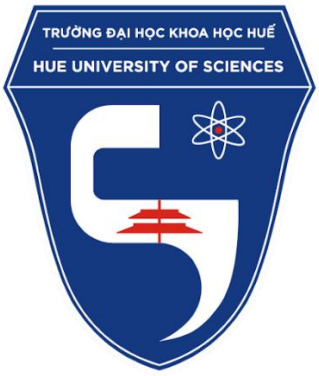
TRƯỜNG ĐẠI HỌC KHOA HỌC



SỐ PHÁCH:.....

Xây dựng trạm thời tiết mini với ESP32

Môn: PHÁT TRIỂN ỨNG DỤNG IOT

Sinh viên thực hiện:Nguyễn Đức Huy

Giảng viên hướng dẫn: Ths.Võ Việt Dũng

HUẾ, THÁNG 4 NĂM 2025.

MỤC LỤC

Chương 1: Giới thiệu đề tài

1. Giới thiệu

1.1. Lý do chọn đề tài

Trong bối cảnh hiện nay, việc giám sát các thông số môi trường như nhiệt độ, độ ẩm và áp suất đóng vai trò quan trọng trong nhiều lĩnh vực, từ nông nghiệp đến quản lý tòa nhà thông minh. Ban đầu, cảm biến BME280 thường được sử dụng cho mục đích này do khả năng đo đồng thời cả ba thông số trên. Tuy nhiên, trong một số trường hợp, việc sử dụng cảm biến BME280 có thể gặp hạn chế về mặt sẵn có hoặc chi phí. Do đó, việc thay thế bằng cảm biến DHT22 để đo nhiệt độ và độ ẩm, kết hợp với việc sinh giá trị áp suất ngẫu nhiên, là một giải pháp khả thi và kinh tế.

Cảm biến DHT22 nổi bật với độ chính xác cao và phạm vi đo rộng, cho phép đo nhiệt độ từ -40°C đến 80°C với độ chính xác ±0.5°C, và độ ẩm từ 0% đến 100% với độ chính xác 2-5% . Ngoài ra, việc tích hợp hệ thống với nền tảng Blynk giúp người dùng có thể giám sát và điều khiển từ xa thông qua ứng dụng di động, mang lại sự tiện lợi và hiệu quả trong việc quản lý các thông số môi trường .

1.2. Mục tiêu của đề tài

Đề tài hướng đến việc xây dựng một hệ thống giám sát môi trường với các mục tiêu cụ thể sau:

- **Giám sát nhiệt độ, độ ẩm và áp suất:** Sử dụng cảm biến DHT22 để đo nhiệt độ và độ ẩm, trong khi giá trị áp suất được sinh ngẫu nhiên để mô phỏng dữ liệu.
- **Hiển thị trực tiếp trên màn hình OLED:** Kết quả đo được hiển thị trên màn hình OLED SSD1306, cho phép người dùng quan sát trực tiếp các thông số môi trường .
- **Cảnh báo khi vượt ngưỡng bằng đèn LED:** Hệ thống sẽ kích hoạt đèn LED cảnh báo khi nhiệt độ hoặc độ ẩm vượt ngưỡng cho phép, giúp người dùng nhận biết nhanh chóng tình trạng môi trường.
- **Thay đổi ngưỡng qua ứng dụng Blynk:** Người dùng có thể thiết lập và điều chỉnh các ngưỡng cảnh báo thông qua ứng dụng Blynk trên điện thoại di động, tăng tính linh hoạt và tiện dụng .

1.3. Phạm vi thực hiện

Đề tài được thực hiện trong phạm vi mô phỏng trên nền tảng Wokwi với các thành phần chính sau:

Cảm biến DHT22: Đo nhiệt độ và độ ẩm môi trường với độ chính xác cao

Màn hình OLED SSD1306: Hiển thị các thông số đo được một cách rõ ràng và trực quan .

Nền tảng Wokwi: Sử dụng để mô phỏng toàn bộ hệ thống, cho phép kiểm tra và đánh giá hiệu suất trước khi triển khai thực tế .

Giá trị áp suất: Được sinh ngẫu nhiên để mô phỏng dữ liệu, thay thế cho cảm biến BME280 trong trường hợp không khả dụng.

Việc sử dụng nền tảng Wokwi giúp giảm thiểu chi phí phần cứng và cho phép thử nghiệm, điều chỉnh hệ thống một cách linh hoạt trước khi triển khai thực tế.

