**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH**

**KHOA KỸ THUẬT ĐIỆN TỬ II**



**ĐỒ ÁN CUỐI KỲ MẠNG CẢM BIẾN**

**GIÁM SÁT NHIỆT ĐỘ, ĐỘ ẨM**

**NIÊN KHÓA: 2020 – 2025**

**Giáo viên hướng dẫn : Hồ Nhựt Minh**

**Sinh viên thực hiện:**

|  |  |  |
| --- | --- | --- |
| Họ và Tên | Mã số sinh viên | Lớp |
| Trần Hào Phóng | N20DCDT016 | D20CQDT01-N |
| Võ Minh Thông | N20DCDT028 | D20CQDT01-N |

*TP Hồ Chí Minh , Tháng 10, năm 2023*

MỤC LỤC

[PHẦN MỞ ĐẦU 6](#_Toc149119574)

[**1. Lý do chọn đề tài** 6](#_Toc149119575)

[**2. Mục tiêu và nhiệm vụ nghiên cứu** 6](#_Toc149119576)

[**4. Phương pháp nghiên cứu** 7](#_Toc149119577)

[**5. Cấu trúc của luận văn** 8](#_Toc149119578)

[PHẦN NỘI DUNG 10](#_Toc149119579)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT PHẦN MỀM 10](#_Toc149119580)

[**1.1 Tổng quan về Arduino IDE** 10](#_Toc149119581)

[**1.1.1 Giới thiệu về Arduino IDE** 10](#_Toc149119582)

[**1.1.2 Giao diện người dùng (UI) và ngôn ngữ lập trình** 10](#_Toc149119583)

[**1.1.3 Bo mạch hỗ trợ và cách thức tải mã lên bo mạch** 13](#_Toc149119584)

[**1.2 Tổng quan về LabVIEW** 17](#_Toc149119585)

[**1.2.1. Giới thiệu về LabVIEW** 17](#_Toc149119586)

[**1.2.2 Ngôn ngữ lập trình đồ họa và Virtual Instrument (VI)** 17](#_Toc149119587)

[**1.2.3 Ứng dụng đa lĩnh vực và khả năng phát triển ứng dụng thời gian thực** 18](#_Toc149119588)

[**1.2.4 Khả năng hỗ trợ đa nền tảng và tương thích nhiều phần cứng** 19](#_Toc149119589)

[**1.3 Tổng quan về nền tảng Thingspeak** 21](#_Toc149119590)

[**1.3.1 Giới thiệu Thingspeak** 21](#_Toc149119591)

[**1.3.2 Khả năng lưu trữ, quản lí, hiển thị và phân tích dữ liệu** 21](#_Toc149119592)

[**1.3.3 Gửi, cập nhật dữ liệu và các khả năng khác** 22](#_Toc149119593)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT PHẦN CỨNG 24](#_Toc149119594)

[**2.1.ESP8266 WIFI NodeMCU** 24](#_Toc149119595)

[**2.1.1.Thông số kỹ thuật** 24](#_Toc149119596)

[**2.1.2.Sơ đồ chân và chức năng** 25](#_Toc149119597)

[**2.1.3.Thư viện và tài liệu hỗ trợ** 27](#_Toc149119598)

[**2.2.Cảm biến nhiệt độ - độ ẩm (DHT11)** 28](#_Toc149119599)

[**2.2.1.Thông số kỹ thuật** 28](#_Toc149119600)

[**2.2.2.Sơ đồ chân** 29](#_Toc149119601)

[**2.2.3.Nguyên lí hoạt động** 29](#_Toc149119602)

[**2.3. LED** 30](#_Toc149119603)

[**2.3.1. Thông số kĩ thuật:** 30](#_Toc149119604)

[**2.3.2. Nguyên lí hoạt động** 31](#_Toc149119605)

[CHƯƠNG 3: TIẾN HÀNH ĐỒ ÁN 32](#_Toc149119606)

[**3.1. Xây dựng kênh dữ liệu trên Thingspeak** 32](#_Toc149119607)

[**3.2.Lập trình phần cứng trên Arduino IDE** 35](#_Toc149119608)

[**3.2.1.Hàm set up() và loop()** 36](#_Toc149119609)

[**3.2.2.Các hàm con** 36](#_Toc149119610)

[**3.3. Xử lí dữ liệu trên LabVIEW** 42](#_Toc149119611)

[**3.3.1.Xây dưng các subVI hỗ trợ** 42](#_Toc149119612)

[**3.3.2.Thiết kế giao diện LabVIEW** 44](#_Toc149119613)

[**3.4.Chạy thực tế đồ án** 47](#_Toc149119614)

[**3.4.1. Gửi dữ liệu lên Thingspeak và gửi email cảnh báo:** 47](#_Toc149119615)

[**3.4.2.Web cá nhân để theo dõi và điều khiển phần cứng từ xa** 48](#_Toc149119616)

[PHẦN KẾT LUẬN 51](#_Toc149119617)

**MỤC LỤC HÌNH ẢNH**

**Chương 1**

[Hình 1. 1: Logo Biểu tượng của Arduino IDE 10](#_Toc149119618)

[Hình 1. 2: Menu Bar của Arduino IDE 11](#_Toc149119619)

[Hình 1. 3: Code Editor của Arduino IDE 11](#_Toc149119620)

[Hình 1. 4: Message Area của Arduino IDE 11](#_Toc149119621)

[Hình 1. 5: Toolbar của Arduino IDE 12](#_Toc149119622)

[Hình 1. 6: Hàm setup() của Arduino IDE 12](#_Toc149119623)

[Hình 1. 7: Hàm loop() của Arduino IDE 12](#_Toc149119624)

[Hình 1. 8: Thanh bên trái của Arduino IDE 13](#_Toc149119625)

[Hình 1. 9: Bo mạch Arduino Uno 14](#_Toc149119626)

[Hình 1. 10: Bo mạch Arduino Mega 14](#_Toc149119627)

[Hình 1. 11: Bo mạch Arduino Nano 15](#_Toc149119628)

[Hình 1. 12: Bo mạch Arduino Due 15](#_Toc149119629)

[Hình 1. 13: Bo mạch Arduino Leonardo 16](#_Toc149119630)

**Chương 2**

[Hình 2. 1: Bo mạch ESP8266 24](#_Toc149119631)

[Hình 2. 2: Sơ đồ chân của ESP8266 26](#_Toc149119632)

[Hình 2. 3: Module cảm biến DHT11 28](#_Toc149119633)

[Hình 2. 4: Sơ đồ chân của cảm biến và module cảm biến DHT11 29](#_Toc149119634)

[Hình 2. 5: Một số LED với nhiều màu sắc 30](#_Toc149119635)

[Hình 2. 6: Cấu tạo LED 31](#_Toc149119636)

**Chương 3**

[Hình 3. 1: Đăng ký tài khoản Thingspeak 32](#_Toc149119637)

[Hình 3. 2: Đăng nhập vào tài khoản Thingspeak 32](#_Toc149119638)

[Hình 3. 3: Truy cập trang chính của Thingspeak 33](#_Toc149119639)

[Hình 3. 4: Tạo kênh mới trên Thingspeak 33](#_Toc149119640)

[Hình 3. 5: Điền các thông tin cơ bản của channel 34](#_Toc149119641)

[Hình 3. 6: Ảnh chụp Arduino IDE– Thư viện sử dụng 35](#_Toc149119642)

[Hình 3. 7: Ảnh chụp Arduino IDE– Các biến và hằng số sử dụng 35](#_Toc149119643)

[Hình 3. 8: Ảnh chụp Arduino IDE– Hàm setup() 36](#_Toc149119644)

[Hình 3. 9: Ảnh chụp Arduino IDE– Hàm loop() 36](#_Toc149119645)

[Hình 3. 10: Ảnh chụp Arduino IDE– Hàm con CollectDataSensor() 37](#_Toc149119646)

[Hình 3. 11: Ảnh chụp Arduino IDE– Hàm con DataCenter() 37](#_Toc149119647)

[Hình 3. 12: Ảnh chụp Arduino IDE– Hàm con ControlCenter() 37](#_Toc149119648)

[Hình 3. 13: Ảnh chụp Arduino IDE– Hàm con UploadWeb() 38](#_Toc149119649)

[Hình 3. 14: Giao diện Split.vi (Front Panel) 43](#_Toc149119650)

[Hình 3. 15: Giao diện Split.vi (Block Diagram) 43](#_Toc149119651)

[Hình 3. 16: Giao diện SendEmail.vi (Front Panel) 44](#_Toc149119652)

[Hình 3. 17: Giao diện SendEmail.vi (Block Diagram) 44](#_Toc149119653)

[Hình 3. 18: Giao diện thiết kế(Front Panel) 45](#_Toc149119654)

[Hình 3. 19: Giao diện thiết kế - Đọc dữ liệu từ Serial Port thông qua giao thức VISA 45](#_Toc149119655)

[Hình 3. 20: Giao diện thiết kế - Hiện biểu đồ và thiết kế gửi email khi cần 46](#_Toc149119656)

[Hình 3. 21: Giao diện thiết kế - Hiện biểu đồ và thiết kế gửi email khi cần 46](#_Toc149119657)

[Hình 3. 22: Giao diện thiết kế - Đưa dữ liệu lên ThingSpeak 46](#_Toc149119658)

[Hình 3. 23: Giao diện thiết kế (Block Diagram) 47](#_Toc149119659)

[Hình 3. 24: LabVIEW gửi email khi đạt điều kiện cho trước 47](#_Toc149119660)

[Hình 3. 25: Email chỉ gửi đi 1 lần cho mỗi lần nhiệt độ vượt quá giới hạn 48](#_Toc149119661)

[Hình 3. 26: Giao diện thực tế khi chạy LabVIEW 48](#_Toc149119662)

[Hình 3. 27: Giao diện trang web cơ bản 49](#_Toc149119663)

[Hình 3. 28: Giao diện chờ khi thực hiện nhấn nút trên giao diện 50](#_Toc149119664)

# PHẦN MỞ ĐẦU

Trong một thế giới ngày càng phụ thuộc vào sự công nghệ hóa và tự động hóa, việc giám sát nhiệt độ và độ ẩm trở thành một phần không thể thiếu của cuộc sống và sản xuất hàng ngày. Hai yếu tố quan trọng này không chỉ ảnh hưởng đến sự an toàn và chất lượng của sản phẩm mà còn đóng vai trò quyết định trong nhiều khía cạnh của cuộc sống con người và môi trường xung quanh.

Nhiệt độ và độ ẩm có vai trò to lớn trong các lĩnh vực đa dạng, từ nông nghiệp đến công nghiệp thực phẩm, y tế, và quản lý môi trường. Sự thay đổi nhỏ trong nhiệt độ hoặc độ ẩm có thể có tác động lớn đến sự ổn định và an toàn của quy trình sản xuất, sự tươi ngon và an toàn của thực phẩm, sức kháng của hệ thống y tế, và sự tồn vẹn của môi trường tự nhiên.

## **1. Lý do chọn đề tài**

Nhiệt độ và độ ẩm có ảnh hưởng đến sự an toàn và chất lượng của sản phẩm trong ngành thực phẩm và y tế. Đồng thời, việc theo dõi chúng cũng giúp tiết kiệm năng lượng và tài nguyên trong công nghiệp. Sự kết hợp của công nghệ thông tin và trí tuệ nhân tạo cũng tạo ra những giải pháp thông minh và tự động hóa trong giám sát. Điều này không chỉ giúp cải thiện hiệu suất hoạt động mà còn mở ra cơ hội cho nghiên cứu và phát triển công nghệ mới. Bên cạnh đó, đề tài này còn đặt ra nhiều thách thức kỹ thuật cũng như cơ hội sáng tạo để đáp ứng các yêu cầu ngày càng phức tạp trong việc quản lý nhiệt độ và độ ẩm.

## **2. Mục tiêu và nhiệm vụ nghiên cứu**

* Mục tiêu nghiên cứu:
  + - * + Nâng cao hiệu suất giám sát: Phát triển các cảm biến và thiết bị giám sát tiên tiến để cung cấp độ chính xác cao, độ tin cậy và thời gian thực cho việc đo lường nhiệt độ và độ ẩm.
        + Phân tích dữ liệu: Nghiên cứu và phát triển các phương pháp phân tích dữ liệu để hiểu sâu hơn về biến đổi nhiệt độ và độ ẩm trong các môi trường cụ thể, bao gồm cả việc theo dõi xu hướng, biến đổi và nguy cơ liên quan
        + Ứng dụng thực tiễn: Áp dụng kết quả nghiên cứu vào các lĩnh vực cụ thể như nông nghiệp, công nghiệp, y tế, và môi trường để cải thiện quản lý và hiệu suất trong các ngành này
        + Tích hợp công nghệ mới: Nghiên cứu và áp dụng các công nghệ mới như IoT (Internet of Things), trí tuệ nhân tạo (AI), và hệ thống thông tin địa lý (GIS) để nâng cao khả năng giám sát và quản lý nhiệt độ và độ ẩm
* Nhiệm vụ nghiên cứu:
  + - * + Nghiên cứu về cảm biến và thiết bị: Tiến hành nghiên cứu để phát triển hoặc cải tiến các cảm biến và thiết bị đo nhiệt độ và độ ẩm với độ chính xác cao và khả năng kết nối mạng.
        + Phân tích dữ liệu: Thu thập và phân tích dữ liệu từ các cảm biến để hiểu sâu hơn về biến đổi trong nhiệt độ và độ ẩm, bao gồm cả việc xác định các nguy cơ tiềm ẩn hoặc xu hướng quan trọng.
        + Phát triển phần mềm và hệ thống: Xây dựng các phần mềm và hệ thống giám sát để quản lý dữ liệu, cung cấp thông báo cảnh báo khi có sự biến đổi đáng kể, và tạo ra các bản đồ và báo cáo liên quan.
        + Kiểm tra ứng dụng thực tiễn: Áp dụng hệ thống giám sát vào các môi trường thực tế và đánh giá hiệu suất của nó trong việc quản lý nhiệt độ và độ ẩm.
        + Nghiên cứu và phát triển ứng dụng đặc thù: Tùy chỉnh và phát triển ứng dụng cụ thể cho các ngành như nông nghiệp, quản lý tài nguyên nước, y tế, và sản xuất công nghiệp.
        + Đối tượng nghiên cứu: Nghiên cứu có thể tập trung vào các mô hình quản lý nhiệt độ và độ ẩm trong môi trường nông nghiệp, kho lạnh, y tế, hệ thống cung cấp nước, hoặc các ứng dụng khác dựa trên yêu cầu cụ thể..

