



**TRƯỜNG ĐẠI HỌC BÁCH KHOA**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CUỐI KỲ**  
**HỌC PHẦN: VI ĐIỀU KHIỂN**

**TÊN ĐỀ TÀI**  
**HỆ THỐNG GIÁM SÁT VÀ BẢO VỆ RỪNG**

<b>Nhóm</b>	<b>12</b>
<b>Họ Và Tên Sinh Viên</b>	<b>Lớp Học Phần</b>
Thái Viết Quốc Hưng	22Nh12
Đặng Phúc Long	
Phạm Nhật Thành	

**ĐÀ NẴNG, 05/2025**

## **TÓM TẮT**

Hiện nay, vấn đề cháy rừng và nạn chặt phá rừng trái phép đang diễn ra ngày càng nghiêm trọng, gây tổn hại lớn đến môi trường, đa dạng sinh học và đời sống con người, trong khi công tác giám sát thủ công gặp nhiều hạn chế do diện tích rừng rộng lớn, nhân lực thiếu hụt và thiếu hệ thống cảnh báo sớm. Để giải quyết vấn đề này, đề tài đã xây dựng một hệ thống giám sát và bảo vệ rừng ứng dụng công nghệ IoT, kết hợp các cảm biến nhiệt độ - độ ẩm, cảm biến âm thanh, cảm biến GPS và động cơ bơm, kết nối qua vi điều khiển ESP32. Hệ thống có khả năng phát hiện sớm nguy cơ cháy rừng dựa vào điều kiện môi trường, cảnh báo nạn chặt phá rừng thông qua phân tích và nhận diện âm thanh tiếng cưa, tự động điều khiển tưới nước khi cần, đồng thời truyền dữ liệu về server để hiển thị trực quan trên giao diện web với biểu đồ nhiệt độ, độ ẩm, bản đồ định vị GPS và có thể nghe lại âm thanh đã thu được. Kết quả đạt được cho thấy hệ thống hoạt động ổn định, thu thập và truyền dữ liệu cảm biến chính xác. Hiển thị được biểu đồ nhiệt độ, độ ẩm và bản đồ GPS real-time trên giao diện web. Cảnh báo kịp thời khi phát hiện tiếng cưa, nhiệt độ bất thường, giúp giảm thiểu thiệt hại do cháy rừng và nạn phá rừng, góp phần nâng cao năng lực bảo vệ và quản lý tài nguyên rừng.

## BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Tự đánh giá
Thái Viết Quốc Hưng	-Viết server Backend xử lý nhiệt độ, độ ẩm. -Viết code Arduino kết nối Cảm biến nhiệt độ, độ ẩm DHT22 -Lắp mạch.	Đã hoàn thành tất cả nhiệm vụ.
Đặng Phúc Long	-Viết server AI nhận diện tiếng cưa. -Viết server Backend xử lý âm thanh. -Viết code Arduino kết nối Cảm biến âm thanh INMP441 I2S Omnidirectional Microphone. -Lắp mạch .	Đã hoàn thành tất cả nhiệm vụ.
Phạm Nhật Thành	-Viết server Backend xử lý tín hiệu từ module định vị GPS NEO 6M. -Viết Frontend. -Viết code Arduino kết nối module định vị GPS NEO 6M và điều khiển động cơ phun nước R385. -Lắp mạch.	Đã hoàn thành tất cả nhiệm vụ.

## MỤC LỤC

I. GIỚI THIỆU .....	6
1. Thực trạng .....	6
2. Các vấn đề cần giải quyết.....	6
3. Đề xuất giải pháp tổng quan.....	7
II. GIẢI PHÁP.....	8
1. Phần cứng.....	8
1.1 Sơ đồ lắp mạch.....	8
1.2 Cảm biến nhiệt độ độ ẩm DHT22 .....	11
1.3 Cảm biến âm thanh INMP441 I2S Omnidirectional Microphone .....	12
1.3 Module GPS NEO 6M.....	14
2. Truyền thông .....	15
2.1 Module truyền thông (ESP32) .....	15
2.2 Giao thức truyền thông (Http và Websocket) .....	15
2.2.1 HTTP .....	15
2.2.2 Websocket .....	16
3. Phần mềm.....	16
3.1 Sơ đồ khối thuật toán xử lý chính .....	16
3.2 Thư viện sử dụng .....	22
III. KẾT QUẢ VÀ ĐÁNH GIÁ .....	23
1. Giám sát Nhiệt độ và Độ ẩm .....	23
2. Phát hiện và Cảnh báo Chặt phá rừng.....	24
3. Định vị và Hiện thị trên Bản đồ GPS.....	25
4. Hệ thống Tưới nước Tự động .....	25
5. Giao diện Web và Khả năng Hiện thị Dữ liệu .....	26
IV. KẾT LUẬN .....	26
1. Các kết quả chính đạt được.....	26
2. Hướng phát triển và cải thiện.....	27
V. TÀI LIỆU THAM KHẢO.....	28

## DANH MỤC HÌNH ẢNH

Hình 1 : Sơ đồ tổng quan hệ thống .....	8
Hình 2: Sơ đồ lắp mạch .....	8
Hình 3: Sơ đồ chân của cảm biến nhiệt độ, độ ẩm DHT22 .....	11
Hình 4: Sơ đồ chân của cảm biến âm thanh INMP441 .....	13
Hình 5: Sơ đồ chân của Module GPS NEO 6M .....	14
Hình 6: Sơ đồ khối thuật toán đọc và truyền dữ liệu âm thanh cảm biến .....	17
Hình 7: Sơ đồ khối thuật toán đọc và truyền dữ liệu nhiệt độ, độ ẩm.....	18
Hình 8: Sơ đồ khối thuật toán điều khiển và tự động bơm nước.....	19
Hình 9: Sơ đồ thuật toán truy xuất dữ liệu nhiệt độ, độ ẩm .....	20
Hình 10: Sơ đồ khối thuật toán phát hiện chặt phá rừng .....	21
Hình 11: Mô hình thiết bị giám sát và bảo vệ rừng .....	23

## I. GIỚI THIỆU

Nạn cháy rừng và chặt phá rừng trái phép đang là những mối đe dọa nghiêm trọng đến hệ sinh thái toàn cầu, gây ra những hậu quả khôn lường về môi trường, suy giảm đa dạng sinh học và ảnh hưởng trực tiếp đến đời sống con người. Việt Nam, với diện tích rừng rộng lớn, cũng không nằm ngoài vòng xoáy của vấn đề này. Công tác giám sát rừng hiện nay chủ yếu vẫn dựa vào phương pháp thủ công, gặp nhiều hạn chế về quy mô, nhân lực và khả năng phản ứng nhanh chóng. Điều này dẫn đến việc khó khăn trong việc phát hiện sớm các nguy cơ cháy rừng hay các hành vi phá hoại, gây tổn thất lớn trước khi có biện pháp can thiệp.

Để giải quyết những thách thức trên, đề tài này đề xuất và triển khai một **Hệ thống giám sát và bảo vệ rừng thông minh** ứng dụng công nghệ **Internet of Things (IoT)**. Hệ thống được thiết kế để cung cấp khả năng giám sát liên tục, cảnh báo sớm và tự động hóa một phần các tác vụ bảo vệ rừng, nhằm nâng cao hiệu quả quản lý và giảm thiểu thiệt hại.

