BỘ THÔNG TIN VÀ TRUYỀN THÔNG
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÁO CÁO BÀI TẬP NHÓM

Môn: Mạng cảm biến

Đề tài: Nghiên cứu và xây dựng giao diện IoT(dashboard) đơn giản theo dõi nhiệt độ và độ ẩm

Giảng viên hướng dẫn: Trần Thị Thu Thủy

Nhóm sinh viên:	**1. Vũ Quang Sáng**	**B18DCDT202**
	2. Dương Công Hòa	**B19DCDT084**
	3. Phan Thế Việt	**B19DCDT253**
	4. Lã Minh Hoàng	**B19DCDT093**

Hà Nội – 2023

MỤC LỤC

LỜI MỞ ĐẦU

Trong bối cảnh cuộc cách mạng công nghiệp 4.0 đang ngày càng trở nên trung tâm trong phát triển của xã hội, Internet of Things (IoT) đã trở thành một trong những xu hướng quan trọng, định hình và thúc đẩy sự tiến bộ của nhiều lĩnh vực khác nhau. Việc kết nối các thiết bị thông minh và thu thập dữ liệu từ chúng mở ra những cơ hội mới trong việc quản lý và kiểm soát môi trường xung quanh chúng ta.

Nghiên cứu này tập trung vào việc phát triển và xây dựng một demo giao diện IoT đơn giản nhưng mạnh mẽ, giúp theo dõi thông tin về nhiệt độ và độ ẩm. Đây không chỉ là một ứng dụng thực tế mà còn là một bước tiến quan trọng trong việc hiểu rõ cách chúng ta có thể tích hợp công nghệ IoT vào cuộc sống hàng ngày.

Giao diện dashboard được thiết kế trong nghiên cứu này không chỉ tập trung vào sự tiện lợi và dễ sử dụng, mà còn chú trọng đến khả năng hiển thị dữ liệu một cách rõ ràng và trực quan. Mục tiêu là tạo ra một trải nghiệm người dùng tốt nhất, giúp họ nhanh chóng và dễ dàng theo dõi các thông số môi trường quan trọng. Bên cạnh đó, nghiên cứu này cũng đề cập đến các thách thức và quyết định thiết kế trong quá trình xây dựng giao diện. Những quyết định này không chỉ ảnh hưởng đến trải nghiệm người dùng mà còn đề cập đến vấn đề an ninh và quản lý dữ liệu, quan trọng trong việc bảo vệ thông tin cá nhân và đảm bảo tính ổn định của hệ thống.

Chúng em xin gửi lời cảm ơn đến cô Trần Thị Thu Thủy – giảng viên bộ môn "Mạng cảm biến" đã trang bị cho chúng em những kiến thức, kỹ năng cơ bản cần có để hoàn thành đề tài này. Trong quá trình nghiên cứu đề tài, do kiến thức còn hạn chế nên chúng em vẫn nhiều thiếu sót khi tìm hiểu, thực hiện và trình bày về đề tài. Rất mong nhận được sự quan tâm, góp ý của cô để đề tài của chúng em hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

CHƯƠNG 1: TỔNG QUAN VỀ MẠNG CẢM BIẾN

1.1. Giới thiệu về mạng cảm biến

1.1.1. Lịch sử của mạng cảm biến

Mạng cảm biến (Sensor Network) là một phần quan trọng trong sự phát triển của công nghệ thông tin và viễn thông. Dưới đây là tổng quan về lịch sử phát triển của mạng cảm biến:

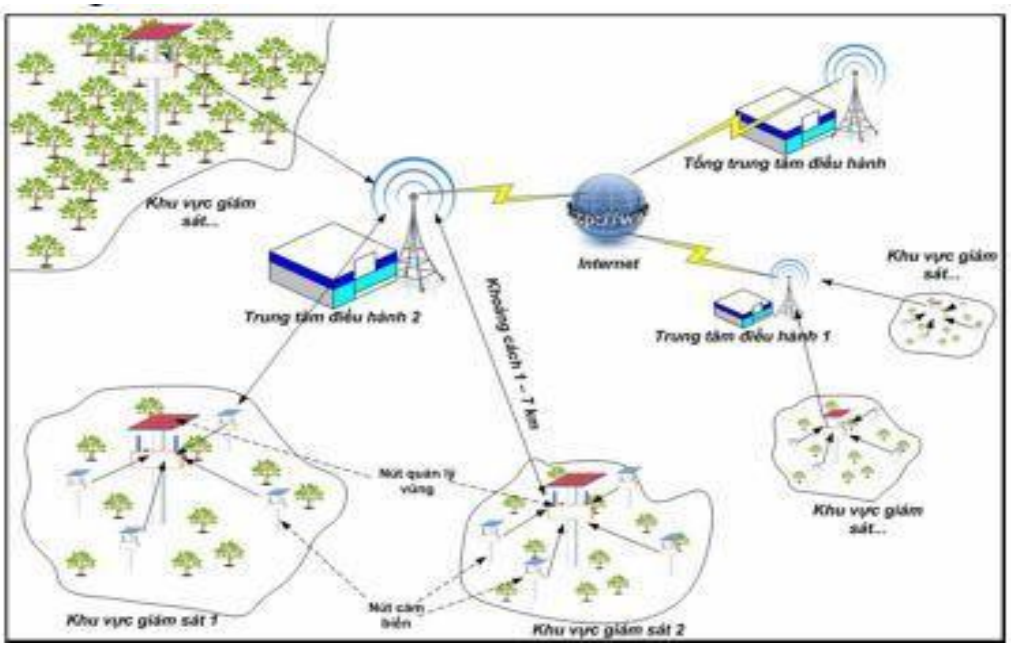
Thập kỷ 1980: mạng cảm biến xuất hiện đầu tiên, mạng cảm biến ban đầu được phát triển để theo dõi môi trường và thu thập dữ liệu về nhiệt độ, ánh sáng, áp suất và các thông số khác. Tuy nhiên chúng thường được triển khai bằng cách kết nối trực tiếp với máy tính cơ bản và không có tính năng mạng lưới.

Thập kỷ 1990: Mạng lưới không dây, công nghệ không dây phát triển nhanh chóng, giúp mạng cảm biến trở nên linh hoạt hơn và dễ dàng triển khai. Điều này dẫn đến sự ra đời của các mạng cảm biến không dây, cho phép nút cảm biến giao tiếp không dây với nhau và truyền dữ liệu đến một trạm cơ sở.

Thập kỷ 2000: Trong thập kỷ này, mạng cảm biến đã trải qua một sự phát triển khá mạnh mẽ trong các khía cạnh kỹ thuật và ứng dụng. Các tiến bộ bao gồm việc cải thiện năng suất và hiệu suất của cảm biến, cải thiện khả năng tự điều chỉnh và sửa lỗi của mạng và tích hợp các cảm biến với hệ thống thông tin đám mây.

Hiện nay: Ngày nay mạng cảm biến được ứng dụng rộng rãi và trở thành một phần quan trọng trong cuộc sống hàng ngày, có nhiều ứng dụng rộng rãi. Chúng được sử dụng trong các lĩnh vực như quản lý môi trường ,giám sát giao thông, y tế, nông nghiệp thông minh, công nghiệp và nhiều ứng dụng khác. Mạng cảm biến không chỉ giúp thu thập dữ liệu mà còn cung cấp thông tin quan trọng cho việc ra quyết định và quản lý.