## **4. Phương pháp nghiên cứu**

* + - * Lựa chọn cảm biến và thiết bị: Chọn các cảm biến nhiệt độ và độ ẩm phù hợp cho mục tiêu nghiên cứu của bạn. Điều này có thể bao gồm cảm biến khác nhau về loại (ví dụ: cảm biến nhiệt độ điện trở, cảm biến độ ẩm điện dung) và tính năng (độ chính xác, kích thước, khả năng kết nối mạng).
      * Thiết kế mạng cảm biến (nếu cần): Nếu nghiên cứu đòi hỏi giám sát nhiệt độ và độ ẩm trên diện rộng hoặc trong nhiều vị trí, bạn có thể cần thiết kế một mạng cảm biến. Điều này liên quan đến việc xác định vị trí cài đặt cảm biến, kết nối mạng, và quản lý dữ liệu.
      * Thu thập dữ liệu: Cài đặt cảm biến và thiết bị tại các vị trí quan trọng và bắt đầu thu thập dữ liệu nhiệt độ và độ ẩm. Đảm bảo rằng quá trình thu thập dữ liệu được tiến hành một cách liên tục hoặc theo lịch trình cụ thể.
      * Phân tích dữ liệu: Xử lý và phân tích dữ liệu thu thập từ các cảm biến. Có thể áp dụng các phương pháp thống kê, machine learning, hoặc các phương pháp khác tùy thuộc vào loại dữ liệu và mục tiêu phân tích.
      * Tích hợp hệ thống thông tin: Tạo ra các hệ thống thông tin địa lý (GIS) hoặc các ứng dụng thông tin để hiển thị dữ liệu nhiệt độ và độ ẩm một cách trực quan và dễ sử dụng.
      * Kiểm tra ứng dụng thực tiễn: Áp dụng hệ thống giám sát nhiệt độ và độ ẩm vào các tình huống thực tế và đánh giá hiệu suất của nó. Điều này bao gồm việc xem xét khả năng dự đoán và đối phó với biến đổi nhiệt độ và độ ẩm.
      * Tổng hợp kết quả và viết báo cáo: Tổng hợp kết quả của nghiên cứu và viết báo cáo nghiên cứu, bao gồm mô tả phương pháp, kết quả, và những phát hiện quan trọng.
      * Đánh giá và đề xuất cải tiến: Cuối cùng, đánh giá hiệu suất của hệ thống và đề xuất các cải tiến hoặc nghiên cứu tiếp theo để nâng cao khả năng giám sát nhiệt độ và độ ẩm trong tương lai.

**5. Cấu trúc của luận văn**

Phần nội dung chính của luận văn gồm có 3 chương:

**Chương 1: Cơ sở lý thuyết phần mềm**

Chương này nói về những ứng dụng, công cụ phần mềm được dung trong xuyên suốt luận văn, đồng thời trình bày tổng quan về những vấn đề nổi bật liên quan đến công cụ, ứng dụng đó.

**Chương 2: Cơ sở lý thuyết phần cứng**

Chương này trình bày tổng quan những gì cần biết về những linh kiện được sử dụng trong luận văn bao gồm thông số kỹ thuật, nguyên lí hoạt động,…

**Chương 3: Tiến hành đồ án**

Chương này tập trung vào trình bày cách thức triển khai đồ án, từ lập trình phần mềm đến kết nối ngoại vi phần cứng.

# PHẦN NỘI DUNG

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT PHẦN MỀM

**1.1 Tổng quan về Arduino IDE**

### **1.1.1 Giới thiệu về Arduino IDE**

Arduino IDE (Integrated Development Environment) là một môi trường phát triển tích hợp cho việc lập trình và phát triển ứng dụng dựa trên nền tảng Arduino. Nó cung cấp một loạt công cụ và tài liệu hữu ích để đơn giản hóa quá trình phát triển dự án điện tử và nhúng sử dụng bo mạch Arduino.



Hình 1. 1: Logo Biểu tượng của Arduino IDE

### **1.1.2 Giao diện người dùng (UI) và ngôn ngữ lập trình**

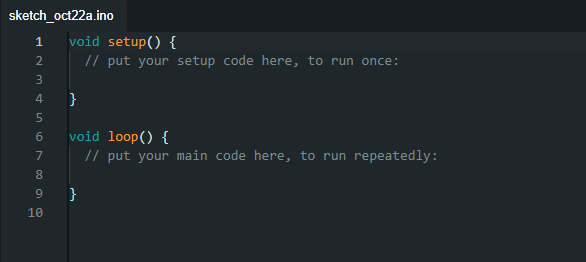
Giao diện người dùng của Arduino IDE được thiết kế đơn giản và trực quan để giúp người dùng lập trình và phát triển ứng dụng Arduino một cách dễ dàng. Dưới đây là mô tả chi tiết về các phần quan trọng của giao diện người dùng:

* + - Khung lệnh (Menu Bar): Giao diện bắt đầu với một khung lệnh ở đầu cửa sổ, bao gồm các menu như “File,” “Edit,” “Sketch,” “Tools,” và “Help.” Trong menu này, bạn có thể thực hiện các tác vụ như tạo mới, mở, lưu, và tải lên chương trình.



Hình 1. 2: Menu Bar của Arduino IDE

* + - Cửa sổ mã nguồn (Code Editor): Cửa sổ mã nguồn là nơi bạn viết và chỉnh sửa mã Arduino của mình. Đây là nơi bạn sẽ viết các hàm, biến, và lệnh để điều khiển bo mạch Arduino.



Hình 1. 3: Code Editor của Arduino IDE

* + - Cửa sổ cảnh báo (Message Area): Cửa sổ cảnh báo hiển thị thông báo liên quan đến quá trình biên dịch và tải lên chương trình. Nó bao gồm các thông báo lỗi và cảnh báo trong quá trình phát triển.



Hình 1. 4: Message Area của Arduino IDE

* + - Thanh công cụ (Toolbar): Thanh công cụ chứa các biểu tượng nhanh cho các tác vụ phổ biến như biên dịch (Compile), tải lên (Upload), và mở Serial Monitor để theo dõi dữ liệu từ bo mạch Arduino.



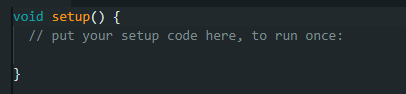
Hình 1. 5: Toolbar của Arduino IDE

* + - Thanh trạng thái (Status Bar): Thanh trạng thái hiển thị thông tin như loại bo mạch Arduino được chọn, cổng kết nối, và trạng thái của quá trình tải lên

Ngôn ngữ lập trình được sử dụng trong Arduino IDE dựa trên C/C++, nhưng có một số sự điều chỉnh và thư viện đặc biệt được hem vào để làm cho việc lập trình cho bo mạch Arduino dễ dàng hơn.

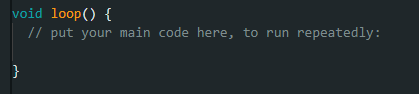
Một số đặc điểm quan trọng của ngôn ngữ lập trình Arduino:

* + - Hàm setup() và loop(): Mỗi chương trình Arduino phải bao gồm ít nhất hai hàm chính: setup() và loop().
      * setup() chạy một lần khi bo mạch khởi động và được sử dụng để cài đặt các cài đặt ban đầu như khai báo biến và cấu hình chân.



Hình 1. 6: Hàm setup() của Arduino IDE

* + - * loop() chạy liên tục sau khi setup() hoàn thành và là nơi chứa mã để thực hiện các nhiệm vụ chính trong ứng dụng.



Hình 1. 7: Hàm loop() của Arduino IDE

* + - Thư viện Arduino: Arduino IDE đi kèm với một loạt thư viện tiêu chuẩn và bên ngoài (được tạo bởi cộng đồng) giúp bạn thực hiện các chức năng phức tạp một cách dễ dàng. Bạn có thể import thư viện bằng cách sử dụng #include và sau đó sử dụng các hàm và lớp được cung cấp bởi thư viện đó trong chương trình của bạn.



Hình 1. 8: Thanh bên trái của Arduino IDE

* + - Biến toàn cục (Global Variables): Các biến toàn cục có thể được khai báo ở đầu chương trình và sử dụng trong cả hàm setup() và loop(). Các biến này có sẵn cho tất cả các hàm và phần trong chương trình.
    - Hàm millis(): Hàm millis() được sử dụng để đo thời gian và thường được sử dụng để tạo các chức năng liên quan đến thời gian trong ứng dụng Arduino. Nó trả về số milliseconds kể từ khi bo mạch bắt đầu chạy.
    - Thư viện Serial: Thư viện Serial cho phép bạn giao tiếp với máy tính hoặc các thiết bị khác thông qua cổng Serial. Điều này thường được sử dụng để gửi và nhận dữ liệu từ và đến bo mạch Arduino để theo dõi và kiểm tra.

### **1.1.3 Bo mạch hỗ trợ và cách thức tải mã lên bo mạch**

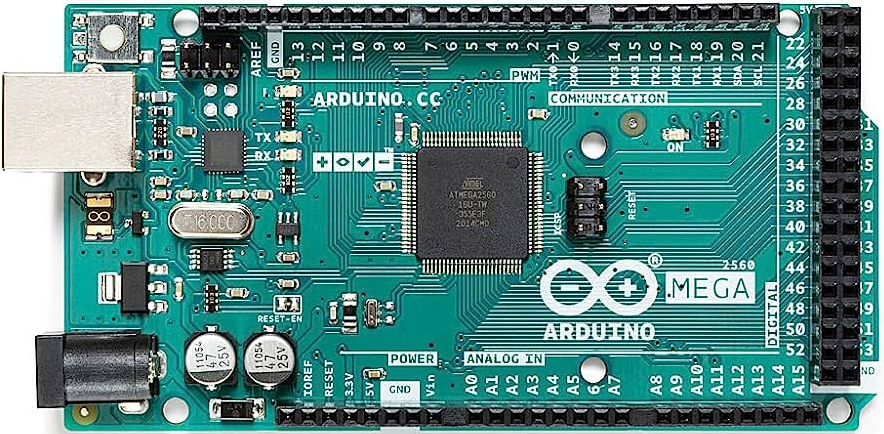
Arduino IDE hỗ trợ nhiều loại bo mạch Arduino khác nhau, bao gồm các phiên bản chính và phiên bản từ bên thứ ba. Điều này giúp đáp ứng các nhu cầu phát triển đa dạng của các dự án.

* + - Arduino Uno: Là bo mạch phổ biến nhất và được sử dụng rộng rãi cho các dự án cơ bản. Arduino Uno dựa trên vi xử lý ATmega328P.



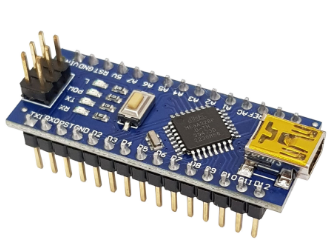
Hình 1. 9: Bo mạch Arduino Uno

* + - Arduino Mega: Được thiết kế cho các dự án phức tạp hơn với nhiều chân IO hơn. Arduino Mega sử dụng vi xử lý ATmega2560.



Hình 1. 10: Bo mạch Arduino Mega

* + - Arduino Nano: Kích thước nhỏ gọn và phù hợp cho các dự án yêu cầu tích hợp trong không gian hạn chế.



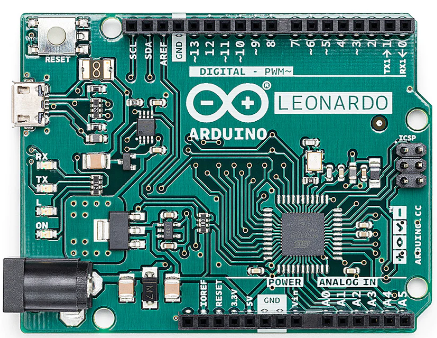
Hình 1. 11: Bo mạch Arduino Nano

* + - Arduino Due: Sử dụng vi xử lý ARM Cortex-M3, Arduino Due thích hợp cho các ứng dụng đòi hỏi hiệu năng cao hơn.



Hình 1. 12: Bo mạch Arduino Due

* + - Arduino Leonardo: Được xây dựng với vi xử lý ATmega32U4 và hỗ trợ tính năng USB HID, cho phép bạn tạo các thiết bị đầu vào như bàn phím và chuột.



Hình 1. 13: Bo mạch Arduino Leonardo

* + - Arduino MKR Series: Dành cho các ứng dụng IoT, Arduino MKR Series bao gồm các bo mạch như MKR1000, MKR1010, và MKR Zero.
    - Bo mạch từ bên thứ ba: Ngoài các bo mạch chính thống, Arduino IDE cũng hỗ trợ nhiều bo mạch từ bên thứ ba (non-official boards) do cộng đồng phát triển, cho phép bạn sáng tạo và sử dụng các bo mạch đa dạng khác.

Quy trình tải mã lập trình của bạn lên bo mạch Arduino sử dụng Arduino IDE:

* + - Kết nối bo mạch Arduino với máy tính: Sử dụng cáp USB, kết nối bo mạch Arduino với máy tính.
    - Chọn loại bo mạch và cổng: Trong Arduino IDE, bạn chọn loại bo mạch mà bạn đang sử dụng từ menu "Tools" và sau đó chọn cổng kết nối USB của bo mạch trong menu "Port." Điều này cho phép IDE xác định đúng bo mạch và cổng để tải lên.
    - Viết mã và kiểm tra lỗi: Sử dụng cửa sổ mã nguồn, viết mã Arduino của bạn. Sau đó, có thể kiểm tra mã bằng cách nhấn nút "Verify" (biểu tượng nút gắn vít) để biên dịch chương trình và kiểm tra lỗi. Nếu có lỗi, chúng sẽ được hiển thị trong cửa sổ cảnh báo.
    - Tải mã lên: Khi mã được tải không có lỗi, nhấn nút "Upload" (biểu tượng mũi tên) để tải mã lên bo mạch Arduino. IDE sẽ biên dịch mã và tải nó lên bo mạch thông qua cổng USB.
    - Kiểm tra kết quả: Sau khi tải lên thành công, kiểm tra kết quả sử dụng Serial Monitor hoặc các phương tiện khác tùy thuộc vào dự án.

## **1.2 Tổng quan về LabVIEW**

### **1.2.1. Giới thiệu về LabVIEW**

LabVIEW là một phần mềm phát triển ứng dụng và hệ thống, được phát triển bởi National Instruments (nay là NI), sử dụng để thiết kế, phát triển và kiểm tra ứng dụng trong các lĩnh vực như tự động hóa, kiểm tra và đo lường, hệ thống nhúng, và nghiên cứu khoa học. LabVIEW là một hệ thống phát triển dựa trên đồ họa và kỹ thuật trực quan, cho phép người sử dụng tạo ra các ứng dụng bằng cách kết hợp các khối chức năng (gọi là "Virtual Instrument" hoặc "VI") bằng cách sử dụng biểu đồ dây nối.

### **1.2.2 Ngôn ngữ lập trình đồ họa và Virtual Instrument (VI)**

Ngôn ngữ lập trình đồ họa

LabVIEW sử dụng một ngôn ngữ lập trình đồ họa duy nhất, được gọi là G Programming Language hoặc G code. Điểm chính của ngôn ngữ này là việc biểu diễn chương trình dưới dạng biểu đồ dây nối (Block Diagram) thay vì mã lệnh văn bản truyền thống. Dưới đây là các thành phần quan trọng của ngôn ngữ này:

* + - Khối chức năng (Function Block): Trong LabVIEW, chương trình được xây dựng bằng cách sử dụng các khối chức năng, mỗi khối đại diện cho một phần của mã lệnh. Ví dụ, một khối chức năng có thể thực hiện tính toán số học, xử lý dữ liệu, hoặc tương tác với thiết bị phần cứng.
    - Dây nối (Wire): Dây nối kết nối các khối chức năng với nhau để truyền dữ liệu và thông tin điều kiện. Mỗi dây nối có màu sắc khác nhau để thể hiện kiểu dữ liệu và giúp dễ dàng theo dõi sự truyền tải dữ liệu trong chương trình.
    - Cấu trúc lặp (Loop) và điều kiện (Conditional): LabVIEW hỗ trợ các cấu trúc lặp (ví dụ: vòng lặp For, While) và cấu trúc điều kiện (ví dụ: câu lệnh If) để kiểm soát luồng chương trình.
    - Biểu đồ dây nối (Block Diagram): Đây là nơi tạo và chỉnh sửa mã lệnh của ứng dụng. Nó hiển thị toàn bộ cấu trúc của chương trình dưới dạng biểu đồ, giúp người dùng dễ dàng theo dõi và hiểu chương trình.