### 1. Thực trạng

- Diện tích rừng rộng lớn: Việt Nam sở hữu hàng triệu hecta rừng, gây khó khăn cho việc giám sát toàn diện bằng phương pháp thủ công.
- Thiếu hụt nhân lực: Lực lượng kiểm lâm còn mỏng, khó đáp ứng được nhu cầu giám sát trên quy mô lớn và liên tục.
- Thiếu hệ thống cảnh báo sớm: Việc phát hiện sớm cháy rừng hoặc các hành vi phá hoại còn hạn chế, dẫn đến việc ứng phó chậm trễ và hậu quả nghiêm trọng.
- Dữ liệu giám sát không liên tục: Thông tin về tình trạng rừng thường được thu thập định kỳ, không cung cấp dữ liệu tức thời để đưa ra quyết định kịp thời.

### 2. Các vấn đề cần giải quyết

Dựa trên thực trạng, hệ thống IoT được đề xuất sẽ tập trung giải quyết các vấn đề chính sau:

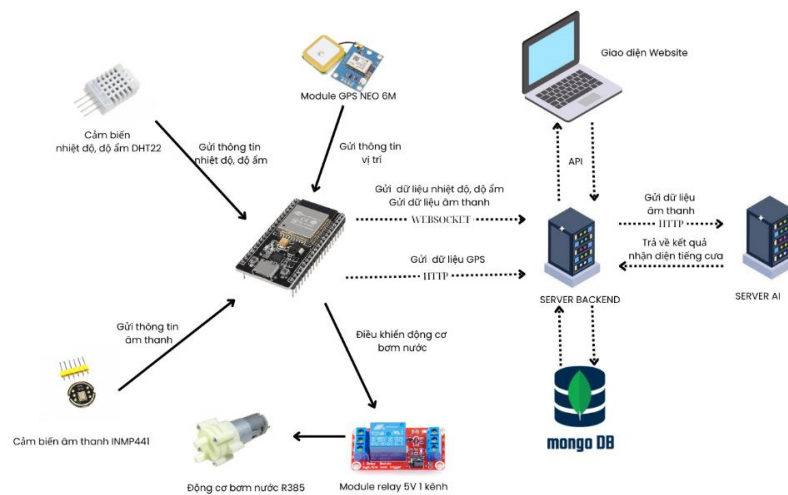
- Phát hiện sớm nguy cơ cháy rừng: Cần một cơ chế cảnh báo tự động khi các điều kiện môi trường như nhiệt độ và độ ẩm vượt ngưỡng nguy hiểm.
- Cảnh báo nạn chặt phá rừng trái phép: Yêu cầu khả năng nhận diện âm thanh đặc trưng của hoạt động khai thác gỗ (tiếng cưa) để kịp thời phát hiện.
- Hỗ trợ công tác chữa cháy ban đầu: Hệ thống cần có khả năng tự động thực hiện một số hành động phòng ngừa hoặc giảm thiểu thiệt hại ban đầu như tưới nước.
- Cung cấp dữ liệu giám sát trực quan và thời gian thực: Đảm bảo cán bộ quản lý có thể theo dõi tình trạng rừng mọi lúc, mọi nơi thông qua giao diện dễ hiểu.

- Tối ưu hóa nguồn lực giám sát: Giảm sự phụ thuộc vào giám sát thủ công, cho phép nhân lực tập trung vào các nhiệm vụ chiến lược hơn.

### 3. Đề xuất giải pháp tổng quan

Vấn đề	Giải pháp đề xuất
Phần cứng	ESP32 Cảm biến nhiệt độ , độ ẩm DHT22 Cảm biến âm thanh INMP441 I2S Omnidirectional Microphone Module GPS NEO 6M Anten GPS 1575.42Mhz SMA Module relay 5V 1 kênh Động cơ bơm nước R385
Ứng dụng web	Sử dụng Framework ReactJS
Server nhận dữ liệu và lưu trữ	Sử dụng Framework Django Rest Framework
Server cung cấp API nhận dạng tiếng cưa	Sử dụng Framework Flask
Cơ sở dữ liệu lưu trữ	Sử dụng MongoDB