Tóm lại, lịch sử của mạng cảm biến đã trải qua một sự phát triển đáng kể từ những thập kỷ đầu tiên cho đến hiện tại. Công nghệ này đã đóng một vai trò quan trọng trong việc cải thiện khả năng thu thập dữ liệu và quản lý thông tin về môi trường và các ứng dụng khác.



Hình 1.1. Mô hình của mạng cảm biến

1.1.2. Vai trò của mạng cảm biến

Trước khi mạng cảm biến xuất hiện thì các mạng như: mạng máy tính, mạng không dây,... đã tồn tại và phát triển. Tuy nhiên mạng cảm biến đã xuất hiện với mục tiêu giải quyết một loạt những vấn đề và ứng dụng cụ thể mà các mạng và hệ thống trước đây không thể thực hiện hiệu quả hoặc phù hợp. Một số lý do chính vì sao mạng cảm biến cần phải tồn tại bao gồm:

- **Giám sát môi trường:** Mạng cảm biến cho phép theo dõi môi trường tự nhiên, chẳng hạn như đo lượng mưa, nhiệt độ, áp suất, chất lượng không khí và nước, mà không cần phải có sự can thiệp của con người.
- **Tiết kiệm năng lượng:** Mạng cảm biến thường được thiết kế để hoạt động với nguồn năng lượng có hạn, chẳng hạn như pin hoặc nguồn năng lượng mặt trời, giúp tiết kiệm năng lượng so với các hệ thống truyền thông truyền thống.
- **Khả năng tự trị và phân tán:** Mạng cảm biến có thể được triển khai phân tán và có khả năng tự điều chỉnh, tự sửa chữa lỗi, tự tổ chức làm cho chúng phù hợp cho các ứng dụng mà yêu cầu sự linh hoạt và khả năng hoạt động trong môi trường khó tiếp cận.

- **Ứng dụng trong các lĩnh vực cụ thể:** Mạng cảm biến có thể được sử dụng trong nhiều lĩnh vực như quản lý môi trường, y tế, an ninh, nông nghiệp và quản lý cơ sở hạ tầng mang lại giá trị và thông tin quan trọng cho các ngành này.

Những lý do trên giúp mạng cảm biến trở thành một công nghệ quan trọng và mạnh mẽ trong việc thu thập và truyền tải dữ liệu từ môi trường và nhiều ứng dụng khác.

1.2. Những ưu điểm và nhược điểm của mạng cảm biến

1.2.1. Ưu điểm của mạng cảm biến

- **Thu thập dữ liệu liên tục và thời gian thực:** Mạng cảm biến cho phép thu thập dữ liệu liên tục từ môi trường hoặc các nguồn cảm biến khác, giúp cung cấp thông tin thời gian thực về sự biến đổi của môi trường.
- **Khả năng tiết kiệm năng lượng:** Nhiều ứng dụng mạng cảm biến hoạt động trên nguồn năng lượng có hạn như pin hoặc năng lượng mặt trời. Mạng cảm biến được thiết kế để tiết kiệm năng lượng và kéo dài tuổi thọ của nguồn năng lượng.
- **Tự tổ chức và tự điều chỉnh:** Mạng cảm biến có khả năng tự tổ chức và tự điều chỉnh. Các nút cảm biến có thể tham gia hoặc rời khỏi mạng mà không làm gián đoạn hoạt động chung . Họ có thể tìm hiểu vị trí của mình và tối ưu hóa định tuyến dữ liệu.
- **Khả năng phát hiện và tự động báo lỗi:** Mạng cảm biến có thể phát hiện lỗi và sự cố tự động, và thậm chí tự động thay thế các nút cảm biến bị hỏng để duy trì tính liên tục của mạng.
- **Triển khai phân tán:** Mạng cảm biến có thể triển khai ở nhiều vị trí khác nhau trong môi trường, cho phép theo dõi và kiểm soát một khu vực rộng lớn hoặc phức tạp.
- **Giá thành thấp:** Các nút cảm biến có thể được sản xuất với giá thành thấp, do đó, có thể triển khai trong số lượng lớn để mục đích theo dõi và thu thập dữ liệu trên diện rộng.
- **Ứng dụng đa dạng:** Mạng cảm biến có nhiều ứng dụng rộng rãi trong các lĩnh vực như quản lý môi trường, giám sát nông nghiệp, y tế, quản lý năng lượng, an ninh, và nhiều ngành công nghiệp khác.

- **Cung cấp thông tin chi tiết:** Mạng cảm biến cung cấp thông tin chi tiết và dữ liệu số hóa, giúp người dùng hiểu rõ hơn về môi trường và các sự kiện diễn ra trong thời gian thực.
- **Tích hợp với hệ thống thông tin đám mây:** Dữ liệu từ mạng cảm biến có thể được truyền tải lên hệ thống thông tin đám mây để lưu trữ, phân tích và chia sẻ dễ dàng.

Những ưu điểm này đã làm cho mạng cảm biến trở thành một công nghệ quan trọng trong việc giải quyết nhiều vấn đề quan trọng trong thế giới hiện đại và mang lại lợi ích đáng kể cho các ứng dụng khác nhau.

1.2.2. Nhược điểm của mạng cảm biến:

- **Tiêu thụ năng lượng cao:** Một số ứng dụng mạng cảm biến yêu cầu sự liên tục trong việc thu thập dữ liệu, và việc truyền dữ liệu từ nút cảm biến đến trạm cơ sở có thể tiêu tốn nhiều năng lượng. Điều này có thể làm giảm tuổi thọ của nguồn năng lượng, đặc biệt là khi sử dụng pin.
- **Thiết bị có hạn:** Các nút cảm biến thường có tài nguyên hạn chế như bộ nhớ, xử lý và năng lượng, điều này giới hạn khả năng xử lý và lưu trữ dữ liệu trên thiết bị.
- **Lỗi và sự cố trong mạng:** Mạng cảm biến có thể gặp phải lỗi và sự cố, bao gồm lỗi phát hiện cảm biến, lỗi định tuyến dữ liệu, và lỗi kết nối. Việc quản lý và duy trì mạng có thể trở nên phức tạp.
- **Bảo mật và quyền riêng tư:** Dữ liệu thu thập bởi các mạng cảm biến thường là nhạy cảm và cần được bảo vệ. Rủi ro về bảo mật và việc xâm nhập vào mạng có thể xảy ra, đặc biệt khi các biện pháp bảo mật không đủ mạnh.
- **Hạn chế về khoảng cách truyền tải:** Tầm hoạt động của các nút cảm biến không thể vượt quá khoảng cách truyền tải tối đa của các kỹ thuật không dây như Wi-Fi hoặc Bluetooth. Điều này có thể giới hạn khả năng triển khai trong các ứng dụng có phạm vi rộng.
- **Phản ứng chậm:** Trong một số trường hợp, mạng cảm biến có thể có thời gian phản ứng chậm do việc thu thập dữ liệu và truyền tải thông tin qua nhiều nút trung gian.

- **Quản lý dữ liệu lớn**: Mạng cảm biến có thể tạo ra lượng dữ liệu lớn và đòi hỏi các hệ thống quản lý dữ liệu mạnh mẽ để lưu trữ, xử lý và phân tích thông tin.
- **Chi phí triển khai và duy trì**: Triển khai và duy trì mạng cảm biến có thể đòi hỏi chi phí đầu tư đáng kể, bao gồm việc mua sắm và lắp đặt cảm biến, quản lý mạng, và thay thế các nút bị hỏng.