Virtual Instrument (VI):

Virtual Instrument (VI) là đối tượng cơ bản của LabVIEW và đóng vai trò quan trọng trong việc phát triển ứng dụng. Một VI bao gồm hai phần chính:

* + - Front Panel: Front Panel là giao diện người dùng của VI, nơi bạn có thể tạo các điều khiển (Controls) như nút bấm, thanh trượt, ô nhập liệu, và hiển thị (Indicators) như biểu đồ, đồ thị, số liệu, để tương tác với chương trình. Front Panel là phần giao diện mà người dùng cuối thấy và tương tác.
    - Block Diagram: Block Diagram là nơi bạn xây dựng chương trình bằng cách kết hợp các khối chức năng và kết nối chúng bằng dây nối. Block Diagram thể hiện cách mà các điều khiển và chỉ dẫn trên Front Panel ảnh hưởng đến luồng chương trình.
    - Khi tạo một VI, trước tiên xây dựng Front Panel để tạo giao diện người dùng và sau đó thêm các khối chức năng và dây nối trên Block Diagram để định nghĩa logic của ứng dụng. Mối liên kết giữa Front Panel và Block Diagram cho phép tạo các ứng dụng trực quan và dễ dàng hiểu.

### **1.2.3 Ứng dụng đa lĩnh vực và khả năng phát triển ứng dụng thời gian thực**

Ứng dụng đa lĩnh vực (Multi-Domain Applications):

* + - Tự động hóa công nghiệp: LabVIEW cho phép phát triển các hệ thống tự động hóa công nghiệp, kiểm soát dây chuyền sản xuất, quản lý và giám sát các quy trình công nghiệp.
    - Kiểm tra và đo lường: LabVIEW là một công cụ mạnh mẽ cho phát triển các ứng dụng kiểm tra và đo lường, bao gồm kiểm tra sản phẩm, đo lường chính xác và thu thập dữ liệu từ các thiết bị đo lường.
    - Nghiên cứu khoa học: LabVIEW thường được sử dụng trong nghiên cứu khoa học để thu thập dữ liệu, kiểm soát thiết bị thí nghiệm, và phân tích dữ liệu.
    - Giảng dạy: LabVIEW cung cấp một môi trường thích hợp để giảng dạy và học hỏi về lập trình và kiểm tra tự động.
    - Hệ thống nhúng: LabVIEW hỗ trợ phát triển các ứng dụng nhúng trên nhiều nền tảng như Raspberry Pi và FPGA, cho phép bạn tạo các hệ thống nhúng và IoT.
    - Robotics: LabVIEW hỗ trợ phát triển và kiểm soát các ứng dụng robotics thông qua tích hợp với ROS (Robot Operating System).
    - Quản lý năng lượng: LabVIEW có thể được sử dụng để quản lý và tối ưu hóa hệ thống quản lý năng lượng trong các ứng dụng duy trì và kiểm soát tiêu thụ năng lượng.

Phát triển ứng dụng thời gian thực (Real-Time Application Development):

* + - Cấu trúc thời gian thực (Real-Time Framework): LabVIEW cung cấp một cấu trúc thời gian thực mạnh mẽ, cho phép bạn xây dựng các phần của ứng dụng chạy với độ tin cậy và độ chính xác cao.
    - Hỗ trợ cho các hệ điều hành thời gian thực: LabVIEW hỗ trợ nhiều hệ điều hành thời gian thực như NI Linux Real-Time, Phar Lap ETS, và VxWorks.
    - Kiểm soát thiết bị phần cứng thời gian thực: LabVIEW có khả năng kiểm soát và giao tiếp với các thiết bị phần cứng thời gian thực như cảm biến và actuators.
    - Xử lý dữ liệu thời gian thực: LabVIEW cho phép xử lý dữ liệu và đưa ra quyết định trong thời gian thực, giúp đáp ứng các yêu cầu của các ứng dụng đòi hỏi phản ứng nhanh và đáng tin cậy.
    - Kiểm tra và giám sát thời gian thực: LabVIEW cung cấp công cụ để kiểm tra và giám sát hệ thống thời gian thực, giúp đảm bảo rằng chúng hoạt động theo cách dự kiến.
    - Phát triển ứng dụng thời gian thực trong LabVIEW là một lựa chọn hấp dẫn cho các ứng dụng yêu cầu độ tin cậy và đáng tin cậy cao, chẳng hạn như hệ thống kiểm tra và kiểm soát trong môi trường công nghiệp và tự động hóa.

### **1.2.4 Khả năng hỗ trợ đa nền tảng và tương thích nhiều phần cứng**

Hỗ trợ nhiều nền tảng (Cross-Platform Support):

LabVIEW hỗ trợ nhiều hệ điều hành và nền tảng khác nhau, cho phép bạn phát triển và chạy ứng dụng trên các môi trường khác nhau. Các nền tảng chính bao gồm:

* + - Microsoft Windows: LabVIEW có phiên bản chính thức cho Windows, cho phép phát triển và chạy ứng dụng trên các phiên bản Windows khác nhau như Windows 7, 8, và 10.
    - macOS: LabVIEW cũng có phiên bản dành riêng cho macOS, cho phép phát triển ứng dụng trên các máy tính Mac.
    - Linux: LabVIEW hỗ trợ nhiều phiên bản của Linux, từ các phiên bản phổ biến như Ubuntu đến các phiên bản tùy chỉnh khác.
    - Real-Time OS: LabVIEW cung cấp hỗ trợ cho các hệ điều hành thời gian thực như NI Linux Real-Time, Phar Lap ETS, và VxWorks, giúp bạn phát triển các ứng dụng đòi hỏi đáng tin cậy và thời gian thực.
    - FPGA: LabVIEW FPGA cho phép phát triển ứng dụng trên các FPGA (Field-Programmable Gate Array) của National Instruments, làm cho nó phù hợp cho các ứng dụng cần xử lý siêu nhanh và tùy chỉnh phần cứng.

Hỗ trợ cho các thiết bị phần cứng (Hardware Support):

* + - DAQ (Data Acquisition): LabVIEW hỗ trợ các thiết bị DAQ để thu thập dữ liệu từ cảm biến và giao tiếp với các tín hiệu đầu vào và đầu ra. Điều này cho phép bạn thực hiện kiểm tra và đo lường dễ dàng.
    - Instrument Control: LabVIEW có khả năng kiểm soát và giao tiếp với các thiết bị điều khiển và đo lường như multimeter, oscilloscope, và spectrum analyzer.
    - PLC (Programmable Logic Controller): LabVIEW hỗ trợ tích hợp với các PLC thông qua các giao thức như Modbus, EtherNet/IP, và PROFINET, cho phép bạn kiểm soát và giám sát các quy trình công nghiệp.
    - Robotics: LabVIEW cung cấp hỗ trợ cho phát triển ứng dụng robotics và kiểm soát robot qua các giao thức như ROS (Robot Operating System).
    - Embedded Systems: LabVIEW hỗ trợ phát triển ứng dụng nhúng trên nhiều nền tảng như Raspberry Pi và Arduino, cho phép bạn tạo các thiết bị IoT (Internet of Things) và hệ thống nhúng

## **1.3 Tổng quan về nền tảng Thingspeak**

### **1.3.1 Giới thiệu Thingspeak**

ThingSpeak là một nền tảng trực tuyến miễn phí cho IoT (Internet of Things), cho phép người dùng thu thập, lưu trữ và hiển thị dữ liệu từ các thiết bị kết nối internet một cách dễ dàng. ThingSpeak được phát triển bởi MathWorks, công ty chuyên về phần mềm và công cụ cho khoa học và công nghệ, và nó đã trở thành một trong những nền tảng phổ biến cho ứng dụng IoT và dự án cùng với các hệ thống khác

### **1.3.2 Khả năng lưu trữ, quản lí, hiển thị và phân tích dữ liệu**

Lưu trữ và Quản lý Dữ liệu (Data Storage and Management):

* + - Kênh Dữ liệu (Channels): Một kênh dữ liệu là một nơi để lưu trữ dữ liệu từ một thiết bị IoT cụ thể hoặc một ứng dụng. Mỗi kênh dữ liệu có một ID duy nhất và chứa nhiều trường dữ liệu (Fields) để lưu trữ thông tin. Ví dụ, bạn có thể tạo một kênh dữ liệu cho việc theo dõi nhiệt độ và độ ẩm từ một cảm biến.
    - Trường Dữ liệu (Fields): Mỗi kênh dữ liệu có nhiều trường dữ liệu để lưu trữ các thông tin cụ thể. Các trường dữ liệu được đặt tên theo ý bạn, chẳng hạn như "Nhiệt độ," "Độ ẩm," "Áp suất," và có thể chứa các giá trị số, văn bản, hoặc dữ liệu khác.
    - Gửi Dữ liệu Lên Kênh Dữ liệu: ThingSpeak cung cấp các giao thức và API cho phép thiết bị IoT gửi dữ liệu lên kênh dữ liệu. Dữ liệu có thể được gửi qua HTTP, MQTT, hoặc TCP/IP. ThingSpeak cũng hỗ trợ cách gửi dữ liệu thông qua email hoặc SMS.
    - Tự Động Ghi Dữ Liệu: ThingSpeak cho phép bạn cài đặt tần suất tự động ghi dữ liệu từ các thiết bị. Bạn có thể chọn cách tự động ghi dữ liệu trong khoảng thời gian cố định, ví dụ như mỗi giây, mỗi phút hoặc theo thời gian thực.

Hiển Thị và Phân Tích Dữ Liệu (Data Visualization and Analysis):

* + - Biểu Đồ và Đồ Thị: Bạn có thể tạo các biểu đồ và đồ thị trực quan để hiển thị dữ liệu từ kênh dữ liệu. ThingSpeak hỗ trợ nhiều loại biểu đồ như đồ thị đường, đồ thị cột, và biểu đồ điểm.
    - Đặt Giới Hạn và Cảnh Báo (Thresholds and Alerts): Bạn có thể thiết lập giới hạn trên dữ liệu và cài đặt các cảnh báo để nhận thông báo khi dữ liệu vượt quá hoặc thấp hơn ngưỡng cụ thể. Điều này hữu ích để cảnh báo khi có sự cố xảy ra.
    - Phân Tích Dữ Liệu: ThingSpeak cho phép bạn thực hiện phân tích dữ liệu cơ bản để hiểu hơn về xu hướng và biểu đồ dữ liệu.
    - Tích Hợp Với Ứng Dụng Ngoại Vi: ThingSpeak có khả năng tích hợp với các ứng dụng và dịch vụ khác thông qua API, cho phép bạn xuất dữ liệu hoặc sử dụng nó trong các ứng dụng bên ngoài..

### **1.3.3 Gửi, cập nhật dữ liệu và các khả năng khác**

Gửi và Cập Nhật Dữ Liệu (Sending and Updating Data):

* + - HTTP POST Requests: Bạn có thể sử dụng các yêu cầu HTTP POST để gửi dữ liệu lên ThingSpeak. Các yêu cầu này có thể được gửi thông qua giao thức HTTP hoặc HTTPS.
    - MQTT (Message Queuing Telemetry Transport): ThingSpeak hỗ trợ giao thức MQTT, một giao thức phát tín hiệu sự kiện nhẹ cho các ứng dụng IoT. Bạn có thể gửi dữ liệu bằng cách xuất bản (publish) các thông điệp MQTT.
    - TCP/IP: ThingSpeak cũng cho phép bạn gửi dữ liệu thông qua kết nối TCP/IP để đẩy thông tin đến kênh dữ liệu của bạn.
    - Email và SMS: ThingSpeak hỗ trợ gửi dữ liệu thông qua email hoặc SMS. Bạn có thể cài đặt các cảnh báo để thông báo qua email hoặc SMS khi có sự kiện quan trọng xảy ra.
    - Tự Động Cập Nhật Dữ Liệu: Bạn có thể cài đặt kênh dữ liệu để tự động cập nhật dữ liệu từ các thiết bị IoT theo thời gian thực hoặc theo lịch trình cố định.

Triggers và Actions (Cảnh Báo và Hành Động):

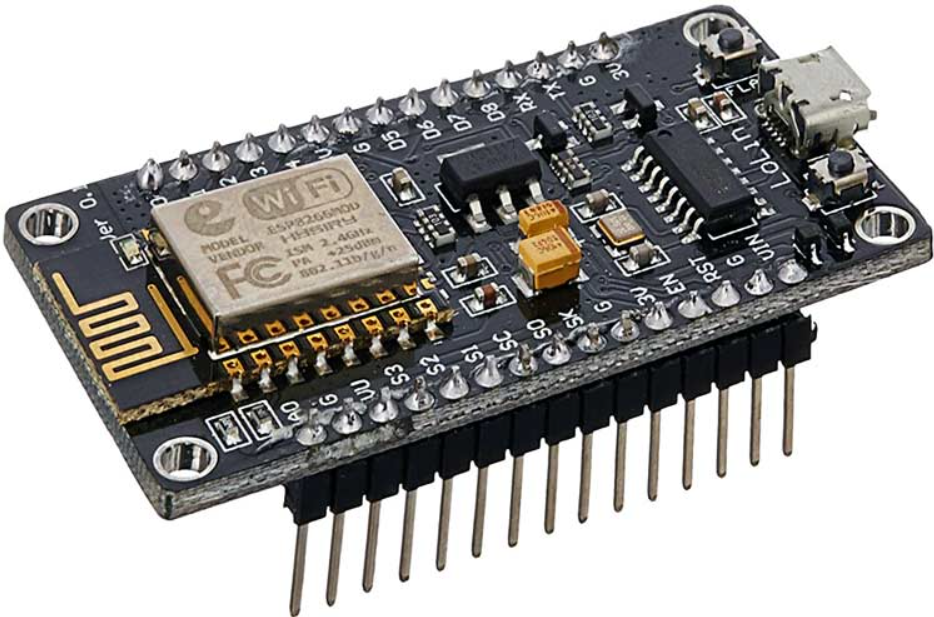
* + - Triggers: Bạn có thể thiết lập các triggers dựa trên dữ liệu trong kênh dữ liệu. Ví dụ, bạn có thể tạo trigger để bật cảnh báo khi nhiệt độ vượt quá một ngưỡng cụ thể.
    - Actions: Khi một trigger được kích hoạt, bạn có thể định nghĩa các hành động để thực hiện. Actions có thể bao gồm gửi email hoặc SMS, cập nhật dữ liệu, hoặc thực hiện các thao tác tùy chỉnh bằng cách sử dụng API.
    - Kết Hợp Trigger và Action: Khi một điều kiện được đáp ứng, trigger sẽ kích hoạt action tương ứng, cho phép bạn thực hiện các tác vụ tự động dựa trên dữ liệu IoT.

Khả năng Mở Rộng (Scalability and Integration):

* + - ThingSpeak là một nền tảng mở rộng và có khả năng tích hợp với nhiều dịch vụ và ứng dụng khác:
    - API: ThingSpeak cung cấp API mạnh mẽ cho phép bạn tạo và quản lý kênh dữ liệu, gửi và lấy dữ liệu từ kênh, và thực hiện nhiều thao tác khác thông qua các yêu cầu HTTP.
    - Tích Hợp Hệ Thống Bên Ngoài: Bạn có thể tích hợp ThingSpeak với các ứng dụng và dịch vụ bên ngoài như MATLAB, Microsoft Azure, Google Sheets, hoặc các dự án và ứng dụng IoT khác.
    - Tùy chỉnh Ứng Dụng Web: Bạn có thể tạo các ứng dụng web tùy chỉnh để hiển thị và quản lý dữ liệu từ kênh dữ liệu của mình, cho phép bạn tạo giao diện người dùng theo ý muốn.

