ĐẠI HỌC BÁCH KHOA HÀ NỘI

**TRƯỜNG CÔNG NGHỆ THÔNG TIN & TUYỀN THÔNG**

----- 🙡 🕮 🙣 -----



**ĐỒ ÁN MÔN HỌC II**

***Đề tài:* Triển khai hệ thống web giám sát thông tin môi trường & điều khiển thiết bị tại vườn**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | **PGS.TS. Nguyễn Thị Hoàng Lan** |
| **Sinh viên thực hiện:** | **Nguyễn Đức Quân** |
| **Mã sinh viên:** | **20204774** |
| **Mã học phần:** | **IT3931** |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Hà Nội, năm 2024**

MỤC LỤC

[MỤC LỤC 2](#_Toc169813073)

[MỞ ĐẦU 4](#_Toc169813074)

[CHƯƠNG I. Giới thiệu đề tài và nhiệm vụ cụ thể 5](#_Toc169813075)

[**I.1. Giới thiệu đề tài** 5](#_Toc169813076)

[I.1.1. Nguyên cớ và ý tưởng 5](#_Toc169813077)

[I.1.2. Mục đích nghiên cứu 5](#_Toc169813078)

[I.1.3. Phương pháp nghiên cứu và triển khai 5](#_Toc169813079)

[I.1.4. Các lợi ích dự kiến 5](#_Toc169813080)

[I.1.5. Định hướng phát triển và mở rộng 5](#_Toc169813081)

[**I.2. Nhiệm vụ cụ thể** 6](#_Toc169813082)

[I.2.1. Hiển thị dữ liệu cảm biến 6](#_Toc169813083)

[I.2.2. Control Panel để điều khiển thiết bị từ xa 6](#_Toc169813084)

[I.2.3. Cảnh báo bất thường 6](#_Toc169813085)

[CHƯƠNG II. Tóm tắt về cơ sở lý thuyết và công nghệ sử dụng 7](#_Toc169813086)

[**II.1. Cơ sở lý thuyết** 7](#_Toc169813087)

[**II.2. Công nghệ sử dụng** 7](#_Toc169813088)

[II.2.1 Node.js & Express 8](#_Toc169813089)

[II.2.2. AWS IoT Core 8](#_Toc169813090)

[II.2.3. AWS EC2 8](#_Toc169813091)

[II.2.4. ESP32 & DHT11 9](#_Toc169813092)

[CHƯƠNG III. Phân tích thiết kế, xây dựng ứng dụng & hệ thống 10](#_Toc169813093)

[**III.1 Phân tích thiết kế** 10](#_Toc169813094)

[III.1.1 Frontend (Web Client) 10](#_Toc169813095)

[III.1.2 Backend Server 10](#_Toc169813096)

[III.1.3 Broker IoT Core 11](#_Toc169813097)

[III.1.4 Thiết bị IoT (ESP32) 11](#_Toc169813098)

[**III.2 Xây dựng ứng dụng & hệ thống** 12](#_Toc169813099)

[III.2.1 Tổng quan hệ thống 12](#_Toc169813100)

[III.2.2 Các luồng hoạt động 12](#_Toc169813101)

[CHƯƠNG IV. Cài đặt chương trình và kết quả thực nghiệm 16](#_Toc169813102)

[**IV.1 Cài đặt chương trình** 16](#_Toc169813103)

[IV.1.1 Môi trường phát triển 16](#_Toc169813104)

[IV.1.2 Quy trình phát triển 16](#_Toc169813105)

[**IV.2 Kết quả thực nghiệm** 17](#_Toc169813106)

[IV.2.1 Mục tiêu thực nghiệm 17](#_Toc169813107)

[IV.2.2 Cấu hình thực nghiệm 18](#_Toc169813108)

[IV.2.3 Tiến hành thực nghiệm 18](#_Toc169813109)

[IV.2.4 Đánh giá kết quả thực nghiệm 19](#_Toc169813110)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 20](#_Toc169813111)

[TÀI LIỆU THAM KHẢO 21](#_Toc169813112)

**DANH SÁCH HÌNH ẢNH**

[Hình 1: Sơ đồ tổng quan hệ thống 11](#_Toc169813004)

[Hình 2: Bảng điều khiển 12](#_Toc169813005)

[Hình 3: Sơ đồ cập nhật trạng thái thiết bị 13](#_Toc169813006)

[Hình 4: Email cảnh báo bất thường 14](#_Toc169813007)

**MỞ ĐẦU**

Trước sự bùng nổ của công nghệ thông tin và nhu cầu ngày càng tăng về quản lý hiệu quả nguồn tài nguyên, việc áp dụng các hệ thống thông minh vào giám sát môi trường và điều khiển thiết bị đã trở thành một xu hướng thiết thực và cần thiết. Đồ án này ra đời với mục đích triển khai một hệ thống web giám sát thông tin môi trường và điều khiển thiết bị tại vườn, nhằm cải thiện quản lý tài nguyên và giảm thiểu tác động tiêu cực đến môi trường.