1.4. Phương pháp thực hiện

Để hoàn thành đề tài này, các bước thực hiện được tiến hành như sau:

- **Tìm hiểu lý thuyết**: Nghiên cứu các tài liệu, sách vở, trang web và datasheet của các linh kiện như ESP32, PIR, LCD I2C, buzzer... để nắm rõ nguyên lý hoạt động và cách lập trình điều khiển.
- **Thiết kế sơ đồ nguyên lý**: Dựa trên nhu cầu hệ thống, tiến hành thiết kế sơ đồ nguyên lý điện bao gồm kết nối giữa các thiết bị phần cứng.
- **Mô phỏng phần cứng trên Wokwi**: Xây dựng mô hình phần cứng trên nền tảng Wokwi để kiểm thử và mô phỏng các hoạt động của hệ thống trong điều kiện không cần thiết bị thật.
- **Lập trình điều khiển**: Viết code bằng Arduino IDE để ESP32 có thể đọc tín hiệu từ cảm biến, xử lý logic và điều khiển các thiết bị cảnh báo, đồng thời hiển thị trạng thái lên LCD.
- **Kiểm thử và đánh giá**: Tiến hành chạy mô phỏng, điều chỉnh thông số và khắc phục lỗi nếu có. Đánh giá hoạt động của hệ thống theo các tiêu chí hiệu quả, độ chính xác và khả năng mở rộng.
- **Tổng kết và đề xuất**: Đưa ra những nhận xét về kết quả đạt được, khó khăn gặp phải trong quá trình thực hiện và hướng phát triển hệ thống trong tương lai.

Chương 2: Tổng quan hệ thống

2.1. Giới thiệu tổng quan

Trong thời đại phát triển mạnh mẽ của công nghệ IoT (Internet of Things), việc xây dựng các hệ thống giám sát và điều khiển thông minh trở nên ngày càng phổ biến. Hệ thống giám sát nhiệt độ, độ ẩm và áp suất mà đề tài này hướng đến là một ví dụ tiêu biểu trong việc ứng dụng vi điều khiển vào thực tiễn.

Hệ thống được thiết kế dựa trên vi điều khiển **ESP32 DevKit V1**, một loại vi điều khiển hiện đại với khả năng kết nối WiFi, phục vụ cho các ứng dụng IoT một cách linh hoạt. Cảm biến **DHT22** được sử dụng để đo các thông số nhiệt độ và độ ẩm môi trường. Kết quả đo được cập nhật và hiển thị lên màn hình **OLED SSD1306**, đồng thời cũng được gửi đến ứng dụng **Blynk** để người dùng có thể theo dõi từ xa trên thiết bị di động.

Ngoài ra, hệ thống còn tích hợp **đèn LED cảnh báo** khi các chỉ số đo vượt ngưỡng do người dùng cài đặt. Người dùng cũng có thể chủ động thay đổi các ngưỡng nhiệt độ và độ ẩm thông qua ứng dụng Blynk, từ đó kiểm soát hệ thống một cách chủ động và linh hoạt hơn.

2.2. Các thành phần phần cứng

2.2.1. ESP32 DevKit V1

ESP32 là một dòng vi điều khiển mạnh mẽ với khả năng xử lý cao, hỗ trợ kết nối WiFi và Bluetooth tích hợp. Trong hệ thống này, **ESP32 DevKit V1** đóng vai trò là bộ xử lý trung tâm, thực hiện các chức năng chính:

Thu thập dữ liệu từ cảm biến DHT22.

Xử lý và hiển thị dữ liệu lên màn hình OLED

Gửi dữ liệu đến nền tảng Blynk để giám sát từ xa qua Internet.

Kích hoạt các thiết bị cảnh báo, như đèn LED, khi các thông số vượt ngưỡng cài đặt.

ESP32 hỗ trợ nhiều chân GPIO, dễ dàng kết nối với nhiều thiết bị ngoại vi, phù hợp với các dự án yêu cầu độ linh hoạt cao.

Hình 2.2.1: ESP32 DevKit V1

2.2.2. Cảm biến DHT22

Cảm biến **DHT22** là một thiết bị đo độ ẩm và nhiệt độ có độ chính xác cao, phổ biến trong các dự án IoT:

Đo nhiệt độ trong khoảng từ -40°C đến +80°C với sai số ±0.5°C.

Đo độ ẩm trong khoảng từ 0% đến 100% RH với sai số ±2% RH.

Cảm biến này truyền dữ liệu theo giao tiếp 1-wire (digital) và tương thích tốt với vi điều khiển ESP32.

So với các loại cảm biến khác như **DHT11**, DHT22 có phạm vi đo rộng hơn và độ chính xác cao hơn, mặc dù giá thành có phần cao hơn.