#### ❖ Sơ đồ tổng quan hệ thống

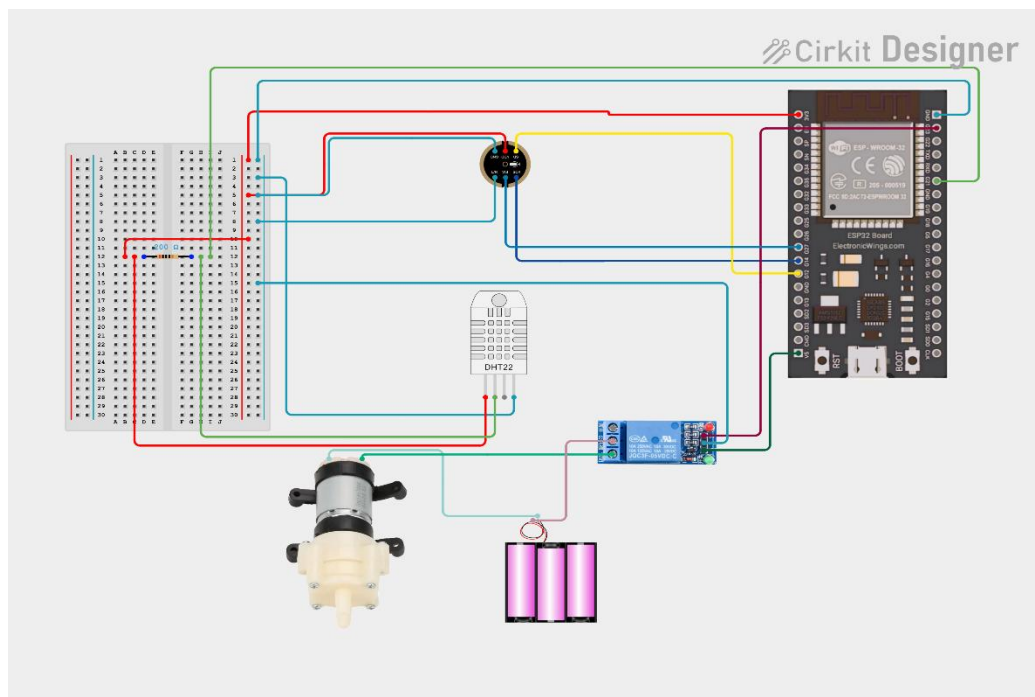


Hình 1 : Sơ đồ tổng quan hệ thống

## II. GIẢI PHÁP


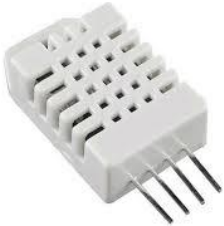
### 1. Phần cứng






#### 1.1 Sơ đồ lắp mạch



Hình 2: Sơ đồ lắp mạch



Tên linh kiện	Thông số kỹ thuật
<p>ESP32</p> 	<ul style="list-style-type: none"> <li>-Vi xử lý: Xtensa Dual-Core LX6 microprocessor</li> <li>-RAM: 520 KB SRAM</li> <li>-Flash: 4MB – 16MB tùy module</li> <li>-Wi-Fi: 2.4 GHz IEEE 802.11 b/g/n</li> <li>-Bluetooth: Bluetooth v4.2 (Classic + BLE)</li> <li>-GPIO: Lên đến 34 chân đa chức năng</li> <li>-ADC / DAC: 12-bit ADC (18 kênh), 2 kênh DAC</li> <li>-Điện áp hoạt động: 3.0 – 3.3V</li> <li>-Tiêu thụ điện năng: ~10 <math>\mu</math>A (deep sleep) – ~250 mA (Wi-Fi hoạt động)</li> <li>-Khoảng cách Wi-Fi: Lên đến 100m (môi trường mở)</li> </ul>
<p>Cảm biến nhiệt độ, độ ẩm DHT22</p> 	<ul style="list-style-type: none"> <li>-Điện áp hoạt động: 3.5V đến 5.5V</li> <li>-Dòng hoạt động: 0,3mA (đo) 60uA (chế độ chờ)</li> <li>-Đầu ra: Dữ liệu nối tiếp</li> <li>-Phạm vi nhiệt độ: -40 ° C đến 80 ° C</li> <li>-Phạm vi độ ẩm: 0% đến 100%</li> <li>-Độ phân giải: Nhiệt độ và Độ ẩm đều là 16-bit</li> <li>-Độ chính xác: <math>\pm 0,5</math> ° C và <math>\pm 1\%</math></li> </ul>
<p>Cảm biến âm thanh INMP441 I2S Omnidirectional Microphone</p>	<ul style="list-style-type: none"> <li>- Điện áp hoạt động: 1.8 – 3.3VDC</li> <li>- Dòng tiêu thụ: 1.4mA</li> <li>- Giao tiếp I2S 24bit</li> <li>- Độ nhạy: -26dBFS</li> <li>- Tích hợp bộ lọc Anti-aliasing</li> <li>- Tần số cảm nhận được: 60Hz – 15kHz</li> </ul>

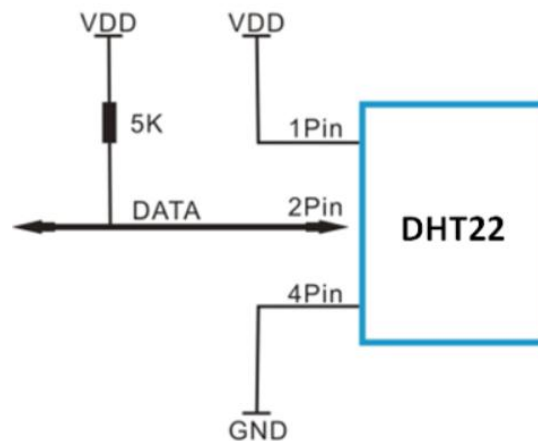
	
<p>Module GPS NEO 6M</p> 	<ul style="list-style-type: none"> <li>- Điện áp hoạt động: 3.3V – 5V</li> <li>- Giao tiếp: UART/TTL</li> <li>- Tốc độ cập nhật: Mặc định 1Hz (tối đa 5Hz)</li> <li>- Độ chính xác vị trí: ~2.5m ngoài trời, có thể tốt hơn với anten chất lượng cao</li> </ul>
<p>Anten GPS 1575.42Mhz SMA</p> 	<ul style="list-style-type: none"> <li>- Tần số hoạt động: 1575.42 MHz</li> <li>-Trở kháng: 50 <math>\Omega</math></li> <li>-Độ rộng băng thông: <math>\pm 10</math> MHz</li> <li>-Độ khuếch đại: <math>28 \pm 2</math> dB</li> </ul>
<p>Module relay 5V 1 kênh</p> 	<ul style="list-style-type: none"> <li>- Điện áp hoạt động: 5V</li> <li>-Dòng kích Relay: 5mA</li> </ul>
<p>Động cơ bơm nước R385</p> 	<ul style="list-style-type: none"> <li>-Điện áp hoạt động: 6V – 12V DC</li> <li>-Dòng điện hoạt động: 0.5~0.7A</li> <li>-Lưu lượng nước: 1~2L / 1 phút</li> <li>-Chiều cao hút tối đa: 1 - 1.5m</li> </ul>

## 1.2 Cảm biến nhiệt độ độ ẩm DHT22

DHT22 là cảm biến nhiệt độ và độ ẩm kỹ thuật số, sử dụng cảm biến độ ẩm điện dung và điện trở nhiệt để đo môi trường xung quanh. Nó xuất tín hiệu kỹ thuật số trực tiếp qua một chân dữ liệu mà không cần ADC. Cảm biến đã được hiệu chuẩn sẵn, tích hợp vi điều khiển 8-bit, độ chính xác cao, tiêu thụ điện năng thấp và có thể truyền dữ liệu ở khoảng cách lên tới 20m. Thiết kế nhỏ gọn, dễ kết nối với 4 chân.

### ❖ Sơ đồ chân

Tên chân	Chức năng
VDD	Cung cấp nguồn điện cho cảm biến, hoạt động trong dải từ 3.5V đến 5.5V
DATA	Chân giao tiếp dữ liệu với vi điều khiển (MCU), xuất dữ liệu nhiệt độ và độ ẩm dưới dạng nối tiếp, dùng bus đơn
GND	Chân nối đất cho mạch
NC	Không kết nối / Không sử dụng (NULL); chỉ có trên cảm biến loại 4 chân



Hình 3: Sơ đồ chân của cảm biến nhiệt độ, độ ẩm DHT22

### ❖ Nguyên lý hoạt động

#### ➤ Thu nhận thông tin môi trường:

- Cảm biến độ ẩm điện dung sẽ thay đổi điện dung khi độ ẩm môi trường thay đổi.
- Thermistor thay đổi giá trị điện trở theo nhiệt độ hiện tại.
- Hai tín hiệu analog này được đọc liên tục bởi IC xử lý.

#### ➤ Xử lý và chuyển đổi tín hiệu:

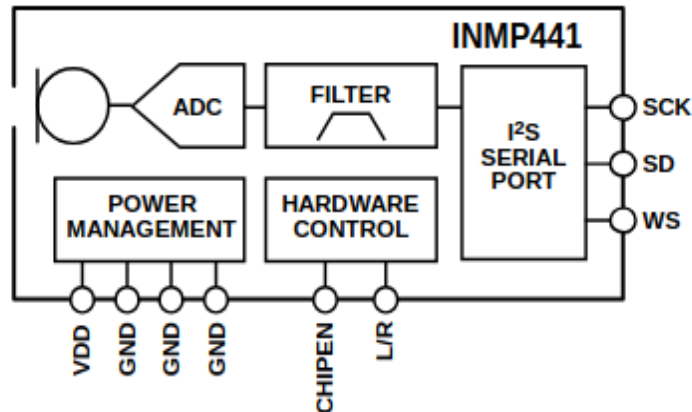
- Dữ liệu analog được chuyển đổi thành tín hiệu số bằng bộ chuyển đổi ADC tích hợp trong IC.
- Tín hiệu sau đó được mã hóa thành một chuỗi bit chuẩn gồm: 16 bit độ ẩm, 16 bit nhiệt độ và 8 bit checksum để xác thực dữ liệu
- **Truyền dữ liệu:**
  - DHT22 sử dụng giao tiếp một dây (single-wire) để truyền dữ liệu đến vi điều khiển.
  - Vi điều khiển sẽ gửi xung khởi tạo (start signal) và cảm biến sẽ phản hồi lại chuỗi dữ liệu trong khoảng 2ms – 5ms, đảm bảo độ ổn định và chính xác.