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT PHẦN CỨNG**

## **2.1.ESP8266 WIFI NodeMCU**



Hình 2. 1: Bo mạch ESP8266

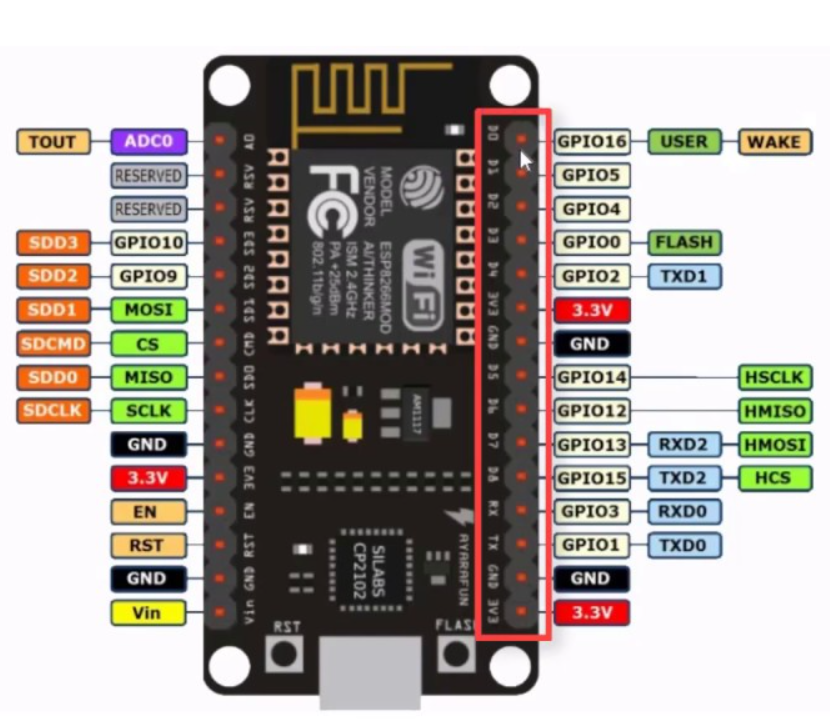
NodeMCU là một nền tảng IoT mã nguồn mở, đã được phát triển bởi một nhóm kỹ sư tại Trung Quốc. Nền tảng này được xây dựng trên cơ sở của vi điều khiển Wi-Fi SoC (System on a Chip) ESP8266, được sản xuất bởi công ty Espressif Systems. NodeMCU cung cấp một bộ SDK cho phép lập trình cho ESP8266 bằng cả ngôn ngữ Lua và C++. Được trang bị với nhiều tính năng hữu ích như Wi-Fi, GPIO, ADC, I2C, SPI, PWM, và nhiều tính năng khác, NodeMCU ESP8266 đã trở thành một công cụ phổ biến và được sử dụng rộng rãi trong các ứng dụng IoT, như kiểm soát thiết bị, thu thập dữ liệu và tương tác với các thiết bị khác.

### **2.1.1.Thông số kỹ thuật**

* + Vi xử lý (CPU): ESP8266EX SoC (System on a Chip) với vi xử lý Tensilica L106 32-bit.
  + Tần số hoạt động: 80 MHz (có thể cấu hình lên đến 160 MHz).
  + Bộ nhớ chương trình (Flash): 4MB (32Mb) Flash SPI.
  + Bộ nhớ RAM: 64KB (512Kb).
  + Chuẩn Wi-Fi: 802.11 b/g/n (Wi-Fi 4).
  + Giao tiếp: GPIO, UART, SPI, I2C, PWM, ADC.
  + Ngôn ngữ lập trình hỗ trợ: Lua, C++, Arduino IDE.
  + Điện áp hoạt động: 3.3V (cần chú ý về điện áp, không thể nạp 5V trực tiếp).
  + Cổng USB: Micro-USB.
  + Kích thước: Thường là 58mm x 31mm.
  + Nguồn cung cấp: Có thể sử dụng pin Lithium-ion hoặc nguồn điện 3.3V bên ngoài.
  + Hỗ trợ UART (Universal Asynchronous Receiver/Transmitter): Có thể sử dụng UART để giao tiếp với máy tính hoặc thiết bị khác.
  + Hỗ trợ GPIO (General-Purpose Input/Output): Có nhiều chân GPIO cho phép bạn kết nối và điều khiển nhiều thiết bị ngoại vi khác nhau.
  + Hỗ trợ Wi-Fi: ESP8266 cho phép kết nối đến mạng Wi-Fi, thực hiện gửi và nhận dữ liệu qua giao thức Wi-Fi

### **2.1.2.Sơ đồ chân và chức năng**

|  |  |  |
| --- | --- | --- |
| Loại chân | Tên chân | Chức năng |
| Power | Micro-USB | NodeMCU có thể được cấp nguồn qua cổng USB |
| 3.3V | Có thể cấp nguồn 3.3V quy định vào chân này để cấp nguồn cho bo mạch |
| GND | Chân nối đất |
| Vin | Nguồn điện bên ngoài |
| Control Pins | EN | Chân và nút reset vi điều khiển |
| RST |
| Analog Pins | A0 | Dùng để đo điện áp analog trong khoảng 0-3,3V |
| GPIO Pins | GPIO1 đến GPIO16 | NodeMCU có 16 chân đầu vào-đầu ra cho mục đích chung trên bo mạch của nó |
| SPI Pins | SPI0 | NodeMCU có sẵn bốn chân để truyền SPI |
| SPI1 |
| CMD |
| CLK |
| UART Pins | RXD0 | NodeMCU có hai kênh UART gồm UART0 (RXD0 & TXD0) và UART1 (RXD1 & TXD1). UART1 được sử dụng để nạp firmware /chương trình |
| TXD0 |
| RXD1 |
| TXD1 |
| I2C Pin |  | NodeMCU có hỗ trợ chức năng I2C. |



Hình 2. 2: Sơ đồ chân của ESP8266

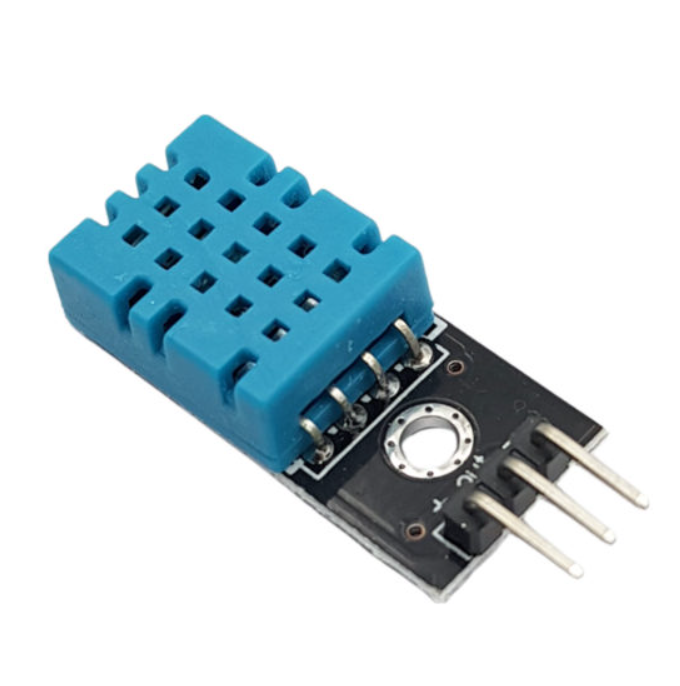
Chi tiết chức năng các chân:

* + 3V3 (3.3V): Nguồn cung cấp 3.3V. Lưu ý rằng NodeMCU hoạt động ở điện áp này.
  + EN (Enable): Pin này được sử dụng để kích hoạt hoặc tắt thiết bị.
  + RST (Reset): Chân này được sử dụng để đặt lại (reset) NodeMCU.
  + GND (Ground): Kết nối đất.
  + D0 (GPIO16): Chân GPIO số 16.
  + D1 (GPIO5): Chân GPIO số 5.
  + D2 (GPIO4): Chân GPIO số 4.
  + D3 (GPIO0): Chân GPIO số 0, cũng là chân mặc định cho chế độ nạp chương trình.
  + D4 (GPIO2): Chân GPIO số 2, cũng là chân TXD1.
  + D5 (GPIO14): Chân GPIO số 14, cũng là chân SCLK cho SPI.
  + D6 (GPIO12): Chân GPIO số 12, cũng là chân MISO cho SPI.
  + D7 (GPIO13): Chân GPIO số 13, cũng là chân MOSI cho SPI.
  + D8 (GPIO15): Chân GPIO số 15, cũng là chân SS cho SPI.
  + RXD (GPIO3): Chân RXD cho UART.
  + TXD (GPIO1): Chân TXD cho UART.
  + SD2 (SDA): Chân SDA cho giao tiếp I2C.
  + SD3 (SCL): Chân SCL cho giao tiếp I2C.
  + A0 (ADC0): Chân đầu vào analog.
  + VIN: Đầu vào nguồn điện ngoại vi (3.3V max).
  + VU (USB Voltage): Điện áp USB từ cổng Micro-USB

### **2.1.3.Thư viện và tài liệu hỗ trợ**

* + Thư viện Arduino cho NodeMCU:
    - Mô tả: Cộng đồng Arduino phát triển rất nhiều thư viện hỗ trợ cho NodeMCU ESP8266. Các thư viện này bao gồm các chức năng tiện ích như kết nối Wi-Fi, giao tiếp với cảm biến, điều khiển các thiết bị ngoại vi, và nhiều chức năng khác.
    - Tài liệu: Các tài liệu hỗ trợ cho thư viện Arduino dành cho NodeMCU có sẵn trên trang web chính thức của Arduino, cùng với ví dụ và hướng dẫn sử dụng.
  + Thư viện Adafruit cho NodeMCU:
    - Mô tả: Adafruit là một trong những nhà sản xuất và cung cấp thư viện phổ biến cho IoT. Họ đã phát triển các thư viện hỗ trợ cho NodeMCU để làm việc với các cảm biến và màn hình hiển thị.
    - Tài liệu: Trang web của Adafruit cung cấp tài liệu hỗ trợ chi tiết, bao gồm ví dụ và hướng dẫn sử dụng cho các thư viện của họ.
  + Thư viện PubSubClient cho MQTT:
    - Mô tả: MQTT (Message Queuing Telemetry Transport) là một giao thức phổ biến cho việc giao tiếp giữa các thiết bị IoT. Thư viện PubSubClient cho NodeMCU giúp bạn thực hiện kết nối MQTT để gửi và nhận dữ liệu từ và đến các máy chủ MQTT.
    - Tài liệu: Thư viện PubSubClient có sẵn trên thư viện Arduino và đi kèm với ví dụ và tài liệu hướng dẫn.
  + Thư viện ESP8266WiFi cho Wi-Fi:
    - Mô tả: Thư viện ESP8266WiFi được tích hợp trong IDE Arduino và cho phép bạn thực hiện kết nối NodeMCU vào mạng Wi-Fi. Điều này rất quan trọng khi bạn cần kiểm soát thiết bị từ xa.
    - Tài liệu: Tài liệu và hướng dẫn về cách sử dụng thư viện ESP8266WiFi có sẵn trên trang web chính thức của Arduino.
  + Tài liệu hướng dẫn chính thức từ Espressif Systems:
    - Mô tả: Espressif Systems, nhà sản xuất của ESP8266, cung cấp tài liệu hướng dẫn chi tiết về việc sử dụng và lập trình NodeMCU ESP8266. Đây là nguồn thông tin chính thống và đáng tin cậy.
    - Tài liệu: Tài liệu hướng dẫn, bản datasheet, và ví dụ ứng dụng có sẵn trên trang web chính thức của Espressif Systems.
  + Diễn đàn và Cộng đồng trực tuyến:
    - Mô tả: Có nhiều diễn đàn và cộng đồng trực tuyến như Reddit, Arduino Forum, và ESP8266 Community trên GitHub, nơi bạn có thể tìm kiếm câu trả lời cho các câu hỏi cụ thể và chia sẻ kinh nghiệm với các thành viên khác.
    - Tài liệu: Diễn đàn và cộng đồng trực tuyến thường chứa rất nhiều thông tin hữu ích và hỗ trợ từ cộng đồng người dùng.

## **2.2.Cảm biến nhiệt độ - độ ẩm (DHT11)**



Hình 2. 3: Module cảm biến DHT11

Cảm biến DHT11 là một loại cảm biến đo nhiệt độ và độ ẩm môi trường. Nó được sử dụng rộng rãi trong các ứng dụng điện tử và IoT (Internet of Things) để giám sát và điều khiển môi trường xung quanh.

### **2.2.1.Thông số kỹ thuật**

Điện áp hoạt động: 5VDC

Chuẩn giao tiếp: TTL, 1 wire.

Khoảng đo độ ẩm: 20%-80%RH sai số ± 5%RH

Khoảng đo nhiệt độ: 0-50°C sai số ± 2°C

Tần số lấy mẫu tối đa 1Hz (1 giây / lần)

Kích thước: 28mm x 12mm x10m

### **2.2.2.Sơ đồ chân**



Hình 2. 4: Sơ đồ chân của cảm biến và module cảm biến DHT11

|  |  |  |
| --- | --- | --- |
| STT | Tên chân | Mô tả |
| 1 | VCC | Nguồn điện 3,5V đến 5,5V |
| 2 | GND | Chân nối đất |
| 3 | Data | Chân xuất dữ liệu Nhiệt độ và Độ ẩm |
| 4 | NC (Not Connect) | KHÔNG DÙNG |

### **2.2.3.Nguyên lí hoạt động**

* + Yêu cầu dữ liệu: Để yêu cầu dữ liệu từ cảm biến DHT11, vi điều khiển sẽ gửi một xung thấp (LOW) trên chân dữ liệu của cảm biến trong ít nhất 18ms để khởi đầu quá trình truyền dữ liệu.
  + Phản hồi từ cảm biến: Sau khi nhận được yêu cầu, cảm biến DHT11 sẽ trả lời bằng cách trả về một xung cao (HIGH) trong khoảng 20-40μs để xác nhận yêu cầu.
  + Truyền dữ liệu nhiệt độ và độ ẩm: Cảm biến DHT11 sau đó bắt đầu truyền dữ liệu nhiệt độ và độ ẩm dưới dạng tín hiệu số. Dữ liệu này được truyền bit by bit, với mỗi bit được biểu diễn bằng thời gian của xung cao (HIGH). Một bit 0 được biểu diễn bằng xung cao trong khoảng 26-28μs, trong khi một bit 1 được biểu diễn bằng xung cao trong khoảng 70μs.
  + Gói dữ liệu: Cảm biến DHT11 truyền dữ liệu theo một thứ tự cụ thể, thường là độ ẩm trước và sau đó là nhiệt độ. Dữ liệu truyền có độ dài 40 bit: 16 bit độ ẩm, 16 bit nhiệt độ, và 8 bit kiểm tra độ tin cậy.
  + Kiểm tra độ tin cậy: Vi điều khiển nhận dữ liệu và sử dụng 8 bit kiểm tra độ tin cậy để kiểm tra tính toàn vẹn của dữ liệu. Kiểm tra này được thực hiện bằng cách tính tổng 8 bit dữ liệu và so sánh nó với 8 bit kiểm tra độ tin cậy được truyền từ cảm biến.
  + Chuyển đổi dữ liệu: Sau khi kiểm tra độ tin cậy thành công, vi điều khiển chuyển đổi dữ liệu từ dạng bit sang giá trị độ ẩm và nhiệt độ tương ứng. Các giá trị này có thể được sử dụng cho các ứng dụng hiển thị hoặc điều khiển.

