Figure 5 Breadboard........................................................................................... 7

Figure 6 Led and resistors ................................................................................. 7

Figure 7 Breadboard Plug Wire ......................................................................... 8

Figure 8 Create Blynk Accout .................................................................. 9

Figure 9 New Channel ......................................................................................10

Figure 10 Enter basic information and save channel ....................................... 10

Figure 11 Channel ID and API Key on Blynk .........................................11

Figure 12 Completed model .............................................................................13

Figure 13 Blynk results ...........................................................................16

Figure 14 When temperature exceeds 30 degrees ............................................17

Figure 15 When temperature below 30 degrees ...............................................18

Table 1 ESP8266 and DHT11 ..........................................................................11

Table 2 ESP8266 and LCD ............................................................................. 12

Table 3 ESP8266 and LED ..............................................................................12

# **I. Introduction**

## 1. Project

The project aims to create an Internet of Things (IoT) based environmental monitoring system using the ESP8266 module, DHT11 temperature and humidity sensor, Blynk for data storage and visualization, an LCD screen for real-time display. The system will collect real-time data on temperature and humidity, transmit the information to Blynk for remote monitoring, display the values on an LCD screen. In addition, base on the huminity and temperature given, this system will have some led display and buzzer on for the notification when temperature too high

## 2. ESP8266

 The ESP8266 Wi-Fi module is a system-on-chip (SoC) microchip that is commonly used in the development of Internet of Things (IoT) applications. This module functions as a standalone wireless transceiver that is readily available at a low cost. Its main purpose is to facilitate internet connectivity for various embedded systems applications. With its low cost and ease of use, the ESP8266 is a popular choice for enabling wireless communication in IoT devices.

Figure 1: ESP8266

Its capabilities include the ability to collect and transmit data from sensors, interact with other devices over Wi-Fi networks, and support remote firmware updates. Overall, the ESP8266 module is a versatile and cost-effective solution for IoT development projects.

## 3. Temperature Sensor

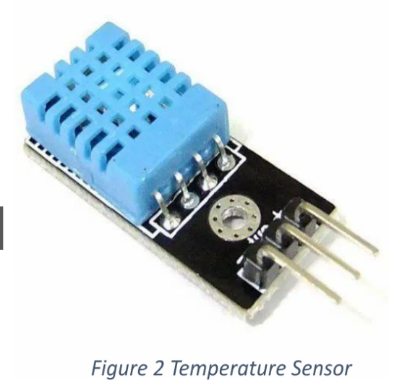
 A temperature sensor is an electronic device designed to detect and measure changes in temperature. It works by converting the physical parameter of temperature into an electrical signal that can be processed and analyzed by a computer or other electronic device. Temperature sensors are commonly used in a wide range of applications, including environmental monitoring, process control, and home automation systems. There are many different types of temperature sensors available, each with its own unique

Figure 2: DHT11

characteristics and applications. Some of the most common types include thermocouples, resistance temperature detectors (RTDs), and thermistors. Regardless of the specific type, all temperature sensors play a critical role in many different fields and industries by providing accurate and reliable temperature measurements.

## 4. LCD

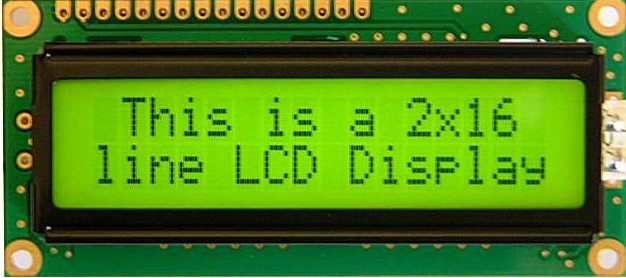
LCD (Liquid Crystal Display) is a type of flat panel display which uses liquid crystals in its primary form of operation. LEDs have a large and varying set of use cases for consumers and businesses, as they can be commonly found in smartphones, televisions, computer monitors and instrument panels.

Figure 3: LCD

## 5. Led RGB

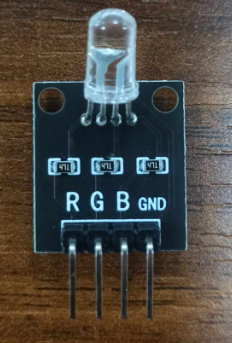
LED, in electronics, a semiconductor device that emits infrared or visible light when charged with an electric current. Visible LEDs are used in many electronic devices as indicator lamps, in automobiles as rear-window and brake lights, and on billboards and signs as alphanumeric displays or even full-colour posters

Figure 4: LED RGB

## 6. Buzzer

An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.