### 1.3 Cảm biến âm thanh INMP441 I2S Omnidirectional Microphone

INMP441 là một loại micro kỹ thuật số thông minh, cực kỳ nhỏ gọn, có khả năng thu âm từ mọi hướng (đa hướng). Điểm đặc biệt của nó là sử dụng công nghệ MEMS tiên tiến và truyền dữ liệu âm thanh dưới dạng kỹ thuật số qua giao diện I2S 24-bit. Điều này giúp micro cho ra âm thanh chất lượng cao, ít nhiễu và dễ dàng kết nối trực tiếp với các bộ xử lý mà không cần thêm bộ chuyển đổi phức tạp.

#### ❖ Sơ đồ chân

Tên chân	Chức năng
VCC	Nguồn vào, hoạt động trong dải từ 1.8V đến 3.3V
GND	Chân nối đất
SD	Serial Data Output (Dữ liệu ra dạng I <sup>2</sup> S)
WS	Word Select (còn gọi là LRCLK, xác định kênh trái/phải)
SCK	Serial Clock (còn gọi là BCLK, xung đồng hồ dữ liệu)
L/R	Left/Right Channel Select (chân cấu hình kênh, không phải tín hiệu đang truyền, thường được nối GND hoặc VCC để cố định bên nào)



Hình 4: Sơ đồ chân của cảm biến âm thanh INMP441

#### ❖ Nguyên lý hoạt động

##### ➤ Thu nhận âm thanh:

- Khi có âm thanh (sóng áp suất không khí) đến, màng rung siêu nhỏ trong cảm biến MEMS sẽ dao động. Sự dao động này làm thay đổi điện dung của một tụ điện nhỏ bên trong, từ đó tạo ra một tín hiệu điện tương ứng với âm thanh.
- Micro có đặc tính đa hướng, nghĩa là nó thu nhận âm thanh đồng đều từ mọi hướng.

##### ➤ Xử lý tín hiệu:

- **Chuyển đổi và khuếch đại:** Tín hiệu điện analog từ cảm biến được khuếch đại và ngay lập tức chuyển đổi thành tín hiệu số bởi bộ chuyển đổi ADC.

##### ➤ Lọc âm thanh (Bộ lọc số):

- **Lọc bỏ tiếng ồn tần số thấp:** Một bộ lọc đặc biệt (High-Pass Filter) sẽ loại bỏ các âm thanh quá trầm hoặc tiếng ồn nền không mong muốn (ví dụ như tiếng gió, tiếng ồn từ máy móc).
- **Lọc bỏ nhiễu tần số cao:** Một bộ lọc khác (Low-Pass Filter) sẽ loại bỏ các nhiễu tần số cao nằm ngoài dải nghe được của tai người, giúp tín hiệu âm thanh trở nên "sạch" hơn.
- Nhờ các bộ lọc này, INMP441 có thể cung cấp âm thanh với độ rõ nét cao (SNR 61 dBA) và thu được dải tần số rộng từ 60 Hz đến 15 kHz một cách đồng đều.

##### ➤ Giao diện I2S:

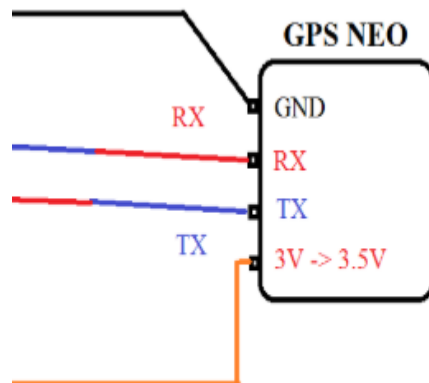
- Sau khi được xử lý và số hóa, dữ liệu âm thanh được gửi ra ngoài qua giao diện I2S dưới dạng 24-bit.

### 1.3 Module GPS NEO 6M

**GPS (Global Positioning System)** là hệ thống định vị toàn cầu, hoạt động dựa trên mạng lưới các vệ tinh nhân tạo quay quanh Trái Đất. GPS xác định vị trí của một thiết bị bằng cách thu tín hiệu từ nhiều vệ tinh và tính toán khoảng cách đến từng vệ tinh, sau đó sử dụng nguyên lý toán học trilateration để xác định tọa độ vị trí (kinh độ, vĩ độ, độ cao). Để cải thiện hiệu suất thu sóng, nhóm đã nâng cấp sử dụng ăng-ten rời **Antenna GPS 1575.42 MHz**, giúp tăng cường khả năng thu tín hiệu GPS, từ đó cải thiện độ chính xác và tốc độ định vị của hệ thống.

#### ❖ Sơ đồ chân

Tên chân	Chức năng
VCC	Cấp nguồn (3.3V hoặc 5V)
GND	Chân nối đất
TX	Chân truyền dữ liệu
RX	Chân nhận dữ liệu



Hình 5: Sơ đồ chân của Module GPS NEO 6M

#### ❖ Nguyên lý hoạt động

##### ➤ Thu tín hiệu từ vệ tinh

- NEO-6M sử dụng ăng-ten GPS để thu tín hiệu vô tuyến từ ít nhất 4 vệ tinh GPS trên quỹ đạo.
- Mỗi vệ tinh phát sóng liên tục, mang theo thời gian truyền và vị trí hiện tại của vệ tinh

##### ➤ Tính khoảng cách đến các vệ tinh

- Dựa vào thời gian tín hiệu truyền từ vệ tinh đến module, NEO-6M tính được khoảng cách đến từng vệ tinh.
- **Xác định vị trí bằng Trilateration**
  - Khi biết khoảng cách đến ít nhất 3 vệ tinh, NEO-6M dùng trilateration để xác định kinh độ, vĩ độ.
  - Nếu có 4 vệ tinh trở lên, module còn có thể tính được cả độ cao (altitude) và đồng bộ thời gian chính xác.
- **Xuất dữ liệu qua UART**
  - Sau khi tính toán, module truyền dữ liệu vị trí ra dưới dạng chuỗi NMEA qua giao tiếp UART (TX/RX) đến vi điều khiển.

## 2. Truyền thông

### 2.1 Module truyền thông (ESP32)

**ESP32** là một module vi điều khiển tích hợp sẵn **Wi-Fi** và **Bluetooth**, do Espressif Systems phát triển. Đây là lựa chọn lý tưởng cho các hệ thống IoT, nhúng, điều khiển từ xa, và các ứng dụng thời gian thực nhờ khả năng giao tiếp đa dạng, mạnh mẽ và tiết kiệm năng lượng.