Lưu ý rằng việc đọc dữ liệu từ cảm biến DHT11 yêu cầu vi điều khiển có thể thực hiện giao tiếp theo giao thức này. Một số thư viện và mã nguồn mở đã được phát triển để giúp đơn giản hóa việc đọc dữ liệu từ cảm biến DHT11 trên các nền tảng như Arduino và Raspberry Pi.

## **2.3. LED**



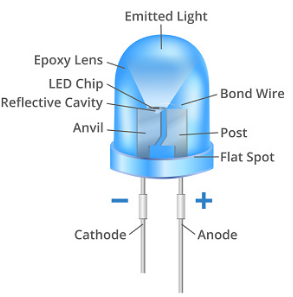
Hình 2. 5: Một số LED với nhiều màu sắc

LED, viết tắt của "Light Emitting Diode," là một loại thiết bị điện tử chuyên dùng để phát sáng. LED đã có sự phát triển đáng kể và trở thành một công nghệ chiếu sáng phổ biến trong nhiều ứng dụng khác nhau như chiếu sáng trong gia đình, màn hình điện thoại di động, màn hình TV, đèn giao thông, và nhiều ứng dụng công nghiệp khác. Điều này là do LED có hiệu suất cao, tuổi thọ dài, tiết kiệm năng lượng và khả năng tạo ra nhiều màu sắc..

### **2.3.1. Thông số kĩ thuật:**

* + Điện Áp (Voltage): LED hoạt động ở điện áp thấp, thường từ 1.5V đến 3.5V tùy thuộc vào loại LED.
  + Dòng Điện (Current): Dòng điện được cung cấp cho LED để phát sáng thường là dòng liên tục và phụ thuộc vào loại LED và màu sắc.
  + Công Suất (Power): Công suất của LED thường được đo bằng mW (miliwatt) hoặc W (watt) và thường dao động từ vài mW đến vài W.
  + Màu Sắc (Color): LED có khả năng phát ra nhiều màu sắc khác nhau, bao gồm đỏ, xanh, lam, trắng, và nhiều màu sắc khác. Màu sắc được định bởi chất liệu bán dẫn bên trong LED.
  + Tuổi Thọ (Lifespan): Tuổi thọ của LED rất cao, thường từ 25.000 đến 100.000 giờ hoặc hơn, tùy thuộc vào điều kiện sử dụng.
  + Góc Phát Sáng (Viewing Angle): Góc mà LED phát sáng được đo bằng độ và có thể là 120 độ, 90 độ, hoặc giá trị khác tùy thuộc vào loại LED.
  + Nhiệt Độ Hoạt Động (Operating Temperature): LED thường hoạt động tốt trong khoảng nhiệt độ từ -40°C đến 85°C.

### **2.3.2. Nguyên lí hoạt động**



Hình 2. 6: Cấu tạo LED

LED là một dạng của bán dẫn (semiconductor) chứa hai lớp bán dẫn khác nhau, thường là bán dẫn đáy (n-type) và bán dẫn lỗ (p-type), nối lại với nhau thông qua một kết nối được gọi là cầu (p-n junction). Khi dòng điện chạy qua cầu p-n junction, các electron từ lớp n-type chuyển qua lớp p-type và kết hợp với lỗ. Trong quá trình này, các electron giảm năng lượng và phát ra photon, tạo ra ánh sáng. Sự kết hợp của các electron và lỗ này tạo ra sự phát sáng trong LED.

Sự khác nhau về chất liệu bán dẫn trong cầu và các vật liệu phủ bên ngoài LED sẽ quyết định màu sắc cụ thể của ánh sáng mà LED phát ra. Ví dụ, sử dụng bán dẫn gallium nitride (GaN) với phốtpho (phosphor) sẽ tạo ra ánh sáng màu trắng.

Điều khiển dòng điện qua LED là cách để kiểm soát độ sáng và màu sắc, và thường được thực hiện thông qua mạch điện tử điều khiển.

**CHƯƠNG 3: TIẾN HÀNH ĐỒ ÁN**

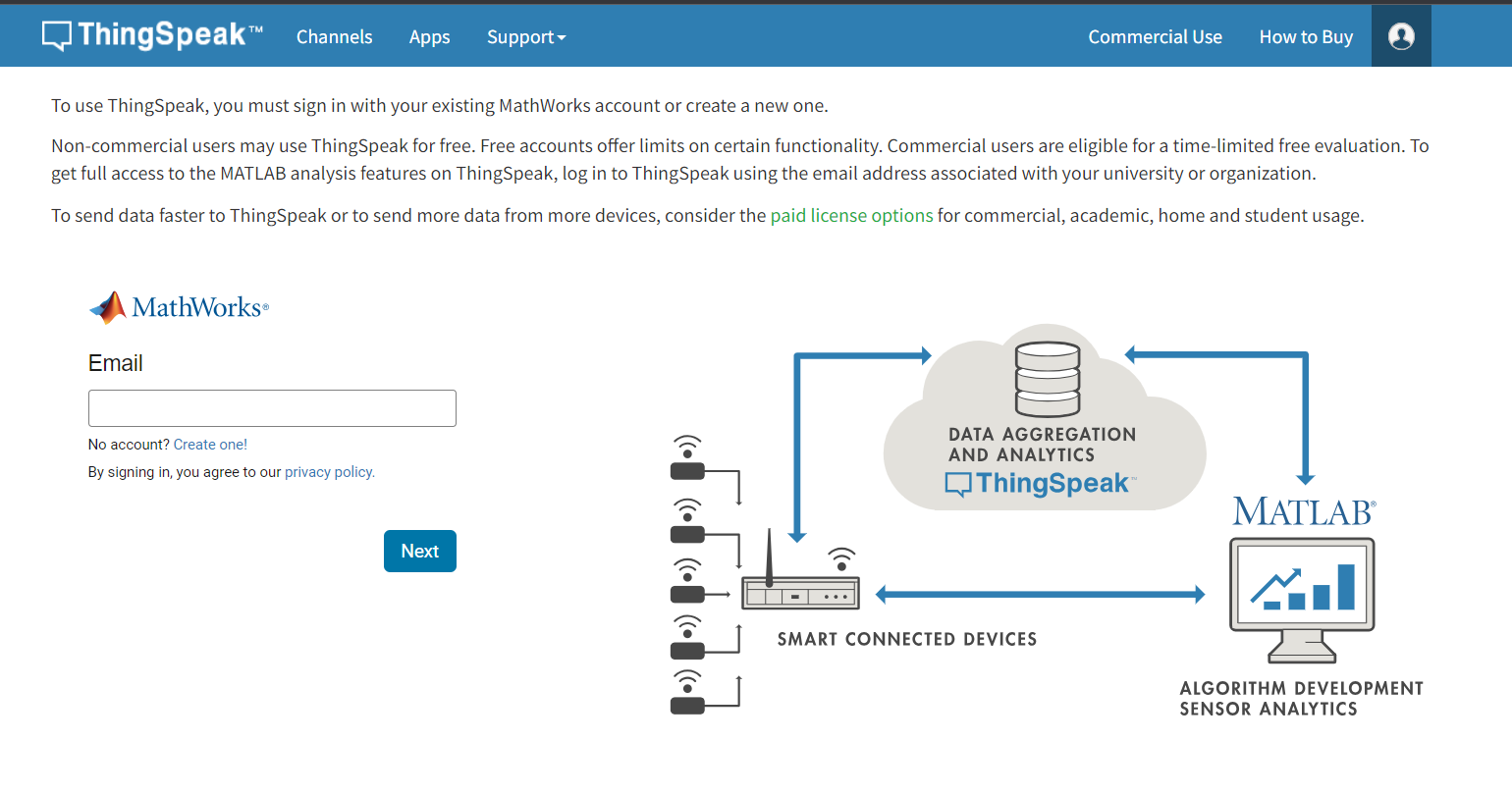
## **3.1. Xây dựng kênh dữ liệu trên Thingspeak**

Thingspeak là một nền tảng dựa trên đám mây cho Internet of Things (IoT) cho phép bạn thu thập, lưu trữ và hiển thị dữ liệu từ các thiết bị IoT của bạn. Để tạo một kênh trên Thingspeak, bạn cần thực hiện các bước sau:

* Đăng ký tài khoản Thingspeak:

Truy cập trang web Thingspeak: https://thingspeak.com/.

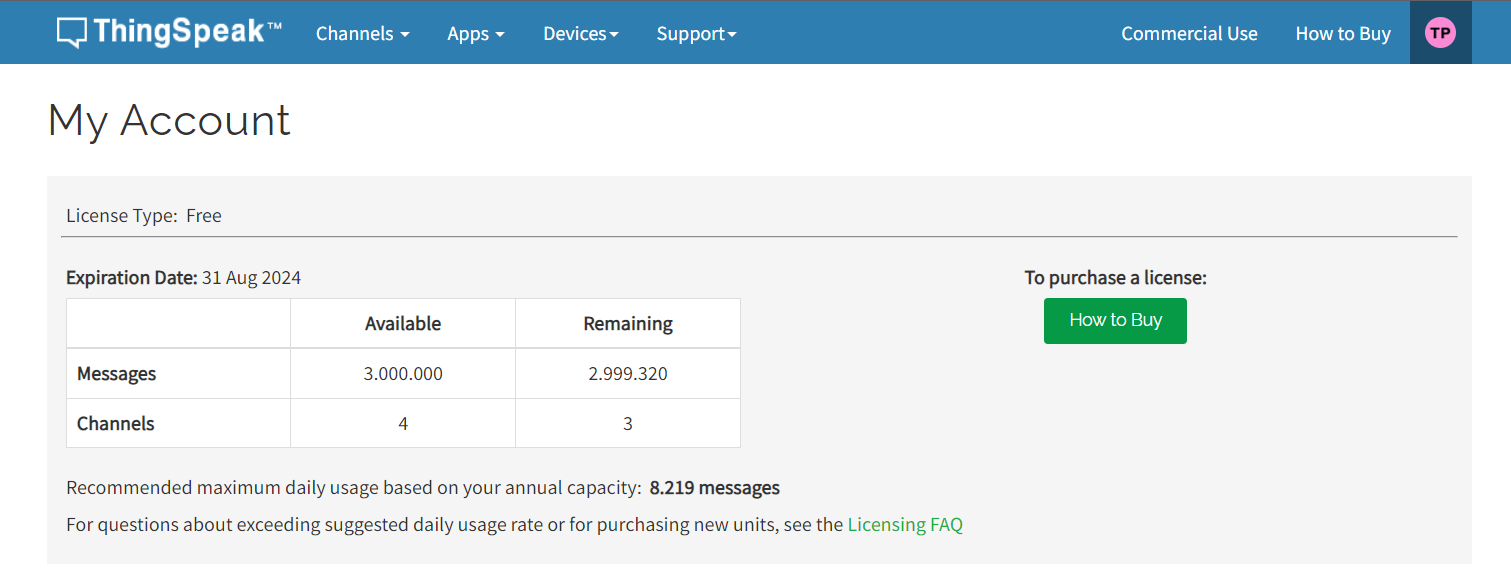
Nhấn vào nút "Create one" để tạo tài khoản nếu bạn chưa có.



Hình 3. 1: Đăng ký tài khoản Thingspeak

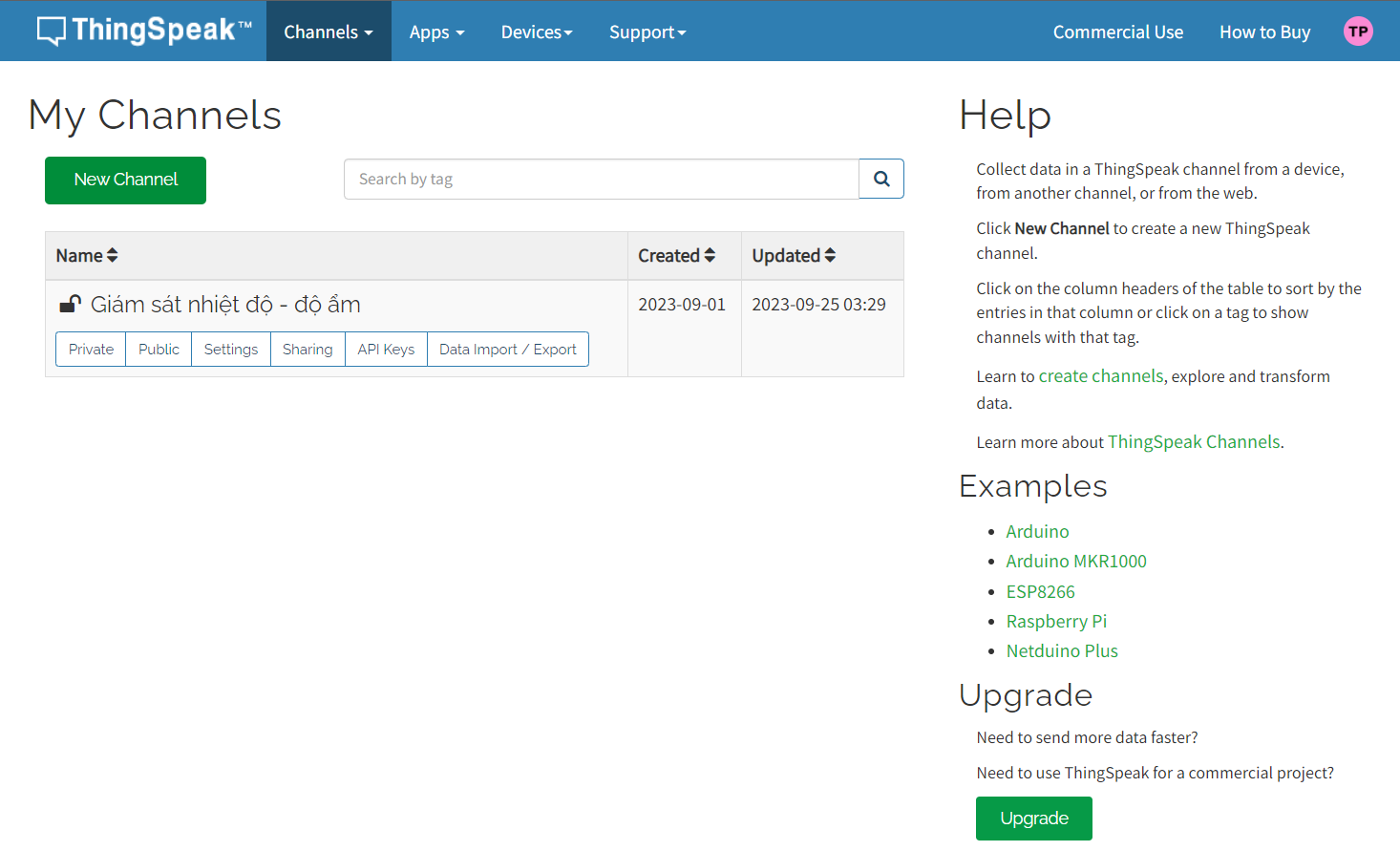
* Đăng nhập vào tài khoản Thingspeak:

Sử dụng tên người dùng và mật khẩu của bạn để đăng nhập.