Hình 2.2.2: Cảm biến DHT22

2.2.3. Màn hình OLED SSD1306 (I2C)

Màn hình OLED **SSD1306** là loại màn hình có kích thước nhỏ gọn, tiêu thụ ít điện năng và cho khả năng hiển thị tốt, rất phù hợp với các dự án nhúng:

Kích thước: 0.96 inch

Độ phân giải: 128x64 pixel

Giao tiếp: I2C (chỉ sử dụng 2 chân: SDA và SCL)

Màn hình OLED SSD1306 được dùng để hiển thị các thông tin như: nhiệt độ, độ ẩm, áp suất (mô phỏng), và trạng thái hệ thống (báo động, ngưỡng cài đặt...).

Hình 2.2.3: màn hình OLED SSD1306 (I2C)

2.2.4. Đèn LED cảnh báo

Hệ thống sử dụng **hai đèn LED** với chức năng cảnh báo:

LED đỏ (tempLED): Sáng khi nhiệt độ vượt quá ngưỡng cho phép.

LED xanh lá (pumpLED): Có thể được sử dụng để cảnh báo độ ẩm cao hoặc báo hiệu trạng thái đang kích hoạt quạt/tưới tự động (tùy cấu hình hệ thống).

Mỗi đèn LED được nối tiếp với một **điện trở 220Ω** nhằm giới hạn dòng điện và bảo vệ thiết bị.

Hình 2.2.4: Đèn LED

2.2.5. Điện trở

Điện trở được dùng để hạn dòng điện đi qua LED nhằm tránh hư hỏng:

Mỗi LED được mắc nối tiếp với **điện trở 220Ω**, đảm bảo dòng điện không vượt quá giới hạn cho phép.

Điện trở giúp bảo vệ linh kiện LED khỏi quá dòng và tăng tuổi thọ linh kiện.

Hình 2.2.5: Điện trở

2.3. Mô phỏng phần mềm

2.3.1. Mô phỏng trên nền tảng Wokwi

Wokwi là một công cụ mô phỏng hệ thống nhúng trực tuyến, cho phép người dùng mô phỏng phần cứng và chạy mã Arduino/ESP32 một cách trực quan. Trong dự án này, Wokwi được sử dụng để mô phỏng toàn bộ hệ thống bao gồm:

- ESP32
- Cảm biến DHT22
- Màn hình OLED SSD1306
- Đèn LED cảnh báo

Việc sử dụng Wokwi mang lại nhiều lợi ích:

Không cần phần cứng thực tế, tiết kiệm chi phí và thời gian.

Dễ dàng kiểm thử và sửa lỗi phần mềm ngay trên trình duyệt.

Quan sát trực quan, dễ hiểu cách các linh kiện hoạt động và kết nối với nhau.

Chia sẻ dễ dàng, vì Wokwi có thể lưu dự án và gửi link cho người khác.

Hình 2.3.1: nền tảng wokwi

2.3.2. Ứng dụng Blynk

Blynk là một nền tảng phát triển ứng dụng IoT cho phép điều khiển và giám sát thiết bị từ xa thông qua giao diện người dùng trên điện thoại. Trong hệ thống này, Blynk đảm nhận các chức năng:

Hiển thị dữ liệu từ cảm biến: nhiệt độ, độ ẩm (áp suất mô phỏng).

Cài đặt ngưỡng cảnh báo thông qua các widget (Slider, Numeric Input).

Nhận cảnh báo từ xa khi các chỉ số vượt ngưỡng.

Kích hoạt hoặc tắt các thiết bị từ xa (như đèn LED, quạt, máy bơm... nếu có).

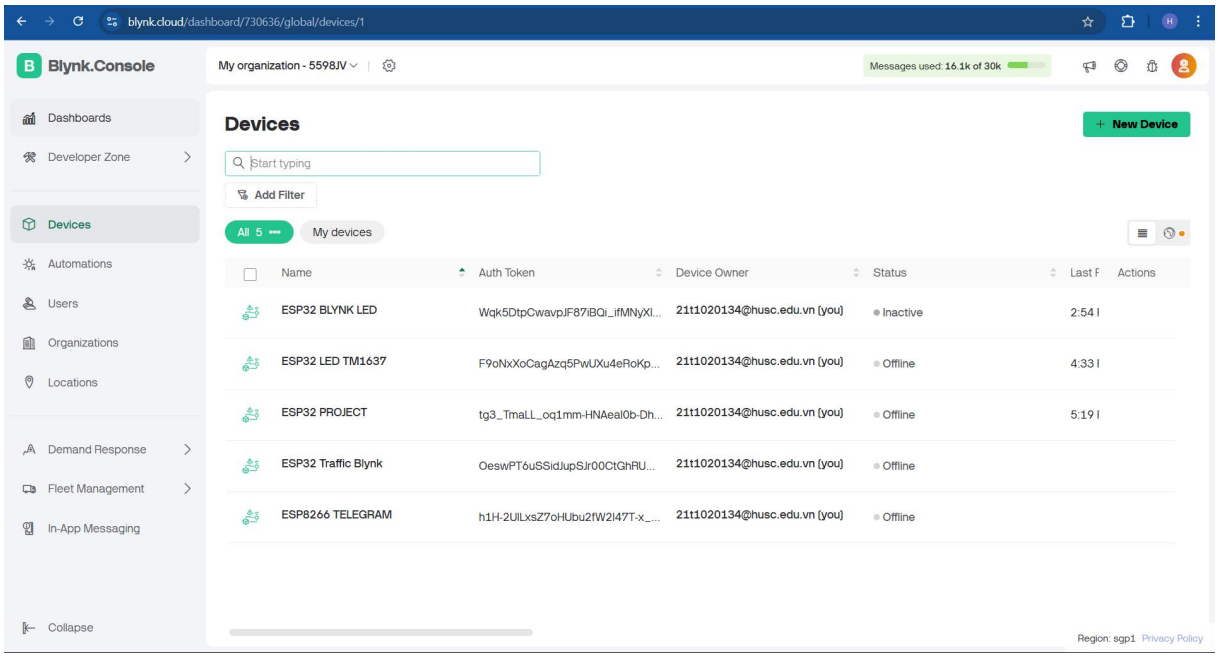
Ưu điểm của Blynk:

Giao diện **kéo – thả**, dễ thiết kế và tùy biến.

Đa nền tảng: hỗ trợ Android, iOS và cả giao diện Web.

Có **API mạnh mẽ**, hỗ trợ Arduino, ESP32, Raspberry Pi…

Phù hợp với **các dự án IoT quy mô nhỏ đến vừa**.



Hình 2.3.2: Ứng dụng Blynk

Chương 3: Thiết kế và triển khai hệ thống

3.1. Sơ đồ nguyên lý kết nối

Sơ đồ nguyên lý mô tả cách kết nối các thành phần phần cứng với nhau như sau:

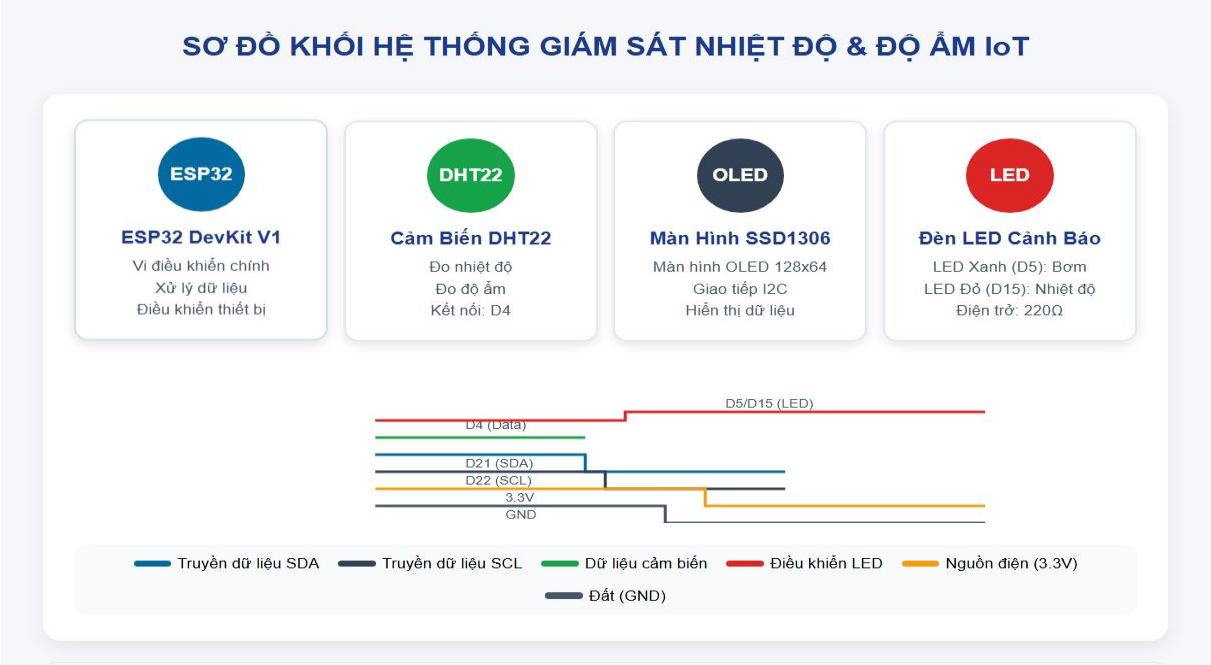
ESP32 DevKit V1: Vi điều khiển trung tâm, đảm nhiệm việc thu thập dữ liệu, xử lý và truyền thông tin đến các thiết bị ngoại vi cũng như lên ứng dụng Blynk.