Figure 5: Buzzer

# **II. Equipment specifics**

## 1. Hardware

### 1.1. Temperature and humidity sensor DHT11

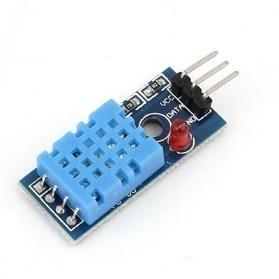
The DHT11 is a low-cost digital sensor that detects temperature and humidity. It can be easily connected to micro-controllers like Arduino and Raspberry Pi to measure humidity and temperature in real-time. The DHT11 is available both as a standalone sensor and as a module. 

Figure 6: Sensor

### 1.2. Breadboard

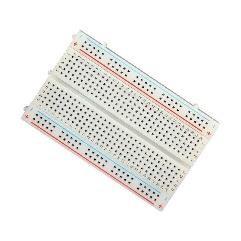
A breadboard is a solderless construction base used to build and test electronic circuits and wiring for projects with microcontroller boards like Arduino. Despite its ubiquity, it can be intimidating for beginners to use. Familiarity with 

Figure 7: Board

breadboards and their usage is essential for developing and testing electronics projects.

### 1.3. Led and Resistors

Resistors are used to limit the amount of current flowing through an LED, preventing it from drawing too much power and potentially damaging the LED or other components in the circuit. For example, if an LED is rated for 30mA and is 

supplied with 5V by an Arduino pin, a 166.6 ohm resistor can be used to ensurethat the LED does not exceed its rated

Figure 8: Led

current. This is a critical component of electronic circuit design and helps to protect the components from damage

### 1.4. Breadboard Plug Wire

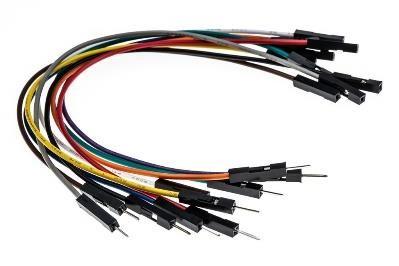
Breadboards are a fundamental tool for constructing and wiring electronic circuits in projects involving microcontroller boards like Arduino. However, beginners in electronics may find the process of using a breadboard daunting, despite its widespread use in the field. With proper guidance and practice, however, it is possible to quickly become familiar with breadboards and effectively utilize them in the dev .elopment of electronic projects.

Figure 9: Wire Connection

## 2. Software

Library DHT11, LCD and ESP8266Wifi

# **III. Practical Implementation**

## 1. Set up on Blynk and get the API Key

### 1.1. Create a Blynk account

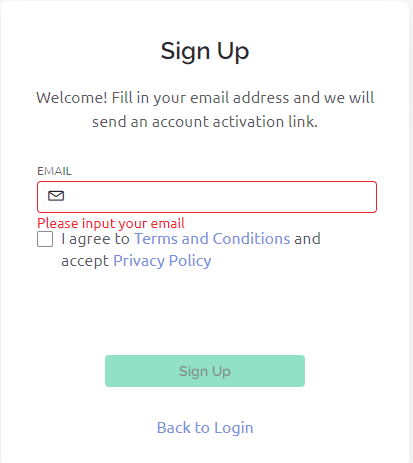
 To get started with Blynk, go to the platform's website at https://blynk.cloud/ and create an account by providing your email address and a password. Once you have created an account, log in to your account by entering your login credentials. You will be directed to the Blynk dashboard, where you can create new channels for your IoT devices, visualize and analyze data, and manage your account settings. Before you can start sending data to Blynk from your IoT devices, you need to obtain an API key, which serves as a unique identifier for your account and is necessary for accessing the various functionalities of Blynk's API.

Figure 10: Create Blynk account

### 1.2. Create a new Template

To create a new channel on Blynk, click on the Developer Zone' button on the Blynk dashboard and next click to “New template”. You will be prompted to enter a name and description for your channel,as well as chose the hardware and connection type. For this project I chose hardware is ESP8266 and connection is