**Các giao thức truyền thông được sử dụng trong dự án:**

Giao thức	Mô tả và ứng dụng chính
Wi-Fi	Chuẩn 802.11 b/g/n, cho phép ESP32 kết nối Internet để gửi/nhận dữ liệu từ server hoặc cloud qua các giao thức như HTTP, MQTT, WebSocket,...
UART	Giao tiếp nối tiếp hai chiều. Giao tiếp với GPS, module GSM, cảm biến nối tiếp,...
I2S	Giao tiếp âm thanh số. Kết nối micro kỹ thuật số (như INMP441), DAC âm thanh,...

### 2.2 Giao thức truyền thông (Http và Websocket)

#### 2.2.1 HTTP

HTTP là giao thức truyền thông lớp ứng dụng hoạt động theo mô hình **request–response**

- Đặc điểm:
  - Đơn hướng: client gửi yêu cầu, server trả lời.
  - Không duy trì kết nối liên tục (stateless).

- Dễ triển khai, phù hợp với các hệ thống gửi dữ liệu theo chu kỳ.
- Ưu điểm:
  - Phổ biến, dễ dùng.
  - Tương thích tốt với hầu hết thiết bị và nền tảng.
- Nhược điểm:
  - Không phù hợp cho truyền dữ liệu thời gian thực.
  - Cần gửi nhiều request liên tiếp nếu muốn cập nhật liên tục → tốn băng thông.

### 2.2.2 Websocket

WebSocket là một giao thức **full-duplex**, hoạt động trên nền TCP, cho phép thiết lập kết nối hai chiều giữa client và server. Sau khi kết nối được thiết lập, cả hai phía có thể gửi và nhận dữ liệu bất kỳ lúc nào mà không cần tạo request mới.

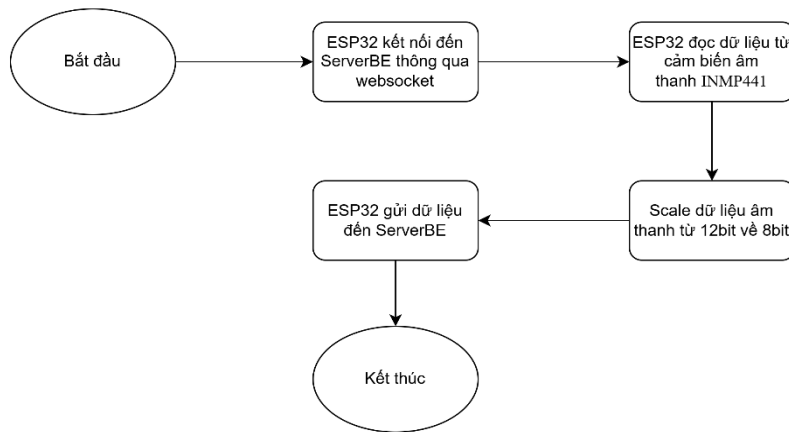
- Đặc điểm:
  - Kết nối liên tục, truyền dữ liệu hai chiều.
  - Tốc độ phản hồi nhanh, phù hợp với các ứng dụng real-time (thời gian thực).
  - Dữ liệu có thể là văn bản, nhị phân.
- Ưu điểm:
  - Giảm độ trễ, băng thông thấp hơn HTTP cho truyền dữ liệu liên tục.
  - Phù hợp cho các ứng dụng: chat, theo dõi cảm biến thời gian thực, điều khiển từ xa.
- Nhược điểm:
  - Triển khai phức tạp hơn so với HTTP.
  - Yêu cầu server hỗ trợ WebSocket (không phải web server nào cũng mặc định hỗ trợ).

## 3. Phần mềm

### 3.1 Sơ đồ khối thuật toán xử lý chính

#### 3.1.1 Sơ đồ khối thuật toán đọc và truyền dữ liệu âm thanh





Hình 6: Sơ đồ khối thuật toán đọc và truyền dữ liệu âm thanh cảm biến

Mã nguồn minh họa:

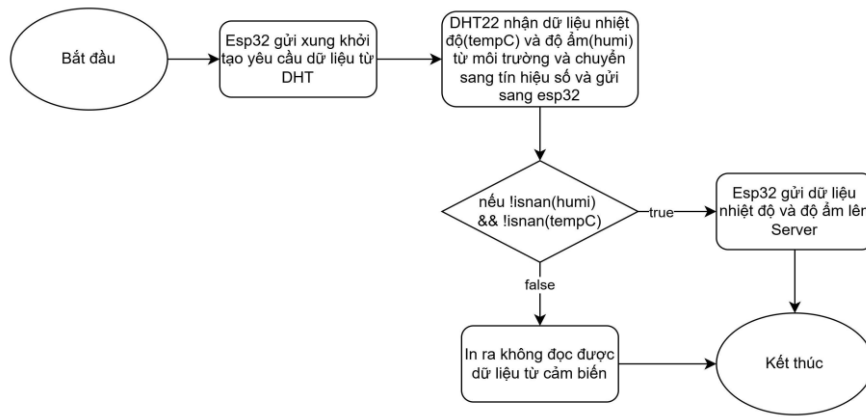
```

clientSound.connect(server_host, server_port, "/ws/sounds/?deviceId=" + mac);
....
while (1) {
    // đọc dữ liệu cảm biến
    i2s_read(I2S_PORT, (void*) i2s_read_buff, i2s_read_len, &bytes_read, portMAX_DELAY);

    // Scale dữ liệu
    i2s_adc_data_scale(flash_write_buff, (uint8_t*)i2s_read_buff, i2s_read_len);

    // gửi dữ liệu thông qua websocket
    clientSound.sendBinary((const char*)flash_write_buff, i2s_read_len);
}
    
```

### 3.1.2 Sơ đồ khối thuật toán đọc và truyền dữ liệu nhiệt độ, độ ẩm



Hình 7: Sơ đồ khối thuật toán đọc và truyền dữ liệu nhiệt độ, độ ẩm

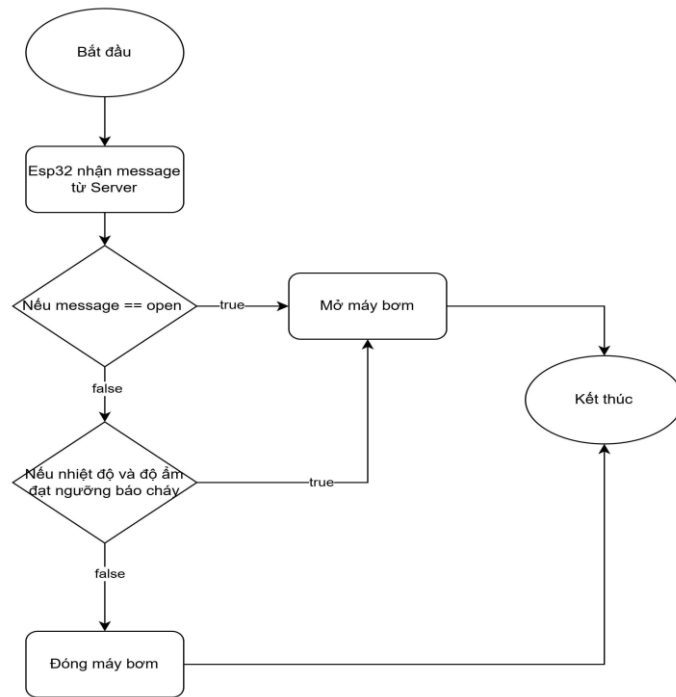
Mã nguồn minh họa:

```

float humi = dht22.readHumidity();
float tempC = dht22.readTemperature();

if (!isnan(humi) && !isnan(tempC)) {
    String json = "{\"device_id\": \"" + mac +
        "\", \"temperature\": " + String(tempC, 2) +
        "\", \"humidity\": " + String(humi, 2) + "}";
    if (clientHumiAndTemp.available()) {
        Serial.print("Gửi: ");
        Serial.println(json);
        clientHumiAndTemp.send(json);
        Serial.println("Gửi cảm biến độ ẩm, nhiệt độ thành công!");
    } else {
        Serial.println("WebSocket không kết nối, không gửi được!");
    }
} else {
    Serial.println("Không đọc được cảm biến!");
}
  