Cảm biến DHT22: Đo nhiệt độ và độ ẩm môi trường, truyền dữ liệu đến ESP32.

Màn hình OLED SSD1306: Hiển thị dữ liệu nhiệt độ, độ ẩm và các trạng thái cảnh báo trực tiếp trên thiết bị.

Đèn LED cảnh báo: Báo hiệu khi nhiệt độ hoặc độ ẩm vượt quá ngưỡng an toàn do người dùng cài đặt.

Điện trở 220Ω: Giới hạn dòng điện đi qua LED, giúp bảo vệ thiết bị khỏi hư hỏng.



Hình 3.1: Sơ đồ nguyên lý kết nối

3.2. Mô tả chi tiết kết nối phần cứng

3.2.1. Kết nối ESP32 với cảm biến DHT22

Thành phần	Chân kết nối trên ESP32
VCC (DHT22)	3.3V
GND (DHT22)	GND
DATA (DHT22)	GPIO27

3.2.2. Kết nối ESP32 với màn hình OLED SSD1306

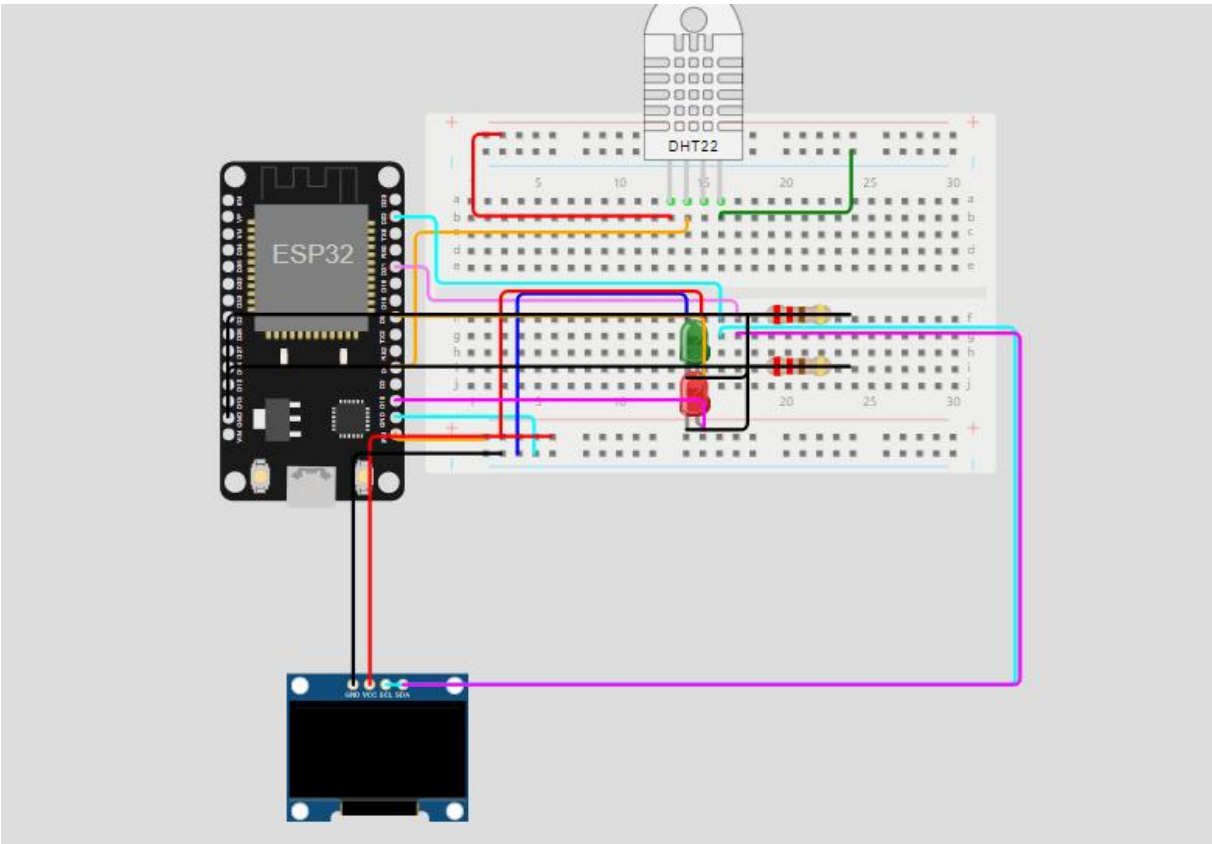
Màn hình OLED SSD1306 sử dụng giao thức truyền thông I2C:

Thành phần	Chân kết nối trên ESP32
VCC (OLED)	3.3V
GND (OLED)	GND
SCL (OLED)	GPIO22
SDA (OLED)	GPIO21