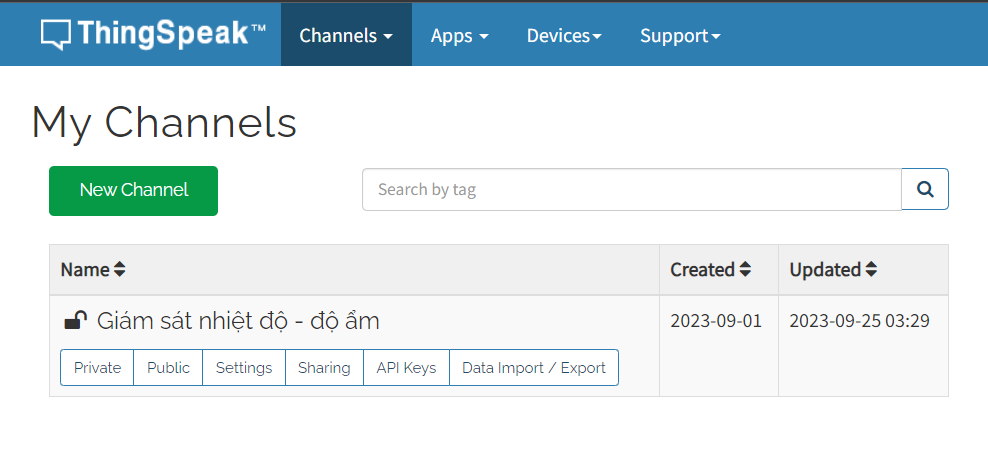
Hình 3. 2: Đăng nhập vào tài khoản Thingspeak

* Tạo kênh mới:
  + Sau khi đăng nhập, truy cập trang chính của Thingspeak và nhấn vào "Channels" ở thanh điều hướng trên cùng.



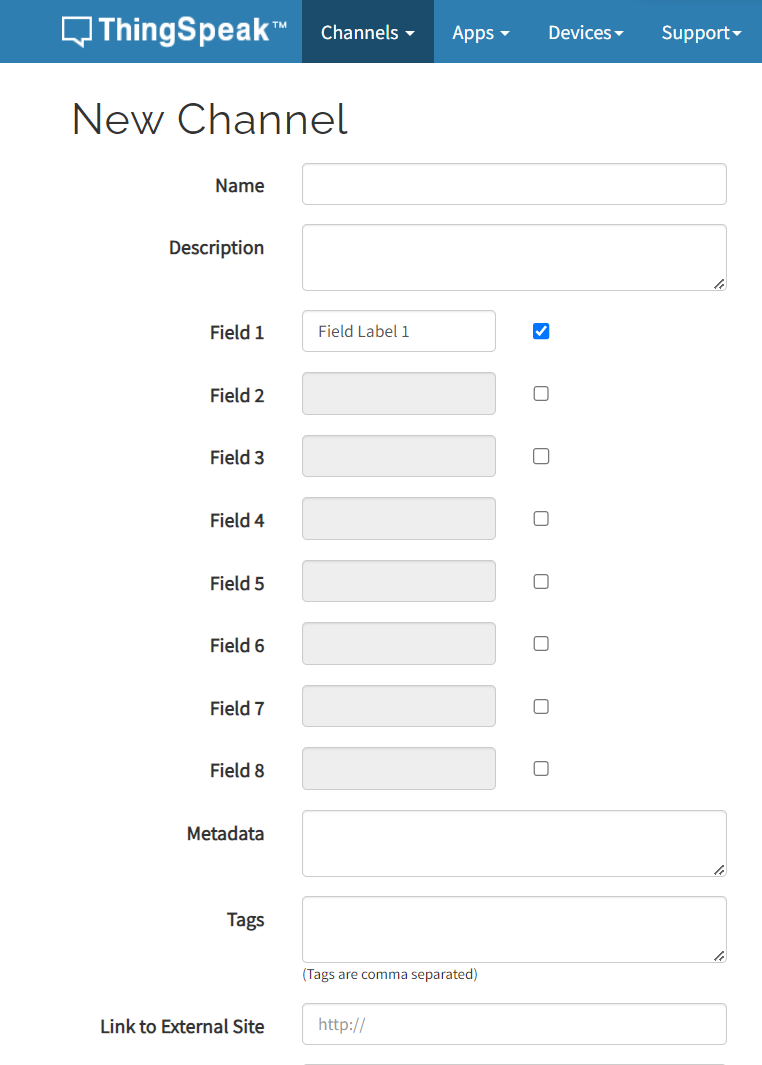
Hình 3. 3: Truy cập trang chính của Thingspeak

* + Nhấn vào nút "New Channel" để tạo kênh mới.



Hình 3. 4: Tạo kênh mới trên Thingspeak

* Cấu hình chi tiết kênh:
  + Điền các thông tin cơ bản như tên kênh, mô tả và dạng dữ liệu (dạng số, dạng văn bản, v.v.).



Hình 3. 5: Điền các thông tin cơ bản của channel

* + Bạn cũng có thể cấu hình quyền riêng tư của kênh và xác định liệu dữ liệu có thể được ghi vào kênh hay không.
* Tạo các trường dữ liệu (Fields):

Bạn có thể thêm các trường dữ liệu (Fields) cho kênh của mình bằng cách nhấn "Add a Field". Điều này cho phép bạn thu thập nhiều thông tin từ nhiều thiết bị khác nhau trên cùng một kênh.

* Lưu cấu hình kênh:

Sau khi cấu hình kênh và các trường dữ liệu, nhấn "Save Channel" để lưu cấu hình.

* Ghi lại thông tin cơ bản về kênh:

Trước khi bạn bắt đầu gửi dữ liệu lên kênh, bạn có thể điền các thông tin cơ bản như tên kênh, mô tả, và các thông tin liên quan khác.

* Nhập dữ liệu vào kênh:

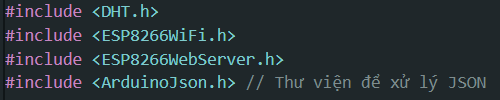
Sử dụng các thiết bị IoT hoặc ứng dụng để gửi dữ liệu vào kênh mà bạn vừa tạo. Thông tin này sẽ được hiển thị và lưu trữ trên kênh của bạn.

* Truy cập và quản lý dữ liệu:

Bạn có thể xem dữ liệu thu thập được từ kênh của mình thông qua giao diện web Thingspeak hoặc sử dụng API của Thingspeak để truy xuất dữ liệu từ ứng dụng khác.

## **3.2.Lập trình phần cứng trên Arduino IDE**

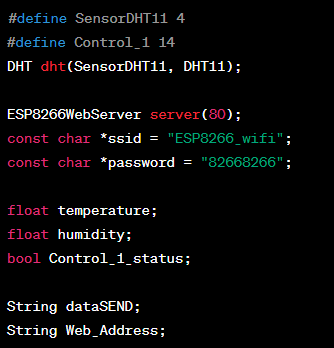
Đầu tiên,sử dụng các thư viện:



Hình 3. 6: Ảnh chụp Arduino IDE– Thư viện sử dụng

* DHT.h: Thư viện để làm việc với cảm biến DHT.
* ESP8266WiFi.h: Thư viện để làm việc với mô-đun ESP8266 và kết nối Wi-Fi.
* ESP8266WebServer.h: Thư viện để tạo máy chủ web.
* ArduinoJson.h: Thư viện để xử lý JSON.

Kế tiếp, định nghĩa các biến và hằng số cần thiết:

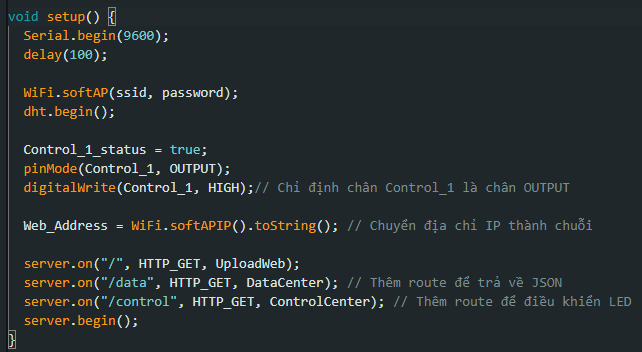


Hình 3. 7: Ảnh chụp Arduino IDE– Các biến và hằng số sử dụng

* SensorDHT11 và Control\_1 định nghĩa các chân GPIO của ESP8266 mà cảm biến DHT11 và LED kết nối tới.
* dht là đối tượng cảm biến DHT11.
* ESP8266WebServer server(80): Khởi tạo đối tượng server với cổng 80 (HTTP).
* ssid và password là thông tin mạng Wi-Fi mà ESP8266 sẽ tạo ra (Access Point).
* Các biến temperature, humidity, và Control\_1\_status lưu trữ giá trị nhiệt độ, độ ẩm, và trạng thái của Control\_1.
* dataSEND: biến chuỗi để gửi dữ liệu qua Serial.
* Web\_Address lưu trữ địa chỉ IP của ESP8266 dưới dạng chuỗi..

### **3.2.1.Hàm set up() và loop()**

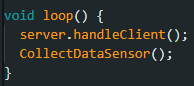
Trong hàm setup():



Hình 3. 8: Ảnh chụp Arduino IDE– Hàm setup()

* Khởi tạo giao tiếp Serial với tốc độ 9600 baud.
* Tạo một mạng Wi-Fi với tên mạng ssid và mật khẩu password.
* Khởi tạo cảm biến DHT11.
* Khởi tạo chân Control\_1 là chân OUTPUT và bật HIGH.
* Lấy địa chỉ IP của máy chủ web ESP8266 và lưu trữ nó trong biến Web\_Address.
* Đăng ký 3 route trên máy chủ web:
  + Route /: Điều này sẽ xử lý trang chính hiển thị thông tin nhiệt độ và độ ẩm.
  + Route /data: Điều này sẽ xử lý yêu cầu để trả về dữ liệu JSON nhiệt độ và độ ẩm.
  + Route /control: Điều này sẽ xử lý yêu cầu để nhận và xử lí dữ liệu điều khiển từ web

Trong hàm loop():

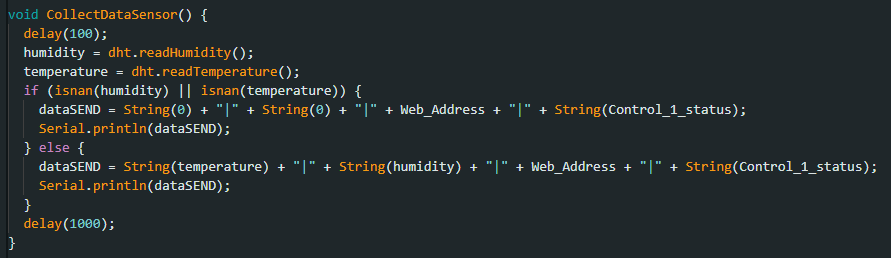


Hình 3. 9: Ảnh chụp Arduino IDE– Hàm loop()

* Hàm server.handleClient() được gọi để xử lý các yêu cầu từ máy khách (trình duyệt).
* Hàm CollectDataSensor() được gọi để đọc dữ liệu từ cảm biến DHT11 và cập nhật biến temperature và humidity, sau đó gửi dữ liệu về Serial Monitor.

### **3.2.2.Các hàm con**

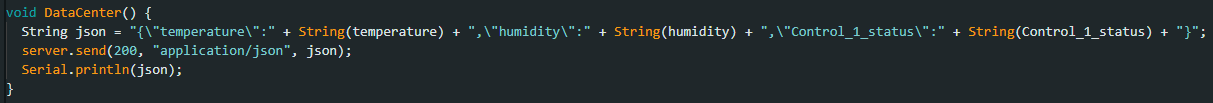
Trong hàm CollectDataSensor():



Hình 3. 10: Ảnh chụp Arduino IDE– Hàm con CollectDataSensor()

* Hàm này đọc dữ liệu từ cảm biến DHT11, cập nhật biến temperature và humidity, sau đó tạo một chuỗi dữ liệu dataSEND chứa nhiệt độ, độ ẩm, địa chỉ IP của ESP8266 và trạng thái Control\_1 (phục vụ giao tiếp VISA với LabVIEW).
* Dữ liệu này được gửi đến Serial Monitor.

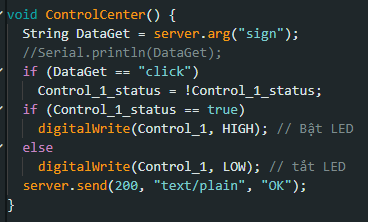
Trong hàm DataCenter():



Hình 3. 11: Ảnh chụp Arduino IDE– Hàm con DataCenter()

* Hàm này được gọi khi có yêu cầu truy cập route "/data".
* Nó tạo một JSON chứa nhiệt độ, độ ẩm và trạng thái Control\_1, sau đó gửi dữ liệu JSON đó.

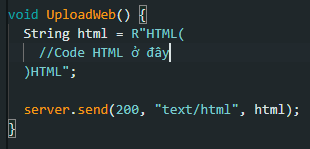
Trong hàm ControlCenter():



Hình 3. 12: Ảnh chụp Arduino IDE– Hàm con ControlCenter()

* Hàm này đầu tiên trích xuất giá trị của tham số "sign" từ yêu cầu truy cập route “/control”
* Nếu là "click", chương trình sẽ thực hiện việc điều khiển đảo trang thái bật tắt tại Control\_1.
* Cuối cùng, hàm gửi một phản hồi HTTP với mã trạng thái 200 (OK) và nội dung "OK" để thông báo rằng yêu cầu điều khiển đã được xử lý thành công.

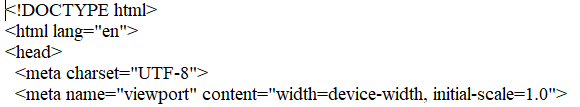
Trong hàm UploadWeb():

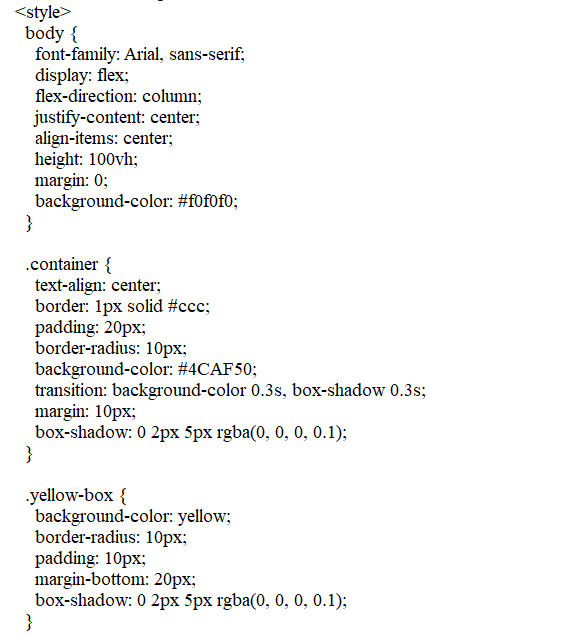


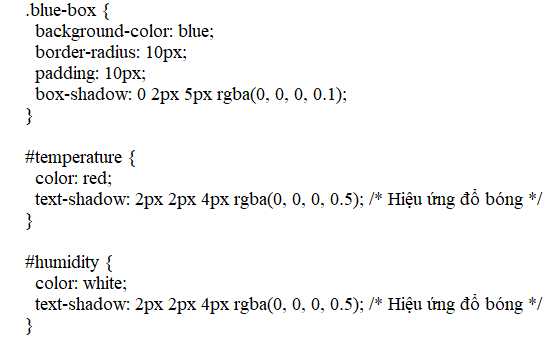
Hình 3. 13: Ảnh chụp Arduino IDE– Hàm con UploadWeb()

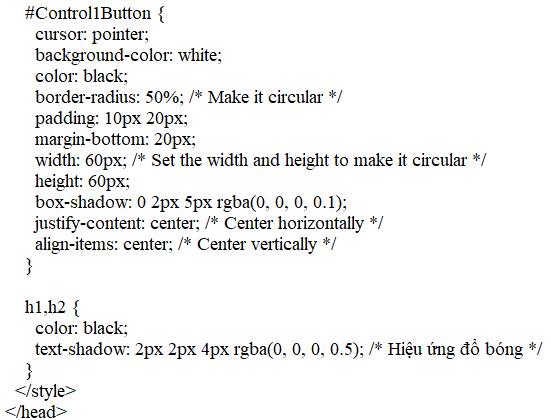
* Hàm này được gọi khi có yêu cầu truy cập trang web gốc ("/").
* Nó gửi đi một trang HTML.