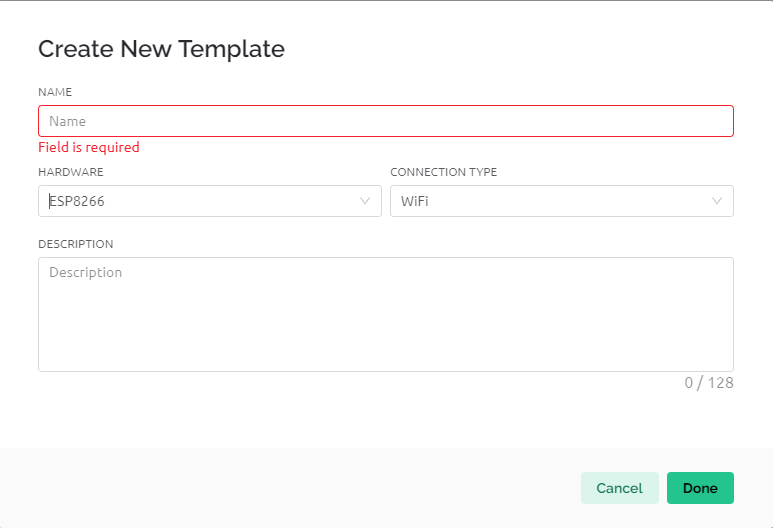


Figure 11: Create new template

*Figure 9 New Channel*

### 1.3. Setup for this template

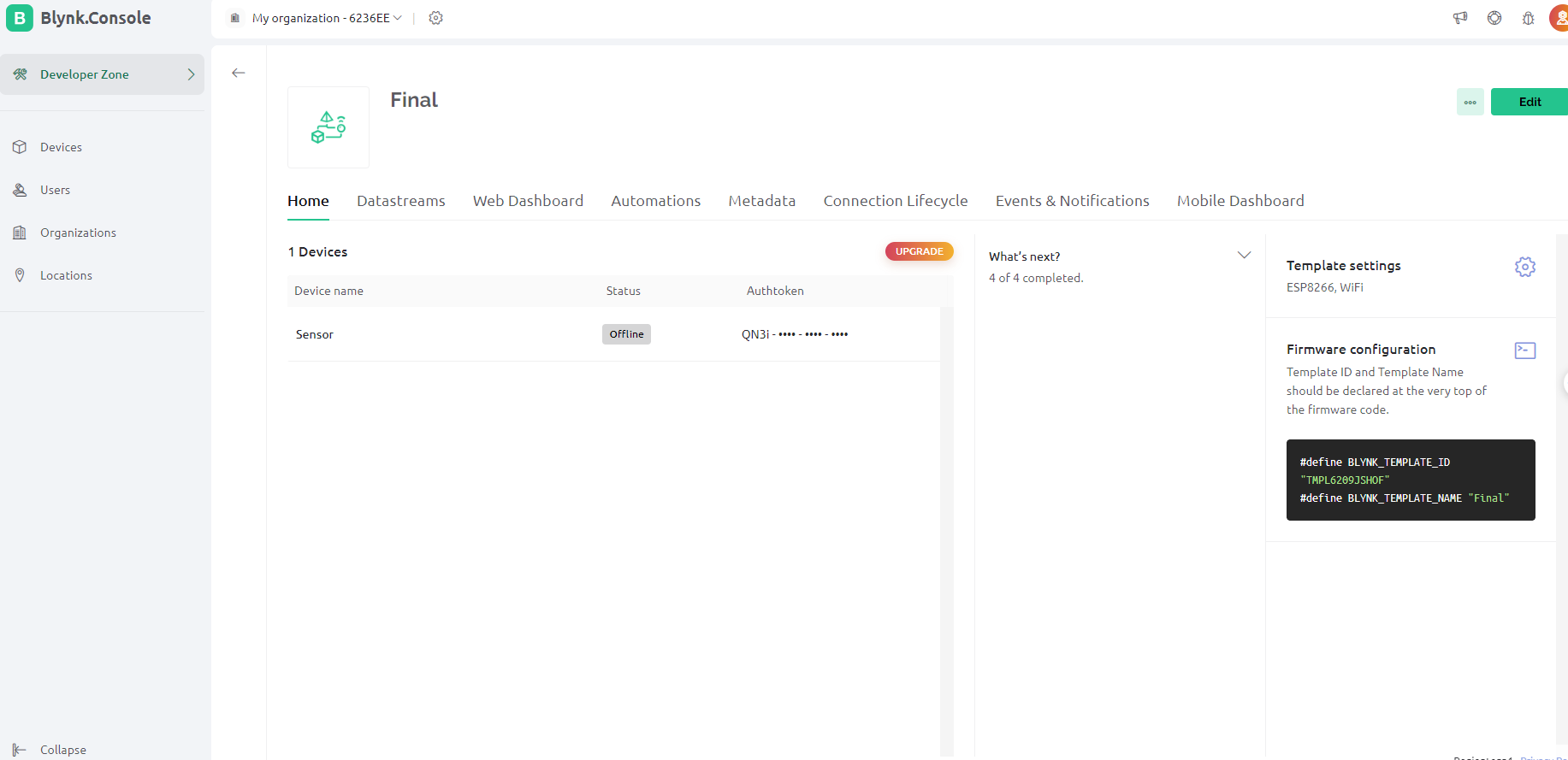
Once you created a template, you need to setup what your template have to do. Like this picture below. After complete setting, you can add a unique channel ID and write API key that you can use to send data to your channel from your IoT devices.