Nhược điểm này cần được xem xét cẩn thận khi lên kế hoạch triển khai mạng cảm biến và chọn công nghệ phù hợp để đảm bảo hiệu suất và tính bảo mật của mạng.

1.3. Ứng dụng của mạng cảm biến

Mạng cảm biến (Sensor Network) là một hệ thống gồm nhiều cảm biến phân tán và kết nối với nhau thông qua mạng truyền thông. Ứng dụng của mạng cảm biến là rất đa dạng và lan rộng, có ảnh hưởng đến nhiều lĩnh vực khác nhau. Dưới đây là một số ứng dụng phổ biến của mạng cảm biến:

- **Quản lý Môi trường:**

o Theo dõi Nhiệt độ và Độ ẩm: Sử dụng cảm biến để giám sát nhiệt độ và độ ẩm trong các môi trường như nhà máy, nhà kho, hoặc nơi lưu trữ thực phẩm.

o Đo lường Chất lượng không khí: Cảm biến khí có thể được triển khai để theo dõi chất lượng không khí trong các thành phố hoặc khu vực công nghiệp.

- **Quản lý Năng lượng:**

o Tối ưu hóa Tiêu thụ Năng lượng: Sử dụng cảm biến để đo lường và quản lý tiêu thụ năng lượng trong các hệ thống như nhà máy, văn phòng, và ngôi nhà thông minh.

- **Giám sát và Kiểm soát Giao thông:**

o Theo dõi Lưu lượng Giao thông: Các cảm biến có thể được đặt ở các điểm quan trọng để theo dõi lưu lượng giao thông và cung cấp thông tin quan trọng cho việc quản lý giao thông.

- **Y tế và Chăm sóc Sức khỏe:**

o Theo dõi Dấu vết Sức khỏe: Sử dụng cảm biến để theo dõi các dấu vết sức khỏe của bệnh nhân và cung cấp thông tin đối với các chăm sóc y tế từ xa hoặc theo dõi các thông số y tế cơ bản.

- **Quản lý Nước**:

o Theo dõi và Dự báo Mực nước: Sử dụng mạng cảm biến để theo dõi mực nước trong các hồ, sông, và các nguồn nước khác để đưa ra dự báo và quản lý tài nguyên nước.

- **An ninh và Quản lý Thông tin:**

o Giám sát An ninh: Các cảm biến có thể được sử dụng để giám sát và phát hiện các sự cố an ninh trong các khu vực quan trọng như cảng biển, sân bay, hoặc các khu vực quân sự.

- **Nông nghiệp thông minh:**

o Theo dõi Điều kiện Nông nghiệp: Sử dụng cảm biến để đo lường độ ẩm đất, nhiệt độ, và các thông số khác để cung cấp thông tin hữu ích cho nông dân và quản lý nông nghiệp.

- **Quản lý Rác thải:**

o Theo dõi Lượng Rác thải: Sử dụng cảm biến để theo dõi lượng rác thải trong các thùng rác và tối ưu hóa quá trình thu gom rác.

Những ứng dụng này chỉ là một số ví dụ và có thể được mở rộng và tùy chỉnh theo nhu cầu cụ thể của từng lĩnh vực và ứng dụng cụ thể. Mạng cảm biến đóng vai trò quan trọng trong việc cung cấp thông tin chính xác và theo thời gian thực, góp phần quan trọng vào sự tiến bộ và hiệu quả của nhiều hệ thống và ngành công nghiệp

1.4. Ứng dụng của mạng cảm biến đo độ ẩm và nhiệt độ

Mạng cảm biến đo độ ẩm và nhiệt độ có rất nhiều ứng dụng quan trọng trong nhiều lĩnh vực và sự linh hoạt của mạng cảm biến đo độ ẩm và nhiệt độ là vô hạn. Việc sử dụng cảm biến này giúp cải thiện hiệu suất, tiết kiệm tài nguyên, và đảm bảo an toàn và an ninh trong nhiều lĩnh vực khác nhau.

Dưới đây là một số ứng dụng chính:

- **Nông nghiệp Thông minh:**

o Quản lý Tưới tiêu: Mạng cảm biến có thể giúp nông dân theo dõi độ ẩm đất và nhiệt độ, từ đó tối ưu hóa lịch trình tưới tiêu, giúp tiết kiệm nước và năng lượng.

o Dự báo Mùa vụ: Theo dõi độ ẩm và nhiệt độ trong lòng đất để dự báo mùa vụ nông sản, giúp nông dân lên kế hoạch và quản lý tốt hơn.

Hình 2.1. Hệ thống trang trại thông minh

- **Quản lý Môi trường và Rừng:**

o Giám sát Điều kiện Rừng**:** Sử dụng cảm biến để đo độ ẩm và nhiệt độ trong rừng, cung cấp thông tin quan trọng để quản lý và bảo vệ môi trường tự nhiên.

o Kiểm soát hỏa hoạn: Mạng cảm biến có thể cung cấp dữ liệu liên tục về độ ẩm và nhiệt độ, giúp dự báo và phòng tránh hỏa hoạn trong các khu vực rừng.

- **Quản lý Hệ thống Thủy lợi:**

o Theo dõi Độ ẩm Đất: Đo độ ẩm đất và nhiệt độ để giúp quản lý hệ thống thủy lợi, đảm bảo sự phân phối nước hiệu quả và giảm thiểu lãng phí.

- **Công nghiệp Năng lượng:**

o Giám sát Nhiệt độ Máy móc: Sử dụng cảm biến để theo dõi nhiệt độ của máy móc và thiết bị trong các nhà máy sản xuất năng lượng, giúp ngăn chặn quá trình quá nhiệt và hỏng hóc.

o Quản lý Hệ thống Năng lượng: Theo dõi nhiệt độ và độ ẩm trong các hệ thống điều hòa không khí và hệ thống làm mát, giúp tối ưu hóa hiệu suất và tiết kiệm năng lượng.

- **Y tế và Chăm sóc Sức khỏe:**

o Theo dõi Môi trường Bệnh viện: Đo độ ẩm và nhiệt độ trong các phòng bệnh để đảm bảo môi trường thuận lợi cho sức khỏe bệnh nhân.

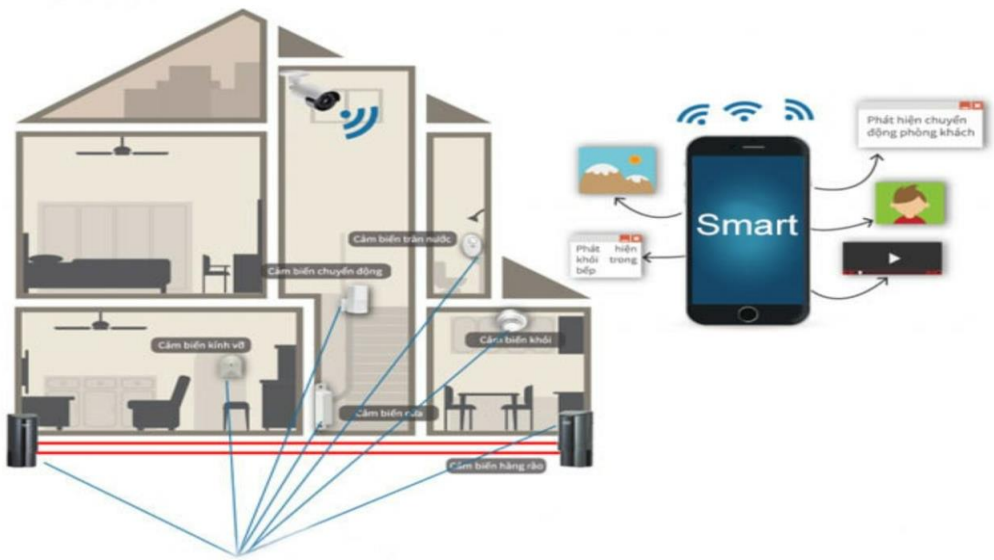
• **Quản lý Chuỗi Cung ứng Thực phẩm:**

o Kiểm soát Nhiệt độ và Độ ẩm: Mạng cảm biến có thể theo dõi điều kiện lưu trữ thực phẩm trong các kho hàng và vận chuyển, giúp đảm bảo chất lượng và an toàn thực phẩm.