* Đề tài nghiên cứu: Tập trung vào việc nghiên cứu và triển khai một hệ thống hoàn chỉnh giám sát thông tin môi trường tại vườn và điều khiển thiết bị, nhằm tăng cường khả năng quản lý và sử dụng hiệu quả nguồn tài nguyên tự nhiên.
* Phương pháp thực hiện: Việc xây dựng hệ thống sử dụng các công nghệ hiện đại bao gồm cả cảm biến môi trường, thiết bị điều khiển từ xa và nền tảng web để thu thập dữ liệu và cung cấp các chức năng điều khiển thông minh.
* Lý do lựa chọn đề tài: Đồ án được lựa chọn nhằm đáp ứng nhu cầu thực tiễn trong việc quản lý môi trường và tài nguyên tại các vườn trồng cây, giúp nâng cao hiệu quả sản xuất và bảo vệ môi trường.
* Tính thực tế của đồ án: Qua quá trình triển khai và chạy thử dự án hoàn toàn có thể rút ra được dự án hoàn toàn khả thi và sử dụng được thực tế.
* Định hướng phát triển mở rộng: Đồ án có tiềm năng phát triển mở rộng quy mô mô hình và mở rộng số lượng người dùng, nhằm cải thiện tính hiệu quả và ứng dụng của hệ thống không chỉ trong lĩnh vực vườn cây trồng mà còn trong các lĩnh vực khác như quản lý đô thị, nông nghiệp chính xác, và giám sát môi trường công nghiệp.
* Các kiến thức và kỹ năng đạt được: Thiết kế hệ thống hoàn thiện, thành thạo sử dụng các công cụ liên qua tới thiết kế, xây dựng, triển khai và vận hành hệ thống web.

**CHƯƠNG I. Giới thiệu đề tài và nhiệm vụ cụ thể**

**I.1. Giới thiệu đề tài**

Đồ án này bắt nguồn từ một nhu cầu cá nhân vô cùng cụ thể và thiết thực trong cuộc sống hiện đại - sự mong muốn kiểm soát và quản lý khu vườn nhà từ xa thông qua internet. Với sự phát triển nhanh chóng của công nghệ thông tin và truy cập internet, việc có thể giám sát và điều khiển các thiết bị từ xa không chỉ đơn giản là một ý tưởng, mà còn là một thực tế có thể áp dụng hiệu quả vào thực tiễn.

I.1.1. Nguyên cớ và ý tưởng

Việc quản lý khu vườn không phải là một công việc đơn giản, đặc biệt là khi chủ nhân cần phải đảm bảo các yếu tố như cung cấp nước, điều khiển ánh sáng và nhiệt độ, và theo dõi sự phát triển của cây trồng. Tuy nhiên, với cuộc cách mạng công nghệ hiện nay, sự kết hợp giữa cảm biến, thiết bị điều khiển và internet đã mở ra những cánh cửa mới cho việc giám sát và điều khiển từ xa, mang lại sự tiện lợi và hiệu quả cao hơn cho người dùng.

I.1.2. Mục đích nghiên cứu

Mục tiêu chính của đồ án là nghiên cứu và triển khai một hệ thống web giám sát thông tin môi trường và điều khiển thiết bị tại khu vườn. Qua đó, chúng tôi mong muốn cung cấp cho người dùng một công cụ hiệu quả để quản lý và tối ưu hóa các hoạt động trong khu vườn, từ đó cải thiện năng suất và bảo vệ môi trường.

I.1.3. Phương pháp nghiên cứu và triển khai

Đồ án sẽ sử dụng một phương pháp tiếp cận kết hợp giữa lý thuyết và thực tiễn. Từ việc nghiên cứu các công nghệ và thiết bị phù hợp cho đến việc thiết kế và triển khai hệ thống, chúng tôi sẽ đảm bảo rằng mọi giải pháp được đưa ra đều phù hợp và áp dụng được trong môi trường thực tế.

I.1.4. Các lợi ích dự kiến

Khi triển khai thành công, hệ thống sẽ giúp người dùng giảm thiểu thời gian và chi phí cho việc quản lý vườn, đồng thời cải thiện hiệu quả sử dụng tài nguyên tự nhiên và giảm thiểu tác động tiêu cực đến môi trường.

I.1.5. Định hướng phát triển và mở rộng

Ngoài việc triển khai cho khu vườn cá nhân, đồ án còn đề xuất các hướng phát triển mở rộng trong các lĩnh vực khác như nông nghiệp chính xác, quản lý đô thị thông minh, và các ứng dụng công nghiệp khác. Điều này nhằm mở rộng phạm vi ứng dụng của hệ thống và tăng cường tính ứng dụng của nó trong các ngành công nghiệp khác nhau.

**I.2. Nhiệm vụ cụ thể**

I.2.1. Hiển thị dữ liệu cảm biến

Để giám sát các điều kiện môi trường trong vườn, đồ án sử dụng cảm biến DHT để thu thập dữ liệu về nhiệt độ và độ ẩm. Dữ liệu này sau đó được hiển thị trực quan và dễ hiểu thông qua các biểu đồ thời gian thực trên giao diện web. Người dùng có thể quan sát biến động của các chỉ số này theo thời gian, từ đó có thể đưa ra các quyết định quản lý hiệu quả cho vườn cây của mình.