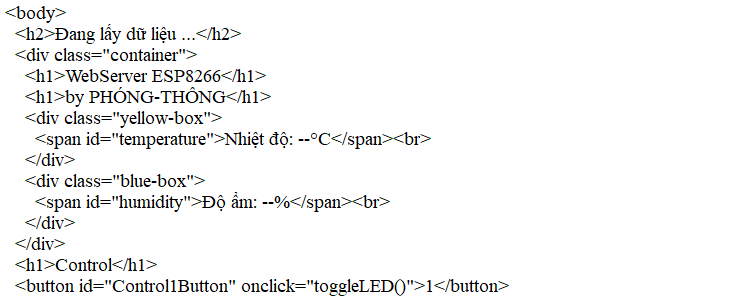
Mã HTML được nhúng:

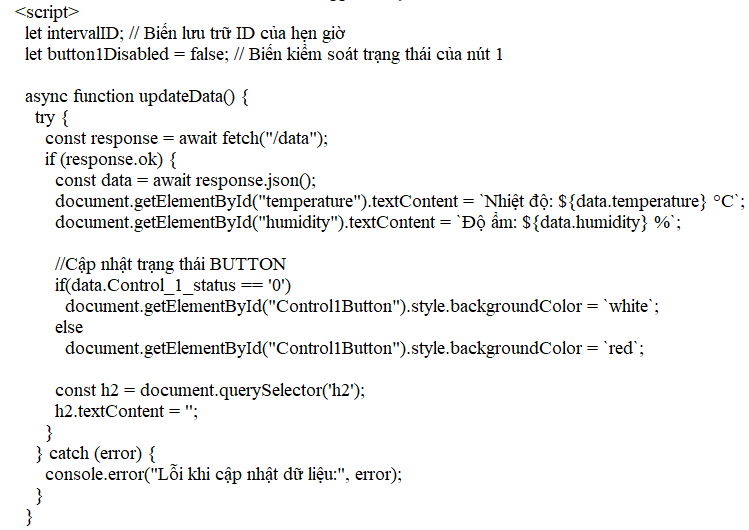


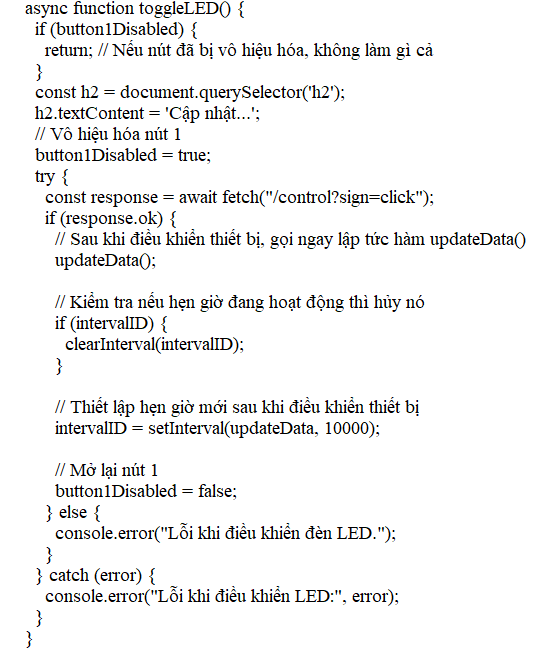


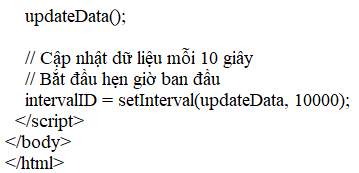












## **3.3. Xử lí dữ liệu trên LabVIEW**

### **3.3.1.Xây dưng các subVI hỗ trợ**

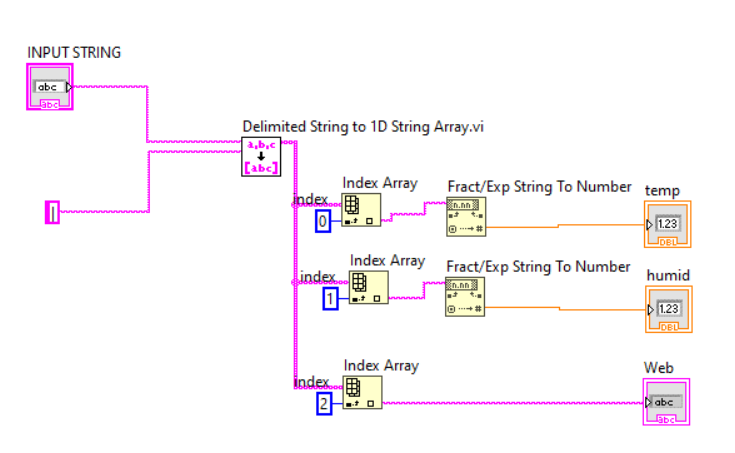
-SubVI thứ 1: Tách dữ liệu thu được

Trong phần lập trình Arduino, ta thực hiện truyền dữ liệu bao gồm giá trị nhiệt độ, giá trị độ ẩm, địa chỉ IP của máy chủ web ESP8266; các phần tử được ngăn cách bởi kí tự “|”

Để truy cập các dữ liệu thu được về LabVIEW, ta tiến hành tạo 1 subVI thực hiện việc tách và trả về kiểu chính xác của chúng.



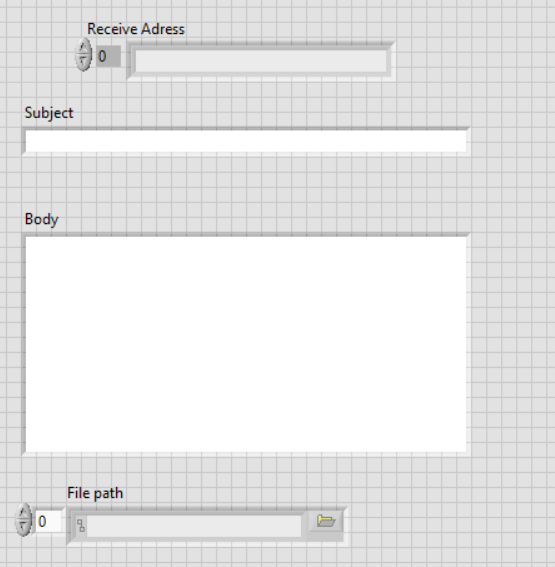
Hình 3. 14: Giao diện Split.vi (Front Panel)



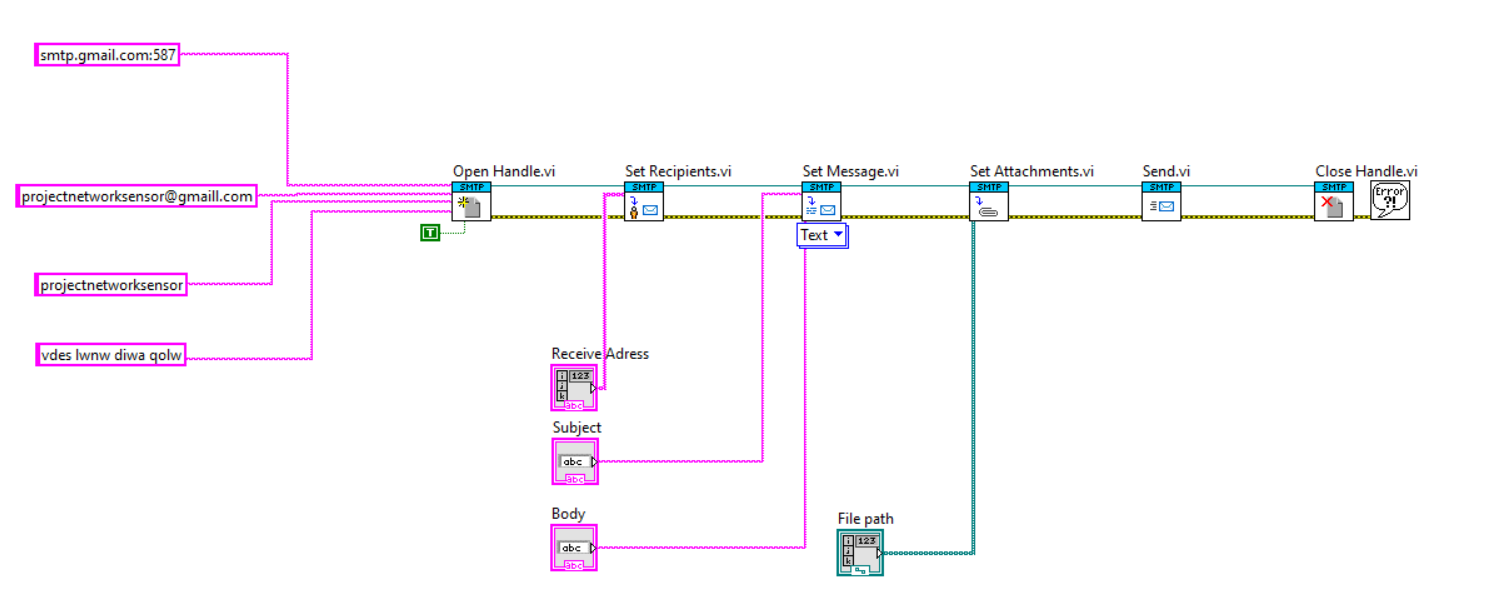
Hình 3. 15: Giao diện Split.vi (Block Diagram)

-SubVI thu 2: Gửi email từ LabVIEW

Để gửi được email, ta cần tạo 1 email cá nhân và đặt làm email nguồn cho subVI



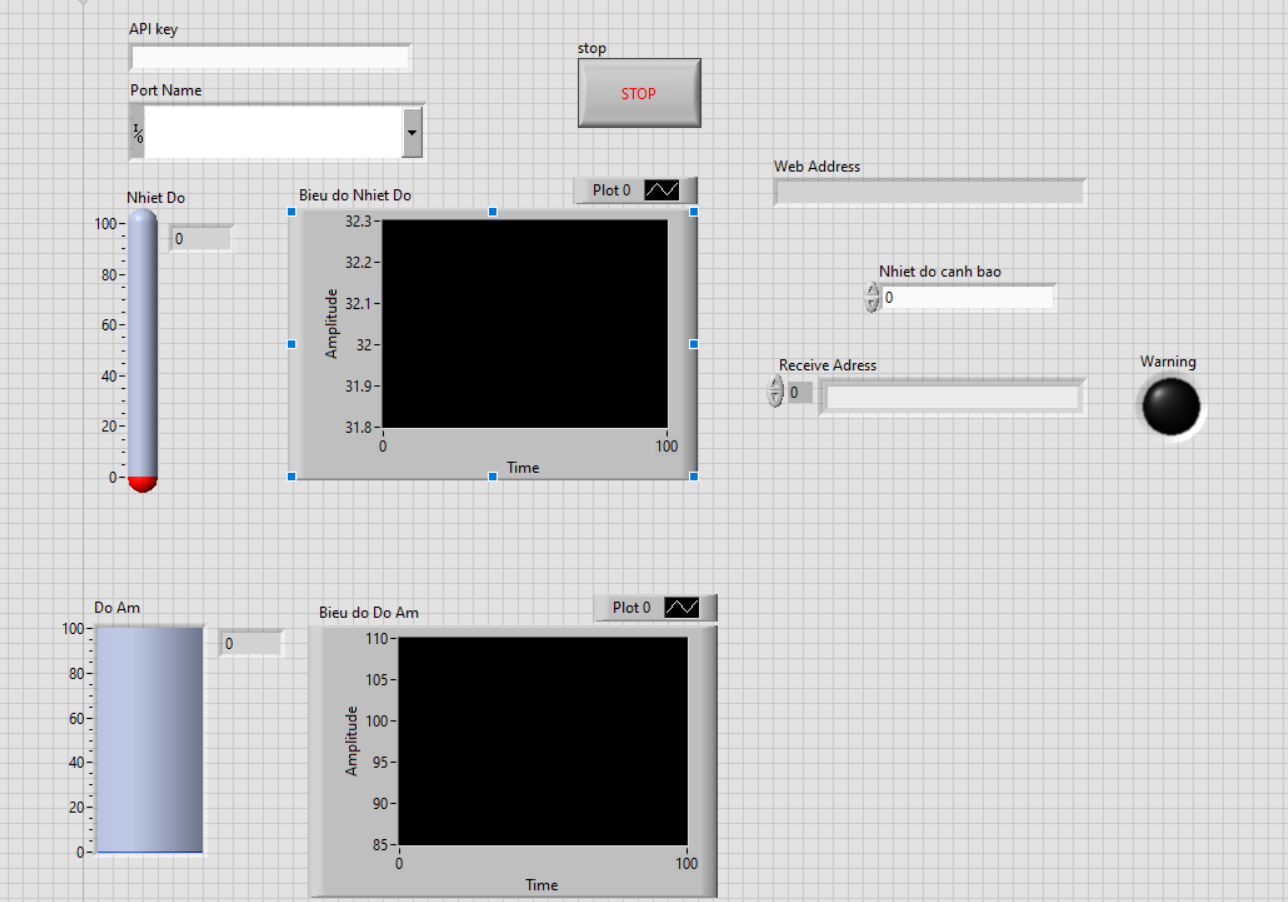
Hình 3. 16: Giao diện SendEmail.vi (Front Panel)



Hình 3. 17: Giao diện SendEmail.vi (Block Diagram)

### **3.3.2.Thiết kế giao diện LabVIEW**

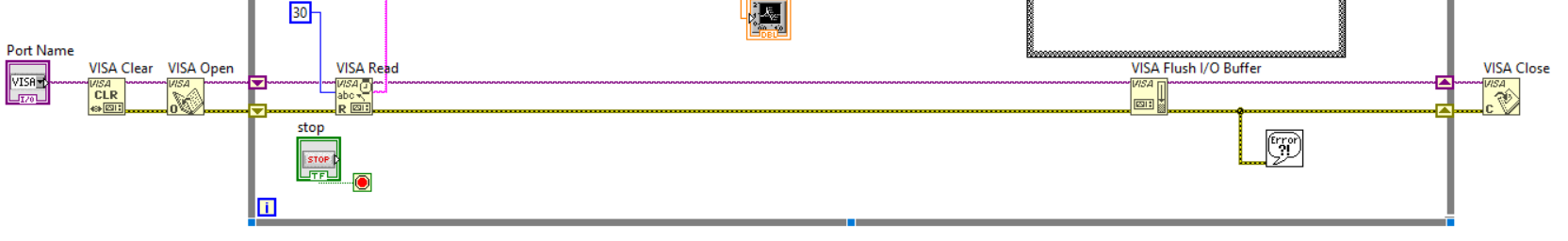
Kéo thả các khung nhập API key (dùng để truy cập đến channel đã tạo trên Thingspeak), PortName (dùng để chọn Port cho giao thức VISA); themeter Nhiet Do, Do Am; Wave Chart Bieu Do Nhiet Do, Bieu Do Do Am; khung nhập Web Adress (dùng để truy cập web local được tạo bởi ESP8266); khung nhập Nhiet do canh bao, Receive Adress (dùng để gửi email khi nhiệt độ đạt giới hạn cần cảnh báo).