Hình 3.1. Hệ thống giám sát nhiệt độ, độ ẩm trong môi trường sản xuất thực phẩm

• **Ngôi nhà Thông minh:**

o Điều khiển Hệ thống Nhiệt: Theo dõi nhiệt độ và độ ẩm trong nhà để tự động điều chỉnh hệ thống sưởi và làm mát, tạo ra môi trường số thoải mái và tiết kiệm năng lượng.



Hình 3.1. Hệ thống nhà thông minh

• **Quản lý Môi trường Đô thị:**

○ Giám sát Nhiệt độ Đô thị: Sử dụng mạng cảm biến để theo dõi nhiệt độ và độ ẩm trong các khu vực đô thị, hỗ trợ quản lý môi trường và dự báo thời tiết đô thị.

Những ứng dụng này chỉ là một số ví dụ, và sự linh hoạt của mạng cảm biến đo độ ẩm và nhiệt độ là vô hạn. Việc sử dụng cảm biến này giúp cải thiện hiệu suất, tiết kiệm tài nguyên, và đảm bảo an toàn và an ninh trong nhiều lĩnh vực khác nhau.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu về cảm biến nhiệt độ, độ ẩm (DHT11)

2.1.1. Module DHT11

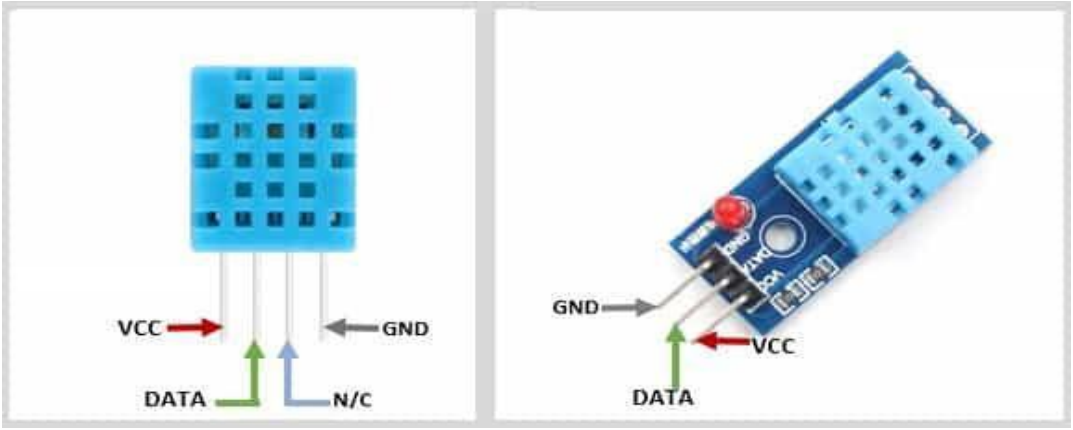
Module DHT11 là một cảm biến kỹ thuật số để cảm nhận nhiệt độ và độ ẩm. Cảm biến này có thể dễ dàng giao tiếp với bất kỳ bộ vi điều khiển vi nào như Arduino, Raspberry Pi, ... để đo độ ẩm và nhiệt độ ngay lập tức.

Module DHT11 để đo không khí xung quanh, cảm biến này sử dụng một điện trở nhiệt và một cảm biến độ ẩm điện dung.

2.1.2. Cấu tạo cảm biến nhiệt độ độ ẩm Module DHT11

2.1.2.1. Thông số kỹ thuật DHT11

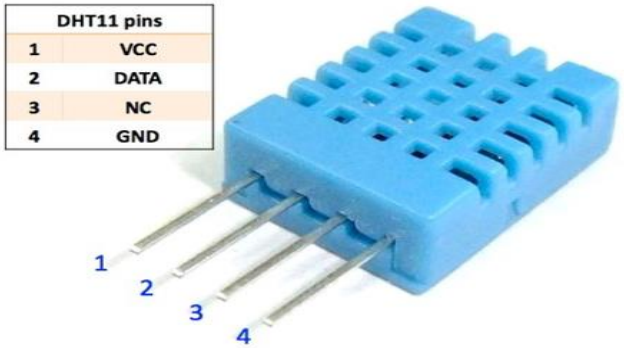
- Điện áp hoạt động: 3V - 5V DC
- Dòng điện tiêu thụ: 2.5mA
- Phạm vi cảm biến độ ẩm: 20% - 90% RH, sai số ±5%RH
- Phạm vi cảm biến nhiệt độ: 0°C ~ 50°C, sai số ±2°C
- Tần số lấy mẫu tối đa: 1Hz (1 giây 1 lần)
- Kích thước: 23 * 12 * 5 mm



Hình 1.2. Cảm biến nhiệt độ, độ ẩm DHT11

2.1.2.2. Datasheet sensor DHT11

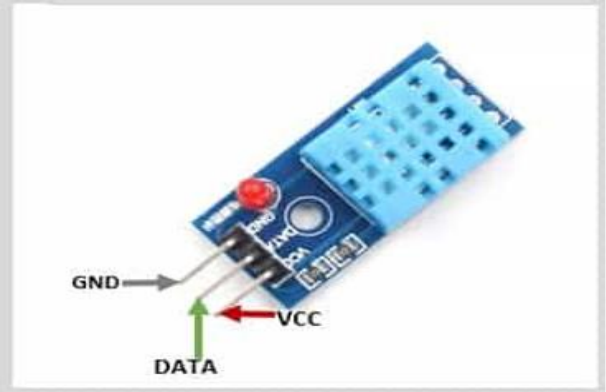
Số chân	Tên chân	Chức năng
1	VCC	Cấp nguồn cho cảm biến 3.3VDC – 5VDC
2	Data	Đầu ra tín hiệu của cảm biến
3	NC	Không có tín hiệu
4	Ground	Nối đất



Hình 2.2. Datasheet cảm biến DHT11

2.1.2.3. Datasheet module sensor DHT11

Số chân	Tên chân	Chức năng
1	GND	Nối đất
2	Data	Đầu ra tín hiệu của cảm biến
3	VCC	Cấp nguồn cho cảm biến 3.3VDC – 5VDC