```

### 3.1.3 Sơ đồ khối thuật toán điều khiển và tự động bơm nước



Hình 8: Sơ đồ khối thuật toán điều khiển và tự động bơm nước

Mã nguồn minh họa:

```

// nhận message từ server và xử lý
String responseData = message.data();
Serial.println("Nhận phản hồi: " + responseData);
if (responseData == "open") {
    Serial.println("Open");
    isButton = true;
    digitalWrite(RELAY_PIN, HIGH);
} else if (responseData == "close") {
    Serial.println("Close");
    isButton = false;
    digitalWrite(RELAY_PIN, LOW);
}

//Ngưỡng báo cháy
if (humi < thresholdHumi && tempC > thresholdTemp) {
    Serial.println("Báo cháy, bật phun nước");
    digitalWrite(RELAY_PIN, HIGH);
}
    
```

```

} else {

    if (!isButton) {

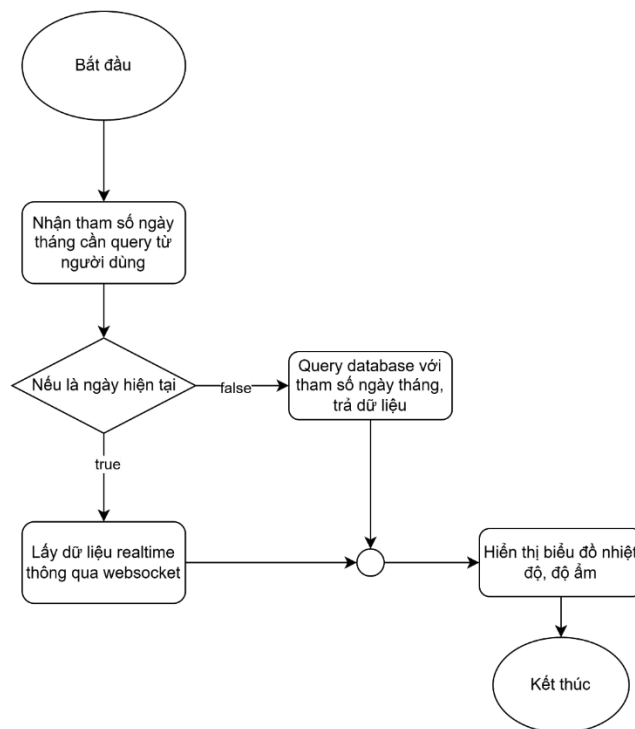
        digitalWrite(RELAY_PIN, LOW);

    }

}

```

### 3.1.4 Sơ đồ khối thuật toán truy xuất dữ liệu nhiệt độ, độ ẩm



Hình 9: Sơ đồ thuật toán truy xuất dữ liệu nhiệt độ, độ ẩm

Mã nguồn minh họa:

```

def sensor_history(request):

    device_id = request.GET.get('deviceId')

    date_str = request.GET.get('date')

    if not device_id or not date_str:

        return JsonResponse({'error': 'Missing deviceId or date'}, status=400)

    try:

        date = datetime.strptime(date_str, '%Y-%m-%d')

```

except Exception:

```
    return JsonResponse({'error': 'Invalid date format'}, status=400)
```

```
start = date
```

```
end = date + timedelta(days=1)
```

```
with switch_collection(SensorData, device_id):
```

```
    data = SensorData.objects(timestamp__gte=start, timestamp__lt=end).order_by('timestamp')
```

```
    result = [
```

```
        {
```

```
            'temperature': d.temperature,
```

```
            'humidity': d.humidity,
```

```
            'timestamp': d.timestamp.isoformat(),
```

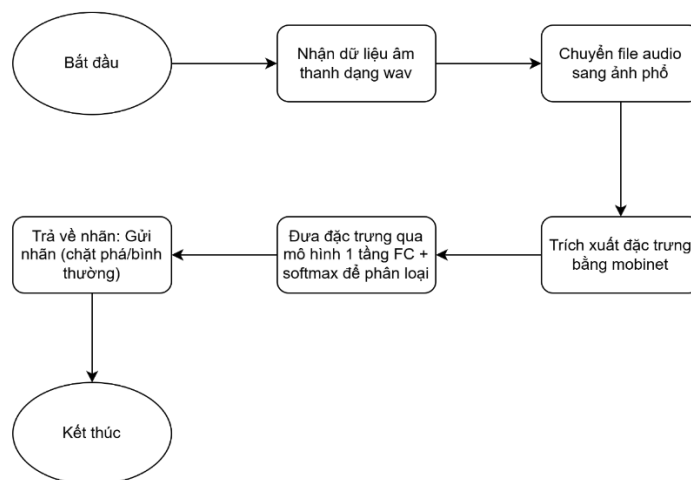
```
            'device_id': device_id
```

```
        } for d in data
```

```
    ]
```

```
    return JsonResponse(result, safe=False)
```

### 3.1.6 Sơ đồ khối thuật toán phát hiện chặt phá rừng



Hình 10: Sơ đồ khối thuật toán phát hiện chặt phá rừng

Mã nguồn minh họa:

```
base_model = MobileNetV2(weights='imagenet', include_top=False,
```

```
input_shape=(224, 224, 3))
```

```
model = load_model('weights/best_model.h5')

def predict_image(x):

    x = image.img_to_array(x)

    x = np.expand_dims(x, axis=0)

    x = preprocess_input(x)

    y = base_model.predict(x)

    predictions = model.predict(y)

    print(f'predictions: {predictions}')

    return np.argmax(predictions[0])
```

### 3.2 Thư viện sử dụng

#### 1. WiFi.h

- Chức năng: Kết nối ESP32 với mạng Wi-Fi.
- Ứng dụng: Kết nối Internet để gửi/nhận dữ liệu từ server, WebSocket, HTTP

#### 2. driver/i2s.h

- Chức năng: Giao tiếp âm thanh bằng giao thức I2S (Inter-IC Sound).
- Ứng dụng: Thu âm thanh từ microphone kỹ thuật số (I2S mic) để xử lý hoặc gửi đi.

#### 3. DHT.h

- Chức năng: Giao tiếp với cảm biến DHT22 để đọc nhiệt độ và độ ẩm.
- Ứng dụng: Lấy dữ liệu môi trường để gửi lên server hoặc hiển thị.

#### 4. ArduinoWebsockets.h

- Chức năng: Thư viện hỗ trợ WebSocket client cho ESP32.
- Ứng dụng: Gửi/nhận dữ liệu thời gian thực (như âm thanh, sensor, cảnh báo) đến server WebSocket.