Figure 12: Blynk Console

### 1.4. Setup Web Dashboard

After created, click to web dashboard to continute, Here I chose 3 tool that include 1 Switch and 2 Gauge. Below is the setting I set for each tool

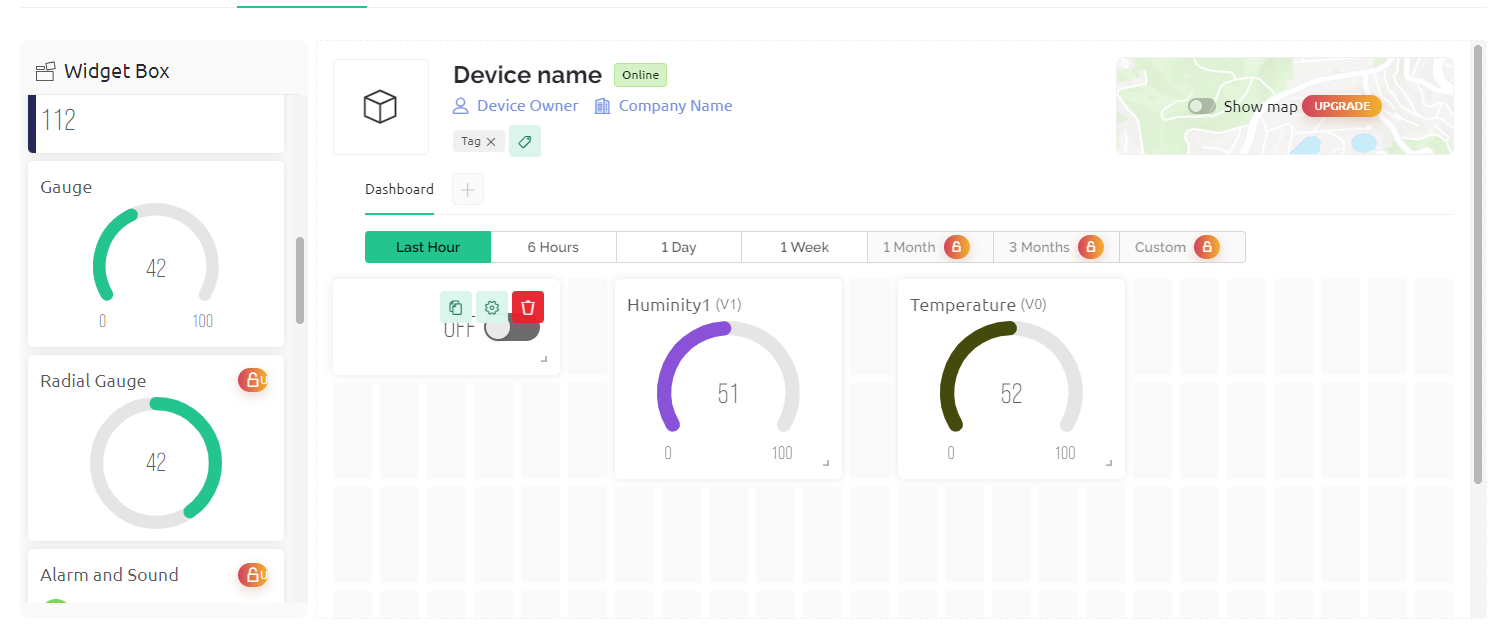


Figure 13: Template setting

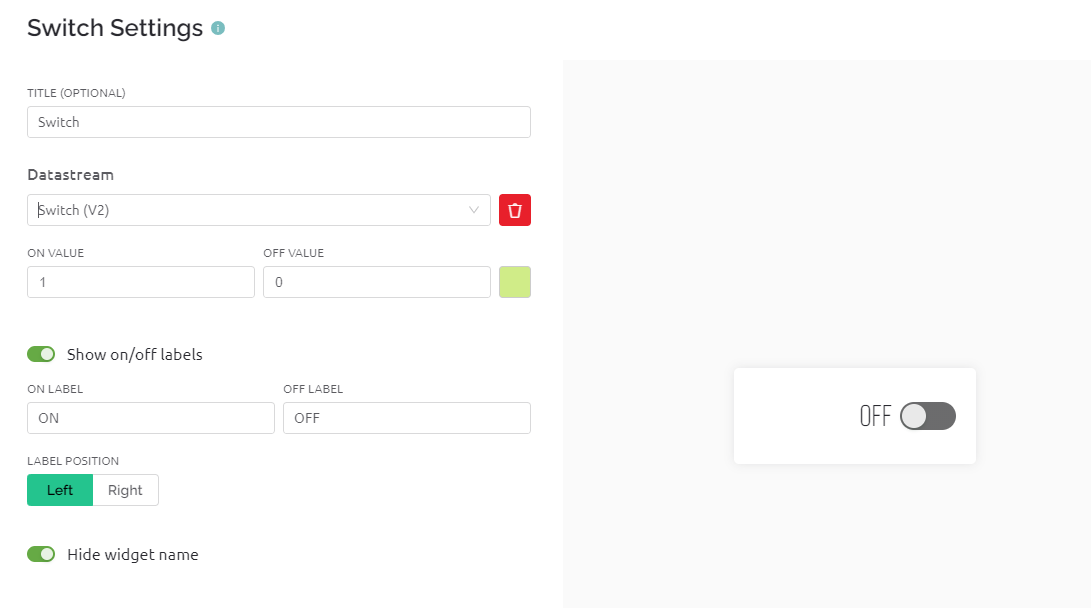
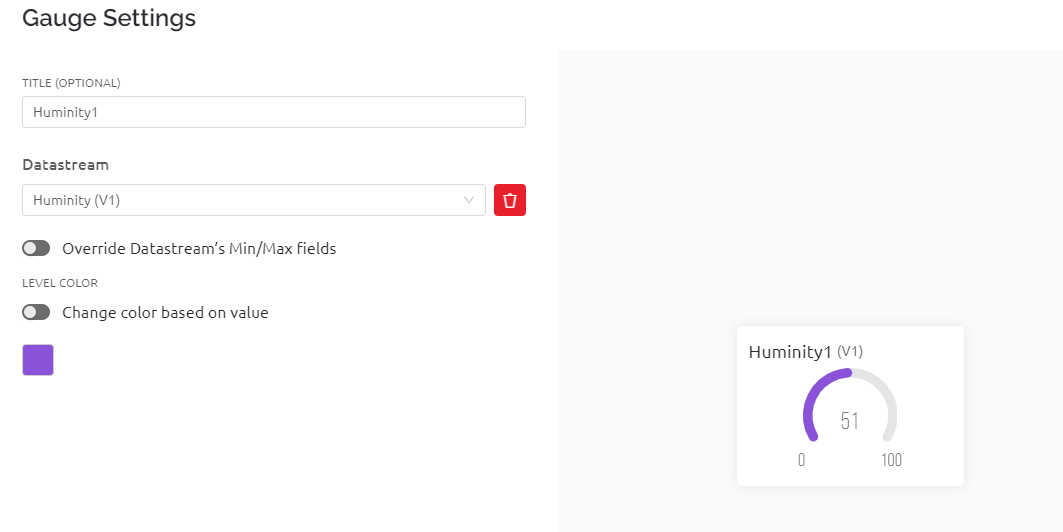
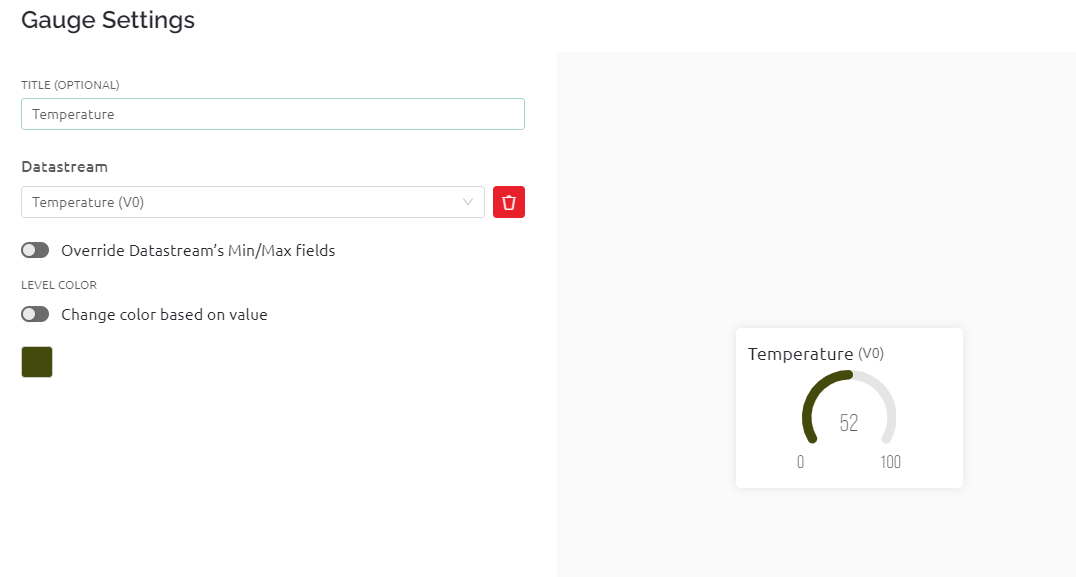


Figure 14: Switch Button Setting





## 2. Connecting IoT Devices

### 2.1. Module ESP8266 and DHT11

|  |  |
| --- | --- |
| DHT | ESP |
| VCC | 3V |
| DATA | D4 |
| GND | GND |

*Table 1 ESP8266 and DHT11*

* Connect VCC pin of DHT11 to the 3V pin of the ESP8266 module.
* Connect DATA pin of DHT11 to the D4 pin of the ESP8266 module.
* Connect GND pin of DHT11 to the GND pin of the ESP8266 module.