Hình 3.2. Datasheet module cảm biến DHT11

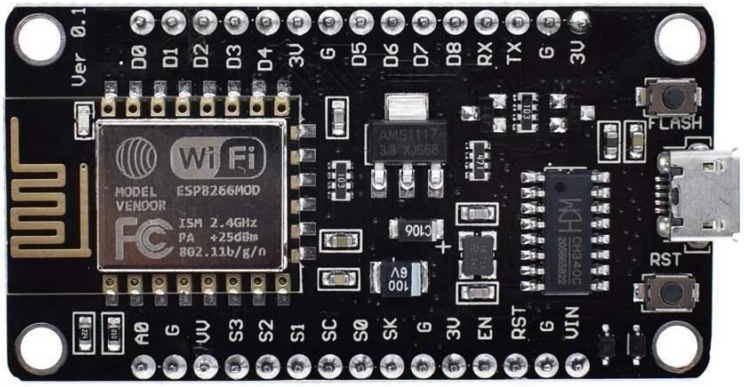
2.1.3. Các ứng dụng

- Đo nhiệt độ và độ ẩm
- Đài thời tiết địa phương

- Kiểm soát khí hậu tự động
- Giám sát môi trường

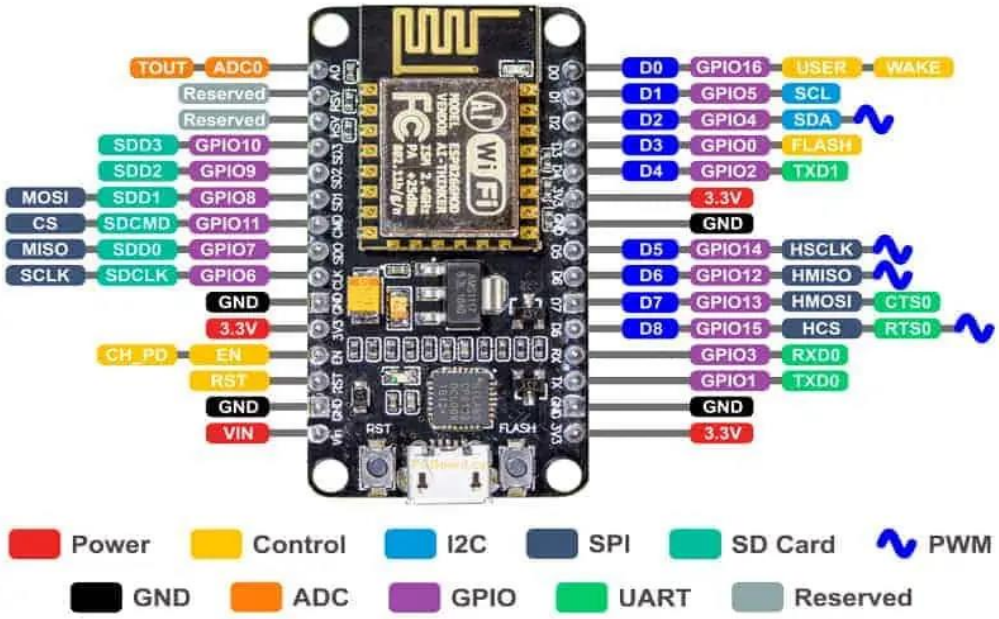
2.2. Giới thiệu về module wifi ESP8266

ESP8266 là một module Wi-Fi với khả năng kết nối Internet và được tích hợp sẵn trên một số board nhúng như NodeMCU, Wemos, và ESP-01. ESP8266 có thể hoạt động như một điểm truy cập (access point), một client kết nối đến một điểm truy cập khác, hoặc cả hai đều được. Nó được sử dụng rộng rãi trong các ứng dụng IoT (Internet of Things) như cảm biến thông minh, hệ thống kiểm soát thiết bị, hoặc các ứng dụng điều khiển từ xa. Module này có giá thành rẻ và rất dễ sử dụng, cùng với đó là khả năng tương thích với nhiều loại vi điều khiển khác nhau.



Hình 4.2. Vi điều khiển ESP8266

2.2.1. Datasheet ESP826



Hình 5.2. Datasheet ESP8266

2.2.2. Thông số kỹ thuật :

- Microcontroller: ESP8266EX
- Điện áp hoạt động: 3.3V DC
- Số chân I/O: 17 chân GPIO
- Kết nối mạng: WiFi 802.11 b/g/n
- Giao diện mạng: TCP/IP
- Đồng hồ thời gian thực (RTC): không tích hợp
- Bộ nhớ trong: 4MB
- RAM: 80KB
- Cổng nạp: Micro-USB
- Hỗ trợ các giao thức: MQTT, CoAP, HTTP/HTTPS
- Kích thước: 49 x 24.5 x 13mm

2.2.3. Thiết bị ngoại vi và chân I/O ESP8266

NodeMCU ESP8266 là một module IoT dựa trên vi điều khiển ESP8266. Nó được tích hợp sẵn các chân I/O (Input/Output) và hỗ trợ các thiết bị ngoại vi để kết nối và tương tác với các linh kiện và cảm biến khác. Dưới đây là một số thiết bị ngoại vi và các chân I/O quan trọng trên ESP8266 NodeMCU:

17 chân GPIO	Được sử dụng để đọc dữ liệu từ các cảm biến, điều khiển các thiết bị đầu ra, hoặc giao tiếp với các thiết bị khác như LED, động cơ, nút nhấn, v.v.
1 kênh ADC	1 kênh ADC có độ chính xác 10 bit theo công nghệ SAR ADC.
2 giao tiếp UART	2 giao tiếp UART hỗ trợ điều khiển dòng dữ liệu.
4 đầu ra PWM	4 chân PWM để điều khiển tốc độ động cơ hoặc độ sáng của đèn LED.

2 giao tiếp SPI và 1 giao tiếp I2C	2 giao tiếp SPI và một giao tiếp I2C để kết nối các cảm biến và thiết bị ngoại vi khác.
Giao tiếp I2S	Một giao tiếp I2S để thêm âm thanh vào dự án của bạn.

2.3. Các công cụ tạo lập trong chương trình

2.3.1. NodeJS

2.3.1.1. Giới thiệu

NodeJS là một mã nguồn mở, đa nền tảng, chạy trên môi trường JavaSript, được xây dựng trên V8 JavaScript engine của Chrome - V8 thực thi mã JavaScript bên ngoài trình duyệt. Nó được tạo ra vào năm 2009 đi kèm với một lợi thế chính - NodeJS cho phép thực hiện lập trình bất đồng bộ.

NodeJS là một nền tảng được xây dựng trên JavaScript runtime của Chrome với mục đích xây dựng các ứng dụng mạng nhanh chóng và có thể mở rộng được một cách dễ dàng hơn. NodeJS sử dụng mô hình I/O lập trình theo sự kiện, non-blocking, do đó nodeJS khá gọn nhẹ và hiệu quả - công cụ hoàn hảo cho các ứng dụng chuyên sâu về dữ liệu theo thời gian thực chạy trên các thiết bị phân tán.

NodeJS là môi trường runtime mã nguồn mở đa nền tảng, được sử dụng để phát triển các ứng dụng mạng và ứng dụng server-side. Các ứng dụng NodeJS được viết bằng JavaScript và có thể chạy trong NodeJS runtime trên OS X, Microsoft Windows và Linux.

NodeJS cũng cung cấp một thư viện bao gồm rất nhiều các module JavaScript khác nhau nhằm đơn giản hóa việc phát triển các ứng dụng web, qua đó giảm thiểu tình trạng sử dụng quá nhiều Node.js.

2.3.1.2. Các tính năng

- Lập trình hướng sự kiện và không đồng bộ: Toàn bộ API trong thư viện NodeJS đều không đồng bộ, hay không bị chặn. Về cơ bản điều này có nghĩa là một server sử dụng NodeJS sẽ không bao giờ chờ một API trả về dữ liệu. Server sẽ chuyển sang API kế tiếp sau khi gọi API đó và cơ chế thông báo của Events trong NodeJS giúp server nhận được phản hồi từ lần gọi API trước.