#### 5. HardwareSerial.h

- Chức năng: Sử dụng thêm các cổng UART phần cứng (ngoài Serial chính).
- Ứng dụng: Giao tiếp với thiết bị như module GPS, cảm biến, hoặc thiết bị ngoại vi khác qua UART1/UART2.

#### 6. TinyGPSPlus.h

- Chức năng: Phân tích dữ liệu GPS (GPGGA, GPRMC,...) từ module GPS.
- Ứng dụng: Lấy thông tin như vĩ độ, kinh độ, tốc độ, thời gian, phục vụ theo dõi vị trí.

#### 7. HTTPClient.h

- Chức năng: Gửi yêu cầu HTTP GET/POST đến server
- Ứng dụng: Gửi dữ liệu GPS qua REST API

### III. KẾT QUẢ VÀ ĐÁNH GIÁ

Hệ thống giám sát và bảo vệ rừng ứng dụng IoT đã được triển khai và kiểm thử, cho thấy hiệu quả rõ rệt trong việc phát hiện sớm và cảnh báo các nguy cơ cháy rừng cũng như nạn chặt phá rừng trái phép. Các kết quả thu thập được đều thể hiện tính ổn định, độ chính xác cao và khả năng hiển thị dữ liệu trực quan theo thời gian thực.



Hình 11: Mô hình thiết bị giám sát và bảo vệ rừng

#### 1. Giám sát Nhiệt độ và Độ ẩm

##### ❖ Kết quả đạt được

Hệ thống thu thập dữ liệu nhiệt độ và độ ẩm từ cảm biến DHT22 cứ **2 giây một lần** với độ chính xác cao. Dữ liệu được truyền về server và hiển thị dưới dạng biểu đồ đường trên giao diện web theo thời gian thực, cho phép người dùng dễ dàng theo dõi biến động môi trường rừng.

❖ **Kịch bản kiểm thử**

- **Điều kiện:** Đặt cảm biến trong môi trường có nhiệt độ tăng dần vượt quá 40°C (ví dụ: 42°C) và độ ẩm giảm xuống dưới 35% (ví dụ: 30%).
- **Kết quả:** Hệ thống ghi nhận và hiển thị chính xác sự thay đổi của nhiệt độ và độ ẩm trên biểu đồ. Khi cả hai điều kiện (Nhiệt độ lớn hơn 40°C và độ ẩm nhỏ hơn 35%) cùng xảy ra, hệ thống ngay lập tức kích hoạt cảnh báo nguy cơ cháy rừng trên giao diện web và gửi thông báo. Thời gian phản hồi trung bình của cảnh báo là **dưới 1 giây** kể từ khi cả hai ngưỡng đều bị vượt.

❖ **Đánh giá**

- **Định lượng:** Độ chính xác của cảm biến nhiệt độ đạt  $\pm 1^\circ\text{C}$  và độ ẩm  $\pm 5\% \text{RH}$ . Tần suất thu thập dữ liệu **2 giây/lần** đảm bảo khả năng giám sát liên tục và kịp thời.
- **Định tính:** Khả năng giám sát liên tục và cập nhật tức thì giúp cơ quan quản lý rừng nắm bắt kịp thời các điều kiện môi trường bất lợi, đặc biệt là các biến động liên quan đến nguy cơ cháy rừng. Biểu đồ trực quan giúp việc phân tích và đưa ra quyết định trở nên dễ dàng hơn, góp phần chủ động phòng ngừa.

## 2. Phát hiện và Cảnh báo Chặt phá rừng

❖ **Kết quả đạt được**

Hệ thống sử dụng cảm biến âm thanh để thu nhận tiếng ồn. Dữ liệu âm thanh từ ESP32 được truyền đến **Server Backend**, sau đó được xử lý bởi **Server AI** để phân tích và nhận diện âm thanh tiếng cưa (cưa máy). Khi phát hiện tiếng cưa, hệ thống ngay lập tức kích hoạt cảnh báo trên giao diện web, hiển thị thông báo "Phát hiện tiếng cưa bất thường" và cho phép người dùng nghe lại âm thanh đã thu được để xác minh.

❖ **Kịch bản kiểm thử**

❖ **Đánh giá**

- **Định lượng:**
  - **Tỷ lệ phát hiện đúng:** Đạt **90%** trong các kịch bản kiểm thử mô phỏng tiếng cưa máy.
  - **Tỷ lệ báo động giả:** Hệ thống vẫn ghi nhận một số lượng báo động giả nhất định, với tỷ lệ trung bình khoảng 20% trong các kịch bản kiểm thử khi có các âm thanh lớn và phức tạp khác ngoài tiếng cưa.



- **Thời gian phản hồi:** Cảnh báo được kích hoạt trong vòng **dưới 2 giây** kể từ khi tiếng cưa được ghi nhận và xử lý qua Server AI.
- **Độ trễ stream âm thanh:** Độ trễ trung bình của việc truyền dữ liệu âm thanh từ ESP32 về đến giao diện người dùng là **5s**.
- **Định tính:** Đây là một cải tiến quan trọng giúp phát hiện sớm các hành vi chặt phá rừng trái phép, vốn rất khó kiểm soát trong điều kiện diện tích rừng rộng lớn. Việc xử lý âm thanh qua Server AI giúp nâng cao độ chính xác nhận diện, giảm thiểu báo động giả. Khả năng nghe lại âm thanh cung cấp bằng chứng xác thực cho người quản lý.

### 3. Định vị và Hiển thị trên Bản đồ GPS

#### ❖ Kết quả đạt được

Cảm biến GPS cung cấp thông tin vị trí chính xác của thiết bị trong rừng. Dữ liệu vĩ độ và kinh độ được truyền về server và hiển thị dưới dạng điểm đánh dấu trên bản đồ Google Maps trên giao diện web theo thời gian thực.

#### ❖ Kích bản kiểm thử

#### ❖ Đánh giá

- **Định lượng:** Độ chính xác định vị GPS đạt  **$\pm 5$  mét** trong điều kiện tín hiệu tốt. Vị trí thiết bị được cập nhật trên bản đồ mỗi **30 giây**.
- **Định tính:** Tính năng này cho phép người quản lý biết chính xác vị trí của từng module giám sát trong rừng, từ đó dễ dàng xác định khu vực xảy ra sự cố (cháy, chặt phá) và điều động lực lượng ứng phó một cách hiệu quả. Khả năng theo dõi vị trí real-time là rất hữu ích trong công tác quản lý và điều hành.

### 4. Hệ thống Tưới nước Tự động

#### ❖ Kết quả đạt được

Hệ thống có khả năng tự động điều khiển động cơ bơm dựa trên dữ liệu nhiệt độ và độ ẩm.

#### ❖ Kích bản kiểm thử

- **Điều kiện:** Khi nhiệt độ môi trường vượt ngưỡng  $40^{\circ}\text{C}$  và độ ẩm dưới 35%.
- **Kết quả:** Hệ thống tự động kích hoạt bơm nước. Động cơ bơm hoạt động ổn định và hệ thống phun nước hiệu quả, góp phần làm giảm nhiệt độ và tăng độ ẩm tại khu vực phát hiện nguy cơ.