Make sure to power the ESP8266 module properly, and also connect the required resistors if necessary for signal conditioning. Additionally, you need to write a code to read data from the DHT11 sensor using the ESP8266 module.

### 2.2. Module ESP8266 and led RGB

|  |  |
| --- | --- |
| LED RGB | ESP |
| GND | GND |
| R | D6 |
| G | D7 |
| B | D8 |

*Table 3 ESP8266 and LED*

* Connect the GND pin of the LED RGB to the GND pin of the ESP8266 module.
* Connect R pin of LED RGB to D6 pin of ESP8266
* Connect G pin of LED RGB to D7 pin of ESP8266
* Connect B pin of LED RGB to D8 pin of ESP8266

### 2.3 Module ESP8266 and Buzzer

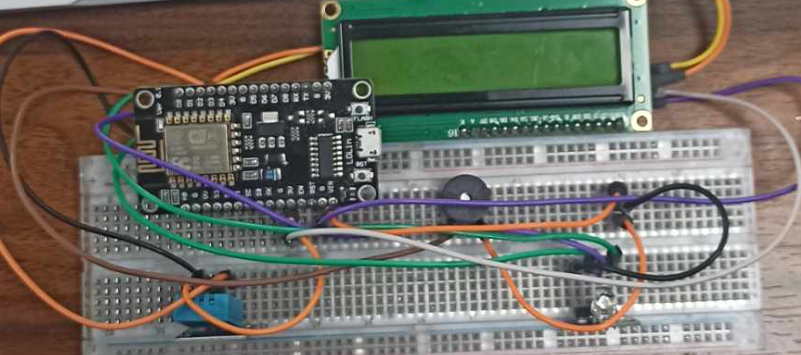
|  |  |
| --- | --- |
| Buzzer | ESP |
| GND | GND |
| VCC | D0 |

* Connect the GND pin of the Buzzer to the GND pin of the ESP8266 module.
* Connect VCC of Buzzer to D0 pin of ESP8266

### 2.4 Module ESP8266 and LCD(through I2C)

|  |  |
| --- | --- |
| I2C | ESP |
| GND | GND |
| VCC | 3V |
| SDA | D2 |
| SCL | D1 |

### 2.5 Complete



*Figure 12 Completed model*

## 3. Upload the Code and Software to the Arduino board

### 3.1. Arduino board

* Download and install the Arduino IDE software on your computer from the official website.
* Connect the ESP8266 module and other components to your Arduino board as per the instructions provided.
* Open the Arduino IDE software on your computer and create a new project.
* Copy and paste the code that you have written for the project into the Arduino IDE software.
* Configure the Arduino IDE software to recognize your Arduino board. In the "Tools" menu, select the board type and the serial port that your board is connected to.
* Verify the code for any errors or warnings using the "Verify" button in the Arduino IDE software.
* Upload the code to the Arduino board by clicking the "Upload" button in the Arduino IDE software.
* Once the code is successfully uploaded, open the serial monitor in the Arduino IDE software to view the data being read from the DHT11 sensor and displayed on the LCD display.
* Test the functionality of the LEDs by varying the temperature and humidity and observing the changes in LED color.

With these steps, you can upload the code and software to your Arduino board and ensure that the project is working as intended.

### 3.2. Uploading the Code to Arduino IDE for the Project

Uploading the code[[1]](#footnote-1) to the Arduino IDE is an important step in the process of completing a project. This step involves transferring the code from the software environment to the physical Arduino board, which allows the project to run on the board. The Arduino IDE provides a simple and straightforward way to upload the code to the board, and this process can be completed in just a few steps.