Hình 3. 18: Giao diện thiết kế(Front Panel)

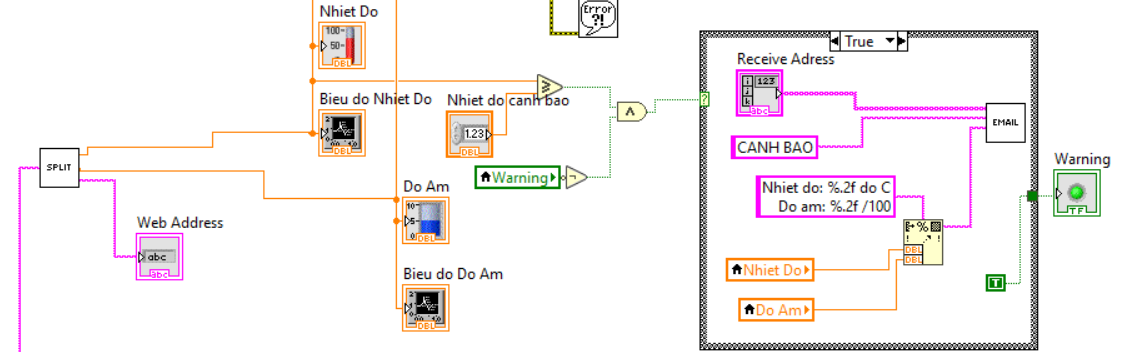
Các phần triển khai:

-Đọc dữ liệu từ Serial Port thông qua giao thức VISA: Sử dụng các khối như VISA Clear, VISA Open, VISA Read, VISA Flush I/O Buffer và VISA Close.

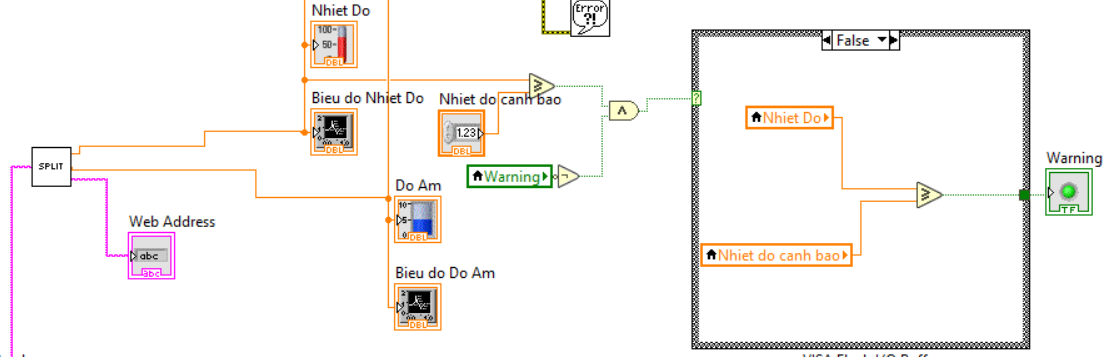


Hình 3. 19: Giao diện thiết kế - Đọc dữ liệu từ Serial Port thông qua giao thức VISA

-Hiện biểu đồ và thiết kế gửi email khi cần: Sử dụng các khung hiển thị và các khối như Wave Chart, Case Structure và 2 SubVI cá nhân là Split và Email.

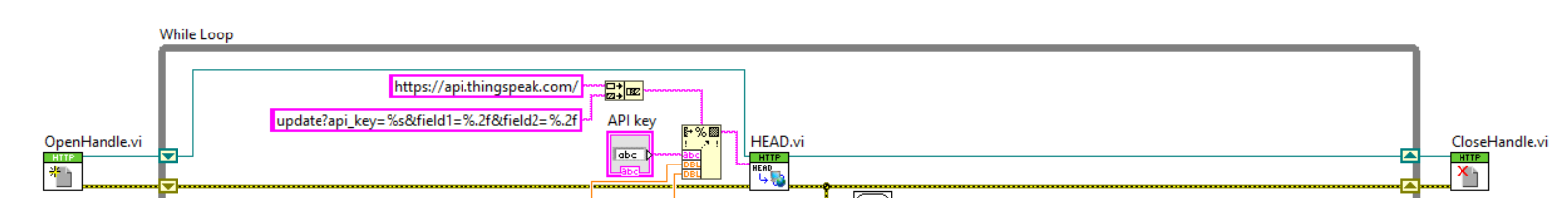


Hình 3. 20: Giao diện thiết kế - Hiện biểu đồ và thiết kế gửi email khi cần



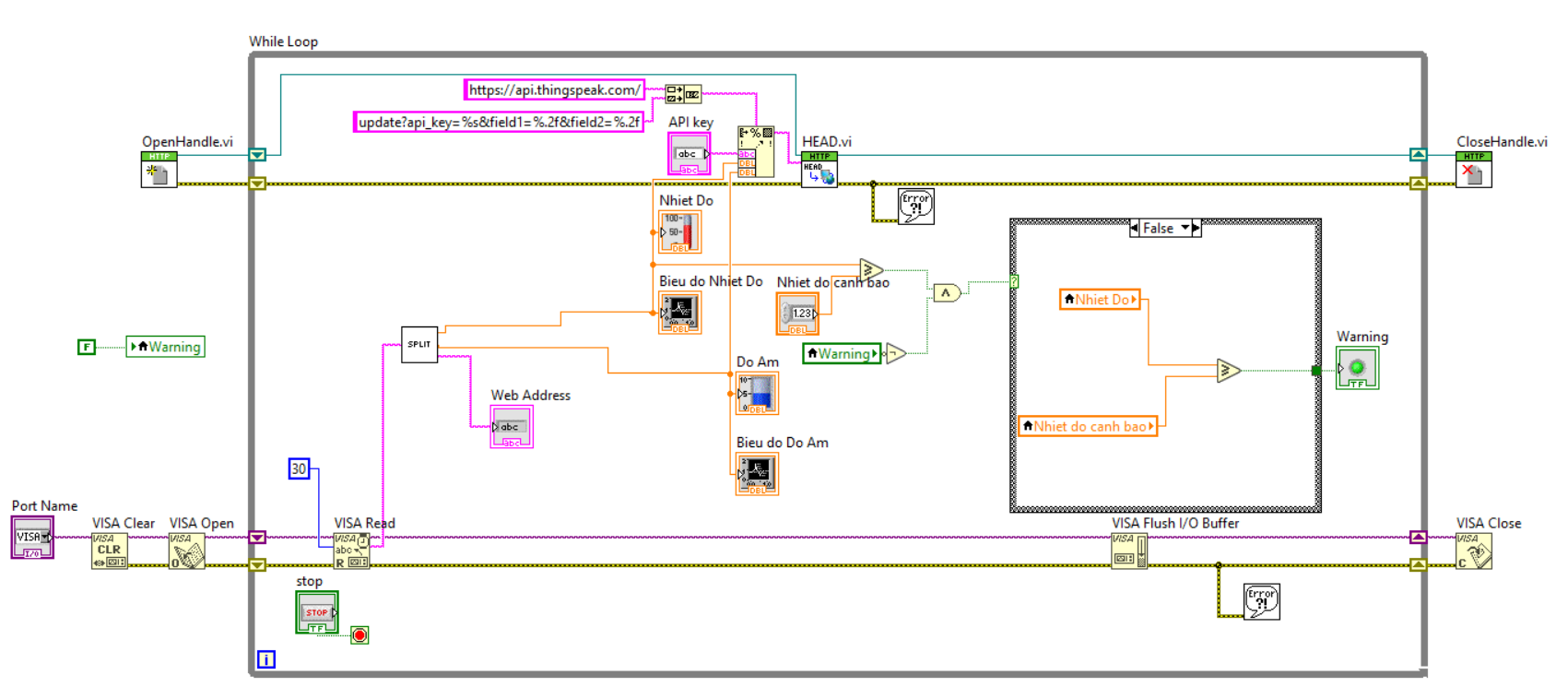
Hình 3. 21: Giao diện thiết kế - Hiện biểu đồ và thiết kế gửi email khi cần

-Đưa dữ liệu lên ThingSpeak: Sử dụng các khối như Open Handle, Head, Close Handle và các khung nhập API key và khối ghép dữ liệu nhiệt độ, độ ẩm.



Hình 3. 22: Giao diện thiết kế - Đưa dữ liệu lên ThingSpeak

Toàn bộ phần triển khai LabVIEW:

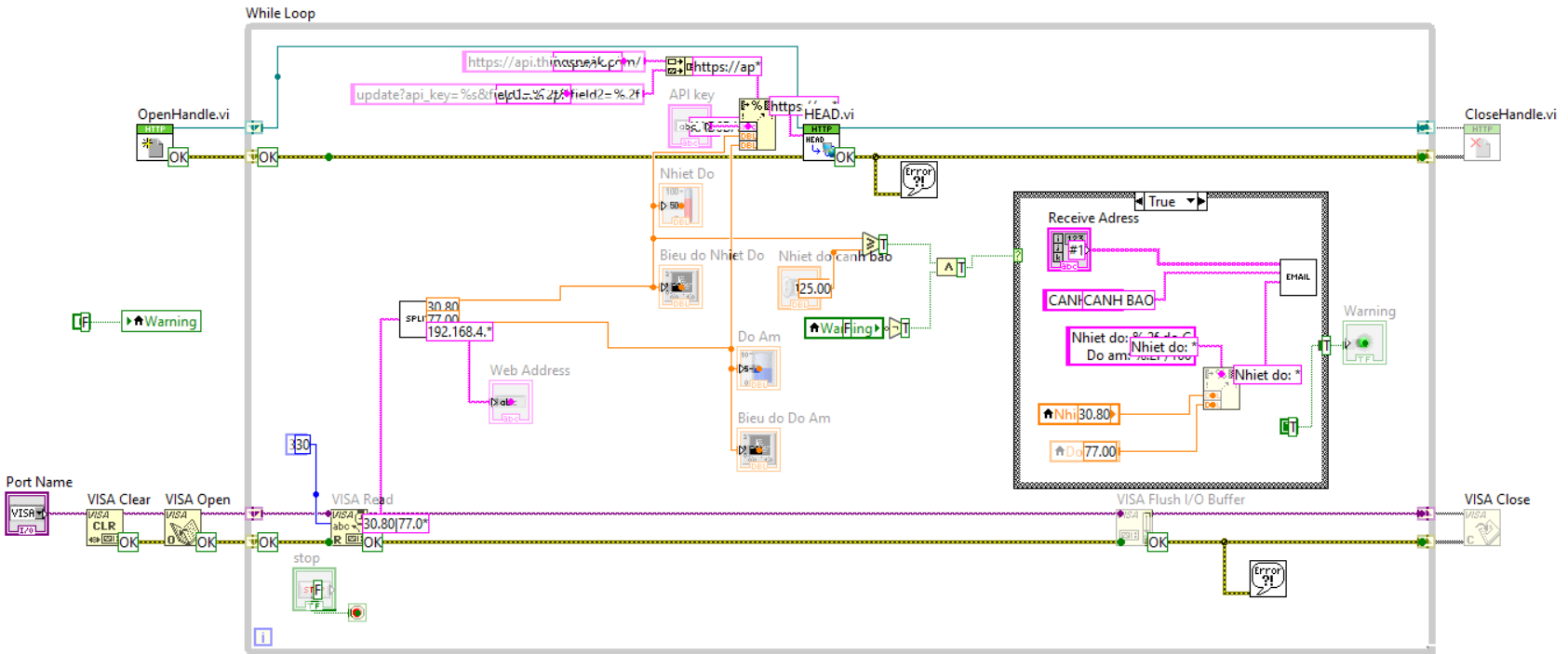


Hình 3. 23: Giao diện thiết kế (Block Diagram)

## **3.4.Chạy thực tế đồ án**

### **3.4.1. Gửi dữ liệu lên Thingspeak và gửi email cảnh báo:**

Quá trình truyền dữ liệu đến LabVIEW thông qua giao tiếp VISA, dữ liệu truyền đến sẽ được sử dụng qua khối SubVI Split để tách dữ liệu; bên cạnh, thiết lập kết nối đến chanel trên Thingspeak thông qua API key được nhập vào. Bên trong phần xử lí trong LabVIEW, khối SubVI gửi email sẽ theo dõi và gửi đi email khi đạt điều kiện (ở đây là điều kiện nhiệt độ). Ngoài ra, để trực quan, các dữ liệu được gửi đến LabVIEW sau khi được tách ra sẽ được hiển thị lên các biểu đồ, khung hiện giá trị.



Hình 3. 24: LabVIEW gửi email khi đạt điều kiện cho trước

A computer screen shot of a diagram

Description automatically generated

Hình 3. 25: Email chỉ gửi đi 1 lần cho mỗi lần nhiệt độ vượt quá giới hạn

A screenshot of a computer program

Description automatically generated

Hình 3. 26: Giao diện thực tế khi chạy LabVIEW

### **3.4.2.Web cá nhân để theo dõi và điều khiển phần cứng từ xa**

ESP8266 sẽ đóng vai trò như server; các clients sẽ chủ động tương tác đến server thông qua 3 route như đã trình bày trong mục trước bao gồm tải trang web, tự cập nhật dữ liệu và gửi điều khiển thông qua nút nhấn trên giao diện web.

Quá trình tương tác cơ bản có thể tóm tắt bao gồm: Đầu tiên clients sẽ kết nối với ssid và password do chính ESP8266 tạo ra; Sau đó, để truy cập trang web theo dõi dữ liệu trên ESP8266, clients tiến hành kết nối với server thông qua đia chỉ IP là 192.168.4.1 (trên giao diện LabVIEW cũng tích hợp hiển thị địa chỉ IP này); Trang web xuất hiện với dữ liệu thu được từ ESP8266 (bao gồm khung hiện nhiệt độ, độ ẩm và một nút có thể đổi màu biểu thị trạng thái của control\_1) và tự động cập nhật dữ liệu sau một khoảng thời gian (cụ thể ở đây là 10 giây) hoặc khi clients thực hiện điều khiển control\_1 bằng việc nhấn nút có trên giao diện.



Hình 3. 27: Giao diện trang web cơ bản



Hình 3. 28: Giao diện chờ khi thực hiện nhấn nút trên giao diện

# 

# PHẦN KẾT LUẬN

Trong đồ án báo cáo này, chúng em đã thực hiện dự án giám sát nhiệt độ và độ ẩm sử dụng mạch thu thập dữ liệu DHT11 và kết nối với mạng Wi-Fi thông qua mạch ESP8266. Dự án này đã giúp chúng em xây dựng một hệ thống giám sát, theo dõi và điều khiển môi trường trong thời gian thực. Bằng việc tự nghiệm thu kết quả đồ án của nhóm, chúng em cũng rút ra được những ưu- nhược điểm của đồ án:

* + Ưu điểm
    - Phối hợp khả năng thu thập dữ liệu và gửi dữ liệu của ESP8266
    - Tạo ra được máy Chủ Web cá nhân, có thể tùy biến cá nhân.
    - Web cá nhân ngoài theo dõi được nhiệt đô – độ ẩm còn có thể tương tác điều khiển lên ESP8266.
  + Hạn chế đồ án
    - Chỉ mới triển khai với quy mô nhỏ, chưa thể giám sát, theo dõi nhiều cảm biến, nhiều địa điểm đồng thời.
    - Lưu trữ dữ liệu thu thập được trên Thingspeak khá cơ bản, chưa thể thực hiện tính toán, phân tích với các dữ liệu trước và hiện tại.
    - Web Local xây dựng còn đơn giản, server tạo ra không đủ mạnh khi số clients truy cập tăng cao.
  + Hướng phát triển của đề tài
    - Nâng quy mô số lượng cảm biến, địa điểm cần theo dõi, giám sát nhiệt độ, độ ẩm.
    - Triển khai server cá nhân lưu trữ dữ liệu thu thập được thay vì dùng Thingspeak.
    - Web local được nâng cấp với nhiều tính năng hơn.

Sinh viên thực hiện Giảng viên hướng dẫn

Trần Hào Phóng Hồ Nhựt Minh

Võ Minh Thông