• Cực kỳ nhanh chóng: Được xây dựng trên Công cụ JavaScript V8 của Google Chrome, thư viện NodeJS có khả năng xử lý mã vô cùng nhanh.

• Đơn luồng/Single thread nhưng có khả năng mở rộng cao: NodeJS sử dụng một mô hình luồng đơn với vòng lặp sự kiện/event. Cơ chế event cho phép máy chủ phản hồi non-blocking và cũng cho phép khả năng mở rộng cao hơn so với các server truyền thống hỗ trợ giới hạn các thread để xử lý yêu cầu. NodeJS sử dụng một chương trình đơn luồng, cùng một chương trình có thể cung cấp dịch vụ cho một số lượng yêu cầu lớn hơn so với các máy chủ truyền thống như Apache HTTP Server.

• Không có buffer - Các ứng dụng NodeJS không có vùng nhớ tạm thời (buffer) cho bất kỳ dữ liệu nào. Các ứng dụng này chỉ đơn giản xuất dữ liệu theo khối.

• License - NodeJS được phát hành theo giấy phép MIT.

2.3.2. Sequelize

Sequelize là một ORM (Object-Relational Mapping) cho NodeJs, giúp quản lý và tương tác với cơ sở dữ liệu quan hệ (relational database). Sequelize hỗ trợ nhiều loại cơ sở dữ liệu phổ biến như MySQL, PostgreSQL, SQLite, và MSSQL.

Tổng quan về Sequelize trong NodeJs:

• ORM (Object-Relational Mapping): Sequelize cho phép bạn tương tác với cơ sở dữ liệu bằng cách sử dụng các đối tượng JavaScript thay vì truy vấn SQL trực tiếp. Điều này giúp giảm sự phức tạp và làm cho việc làm việc với cơ sở dữ liệu trở nên dễ dàng hơn.

• Hỗ trợ nhiều loại cơ sở dữ liệu: Sequelize hỗ trợ nhiều loại cơ sở dữ liệu phổ biến như MySQL, PostgreSQL, SQLite, và MSSQL, giúp bạn có thể chuyển đổi giữa các hệ thống cơ sở dữ liệu một cách dễ dàng.

• Model Definition: Sequelize cho phép định nghĩa các mô hình (models) để mô tả cấu trúc của bảng trong cơ sở dữ liệu. Mỗi model tương ứng với một bảng trong cơ sở dữ liệu.

• Querying: Sequelize cung cấp các phương thức để thực hiện các truy vấn đến cơ sở dữ liệu mà không cần viết SQL trực tiếp. Điều này giúp giảm khả năng phát sinh lỗi và làm tăng khả năng duy trì mã nguồn.

• Associations: Sequelize hỗ trợ định nghĩa các mối quan hệ giữa các bảng (associations), chẳng hạn như mối quan hệ một-nhiều, nhiều-nhiều, và mối quan hệ con-trỏ.

• Migrations: Sequelize có hỗ trợ cho việc quản lý phiên bản cơ sở dữ liệu thông qua migrations. Migrations là cách để đảm bảo rằng cơ sở dữ liệu của bạn được duy trì và cập nhật đồng bộ với các thay đổi trong mã nguồn.

• Hooks và Validators: Sequelize hỗ trợ các hooks (hooks là các hàm được chạy trước hoặc sau khi thực hiện một sự kiện nhất định) và validators (kiểm tra ràng buộc dữ liệu) giúp bạn xử lý các sự kiện và đảm bảo tính nhất quán của dữ liệu.

2.3.3. Socket.IO

Socket.IO là một thư viện JavaScript để phát triển ứng dụng thời gian thực (real-time) thông qua kết nối hai chiều giữa máy khách (client) và máy chủ (server). Nó hoạt động trên nền tảng Node.js và cung cấp một giao diện thuận tiện cho việc xây dựng ứng dụng web thời gian thực.

Tổng quan về Socket.IO:

• Real-time Communication: Socket.IO cho phép truyền dữ liệu theo thời gian thực giữa máy khách và máy chủ. Thay vì phải gửi yêu cầu HTTP để lấy dữ liệu mới, Socket.IO cho phép truyền dữ liệu ngay lập tức khi có sự thay đổi.

• WebSocket và Fallbacks: Socket.IO sử dụng WebSocket nếu trình duyệt và máy chủ hỗ trợ, điều này giúp giảm độ trễ và tăng hiệu suất. Trong trường hợp không hỗ trợ WebSocket, Socket.IO sẽ tự động chuyển sang các kỹ thuật giả mạo như AJAX long polling.

• Sự Kiện (Events): Socket.IO thực hiện truyền dữ liệu thông qua sự kiện. Máy khách và máy chủ có thể kích hoạt và lắng nghe các sự kiện, giúp tạo ra một cơ chế giao tiếp mạnh mẽ.

• Rooms và Namespaces: Socket.IO hỗ trợ việc phân chia người dùng vào các phòng (rooms) và namespaces, giúp quản lý sự kiện và dữ liệu cho các nhóm cụ thể.

• Middleware: Socket.IO có khả năng sử dụng middleware, tương tự như Express.js. Điều này cho phép bạn thực hiện các xử lý trung gian trước khi dữ liệu được gửi hoặc sau khi được nhận.

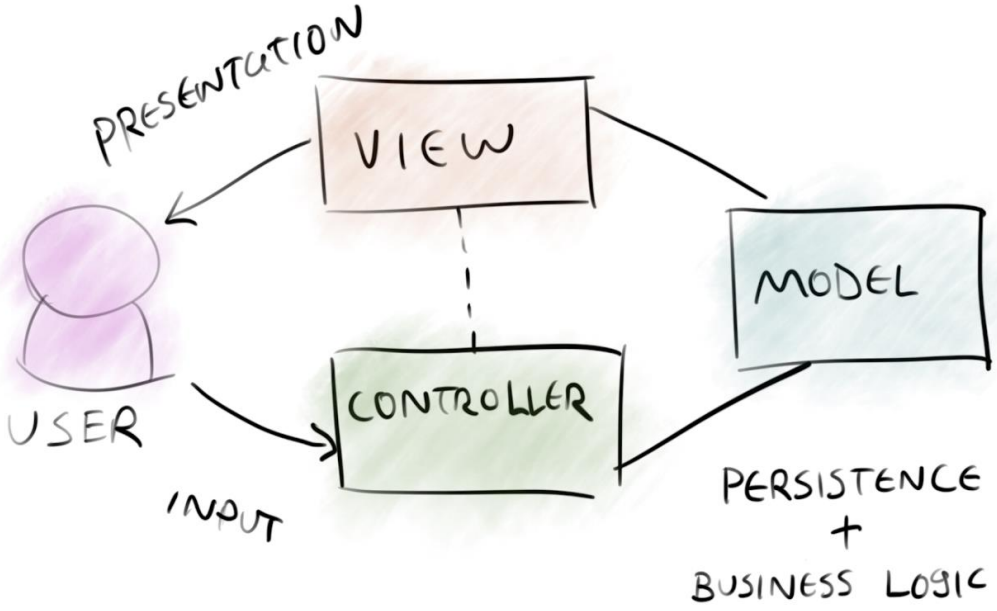
• Hỗ trợ nhiều ngôn ngữ và nền tảng: Socket.IO không chỉ hỗ trợ JavaScript trên máy khách và máy chủ, mà còn hỗ trợ nhiều ngôn ngữ khác như Python, Java, và C#.