3.2.3. Kết nối ESP32 với đèn LED cảnh báo

Thành phần	Kết nối
Anode (LED đỏ)	GPIO15
Anode (LED xanh)	GPIO5
Cathode (LED)	Qua điện trở 220Ω đến GND

3.2.4. Sơ đồ mô phỏng mạch trên Wokwi



Hình 3.2.4: Sơ đồ mô phỏng mạch trên Wokwi

3.3. Lưu đồ hoạt động của hệ thống

Quy trình hoạt động của hệ thống được mô tả như sau:

1. **Khởi động hệ thống**: ESP32 khởi động và thiết lập kết nối với các thiết bị ngoại vi (cảm biến, màn hình OLED, Wi-Fi).
2. **Đọc dữ liệu từ cảm biến DHT22**: ESP32 thu thập nhiệt độ và độ ẩm.
3. **Hiển thị dữ liệu lên màn hình OLED**: Cập nhật liên tục thông tin trên màn hình OLED.
4. **So sánh với ngưỡng cảnh báo**:

   Nếu vượt ngưỡng: ESP32 kích hoạt LED cảnh báo tương ứng.

   Nếu trong giới hạn: LED được tắt.

5. **Gửi dữ liệu lên ứng dụng Blynk**: Thông tin nhiệt độ và độ ẩm được truyền lên điện thoại qua mạng Wi-Fi.
6. **Chờ khoảng thời gian cố định**: Sau một khoảng delay (ví dụ: 2 giây), hệ thống quay lại bước 2.

Chương 4: Kết luận và hướng phát triển

4.1. Kết luận

Hệ thống đã hoạt động đúng theo yêu cầu:

Đọc và hiển thị dữ liệu từ cảm biến.

Gửi dữ liệu lên ứng dụng Blynk để **giám sát và điều chỉnh từ xa**.

Kích hoạt LED cảnh báo khi vượt ngưỡng.

4.2. Hạn chế

Dữ liệu áp suất hiện tại **chưa được lấy từ cảm biến thực**, mà chỉ mô phỏng.

Chưa triển khai ngoài thực tế, mới dừng ở mức mô hình phòng thí nghiệm.

4.3. Hướng phát triển

Tích hợp cảm biến BME280 để đo áp suất thực.

Gửi cảnh báo qua Telegram hoặc Email khi có sự cố.

Điều khiển thiết bị thực như quạt hoặc máy bơm khi vượt ngưỡng.

Tài liệu tham khảo

[1] Espressif Systems. “ESP32 Series Datasheet.”
https://www.espressif.com/en/products/socs/esp32

[2] Adafruit. “DHT22 Temperature-Humidity Sensor Datasheet.”
https://learn.adafruit.com/dht

[3] Blynk IoT Platform – Điều khiển và giám sát thiết bị từ xa.
https://blynk.io/

[4] Arduino – Phần mềm lập trình mã nguồn mở cho vi điều khiển.
https://www.arduino.cc/

[5] Thư viện Adafruit Unified Sensor – Hướng dẫn cài đặt và sử dụng.
https://github.com/adafruit/Adafruit_Sensor

[6] Thư viện DHT sensor library for ESPx by Rob Tillaart.
https://github.com/RobTillaart/DHTNew

[7] Thư viện Adafruit SSD1306 OLED.
https://github.com/adafruit/Adafruit_SSD1306

[8] Thư viện Adafruit GFX – Hỗ trợ hiển thị đồ họa OLED.
https://github.com/adafruit/Adafruit-GFX-Library

[9] Hướng dẫn sử dụng Slider và Virtual Pin trong Blynk.
https://docs.blynk.io/en/getting-started/using-widgets/slider

[10] Diễn đàn Arduino Việt Nam – Cộng đồng chia sẻ kiến thức.
https://arduino.vn/

[11] Trang mua linh kiện và sơ đồ chân ESP32 DevKit V1.
https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

[12] Hướng dẫn tạo dự án IoT với ESP32 và Blynk.
https://randomnerdtutorials.com/esp32-blynk-app/