I.2.2. Control Panel để điều khiển thiết bị từ xa

Bên cạnh việc giám sát, hệ thống cũng cung cấp một control panel cho phép người dùng điều khiển các thiết bị từ xa. Ví dụ như bật tắt hệ thống tưới nước, điều chỉnh độ sáng của đèn chiếu sáng hoặc điều khiển hệ thống phun thuốc.

I.2.3. Cảnh báo bất thường

Để đảm bảo sự ổn định và phát hiện kịp thời các vấn đề trong môi trường vườn, hệ thống được thiết kế để tự động gửi cảnh báo qua email khi phát hiện các dữ liệu cảm biến bất thường. Các trường hợp bất thường có thể bao gồm nhiệt độ hoặc độ ẩm vượt quá ngưỡng cho phép hoặc sự cố trong việc kết nối và gửi dữ liệu. Cảnh báo sẽ được gửi đến người quản lý vườn hoặc người dùng chỉ định, giúp họ có thể can thiệp kịp thời và ngăn chặn các vấn đề tiềm ẩn.

**CHƯƠNG II. Tóm tắt về** **cơ sở lý thuyết và công nghệ sử dụng**

**II.1. Cơ sở lý thuyết**

Để giải quyết bài toán giám sát môi trường và điều khiển thiết bị từ xa trong đồ án này, chúng ta cần kết hợp và áp dụng các công nghệ phù hợp. Cụ thể, để xây dựng một hệ thống có khả năng truyền tải dữ liệu môi trường và điều khiển thiết bị một cách hiệu quả, các công nghệ sau được sử dụng và kết hợp:

1. **API (Application Programming Interface)**: API được sử dụng để cung cấp giao diện cho việc giao tiếp giữa các thành phần khác nhau của hệ thống. Trong đồ án này, API sẽ đóng vai trò quan trọng trong việc cung cấp các dịch vụ và chức năng từ server cho các client, bao gồm cả việc cung cấp dữ liệu môi trường từ các cảm biến và nhận lệnh điều khiển từ client đến thiết bị.
2. **Websocket**: Để đảm bảo việc truyền tải dữ liệu môi trường liên tục và thời gian thực từ server đến web client, websocket được sử dụng. Websocket cho phép thiết lập một kênh liên lạc hai chiều và duy trì kết nối liên tục giữa server và client, từ đó cho phép dữ liệu được truyền tải một cách hiệu quả và nhanh chóng.
3. **MQTT (Message Queuing Telemetry Transport)**: MQTT là một giao thức truyền thông nhẹ được sử dụng để truyền tải dữ liệu môi trường và các lệnh điều khiển giữa các thiết bị nhúng như ESP32 và server. Giao thức này phù hợp với các ứng dụng IoT (Internet of Things) do tính nhẹ và khả năng tiết kiệm năng lượng, đồng thời hỗ trợ độ tin cậy cao trong việc truyền tải dữ liệu.
4. **ESP32**: ESP32 là một nền tảng phổ biến cho các ứng dụng IoT nhờ vào tính năng đa năng và khả năng kết nối internet. Trong đồ án này, ESP32 sẽ được sử dụng để thu thập dữ liệu từ các cảm biến môi trường và thực hiện các hành động điều khiển thiết bị dựa trên các lệnh từ server thông qua giao thức MQTT.

Kết hợp các công nghệ API, Websocket, MQTT và vi xử lý ESP32 sẽ cùng nhau tạo thành một hệ thống giám sát và điều khiển môi trường hiệu quả, đáp ứng được yêu cầu thực tế của đồ án. Các công nghệ này không chỉ cung cấp các phương pháp giao tiếp và truyền dữ liệu hiệu quả mà còn giúp tối ưu hóa hiệu suất và tính ứng dụng của hệ thống trong các môi trường IoT.

**II.2. Công nghệ sử dụng**

### II.2.1 Node.js & Express

1. **Node.js** là một môi trường chạy JavaScript trên server, cho phép xây dựng các ứng dụng web và API mạnh mẽ và có khả năng xử lý các yêu cầu đồng thời một cách hiệu quả.
2. **Express** là một framework web cho Node.js, cung cấp các công cụ mạnh mẽ để xây dựng các API nhanh chóng và dễ dàng. Express giúp quản lý các tuyến đường (routes), middleware, và xử lý các yêu cầu HTTP một cách hiệu quả.
3. Các thư viện Node.js sử dụng trong hệ thống

* mysql2/promise
* nodemailer
* mqtt
* ws

II.2.2. AWS IoT Core

Trong hệ thống giám sát môi trường và điều khiển thiết bị của đồ án, MQTT (Message Queuing Telemetry Transport) được sử dụng để truyền tải dữ liệu môi trường và điều khiển thiết bị giữa các thiết bị nhúng như ESP32 và server. Để triển khai MQTT, chúng ta sử dụng AWS IoT Core, một dịch vụ quản lý thiết bị IoT từ Amazon Web Services, cung cấp các tính năng sau:

1. **Đăng ký và quản lý thiết bị**: AWS IoT Core cho phép đăng ký, quản lý và kết nối hàng triệu thiết bị IoT một cách an toàn và hiệu quả. Các thiết bị nhúng như ESP32 có thể đăng ký với AWS IoT Core để có thể gửi và nhận các tin nhắn MQTT.
2. **Quản lý luồng dữ liệu MQTT**: AWS IoT Core hỗ trợ các luồng dữ liệu MQTT đến và đi giữa các thiết bị và các ứng dụng backend. Điều này giúp đảm bảo tính tin cậy và hiệu quả trong việc truyền tải dữ liệu từ các cảm biến đến server và ngược lại.
3. **Bảo mật và xác thực**: AWS IoT Core cung cấp các cơ chế bảo mật mạnh mẽ như xác thực và ủy quyền để đảm bảo rằng chỉ các thiết bị được ủy quyền mới có thể truy cập và tương tác với dữ liệu và thiết bị khác.
4. **Tích hợp với các dịch vụ AWS khác**: AWS IoT Core có thể tích hợp dễ dàng với các dịch vụ khác trong hệ sinh thái AWS như Lambda, S3, DynamoDB, để xử lý và lưu trữ dữ liệu môi trường thu thập được.

Thông qua việc sử dụng MQTT với AWS IoT Core, hệ thống của đồ án sẽ có thể hoạt động một cách hiệu quả và đáp ứng được các yêu cầu về độ tin cậy, bảo mật và quản lý thiết bị IoT.

II.2.3. AWS EC2

AWS EC2 (Elastic Compute Cloud) là một dịch vụ cung cấp khả năng điện toán đám mây linh hoạt và mở rộng của Amazon Web Services. Trong hệ thống giám sát môi trường và điều khiển thiết bị từ xa của đồ án này, EC2 được sử dụng để triển khai và chạy các thành phần backend của hệ thống. Cụ thể:

1. **Giới thiệu về AWS EC2**:
   * **AWS EC2**: Là một dịch vụ web cung cấp khả năng tính toán có thể mở rộng trong đám mây. Người dùng có thể khởi tạo các phiên bản máy ảo với cấu hình tùy chọn để chạy các ứng dụng và dịch vụ của mình.
   * **Lợi ích**: AWS EC2 cho phép triển khai các ứng dụng một cách nhanh chóng, dễ dàng mở rộng tài nguyên khi cần thiết, và chỉ trả phí cho tài nguyên đã sử dụng, giúp tối ưu hóa chi phí.
2. **Triển khai ứng dụng Node.js và API trên EC2**:
   * **Cài đặt môi trường**: Khởi tạo một phiên bản EC2 với hệ điều hành ubuntu, sau đó cài đặt Node.js, Express và các thư viện cần thiết (mysql2/promise, nodemailer, mqtt, ws).
   * **Triển khai ứng dụng**: Upload mã nguồn của ứng dụng Node.js lên EC2 và chạy ứng dụng trên máy chủ. API server sẽ lắng nghe các yêu cầu từ client và xử lý chúng.

### II.2.4. ESP32 & DHT11

1. **ESP32** là một nền tảng phổ biến cho các ứng dụng IoT nhờ vào tính năng đa năng và khả năng kết nối internet. Trong đồ án này, ESP32 sẽ được sử dụng để thu thập dữ liệu từ các cảm biến môi trường và thực hiện các hành động điều khiển thiết bị dựa trên các lệnh từ server thông qua giao thức MQTT.
2. **DHT11** là một loại cảm biến phổ biến được sử dụng để đo nhiệt độ và độ ẩm. Nó thường được tích hợp trong các hệ thống IoT và các dự án liên quan đến môi trường nhờ vào tính đơn giản và chi phí thấp.

# **CHƯƠNG III. Phân tích thiết kế, xây dựng ứng dụng & hệ thống**

## **III.1 Phân tích thiết kế**

### III.1.1 Frontend (Web Client)

Web client là thành phần giao diện người dùng của hệ thống, cho phép người dùng tương tác với hệ thống và theo dõi các thông số môi trường cũng như điều khiển các thiết bị từ xa.

* **Giao diện người dùng (UI)**: Giao diện được thiết kế trực quan và dễ sử dụng, hiển thị các thông số môi trường như nhiệt độ, độ ẩm thông qua biểu đồ thời gian thực.
* **Công nghệ sử dụng**: HTML, CSS, JavaScript
* **Chức năng chính**:
  + **Hiển thị dữ liệu cảm biến**: Dữ liệu từ cảm biến DHT11 được hiển thị dưới dạng biểu đồ thời gian thực.
  + **Điều khiển thiết bị**: Cho phép người dùng bật/tắt các thiết bị trong vườn từ xa thông qua giao diện web.
  + **Gửi cảnh báo**: Hiển thị thông báo khi có dữ liệu bất thường được phát hiện.