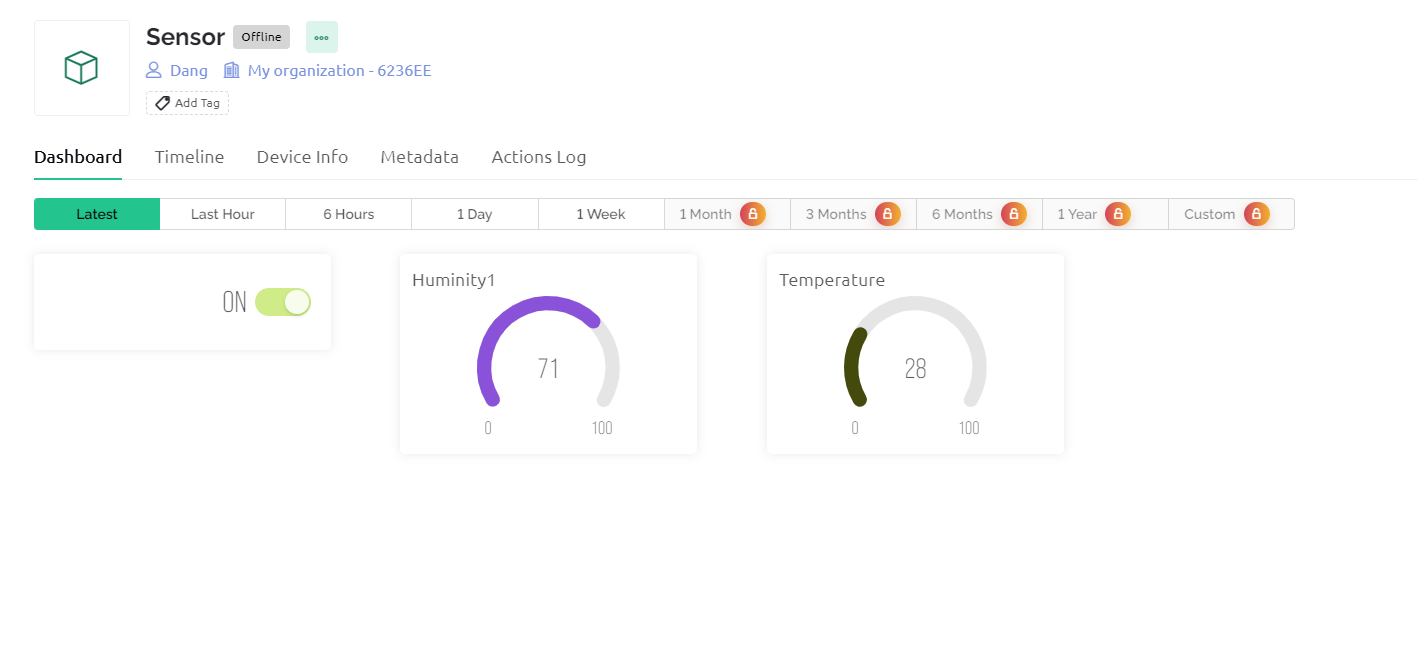
To upload the code to the Arduino IDE, you will need to connect your Arduino board to your computer using a USB cable. Once the board is connected, you can open the Arduino IDE software and create a new sketch by selecting "File" and then "New". You can then copy and paste the code for your project into the new sketch window.

Before uploading the code to your Arduino board, you will need to ensure that the correct board type and serial port are selected. This can be done by going to the "Tools" menu and selecting the appropriate options for your board. Once you have verified that the board type and serial port are correct, you can upload the code to your Arduino board by selecting "Sketch" and then "Upload".

The Arduino IDE software will then compile the code and upload it to your board. Once the upload is complete, you can test the functionality of your project to ensure that it is working as intended. If you encounter any issues during this process, you can refer to the troubleshooting section in the Arduino IDE documentation or seek help from online forums and communities.

# **IV. Results**

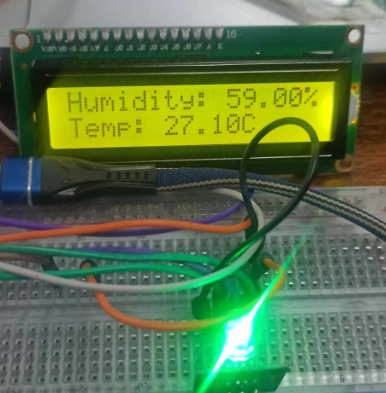
Based on the information provided, the project was successfully completed by connecting the ESP8266 module and the DHT11 temperature and humidity sensor, the ESP8266 module and the LCD, and the ESP8266 module and the LED. The connections were made by following the provided instructions and uploading the appropriate code to the Arduino IDE.



*Figure 13 Blynk results*

The temperature and humidity readings from the DHT11 sensor were successfully displayed on the LCD screen, demonstrating the successful communication between the ESP8266 module and the sensor.

*Figure 14 When temperature exceeds 30 degrees*

 Conversely, the green light will turn on when the temperature is lower than 30 degree C and blue light when higher and the buzzer also ring on the same time

*Figure 15 When temperature below 30 degrees*

Moreover, I also design one more button to control my system. The system only run when this button is on. Otherwise, all the function of this system like sensor, LED, Buzzer will be off

# **V. Conclusion**

In conclusion, utilizing an ESP8266 module, DHT11 sensor, LCD screen, Buzzerand LED lights, we were able to construct a working temperature monitoring system. Temperature and humidity data may be precisely measured by the system and shown on the LCD display in real time. Additionally, it uses LED lights to visually feedback when the temperature is above or below a specific threshold. This project emphasizes the value of temperature monitoring in a variety of applications, including agriculture, HVAC systems, and food storage. It also displays the flexibility and possibilities of the ESP8266 module for constructing IoT projects.