Phụ lục

Code chương trình

Main.cpp

```
// Định nghĩa macro cần thiết cho Blynk Template
#define BLYNK_TEMPLATE_ID "TMPL6cIkzcAaR"
#define BLYNK_TEMPLATE_NAME "ESP32 PROJECT"
#define BLYNK_AUTH_TOKEN "tg3_TmaLL_oq1mm-HNAeal0b-DhX4ymo"

#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>

// ------------------------
// CẤU HÌNH HIỆN THỊ OLED
// ------------------------
#define SCREEN_WIDTH 128   // Chiều rộng OLED (pixel)
#define SCREEN_HEIGHT 64   // Chiều cao OLED (pixel)
Adafruit_SSD1306 oled(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// ------------------------
// CẤU HÌNH WIFI
// ------------------------
const char* ssid     = "Wokwi-GUEST";  // WiFi mặc định trong Wokwi
const char* password = "";             // Trong Wokwi, mật khẩu là rỗng

// ------------------------
// CẤU HÌNH CẢM BIẾN DHT22
// ------------------------
#define DHT_PIN 4         // DHT22 nối tại GPIO4
#define DHT_TYPE DHT22
DHT dht(DHT_PIN, DHT_TYPE);

// ------------------------
// CẤU HÌNH ĐÈN LED
// ------------------------
```

```
#define GREEN_LED_PIN 5    // Dùng GPIO5 để điều khiển LED xanh (mô phỏng máy bơm)
#define RED_LED_PIN 15     // Dùng GPIO15 để điều khiển LED đỏ (cảnh báo nhiệt độ cao)

// ------------------------
// NGƯỠNG ĐIỀU KHIỂN MẶC ĐỊNH
// ------------------------
float humidityThreshold = 40.0;  // Ngưỡng độ ẩm để bật LED xanh (%), có thể thay đổi từ Blynk
float tempThreshold = 35.0;      // Ngưỡng nhiệt độ để bật LED đỏ (°C), có thể thay đổi từ Blynk

// ------------------------
// BIẾN TOÀN CỤC
// ------------------------
float temperature;  // Nhiệt độ (°C)
float humidity;     // Độ ẩm (%)
float pressure;     // Áp suất mô phỏng (hPa)
int counter = 0;    // Biến đếm số lần đọc
const unsigned long sensorInterval = 3000; // Cứ 3 giây cập nhật cảm biến
unsigned long previousMillis = 0;
int greenLedState = LOW;    // Trạng thái hiện tại của LED xanh
int redLedState = LOW;      // Trạng thái hiện tại của LED đỏ
String statusMessage = "";  // Thông báo trạng thái hiện tại

// Điều chỉnh ngưỡng nhiệt độ từ ứng dụng Blynk (Virtual Pin V8)
BLYNK_WRITE(V8) {
  tempThreshold = param.asFloat();
  Serial.print("Ngưỡng nhiệt độ mới: ");
  Serial.print(tempThreshold);
  Serial.println(" °C");
}

// Điều chỉnh ngưỡng độ ẩm từ ứng dụng Blynk (Virtual Pin V9)
BLYNK_WRITE(V9) {
  humidityThreshold = param.asFloat();
  Serial.print("Ngưỡng độ ẩm mới: ");
  Serial.print(humidityThreshold);
  Serial.println(" %");
}

// Khai báo prototype của các hàm
void updateSensors();
void updateOLEDDisplay();
```

```
void setup() {
  Serial.begin(115200);
  Serial.println("ESP32 Weather Station với ngưỡng tùy chỉnh");

  // Thiết lập GPIO4 (DHT_PIN) có pull-up nội
  pinMode(DHT_PIN, INPUT_PULLUP);

  // Khởi tạo cảm biến DHT22
  dht.begin();

  // Thêm thời gian delay để DHT22 khởi động đầy đủ
  delay(2000);

  // Khởi tạo màn hình OLED
  if (!oled.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("OLED init failed!"));
    while (1);
  }
  oled.clearDisplay();
  oled.setTextSize(1);
  oled.setTextColor(WHITE);
  oled.setCursor(0, 0);
  oled.println("ESP32 Weather Station");
  oled.println("Initializing...");
  oled.display();
  delay(2000);

  // Cấu hình các chân điều khiển LED
  pinMode(GREEN_LED_PIN, OUTPUT);
  digitalWrite(GREEN_LED_PIN, greenLedState);  // Ban đầu, LED xanh tắt
  pinMode(RED_LED_PIN, OUTPUT);
  digitalWrite(RED_LED_PIN, redLedState);      // Ban đầu, LED đỏ tắt

  // Kết nối WiFi và Blynk
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);

  // Đặt ngưỡng ban đầu trên Blynk
  Blynk.virtualWrite(V8, tempThreshold);   // Ngưỡng nhiệt độ
  Blynk.virtualWrite(V9, humidityThreshold); // Ngưỡng độ ẩm

  // Khởi tạo các giá trị hiển thị
  Blynk.virtualWrite(V0, 0); // Trạng thái LED xanh: OFF
```

```
  Blynk.virtualWrite(V1, 0); // Trạng thái LED đỏ: OFF