### III.1.2 Backend Server

Backend server là thành phần xử lý chính của hệ thống, chịu trách nhiệm nhận và xử lý các yêu cầu từ web client và các thiết bị IoT, lưu trữ dữ liệu và thực hiện các chức năng quản lý hệ thống.

* **Nền tảng**: Node.js với framework Express.js.
* **Công nghệ sử dụng**:
  + **API**: Sử dụng API để giao tiếp giữa frontend và backend.
  + **WebSocket**: Để truyền dữ liệu thời gian thực từ server đến web client.
  + **MQTT**: Sử dụng AWS IoT Core để giao tiếp với các thiết bị IoT.
  + **Cơ sở dữ liệu**: MySQL để lưu trữ dữ liệu cảm biến và thông tin người dùng.
  + **Thư viện bổ trợ**: mysql2/promise để kết nối cơ sở dữ liệu, nodemailer để gửi email cảnh báo, mqtt để kết nối với MQTT broker, ws để quản lý WebSocket.

### **III.1.3 Broker IoT Core**

Broker của IoT Core là thành phần quan trọng trong hệ thống, đóng vai trò trung gian trong việc truyền tải dữ liệu giữa các thiết bị IoT (như ESP32) và backend server.

* **AWS IoT Core**: AWS IoT Core là một dịch vụ đám mây được quản lý toàn diện, cho phép các thiết bị IoT kết nối an toàn và tương tác với các ứng dụng đám mây khác.
* **MQTT Broker**: MQTT là giao thức nhắn tin nhẹ, lý tưởng cho các thiết bị IoT có băng thông thấp. MQTT Broker của AWS IoT Core giúp đảm bảo việc truyền tải dữ liệu hiệu quả giữa các thiết bị và server.
* **Công dụng trong hệ thống**:
  + **Kết nối an toàn**: AWS IoT Core cung cấp các cơ chế bảo mật mạnh mẽ như TLS để đảm bảo rằng dữ liệu được truyền tải an toàn.
  + **Quản lý thiết bị**: Dịch vụ này cho phép quản lý dễ dàng các thiết bị IoT, bao gồm việc đăng ký thiết bị, theo dõi tình trạng và quản lý cấu hình.
  + **Khả năng mở rộng**: AWS IoT Core có khả năng xử lý hàng triệu kết nối đồng thời, giúp hệ thống dễ dàng mở rộng khi cần thiết.
* **Chức năng chính**:
  + **Nhận dữ liệu từ ESP32**: Các thiết bị ESP32 gửi dữ liệu cảm biến đến MQTT Broker, sau đó dữ liệu này được chuyển tiếp đến backend server.
  + **Truyền lệnh điều khiển**: Lệnh điều khiển từ backend server được gửi qua MQTT Broker đến các thiết bị ESP32 để thực hiện các hành động tương ứng.

### III.1.4 Thiết bị IoT (ESP32)

ESP32 là vi điều khiển được sử dụng để thu thập dữ liệu từ cảm biến và gửi về backend server, đồng thời nhận các lệnh điều khiển từ server để thực hiện các thao tác tương ứng.

* **Cảm biến sử dụng**: DHT11 để đo nhiệt độ và độ ẩm.
* **Giao tiếp với server**:
  + **MQTT**: ESP32 sử dụng giao thức MQTT để gửi dữ liệu cảm biến và nhận các lệnh điều khiển từ broker.
  + **API:** ESP32 sử dụng API để phát tín hiệu cảnh báo gặp dữ liệu bất thường về server
* **Chức năng chính**:
  + **Thu thập dữ liệu**: Đọc dữ liệu nhiệt độ và độ ẩm từ cảm biến DHT11.
  + **Gửi dữ liệu**: Gửi dữ liệu thu thập được đến server thông qua MQTT.
  + **Nhận và thực hiện lệnh điều khiển**: Nhận lệnh từ server để điều khiển các thiết bị trong vườn (ví dụ: bật/tắt hệ thống tưới nước).
  + **Phát tín hiệu thông báo:** gửi thông báo bất thường về server

## **III.2 Xây dựng ứng dụng & hệ thống**

### III.2.1 Tổng quan hệ thống

Sơ đồ tổng quan hệ thống của chúng tôi gồm ba phần chính: Phần lắp đặt tại vườn, Phần triển khai lên cloud, và Phần giao diện người dùng. Mỗi phần đảm nhận những vai trò và chức năng cụ thể, giúp hệ thống hoạt động một cách hiệu quả và liên tục.