2.3.4. Mô hình MVC

2.3.4.1. Giới thiệu

Mô hình Model-View-Controller (MVC) là một mẫu kiến trúc phân tách một ứng dụng thành ba thành phần logic chính Model, View và Controller. Do đó viết tắt MVC. Mỗi thành phần kiến trúc được xây dựng để xử lý khía cạnh phát triển cụ thể của một ứng dụng. MVC tách lớp logic nghiệp vụ và lớp hiển thị ra riêng biệt. Ngày nay, kiến trúc MVC đã trở nên phổ biến để thiết kế các ứng dụng web cũng như ứng dụng di động.

2.3.4.2. Kiến trúc



Hình 6.2. Mô hình MVC

• **Model:** Thành phần model lưu trữ dữ liệu và logic liên quan của nó. Bao gồm các class function xử lý các tác vụ như truy vấn, thêm, sửa hoặc xóa dữ liệu. Ví dụ, một đối tượng Controller sẽ lấy thông tin khách hàng từ cơ sở dữ liệu. Nó thao tác dữ liệu và gửi trở lại cơ sở dữ liệu hoặc sử dụng nó để hiển thị dữ liệu.

• **View:** Trình bày dữ liệu cho người dùng hoặc xử lý tương tác của người dùng.

○ View là một phần của ứng dụng đại diện cho việc trình bày dữ liệu.

o View được tạo bởi các dữ liệu mà chúng ta lấy từ dữ liệu trong model. Một view yêu cầu model cung cấp đầy đủ dữ liệu để nó hiển thị đầu ra cho người dùng.

o View chính là nới chứa những giao diện như một nút bấm, khung nhập, menu, hình ảnh… nó đảm nhiệm nhiệm vụ hiển thị dữ liệu và giúp người dùng tương tác với hệ thống.

- **Controller:** Là phần quan trọng nhất trong mô hình, nó liên kết phần Model và View.

o Controller là một phần của ứng dụng xử lý tương tác của người dùng. Bộ điều khiển diễn giải đầu vào chuột và bàn phím từ người dùng, thông báo cho model và view để thay đổi khi thích hợp.

o Controller là nới tiếp nhận những yêu cầu xử lý được gửi từ người dùng, nó sẽ gồm những class/ function xử lý nhiều nghiệp vụ logic giúp lấy đúng dữ liệu thông tin cần thiết nhờ các nghiệp vụ lớp Model cung cấp và hiển thị dữ liệu đó ra cho người dùng nhờ lớp View.

o Controller gửi các lệnh đến model để làm thay đổi trạng thái của nó (Ví dụ: ta thêm mới 1 user hoặc cập nhật tên 1 user). Controller cũng gửi các lệnh đến view liên quan của nó để thay đổi cách hiển thị của view (Ví dụ: xem thông tin 1 user).

2.3.4.3. Cách thức tương tác

- Controller tương tác với qua lại với View.
- Controller tương tác qua lại với Model.
- Model và View không có sự tương tác với nhau trực tiếp mà nó tương tác với nhau thông qua Controller.

CHƯƠNG 3: XÂY DỰNG DEMO GIAO DIỆN IOT ĐƠN GIẢN THEO DÕI NHIỆT ĐỘ VÀ ĐỘ ẨM

3.1. Linh kiện phần cứng

- Vi xử lý ESP32
- Module DHT11
- Dây kết nối chân
- Dây cáp USB Micro.
- Laptop.

3.2. Thiết kế phần mềm

3.2.1. Phần mềm Arduino IDE

Arduino IDE là một phần mềm với một mã nguồn mở, được sử dụng chủ yếu để viết và biên dịch mã vào module Arduino. Nó bao gồm phần cứng và phần mềm. Phần cứng chứa đến 300,000 board mạch được thiết kế sẵn với các cảm biến, linh kiện. Phần mềm giúp bạn có thể sử dụng các cảm biến, linh kiện ấy của Arduino một cách linh hoạt phù hợp với mục đích sử dụng.

Tổng quan về các chức năng chính của Arduino IDE:

- **Editor Code:** Arduino IDE cung cấp một trình soạn thảo mã nguồn để viết mã Arduino. Người dùng có thể viết mã trong ngôn ngữ lập trình C/C++ và sử dụng các thư viện Arduino.

- **Biên Dịch và Nạp Chương Trình:**

o Sau khi viết mã, người dùng có thể sử dụng Arduino IDE để biên dịch mã nguồn thành mã máy tương ứng với bo mạch Arduino sử dụng trình biên dịch AVR-GCC.

o Arduino IDE cũng hỗ trợ việc nạp chương trình đã biên dịch vào bo mạch Arduino thông qua cổng USB.

- **Quản lý Thư Viện:** Phần mềm này đi kèm với một trình quản lý thư viện, giúp người dùng dễ dàng thêm, xóa và cập nhật thư viện Arduino. Thư viện là những đoạn mã được viết trước có sẵn để giúp người dùng thực hiện các chức năng cụ thể mà không cần phải viết lại từ đầu.

• **Giả Lập Bo Mạch:** Arduino IDE cung cấp khả năng giả lập bo mạch, cho phép người dùng thử nghiệm chương trình trước khi chạy trên bo mạch thực tế.

• **Môi Trường Đa Nền Tảng:** Arduino IDE hỗ trợ đa nền tảng, nghĩa là bạn có thể cài đặt và sử dụng nó trên nhiều hệ điều hành như Windows, macOS và Linux.

• **Cộng Đồng Hỗ Trợ:** Arduino có một cộng đồng lớn và tích cực. Arduino IDE kết nối với cộng đồng này, giúp người dùng dễ dàng chia sẻ mã nguồn, thảo luận và tìm kiếm hỗ trợ.

• **Dự Án Open Source**: Phần mềm Arduino IDE là một dự án mã nguồn mở (open source), cho phép người dùng tự do sửa đổi và tùy chỉnh theo nhu cầu của họ.

Tóm lại, Arduino IDE là một công cụ quan trọng và hiệu quả cho những người làm việc với bo mạch Arduino, giúp họ dễ dàng phát triển ứng dụng và thực hiện các dự án điện tử.

3.2.2. Phần mềm Visual studio code

Visual Studio Code chính là ứng dụng cho phép biên tập, soạn thảo các đoạn code để hỗ trợ trong quá trình thực hiện xây dựng, thiết kế website một cách nhanh chóng. Visual Studio Code hay còn được viết tắt là VS Code. Trình soạn thảo này vận hành mượt mà trên các nền tảng như Windows, macOS, Linux. Hơn thế nữa, VS Code còn cho khả năng tương thích với những thiết bị máy tính có cấu hình tầm trung vẫn có thể sử dụng dễ dàng.

Hình 3.1. Visual Studio Code

Visual Studio Code hỗ trợ đa dạng các chức năng Debug, đi kèm với Git, có Syntax Highlighting. Đặc biệt là tự hoàn thành mã thông minh, Snippets, và khả năng cải tiến mã nguồn. Nhờ tính năng tùy chỉnh, Visual Studio Code cũng cho phép các

lập trình viên thay đổi Theme, phím tắt, và đa dạng các tùy chọn khác. Mặc dù trình soạn thảo Code này tương đối nhẹ, nhưng lại bao gồm các tính năng mạnh mẽ.

VSCode là một trong những Code Editor mạnh mẽ và phổ biến nhất dành cho lập trình viên. Nhờ hỗ trợ nhiều ngôn ngữ lập trình phổ biến, tích hợp đầy đủ các tính năng và khả năng mở rộng, nên VSCode trở nên cực kì thân thuộc với bất kì lập trình viên nào.