# **VI. Appendix**

#define BLYNK\_TEMPLATE\_ID "TMPL6209JSHOF"

#define BLYNK\_TEMPLATE\_NAME "Final"

#define BLYNK\_AUTH\_TOKEN "QN3ib9zHipvjL-JbHtL1wZo39kuu4mg-"

// Đèn RGB

#define LED\_R\_PIN D6

#define LED\_G\_PIN D7

#define LED\_B\_PIN D8

//Cảm biến nhiệt độ, độ ẩm

#define DHT\_SENSOR\_PIN  D4 // The ESP8266 pin D7 connected to DHT11 sensor

#define DHT\_SENSOR\_TYPE DHT11

// #define DHT\_SENSOR\_TYPE DHT22

//Còi

#define BUZZER\_PIN        D1

#include <DHT.h>

#include <BlynkSimpleEsp8266.h>

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

DHT dht\_sensor(DHT\_SENSOR\_PIN, DHT\_SENSOR\_TYPE);

LiquidCrystal\_I2C lcd(0x27, 16, 2); // set the LCD address to 0x27 for a 16 chars and 2 line display

// const char\* ssid = "VNU-IS THMT 02";

const char\* ssid = "VNU-IS Phong Khanh Tiet";

const char\* password = "Vnu.edu.vn";

int buttonState = 0;

void setup() {

  Serial.begin(9600);

  Blynk.begin(BLYNK\_AUTH\_TOKEN, ssid, password);

  dht\_sensor.begin();

  // Initialize LED pins as OUTPUT

  pinMode(LED\_R\_PIN, OUTPUT);

  pinMode(LED\_G\_PIN, OUTPUT);

  pinMode(LED\_B\_PIN, OUTPUT);

  pinMode(BUZZER\_PIN, OUTPUT);

  lcd.begin();         // initialize the lcd

  lcd.backlight();

  lcd.clear(); // Clear the LCD screen

  lcd.setCursor(5, 0);

  lcd.print("HELLO");

  lcd.setCursor(0, 1);

  lcd.print("BUTTON IS OFF");

}

void runDHT() {

  // read humidity

  float humi  = dht\_sensor.readHumidity();

  // read temperature in Celsius

  float temperature\_C = dht\_sensor.readTemperature();

  // read temperature in Fahrenheit

  float temperature\_F = dht\_sensor.readTemperature(true);

  // check whether the reading is successful or not

  if (isnan(temperature\_C) || isnan(temperature\_F) || isnan(humi)) {

    Serial.println("Failed to read from DHT sensor!");

  } else {

    Serial.print("Humidity: ");

    Serial.print(humi);

    Serial.print("%");

    Serial.print("  |  ");

    Serial.print("Temperature: ");

    Serial.print(temperature\_C);

    Serial.print("°C  ~  ");

    Serial.print(temperature\_F);

    Serial.println("°F");

    // Push data to Blynk

    Blynk.virtualWrite(V1, humi);

    Blynk.virtualWrite(V0, temperature\_C);

    //lcd

    lcd.clear();

    lcd.setCursor(0, 0);

    lcd.print("Humidity: ");

    lcd.print(humi);

    lcd.print("%");

    lcd.setCursor(0, 1);

    lcd.print("Temp: ");

    lcd.print(temperature\_C);

    lcd.print("C");

    // Control LED color based on temperature

    if (temperature\_C > 30) {

      // Red color

      analogWrite(LED\_R\_PIN, 0); // Full brightness

      analogWrite(LED\_G\_PIN, 0);   // No green

      analogWrite(LED\_B\_PIN, 255);   // No blue

      // Turn on buzzer

      digitalWrite(BUZZER\_PIN, HIGH);

    }

    else {

      // Blue color

      analogWrite(LED\_R\_PIN, 0);   // No red

      analogWrite(LED\_G\_PIN, 255);   // No green

      analogWrite(LED\_B\_PIN, 0); // Full brightness

      // Turn on buzzer

      digitalWrite(BUZZER\_PIN, LOW);

    }

  }

}

void loop() {

  Blynk.run();

  // Đọc trạng thái của nút và xử lý

  if (buttonState == 1) {

    runDHT();

  } else {

    Serial.println("Button is OFF");

    lcd.clear(); // Clear the LCD screen

    lcd.setCursor(0, 0);

    lcd.print("HELLO");

    lcd.setCursor(0, 1);

    lcd.print("BUTTON IS OFF");

    analogWrite(LED\_G\_PIN, 0);

    analogWrite(LED\_B\_PIN, 0);

    digitalWrite(BUZZER\_PIN, LOW);

  }

  delay(2000);

}

BLYNK\_WRITE(V2) {

  buttonState = param.asInt(); // Đọc trạng thái của nút (0: Tắt, 1: Bật)

}

1. In part 6: Appendix [↑](#footnote-ref-1)