Ảnh có chứa vòng tròn, biểu đồ, thiết kế

Mô tả được tạo tự động

Hình 1:Sơ đồ tổng quan hệ thống

### III.2.2 Các luồng hoạt động

1. Theo dõi dữ liệu và trạng thái thiết bị

* Người dùng truy cập web theo địa chỉ của máy chủ ([http://13.215.50.161](http://13.215.50.161/))
* Lúc truy cập, web client gọi 1 api get để lấy dữ liệu gần nhất (20 bản ghi)
* Hiển thị dữ liệu dưới dạng biểu đồ đường
* Tiếp tục gọi 1 api get để lấy dữ liệu trạng thái hiện tại của các thiết bị, hiển thị lên web dưới dạng nút switch (on/off)

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Mô tả được tạo tự động

Hình 2: Bảng điều khiển

1. Thay đổi trạng thái thiết bị

* Người dùng thay đổi trạng thái switch để thay đổi trạng thái, tạm khóa việc thay đổi trạng thái của switch này lại để đợi phản hồi từ Server
* Lúc này web client gọi 1 api POST lên Server
* Server nhận được request POST, tiếp tục publish 1 message lên broker với nội dung gồm: ID thiết bị, trạng thái thiết lập, mã lệnh cập nhật (phục vụ cho mục đích phân biệt, xử lí, xác nhận các request khác nhau đồng thời)
* ESP32 lắng nghe được message từ broker, xử lí việc điều khiển thiết bị theo nội dung message (ID thiết bị, trạng thái thiết lập), sau đó publish ngược lại broker 1 message (ack) mang mã lệnh cập nhật
* Server lắng nghe ack trong vòng tối đa 10s (time out) kể từ khi nhận request POST:
* Nếu nhận được ack tương ứng mã lệnh cập nhật, server hiểu rằng ESP32 đã cập nhật trạng thái thiết bị thành công, server sẽ gửi dữ liệu trạng thái mới của thiết bị tới tất cả các web client nhằm đồng bộ trạng thái mà các web client hiển thị cho người dùng
* Nếu không nhận được ack hợp lệ, request POST phản hồi lại web client (nguồn gửi request cập nhật) nội dung thông báo cập nhật không thành công, lúc này web client trả lại trạng thái cũ cho switch nhằm báo cho người dùng việc thay đổi trạng thái thất bại

Ảnh có chứa biểu đồ, văn bản, vòng tròn, Kế hoạch

Mô tả được tạo tự động

Hình 3:Sơ đồ cập nhật trạng thái thiết bị

1. Cảnh báo bất thường

* Khi sensor thu được dữ liệu bất thường, ESP32 gọi 1 POST api tới Server với nội dung (dữ liệu hiện tại, mức lệch so với bình thường)
* Bất thường: ở đây, hệ thống lưu giữ lại số liệu trung bình của 20 bản ghi gần nhất (avg). Các bản ghi mới sẽ được so sánh với giá trị avg để đánh giá có bất thường hay là không (Hiện tại đang dùng mức lệch 10% đối với độ ẩm và 5°C đối với nhiệt độ)
* Server nhận được POST request và thực hiện việc gửi thông báo tới địa chỉ email người dùng

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

Hình 4: Email cảnh báo bất thường

# **CHƯƠNG IV. Cài đặt chương trình và kết quả thực nghiệm**

## **IV.1 Cài đặt chương trình**

### IV.1.1 Môi trường phát triển

1. Visual Studio Code (VS Code)

**VS Code** là một trình soạn thảo mã nguồn phổ biến, được phát triển bởi Microsoft. Nó cung cấp nhiều tính năng mạnh mẽ hỗ trợ lập trình viên trong việc viết, gỡ lỗi và triển khai mã nguồn.

* **Hỗ trợ ngôn ngữ đa dạng**: VS Code hỗ trợ nhiều ngôn ngữ lập trình, bao gồm JavaScript, Node.js, HTML, CSS, và nhiều ngôn ngữ khác.
* **Tiện ích mở rộng**: Có một loạt các tiện ích mở rộng có sẵn trong VS Code, giúp tích hợp các công cụ và dịch vụ khác nhau (ví dụ: MySQL, REST Client, MQTT, và nhiều tiện ích khác).
* **Quản lý dự án**: Tính năng quản lý dự án, điều hướng tệp, và tích hợp với Git giúp lập trình viên dễ dàng quản lý và theo dõi mã nguồn của dự án.
* **Debugger**: Tích hợp trình gỡ lỗi mạnh mẽ cho Node.js và JavaScript, giúp dễ dàng phát hiện và sửa lỗi trong quá trình phát triển.

1. Arduino IDE

**Arduino IDE** là một môi trường phát triển tích hợp dành cho lập trình vi điều khiển, đặc biệt là các bo mạch Arduino như ESP32.

* **Viết và tải mã lên ESP32**: Arduino IDE cung cấp các công cụ để viết mã cho ESP32 và tải mã lên bo mạch dễ dàng.
* **Thư viện hỗ trợ**: Arduino IDE đi kèm với nhiều thư viện hỗ trợ việc kết nối và giao tiếp với các cảm biến và thiết bị khác (ví dụ: DHT11, MQTT).
* **Serial Monitor**: Công cụ Serial Monitor giúp kiểm tra và gỡ lỗi mã bằng cách xem dữ liệu được gửi từ ESP32 qua cổng serial.
* **Cộng đồng và tài liệu phong phú**: Arduino IDE có một cộng đồng lớn và nhiều tài liệu hướng dẫn, ví dụ và thư viện có sẵn, giúp dễ dàng tìm kiếm giải pháp cho các vấn đề kỹ thuật.

### **IV.1.2 Quy trình phát triển**

1. Thiết lập và cài đặt môi trường

 **Cài đặt VS Code**: Tải và cài đặt VS Code từ [trang web chính thức](https://code.visualstudio.com/).

 **Cài đặt Arduino IDE**: Tải và cài đặt Arduino IDE từ [trang web chính thức](https://www.arduino.cc/en/software).

1. Cấu hình VS Code

* **Tiện ích mở rộng cần thiết**: Cài đặt các tiện ích mở rộng cần thiết như Node.js, MySQL, MQTT, và bất kỳ tiện ích mở rộng nào hỗ trợ ngôn ngữ và công cụ bạn đang sử dụng.
* **Tạo và quản lý dự án**: Sử dụng VS Code để tạo và quản lý dự án Node.js của bạn, bao gồm các thư mục public, server.js, và bất kỳ tệp nào liên quan đến dự án.

1. Phát triển ứng dụng web với VS Code

* **Tạo cấu trúc thư mục**:

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phông chữ

Mô tả được tạo tự động

* **Viết mã backend và frontend**: Sử dụng VS Code để viết mã cho các phần backend (Node.js, Express.js) và frontend (HTML, CSS, JavaScript).

1. Lập trình ESP32 với Arduino IDE

* **Viết mã cho ESP32**: Sử dụng Arduino IDE để viết mã cho ESP32, bao gồm việc kết nối với cảm biến DHT11, gửi dữ liệu qua MQTT và nhận lệnh điều khiển.
* **Upload mã lên ESP32**: Dùng Arduino IDE để tải mã lên ESP32 và kiểm tra hoạt động của bo mạch.

## **IV.2 Kết quả thực nghiệm**

### IV.2.1 Mục tiêu thực nghiệm

Mục tiêu của phần thực nghiệm này là kiểm tra và đánh giá hiệu quả hoạt động của hệ thống giám sát thông tin môi trường và điều khiển thiết bị từ xa tại vườn. Cụ thể, tiến hành thực nghiệm để:

* Đánh giá độ chính xác của việc thu thập dữ liệu nhiệt độ và độ ẩm từ cảm biến DHT11.
* Kiểm tra khả năng gửi dữ liệu thời gian thực từ ESP32 lên server thông qua MQTT.
* Đánh giá hiệu suất của việc hiển thị dữ liệu và điều khiển thiết bị trên giao diện web.
* Kiểm tra khả năng gửi cảnh báo qua email khi phát hiện dữ liệu bất thường.

### IV.2.2 Cấu hình thực nghiệm

* **Thiết bị phần cứng**: ESP32, cảm biến DHT11, thiết bị điều khiển (ví dụ: bơm nước, đèn).
* **Môi trường phát triển**: VS Code, Arduino IDE.
* **Nền tảng đám mây**: AWS EC2 (triển khai server Node.js và MySQL), AWS IoT Core (MQTT Broker).
* **Phần mềm và công cụ**: Node.js, Express.js, MySQL, WebSocket, MQTT, nodemailer.

### IV.2.3 Tiến hành thực nghiệm

1. Thu thập và truyền dữ liệu từ cảm biến DHT11

* **Quá trình**: ESP32 thu thập dữ liệu nhiệt độ và độ ẩm từ cảm biến DHT11 mỗi 5 giây và gửi dữ liệu này lên server qua MQTT.
* **Kết quả**: Dữ liệu được truyền tải ổn định và chính xác, thể hiện qua các bản ghi trong cơ sở dữ liệu MySQL. Không có sự cố mất dữ liệu hoặc sai lệch đáng kể.

1. Hiển thị dữ liệu thời gian thực trên giao diện web

* **Quá trình**: Sử dụng WebSocket để cập nhật và hiển thị dữ liệu thời gian thực trên giao diện web. Các biểu đồ thời gian thực thể hiện nhiệt độ và độ ẩm được cập nhật mỗi 5 giây.
* **Kết quả**: Giao diện web phản hồi nhanh chóng và hiển thị dữ liệu một cách chính xác. Các biểu đồ thời gian thực hiển thị mượt mà và liên tục, giúp người dùng dễ dàng theo dõi thông tin môi trường.

1. Điều khiển thiết bị từ xa qua giao diện web

* **Quá trình**: Người dùng gửi lệnh điều khiển từ giao diện web, lệnh này được server tiếp nhận và gửi đến ESP32 qua MQTT để điều khiển các thiết bị (ví dụ: bật/tắt bơm nước).
* **Kết quả**: Thiết bị phản hồi nhanh chóng theo lệnh điều khiển từ người dùng. Độ trễ từ khi gửi lệnh đến khi thiết bị phản hồi là không đáng kể, đảm bảo tính hiệu quả trong việc điều khiển từ xa.