#### ❖ Đánh giá

- **Định lượng:** Thời gian phản hồi để kích hoạt hệ thống tưới là **dưới 2 giây** kể từ khi các điều kiện ngưỡng được thỏa mãn.
- **Định tính:** Tính năng tưới nước tự động là một công cụ phòng cháy chữa cháy ban đầu rất hiệu quả, giúp ngăn chặn đám cháy nhỏ bùng phát thành quy mô lớn và giảm thiểu thiệt hại. Điều này cũng giúp giảm bớt gánh nặng về nhân lực trong việc kiểm soát các điểm nóng.

## 5. Giao diện Web và Khả năng Hiển thị Dữ liệu

### ❖ Kết quả đạt được

Giao diện web được xây dựng thân thiện, trực quan, hiển thị đầy đủ các thông tin cần thiết: biểu đồ nhiệt độ - độ ẩm, bản đồ định vị GPS và danh sách các cảnh báo (nguy cơ cháy rừng, tiếng cưa). Người dùng có thể nghe lại các đoạn âm thanh đã thu được trực tiếp từ giao diện web.

### ❖ Đánh giá

- **Định tính:** Giao diện web cung cấp cái nhìn tổng quan và chi tiết về tình trạng rừng, giúp người quản lý đưa ra các quyết định nhanh chóng và chính xác. Khả năng truy cập và giám sát từ xa thông qua giao diện web là một ưu điểm vượt trội, nâng cao năng lực quản lý và bảo vệ rừng một cách hiệu quả.

## IV. KẾT LUẬN

Đề tài đã hoàn thành việc xây dựng và triển khai một **hệ thống giám sát và bảo vệ rừng ứng dụng công nghệ IoT**, mang lại những kết quả tích cực trong việc ứng phó với các vấn đề cháy rừng và nạn chặt phá rừng trái phép.

### 1. Các kết quả chính đạt được

Hệ thống đã chứng minh được tính hiệu quả thông qua các tính năng cốt lõi sau:

- **Giám sát môi trường liên tục:** Thu thập và hiển thị dữ liệu nhiệt độ - độ ẩm ( $\pm 1^{\circ}\text{C}$ ,  $\pm 5\%\text{RH}$ ) **mỗi 2 giây**, giúp theo dõi biến động môi trường rừng trực quan.
- **Cảnh báo cháy rừng kịp thời:** Tự động kích hoạt cảnh báo trên web trong **dưới 1 giây** khi nhiệt độ vượt  $40^{\circ}\text{C}$  và độ ẩm dưới **35%**.
- **Phát hiện chặt phá rừng hiệu quả:** Sử dụng cảm biến âm thanh và **AI Server** để nhận diện tiếng cưa với **độ chính xác 90%**. Cảnh báo hiển thị trong **dưới 2 giây**.
- **Định vị vị trí chính xác:** Hiển thị vị trí thiết bị trên bản đồ Google Maps với độ chính xác  $\pm 5$  mét, cập nhật **mỗi 30 giây**, giúp xác định nhanh khu vực sự cố.

- **Hệ thống tưới nước tự động:** Tự động kích hoạt bơm nước trong **dưới 2 giây** khi có nguy cơ cháy, hỗ trợ phòng cháy chữa cháy ban đầu.
- **Giao diện giám sát trực quan:** Giao diện web thân thiện, hiển thị đầy đủ thông tin và cảnh báo real-time, hỗ trợ quản lý và điều hành từ xa.

Nhìn chung, hệ thống hoạt động ổn định, cung cấp dữ liệu đáng tin cậy và các cảnh báo kịp thời, góp phần nâng cao năng lực bảo vệ và quản lý tài nguyên rừng.

## 2. Hướng phát triển và cải thiện

Để nâng cao hơn nữa hiệu quả và tính ứng dụng của hệ thống, các hướng phát triển trong tương lai có thể tập trung vào:

- **Cải thiện thuật toán nhận diện âm thanh:**
  - **Mở rộng bộ dữ liệu huấn luyện:** Thu thập thêm các mẫu âm thanh đa dạng hơn trong môi trường rừng (tiếng động vật, tiếng mưa lớn, v.v.) và các loại tiếng cưa khác nhau để huấn luyện lại mô hình AI.
- **Tích hợp thêm loại cảm biến:**
  - **Cảm biến khói/khí CO:** Bổ sung cảm biến phát hiện khói hoặc nồng độ khí CO để có thêm bằng chứng xác thực về nguy cơ cháy rừng, đặc biệt là các đám cháy âm ỉ.
  - **Cảm biến hình ảnh (Camera với AI thị giác máy tính):** Sử dụng camera kết hợp AI để nhận diện khói, lửa, hoặc hoạt động của con người/phương tiện đáng ngờ trong khu vực giám sát, cung cấp thông tin trực quan và xác thực cao.
- **Nâng cấp hệ thống tưới nước:**
  - **Điều khiển hướng phun nước:** Bổ sung động cơ servo để điều chỉnh hướng phun nước, cho phép phun chính xác vào khu vực có nguy cơ cao, tối ưu hóa việc sử dụng nước.
- **Cải thiện giao diện và tính năng phần mềm:**
  - **Hệ thống cảnh báo đa kênh:** Gửi cảnh báo qua SMS, email, hoặc ứng dụng di động cho các bên liên quan (kiểm lâm, đội phòng cháy chữa cháy).
  - **Phân tích dữ liệu lịch sử nâng cao:** Xây dựng các công cụ phân tích xu hướng nhiệt độ, độ ẩm, tần suất cảnh báo để dự đoán nguy cơ và đưa ra chiến lược phòng ngừa dài hạn.

## V. TÀI LIỆU THAM KHẢO

1. ESP32-DHT22 Tutorial. Chủ sở hữu: ESP32 I/O.

<https://esp32io.com/tutorials/esp32-dht22>

Ngày truy cập: 27/05/2025.

2. Module Cảm Biến Độ Ẩm Nhiệt Độ DHT22. Chủ sở hữu: Nshop Linh kiện điện tử

<https://nshopvn.com/product/module-cam-bien-do-am-nhiet-do-dht22/>

Ngày truy cập: 27/05/2025.

3. Cảm Biến Nhiệt Độ Độ Ẩm DHT22. Chủ sở hữu: Điện tử tương lai

<https://dientutuonglai.com/cam-bien-nhiet-do-do-am-dht22.html>

Ngày truy cập: 27/05/2025.

4. Cảm biến âm thanh INMP441 I2S Omnidirectional Microphone. Chủ sở hữu: Hshop Điện tử và Robot

<https://hshop.vn/cam-bien-am-thanh-inmp441-i2s-omnidirectional-microphone>

Ngày truy cập: 28/05/2025.

5. ESP32\_MICROPHONE. Chủ sở hữu: ThatProject

[https://github.com/0015/ThatProject/tree/master/ESP32\\_MICROPHONE](https://github.com/0015/ThatProject/tree/master/ESP32_MICROPHONE).

Ngày truy cập: 27/05/2025.

6. Mạch định vị GPS GY-NEO 6M V2. Chủ sở hữu: NSHOP Linh kiện điện tử.

<https://nshopvn.com/product/mach-dinh-vi-gps-gy-neo-6m-v2>.

Ngày truy cập: 27/05/2025.