  Blynk.virtualWrite(V10, "Hệ thống đang khởi động..."); // Thông báo trạng thái

  Serial.println("Đã kết nối với Blynk. Hệ thống sẵn sàng.");
}

void loop() {
  Blynk.run();
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= sensorInterval) {
    previousMillis = currentMillis;
    updateSensors();
  }
}

void updateSensors() {
  // Chờ 50ms để DHT22 sẵn sàng
  delay(50);

  // Đọc dữ liệu từ DHT22
  temperature = dht.readTemperature();
  humidity = dht.readHumidity();

  // Sinh giá trị áp suất ngẫu nhiên từ 900.0 đến 1100.0 hPa
  pressure = random(9000, 11000) / 10.0;

  if (isnan(temperature) || isnan(humidity)) {
    Serial.println("Failed to read from DHT sensor!");
    oled.clearDisplay();
    oled.setCursor(0, 0);
    oled.println("Sensor Error!");
    oled.println("Check wiring/connection");
    oled.display();

    // Gửi thông báo lỗi đến Blynk
    Blynk.virtualWrite(V10, "LỖI: Không đọc được cảm biến DHT!");
    return;
  }

  // Hiển thị thông số ra Serial Monitor
  Serial.print("\n--- Reading #");
  Serial.print(counter);
```

```
  Serial.println(" ---");
  Serial.print("Temperature: ");
  Serial.print(temperature, 1);
  Serial.println(" °C");
  Serial.print("Humidity: ");
  Serial.print(humidity, 1);
  Serial.println(" %");
  Serial.print("Pressure: ");
  Serial.print(pressure, 1);
  Serial.println(" hPa");

  // Cập nhật trạng thái thông báo
  statusMessage = "";

  // Kiểm tra ngưỡng nhiệt độ và điều khiển LED đỏ
  if (temperature > tempThreshold) {
    redLedState = HIGH;
    statusMessage += "CẢNH BÁO: Nhiệt độ cao (" + String(temperature, 1) + "°C) vượt ngưỡng " +
String(tempThreshold, 1) + "°C! ";
  } else {
    redLedState = LOW;
  }
  digitalWrite(RED_LED_PIN, redLedState);

  // Kiểm tra ngưỡng độ ẩm và điều khiển LED xanh
  if (humidity < humidityThreshold) {
    greenLedState = HIGH;
    if (statusMessage != "") statusMessage += "\n";
    statusMessage += "Độ ẩm thấp (" + String(humidity, 1) + "%) dưới ngưỡng " + String(humidityThreshold, 1)
+ "%, máy bơm đang hoạt động.";
  } else {
    greenLedState = LOW;
  }
  digitalWrite(GREEN_LED_PIN, greenLedState);

  // Nếu không có cảnh báo, thêm thông báo bình thường
  if (statusMessage == "") {
    statusMessage = "Tất cả thông số môi trường đều bình thường.";
  }

  Serial.print("Green LED: ");
  Serial.println(greenLedState ? "ON" : "OFF");
```

```
  Serial.print("Red LED: ");
  Serial.println(redLedState ? "ON" : "OFF");
  Serial.print("Status: ");
  Serial.println(statusMessage);

  // Cập nhật hiển thị OLED
  updateOLEDDisplay();

  // Gửi dữ liệu cảm biến đến Blynk
  Blynk.virtualWrite(V2, temperature);
  Blynk.virtualWrite(V3, humidity);
  Blynk.virtualWrite(V4, pressure);
  Blynk.virtualWrite(V5, WiFi.RSSI());

  // Gửi trạng thái của LED lên Blynk
  Blynk.virtualWrite(V0, greenLedState);  // Trạng thái LED xanh trên V0
  Blynk.virtualWrite(V1, redLedState);    // Trạng thái LED đỏ trên V1

  // Gửi thông báo trạng thái lên Blynk
  Blynk.virtualWrite(V10, statusMessage);

  counter++;
}

void updateOLEDDisplay() {
  oled.clearDisplay();

  oled.setTextSize(1);
  oled.setCursor(0, 0);
  oled.println("ESP32 Weather Station");
  oled.drawLine(0, 10, 128, 10, WHITE);

  // Nhiệt độ
  oled.setCursor(0, 16);
  oled.print("Temp: ");
  oled.setCursor(85, 16);
  oled.print(temperature, 1);
  oled.println(" C");

  // Độ ẩm
  oled.setCursor(0, 30);
  oled.print("Hum: ");
```

```
  oled.setCursor(85, 30);
  oled.print(humidity, 1);
  oled.println(" %");

  // Áp suất
  oled.setCursor(0, 44);
  oled.print("Press: ");
  oled.setCursor(85, 44);
  oled.print(pressure, 1);
  oled.println(" hPa");

  // Trạng thái các LED
  oled.setCursor(0, 56);
  oled.print("G:");
  oled.print(greenLedState ? "ON" : "OFF");
  oled.print(" R:");
  oled.print(redLedState ? "ON" : "OFF");

  // Vẽ khung bao quanh OLED
  oled.drawRect(0, 0, 128, 64, WHITE);
  oled.display();
}
```