1. Gửi cảnh báo qua email khi phát hiện dữ liệu bất thường

* **Quá trình**: Server giám sát các giá trị dữ liệu và gửi email cảnh báo qua dịch vụ Gmail khi phát hiện dữ liệu nhiệt độ hoặc độ ẩm vượt ngưỡng quy định.
* **Kết quả**: Hệ thống gửi email cảnh báo thành công mỗi khi phát hiện dữ liệu bất thường. Các email được gửi đi kịp thời, nội dung rõ ràng và dễ hiểu.

### IV.2.4 Đánh giá kết quả thực nghiệm

1. Độ tin cậy và chính xác của hệ thống

Hệ thống hoạt động ổn định, dữ liệu thu thập từ cảm biến DHT11 chính xác và đáng tin cậy. Không có hiện tượng mất dữ liệu hoặc lỗi trong quá trình truyền tải và hiển thị dữ liệu.

1. Tính khả dụng và hiệu suất của giao diện người dùng

Giao diện web thân thiện với người dùng, dễ sử dụng và phản hồi nhanh. Các thao tác điều khiển thiết bị từ xa và hiển thị dữ liệu thời gian thực được thực hiện mượt mà, không gặp bất kỳ sự cố nào.

1. Tính thực tế và khả năng mở rộng của hệ thống

Hệ thống được triển khai thành công và có thể áp dụng thực tế cho việc giám sát và điều khiển thiết bị tại vườn. Ngoài ra, hệ thống có khả năng mở rộng, bao gồm việc tích hợp thêm các loại cảm biến khác và mở rộng quy mô giám sát đến nhiều khu vực khác nhau.

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

**Kết luận**

Dự án "Triển khai hệ thống web giám sát thông tin môi trường & điều khiển thiết bị tại vườn" đã đạt được những mục tiêu quan trọng đề ra ban đầu. Qua quá trình thiết kế, xây dựng, và thực nghiệm, hệ thống đã chứng minh được tính hiệu quả và độ tin cậy trong việc giám sát và điều khiển môi trường tại vườn. Các kết quả thực nghiệm cho thấy hệ thống có thể thu thập và hiển thị dữ liệu môi trường một cách chính xác và thời gian thực, đồng thời cho phép điều khiển thiết bị từ xa một cách nhanh chóng và ổn định. Hơn nữa, hệ thống có khả năng gửi cảnh báo qua email khi phát hiện dữ liệu bất thường, giúp người dùng kịp thời có những biện pháp xử lý thích hợp.

**Hướng phát triển**

* Tăng số lượng cảm biến và thiết bị điều khiển

**Tích hợp thêm nhiều loại cảm biến khác**: Ngoài cảm biến DHT11, có thể tích hợp thêm các loại cảm biến khác như cảm biến ánh sáng, cảm biến đất ẩm, cảm biến CO2, để cung cấp thông tin toàn diện hơn về môi trường.

**Mở rộng số lượng thiết bị điều khiển**: Tăng số lượng thiết bị điều khiển như quạt, hệ thống tưới nước tự động, đèn chiếu sáng để điều khiển nhiều thiết bị hơn trong khu vườn.

* Nâng cao giao diện và chức năng web

**Bổ sung chức năng điều chỉnh thông số**: Thêm các chức năng cho phép người dùng điều chỉnh các thông số hoạt động của hệ thống trực tiếp trên giao diện web, như ngưỡng nhiệt độ/độ ẩm để kích hoạt cảnh báo.

**Cài đặt chế độ tự động**: Tích hợp chế độ tự động, cho phép hệ thống tự động điều khiển các thiết bị dựa trên các thông số cảm biến mà không cần sự can thiệp của người dùng. Ví dụ, tự động bật/tắt hệ thống tưới nước dựa trên độ ẩm của đất.

**Cải thiện giao diện người dùng**: Nâng cấp giao diện người dùng để thân thiện hơn, dễ sử dụng hơn, và có thể hiển thị thêm nhiều thông tin hữu ích như biểu đồ phân tích xu hướng thay đổi của các chỉ số môi trường theo thời gian.

**TÀI LIỆU THAM KHẢO**

|  |  |
| --- | --- |
| [1] | [DHT11 Humidity & Temperature Sensor](https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf) |
| [2] | [ESP32 Documentation](https://docs.espressif.com/projects/esp-idf/en/stable/esp32/index.html) |
| [3] | [Node.js Documentation](https://nodejs.org/docs/latest/api/) |
| [4] | [Express.js Documentation](https://expressjs.com/) |
| [5] | [MySQL Documentation](https://dev.mysql.com/doc/) |
| [6] | [WebSocket API](https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API) |
| [7] | [AWS IoT Core Documentation](https://docs.aws.amazon.com/iot/) |
| [8] | [AWS EC2 Documentation](https://docs.aws.amazon.com/ec2/) |
| [9] | [Deploying Applications on AWS EC2](https://docs.aws.amazon.com/codedeploy/latest/userguide/deployment-steps-server.html) |
| [10] | [Setting up an MQTT Broker on AWS IoT Core](https://www.youtube.com/watch?v=idf-gGXvIu4) |
| [11] | [Sending Emails with Node.js and Nodemailer](https://www.youtube.com/watch?v=QDIOBsMBEI0) |