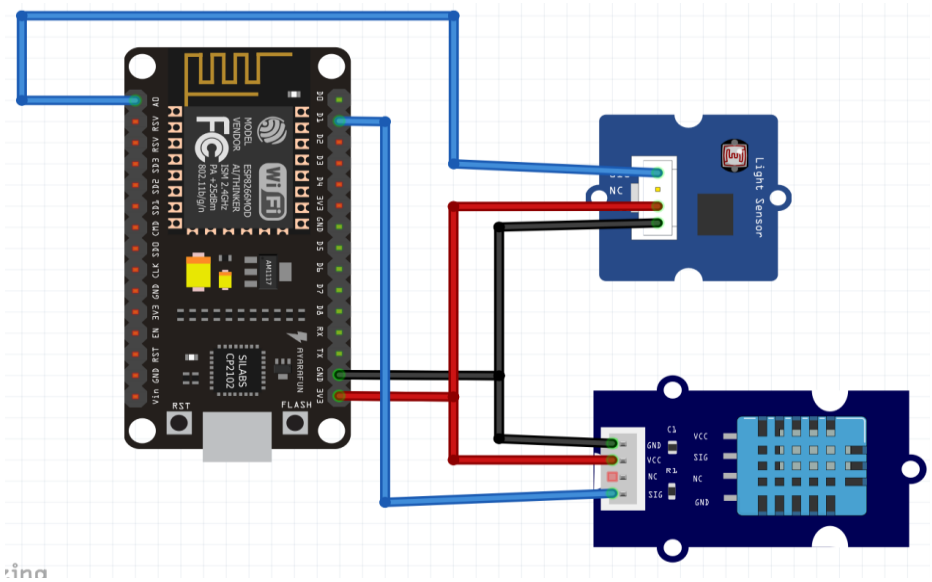
Một số đặc điểm quan trọng của Visual Studio Code:

- Hỗ Trợ Đa Ngôn Ngữ: VSCode cung cấp hỗ trợ cho nhiều ngôn ngữ lập trình, bao gồm C++, Java, Python, JavaScript, HTML, CSS, và nhiều ngôn ngữ khác nữa.
- Mở Rộng và Linh Hoạt: VSCode có một hệ thống mở rộng mạnh mẽ, cho phép người dùng cài đặt các tiện ích mở rộng (extensions) để mở rộng khả năng và tính năng của IDE theo nhu cầu cụ thể.
- IntelliSense: VSCode hỗ trợ IntelliSense, một tính năng giúp nhận biết và tự động hoàn thành mã nguồn, làm tăng tốc quá trình viết mã.
- Gỡ Lỗi Tích Hợp: IDE này tích hợp khả năng gỡ lỗi (debugging) cho nhiều ngôn ngữ, giúp người lập trình dễ dàng theo dõi và sửa lỗi trong mã nguồn.
- Quản Lý Phiên Bản: Visual Studio Code tích hợp tính năng quản lý phiên bản (version control) thông qua các tiện ích như Git.
- Tích Hợp Terminal: VSCode có một terminal tích hợp giúp người dùng thực hiện các lệnh hệ thống mà không cần chuyển sang môi trường dòng lệnh riêng.
- Môi Trường Đa Nền Tảng: VSCode hỗ trợ nhiều hệ điều hành, bao gồm Windows, macOS và Linux.
- Môi Trường Phát Triển Web: IDE này được thiết kế để phục vụ cả các dự án phát triển web, hỗ trợ HTML, CSS, và JavaScript.
- Cộng Đồng Lớn và Hỗ Trợ Đa Dạng: VSCode có một cộng đồng lớn và tích cực, với nhiều người đóng góp extensions và cung cấp hỗ trợ trên các diễn đàn và các nền tảng trực tuyến khác.

• Mã Nguồn Mở: Visual Studio Code là một dự án mã nguồn mở, cho phép người dùng tự do xem, sửa đổi và phân phối lại mã nguồn theo các điều khoản cấp phép của nó.

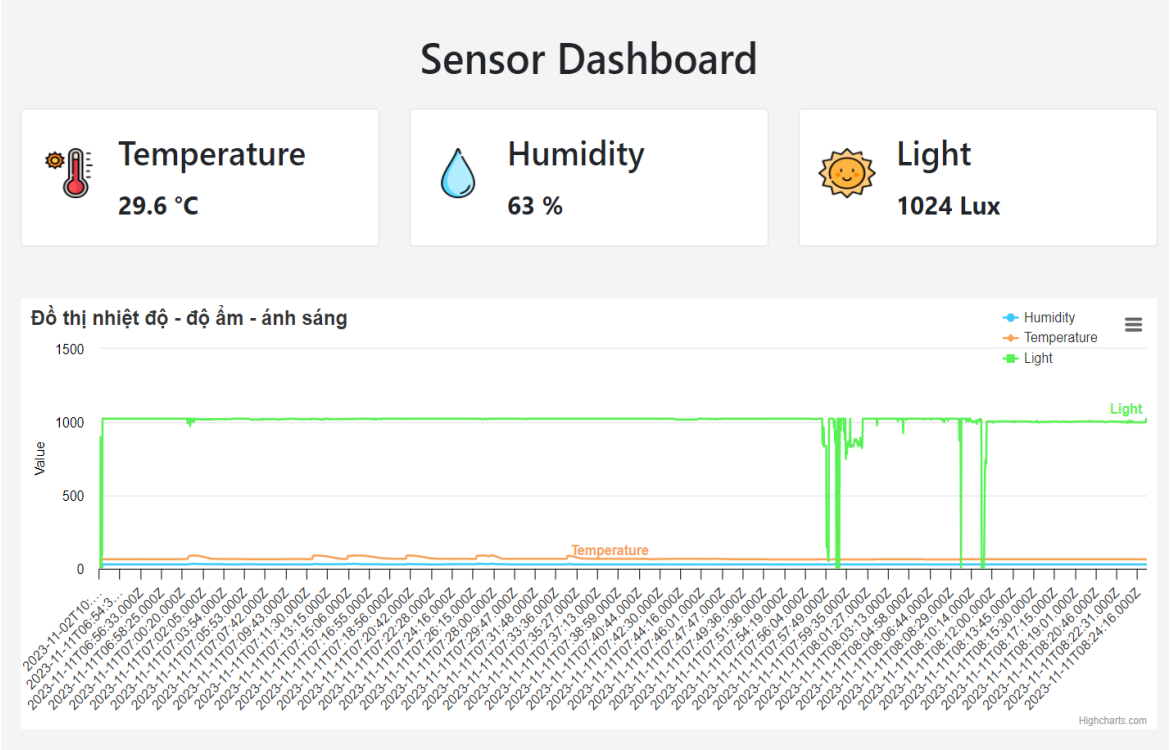
3.3. Xây dựng chương trình

3.3.1. Kết nối phần cứng



Hình 3.2. Kết nối ESP8266 và module DHT11

3.3.2. Giao diện Dashboard



Hình 3.3. Giao diện Dashboard

TÀI LIỆU THAM KHẢO

[1] https://www.theengineeringprojects.com/2019/03/introduction-to-dht11.html#:~:text=DHT11%20is%20a%20low-cost%2C%20small-sized%20%26%20easy-to-operate%20embedded,with%20%C2%B15%25%20accuracy%29%20and%20pro...20output

[2] https://sequelize.org/docs/v6/getting-started/

[3] https://www.highcharts.com/demo/highcharts/line-chart

[4] https://socket.io/docs/v4/

[5] https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf