

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**  
**KHOA CÔNG NGHỆ THÔNG TIN - 1**

# **HỆ THỐNG TƯỚI CÂY THÔNG MINH TÍCH HỢP AI**

**Giảng viên: TS. Kim Ngọc Bách**



## Thành viên Nhóm 6

Mã sinh viên	Họ và tên	Công việc
B22DCCN482	Trịnh Quang Lâm (Nhóm trưởng)	1. <b>Lên kế hoạch</b> , phân chia nhiệm vụ 2. <b>Thuyết trình</b> 3. <b>Viết báo cáo</b> 4. <b>Code Front-end</b>
B22DCCN422	Cao Thị Thu Hương	1. <b>Báo cáo</b> 2. <b>Thuyết trình</b> 3. <b>Code phần cứng</b>
B22DCNN434	Vũ Nhân Kiên	1. <b>Làm slide</b> 2. <b>Thuyết trình</b> 3. <b>Code Back-end</b>
B22DCCN889	Vũ Thế Văn	1. <b>Báo cáo</b> 2. <b>Thuyết trình</b> 3. <b>Code AI</b>

# Nội dung

1. Giới thiệu đề tài
2. Tổng quan hệ thống
3. MQTT và Socket.io
4. Server
5. Thiết bị IoT
6. AI
7. Tổng kết



# 1. GIỚI THIỆU ĐỀ TÀI



## 1.1. Lý do lựa chọn đề tài

- Nông nghiệp hiện nay đang đối mặt với biến đổi khí hậu và thiếu hụt nguồn nước, đòi hỏi giải pháp tưới tiêu thông minh, tiết kiệm và chính xác hơn
  - Việc tưới thủ công tốn thời gian, dễ gây lãng phí nước hoặc ảnh hưởng năng suất cây trồng nếu không đúng thời điểm
  - Sự phát triển của **IoT** và **trí tuệ nhân tạo (AI)** mở ra khả năng tự động hóa và tối ưu lịch tưới dựa trên dữ liệu cảm biến và dự báo thời tiết
- Ứng dụng thực tế, có tính khả thi cao, phù hợp xu hướng nông nghiệp thông minh (Smart Agriculture) trong kỷ nguyên 4.0.



## 1.2. Mục tiêu hệ thống

Dự án được xây dựng nhằm mang lại một giải pháp nông nghiệp thông minh, hiện đại và hiệu quả:

- Tự động hóa giám sát và điều khiển
- Tối ưu hóa việc tưới tiêu bằng AI
- Giám sát và tương tác từ xa



## 1.3. Phạm vi hệ thống

- **Phạm vi triển khai:** hệ thống được thiết kế và triển khai cho quy mô nhỏ như vườn gia đình, ban công hoặc một khu vực canh tác thử nghiệm
- **Phạm vi phần cứng:**
  - Bộ điều khiển trung tâm: vi điều khiển **ESP32**
  - Cảm biến: tích hợp các cảm biến môi trường gồm:
    - Cảm biến đo áp suất: **BME280**
    - Cảm biến độ ẩm đất
  - Thiết bị: 1 bơm nước mini 12V được điều khiển qua relay.
- **Ngoài phạm vi:** Chưa bao gồm chức năng mở rộng như:
  - Bón phân
  - Xem xét sâu bệnh
  - Quy mô lớn hơn
  - FOTA Update Firmware từ xa



## 1.4. Công nghệ sử dụng

- **Database:**
  - **MongoDB Atlas:** Dịch vụ cơ sở dữ liệu NoSQL trên nền tảng đám mây
- **Git (Github Server):** Dùng để quản lý mã nguồn, theo dõi thay đổi và hỗ trợ làm việc nhóm
- **Commucation / Cloud Service:**
  - **MQTT Protocol:** Được sử dụng làm giao thức truyền thông giữa các thiết bị IoT và Server
  - **HiveMQ Cloud:** Nền tảng MQTT broker trên đám mây, dùng để kết nối và quản lý các thiết bị IoT thông qua giao thức MQTT
- **Model AI:** Sử dụng mô hình **XGBoost** (ổn định với dữ liệu dạng bảng, tận dụng tốt đặc trưng, suy luận nhanh)





## 1.4. Công nghệ sử dụng

- **Môi trường phát triển:**
  - **PlatformIO:** công cụ chính để viết mã, biên dịch và quản lý kết nối với phần cứng IoT.
  - **Visual Studio Code:** được sử dụng để phát triển cả frontend và backend, đồng thời dễ dàng quản lý dự án với Git.
- **Frontend**
  - **ReactJS:** Dùng để xây dựng giao diện người dùng (UI)
  - **TypeScript:** Ngôn ngữ chính để viết Logic
- **Backend**
  - **NodeJS & ExpressJS:** framework, môi trường chạy JavaScript phía Server
  - **TypeScript:** Ngôn ngữ chính để viết Logic



## 2. TỔNG QUAN HỆ THỐNG



## 2.1. Kiến trúc tổng thể

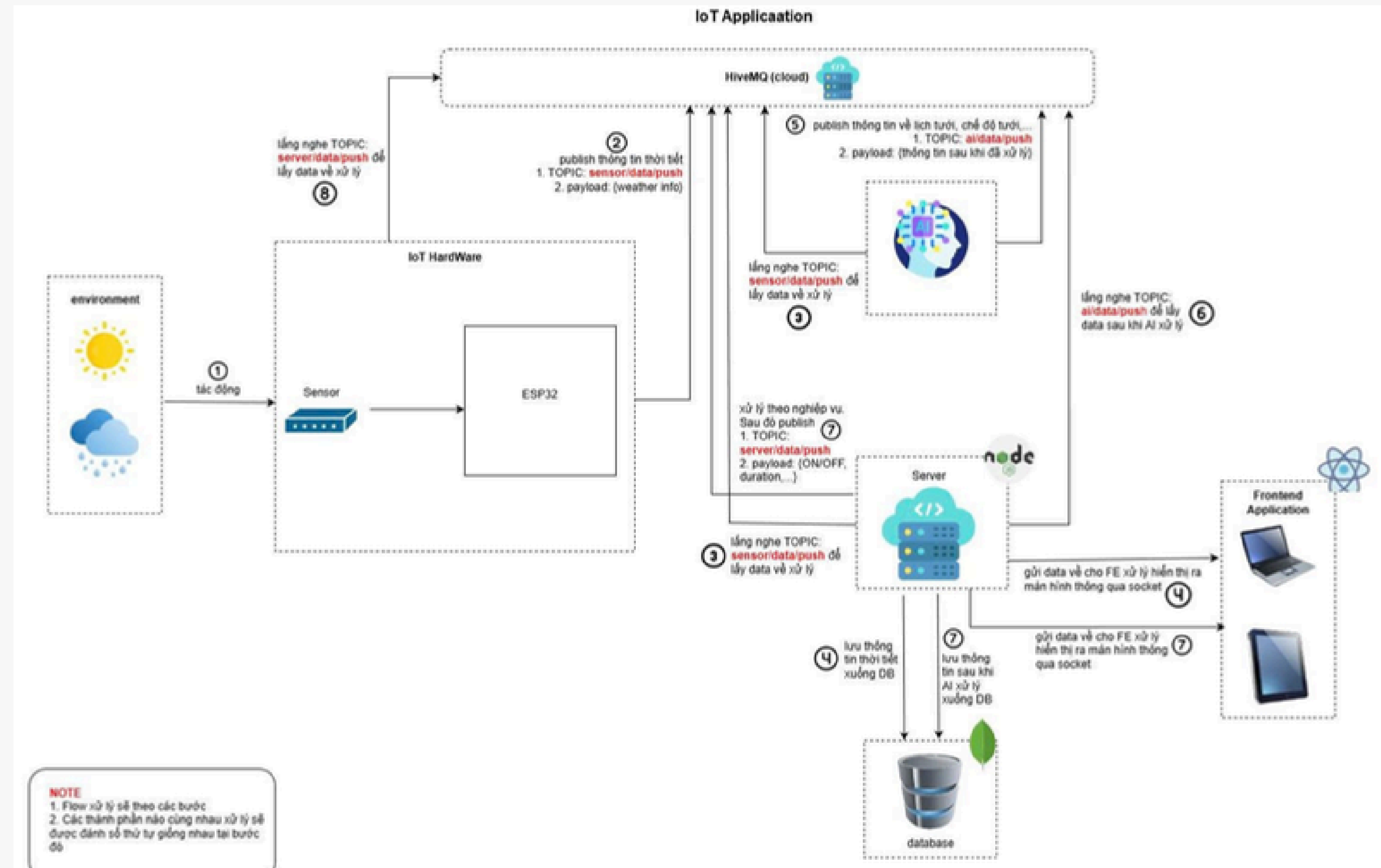
Hệ thống tưới cây thông minh gồm 3 phần chính:

- **Phần cảm biến và điều khiển:** ESP32 kết nối với các cảm biến môi trường (BME280, cảm biến độ ẩm đất) để thu thập dữ liệu
- **Phần xử lý dữ liệu và trí tuệ nhân tạo:** Backend xử lý dữ liệu cảm biến, AI dự đoán mưa bằng mô hình học sâu
- **Phần giao diện người dùng:** Ứng dụng web hiển thị dữ liệu, lịch tưới và cho phép điều khiển bơm từ xa



## 2.1. Kiến trúc tổng thể

### Sơ đồ tổng quan hệ thống



## 2.2. Môi trường và các ràng buộc

### 2.2.1. Môi trường phát triển và môi trường hoạt động

Môi trường phát triển	Môi trường hoạt động
Arduino IDE: Lập trình và nạp chương trình cho ESP32.	Kết nối Wi-Fi nội bộ.
Visual Studio Code: Phát triển frontend (ReactJS) và backend (NodeJS).	Trình duyệt web hiển thị giao diện người dùng.
MongoDB Atlas: Cơ sở dữ liệu NoSQL lưu trữ dữ liệu cảm biến.	Máy chủ NodeJS hoặc Cloud lưu trữ và xử lý dữ liệu
Git/Github: Quản lý mã nguồn và làm việc nhóm.	

## 2.2. Môi trường và các ràng buộc

### 2.2.2. Ràng buộc giả định

- Cảm biến hoạt động ổn định với chu kỳ đo 15 giây.
- Hệ thống yêu cầu kết nối Internet liên tục.
- AI cần ít nhất 60 ngày dữ liệu để học và dự đoán chính xác.
- ESP32 có giới hạn tài nguyên nên chỉ đảm nhận nhiệm vụ thu thập và gửi dữ liệu, không chạy mô hình AI trực tiếp.
- Nếu mất kết nối, hệ thống tự chuyển sang chế độ tưới thủ công theo lịch mặc định, hoặc cho phép người dùng điều khiển thủ công/điều khiển từ xa



## 2.3. Các yêu cầu

### 2.3.1. Yêu cầu chức năng

- Thu thập và gửi dữ liệu cảm biến (nhiệt độ, độ ẩm, áp suất, độ ẩm đất).
- Lưu trữ dữ liệu và phân tích trên server.
- AI dự đoán khả năng mưa và lượng mưa trong 60 phút tới
- Điều khiển tự động bơm tưới theo ngưỡng độ ẩm và kết quả dự báo.
- Cho phép người dùng theo dõi và điều khiển thủ công qua giao diện web.
- Ghi lại lịch sử hoạt động và nhật ký tưới cây



## 2.3. Các yêu cầu

### 2.3.2. Yêu cầu phi chức năng

- Thời gian phản hồi  $< 300\text{ms}$  cho mỗi yêu cầu dự đoán.
- Hệ thống ổn định, tự phục hồi khi kết nối lại.
- Có khả năng mở rộng, thêm cảm biến hoặc khu vực tưới.
- Giao diện thân thiện, dễ sử dụng.
- Đảm bảo bảo mật cơ bản (xác thực truy cập web).





### 3. MQTT VÀ SOCKET.IO



## 3.1. MQTT

- MQTT là một giao thức truyền thông tin nhắn (messaging protocol) được thiết kế đặc biệt cho IoT. Nó hoạt động dựa trên mô hình Publish, Subscribe
- Đặc điểm chính và quan trọng nhất: **Tách rời**
  1. Không gian: Publisher và Subscriber chỉ cần biết địa chỉ của **Broker** (HiveMQ), hoạt động qua các **topic** đăng ký
  2. Thời gian: Publisher và Subscriber không cần phải online cùng một lúc. Publisher có thể gửi tin nhắn ngay cả khi Subscriber đang offline. Broker sẽ giữ tin nhắn đó và giao cho Subscriber ngay khi Subscriber kết nối trở lại.
  3. Đồng bộ: Cả hai bên đều hoạt động bất đồng bộ. Publisher gửi tin xong là quên luôn (fire-and-forget), không cần chờ Subscriber phản hồi.



## 3.2. Socket.io

- Là một thư viện giúp giao tiếp thời gian thực, hai chiều giữa Server và Client.
- Nó hoạt động dựa trên Sự kiện. Thay vì Client liên tục request, Server sẽ chủ động đẩy dữ liệu xuống Client ngay khi có.
- Vai trò trong dự án: Socket.IO đóng vai trò là "Người đưa tin" (Messenger) từ Server đến giao diện người dùng (Dashboard).



## 4. SERVER



## 4.1. Kiến trúc Backend

### - NodeJS(22.17.1) với ExpressJS:

- NodeJS cung cấp môi trường chạy JavaScript phía Server.
- ExpressJS là framework nhẹ trên NodeJS, dùng xây dựng các API RESTful, giúp kết nối frontend với cơ sở dữ liệu và các thiết bị.

### - Vai trò trong dự án:

1. **Tạo máy chủ web:** tạo ra một máy chủ lắng nghe các kết nối từ bên ngoài
2. **Định tuyến các API:** Đây là vai trò quan trọng nhất, nó "chỉ đường" cho các yêu cầu (request)
3. **Xử lý yêu cầu và phản hồi**
4. **Xử lý middleware**



## 4.2. Cơ sở dữ liệu: Mongo Atlas

- **Mongo Atlas:** Dịch vụ cơ sở dữ liệu NoSQL trên nền tảng đám mây
  - Chuyên dùng để lưu trữ dữ liệu dưới dạng tài liệu (document) JSON.
  - Nó giúp có một CSDL chạy 24/7 mà không cần tự mình cài đặt, bảo trì hay sao lưu máy chủ
- Vai trò trong dự án:
  - Lưu trữ dữ liệu từ các thiết bị IoT, lịch tưới và thông tin dự báo cho AI đưa ra
  - Đồng bộ với Backend và hỗ trợ truy xuất nhanh chóng



## 4.3. Vai trò của Server trong dự án

- Server là thành phần duy nhất kết nối tất cả các bộ phận khác (Cảm biến, CSDL, Giao diện Web, và Model AI).
  - **Nhận dữ liệu (MQTT):** Lắng nghe (Subscribe) topic để nhận dữ liệu nhiệt độ, độ ẩm... thô từ ESP32; nhận lịch tưới và dự đoán từ model AI trả về.
  - **Đẩy ra Giao diện (Socket.IO):** Ngay lập tức đẩy (Emit) dữ liệu đã xử lý qua sự kiện cho các client React đang kết nối, giúp cập nhật biểu đồ real-time.
  - **Gửi đến Cảm biến (MQTT):** Gửi (Publish) lệnh điều khiển đến thiết bị thông qua topic
  - **Phản hồi Giao diện:** Cung cấp các API cho các chức năng.
- Cầu nối Cơ sở dữ liệu:
  - Kết nối: Khởi tạo và duy trì kết nối đến MongoDB Atlas bằng Mongoose.
  - Lưu trữ: Là thành phần duy nhất được phép ghi (write) dữ liệu vào CSDL



## 5. THIẾT BỊ IOT



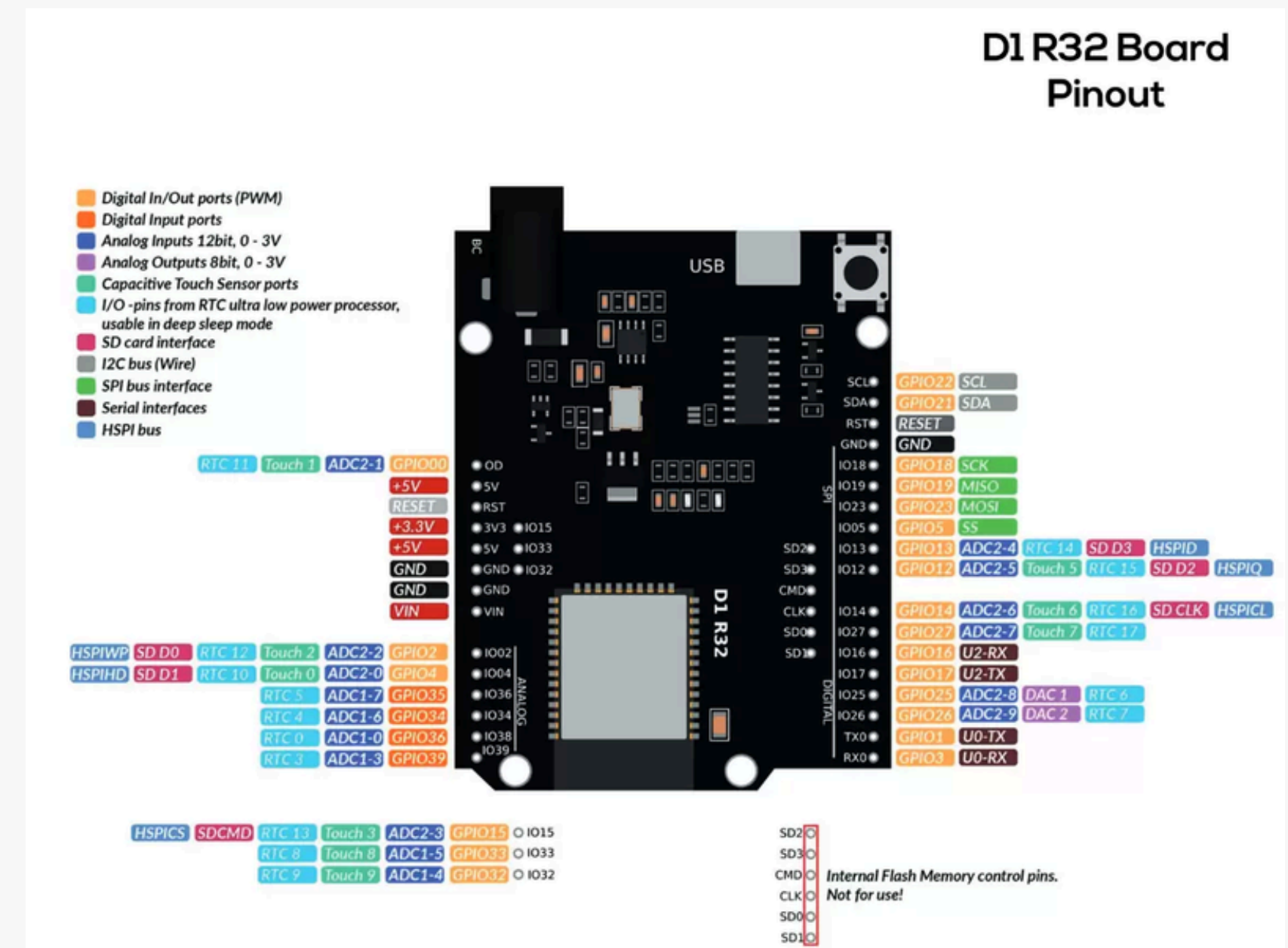


## 5.1. Thành phần phần cứng

### 5.1.1. Vi điều khiển ESP32

Sử dụng bo mạch phát triển **ESP32**  
**WeMos D1 R32** tích hợp vi xử lý  
**ESP32-WROOM-32**

- Hỗ trợ WiFi, Bluetooth, tương thích với Arduino UNO form factor.



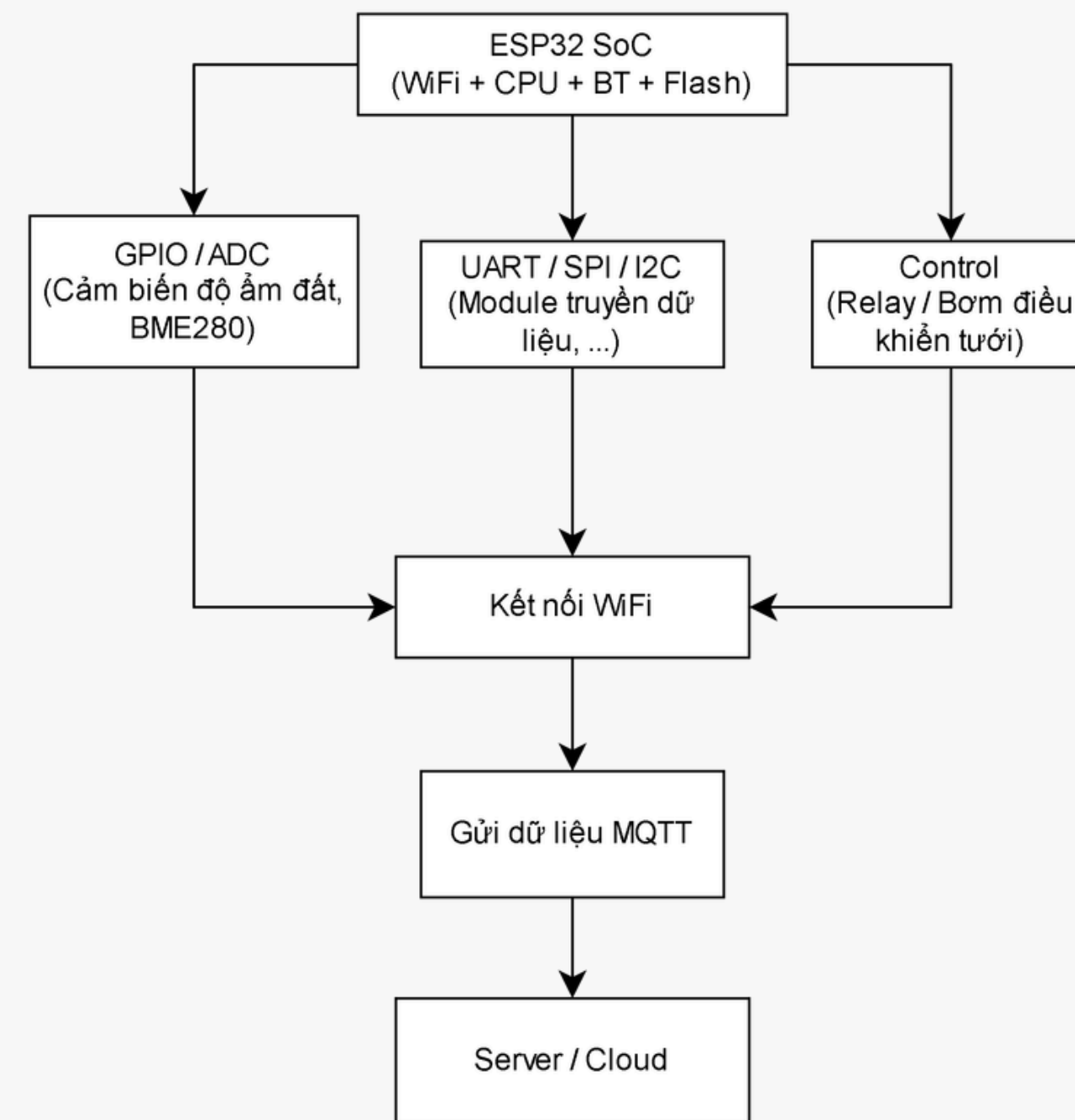
Wemos D1 R32 ESPDuino Pinout

## 5.1. Thành phần phần cứng

### 5.1.1. Vi điều khiển ESP32

Sử dụng bo mạch phát triển **ESP32 WeMos D1 R32** tích hợp vi xử lý **ESP32-WROOM-32**

- Hỗ trợ WiFi, Bluetooth, tương thích với Arduino UNO form factor.

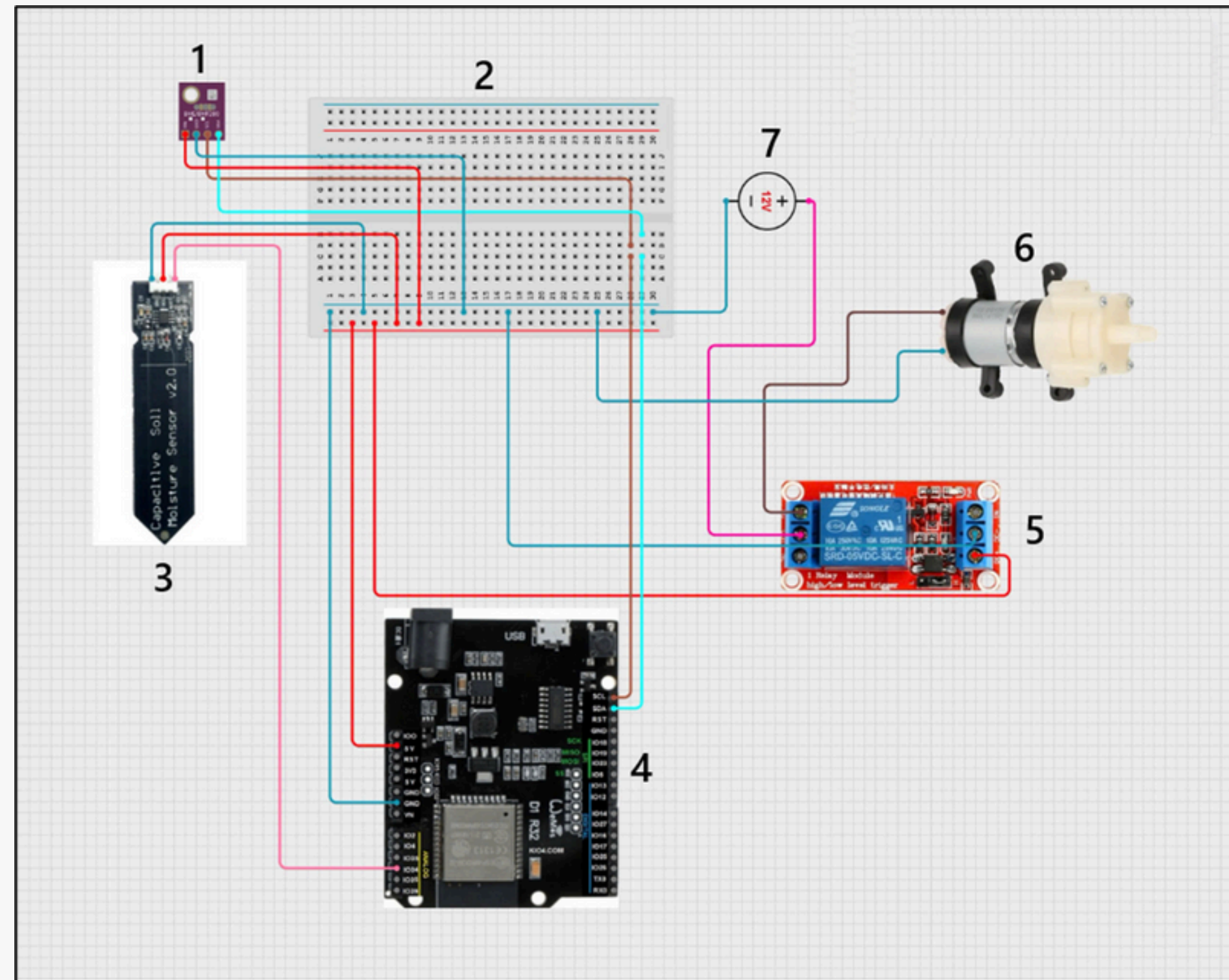


Sơ đồ khối hoạt động cơ bản

## 5.2. Sơ đồ kết nối thiết bị

Trong đó:

- Cảm biến môi trường BME280
- Breadboard (Bo mạch test)
- Cảm biến độ ẩm đất
- ESP32 WeMos D1 R32
- Module Relay 1 kênh (Công tắc điện tử)
- Máy bơm nước Mini
- Nguồn 12V

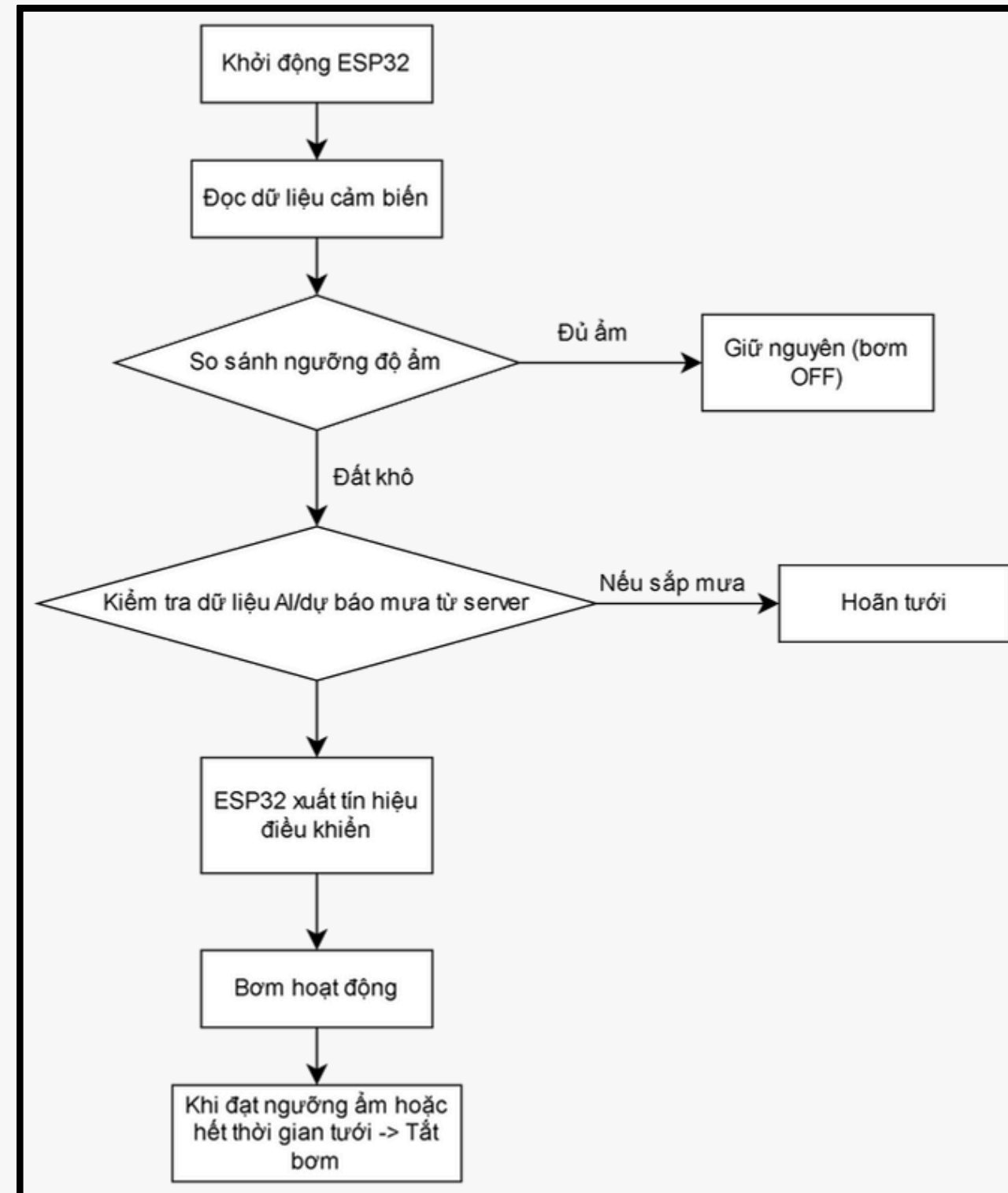


## 5.2. Nguyên lý hoạt động cảm biến

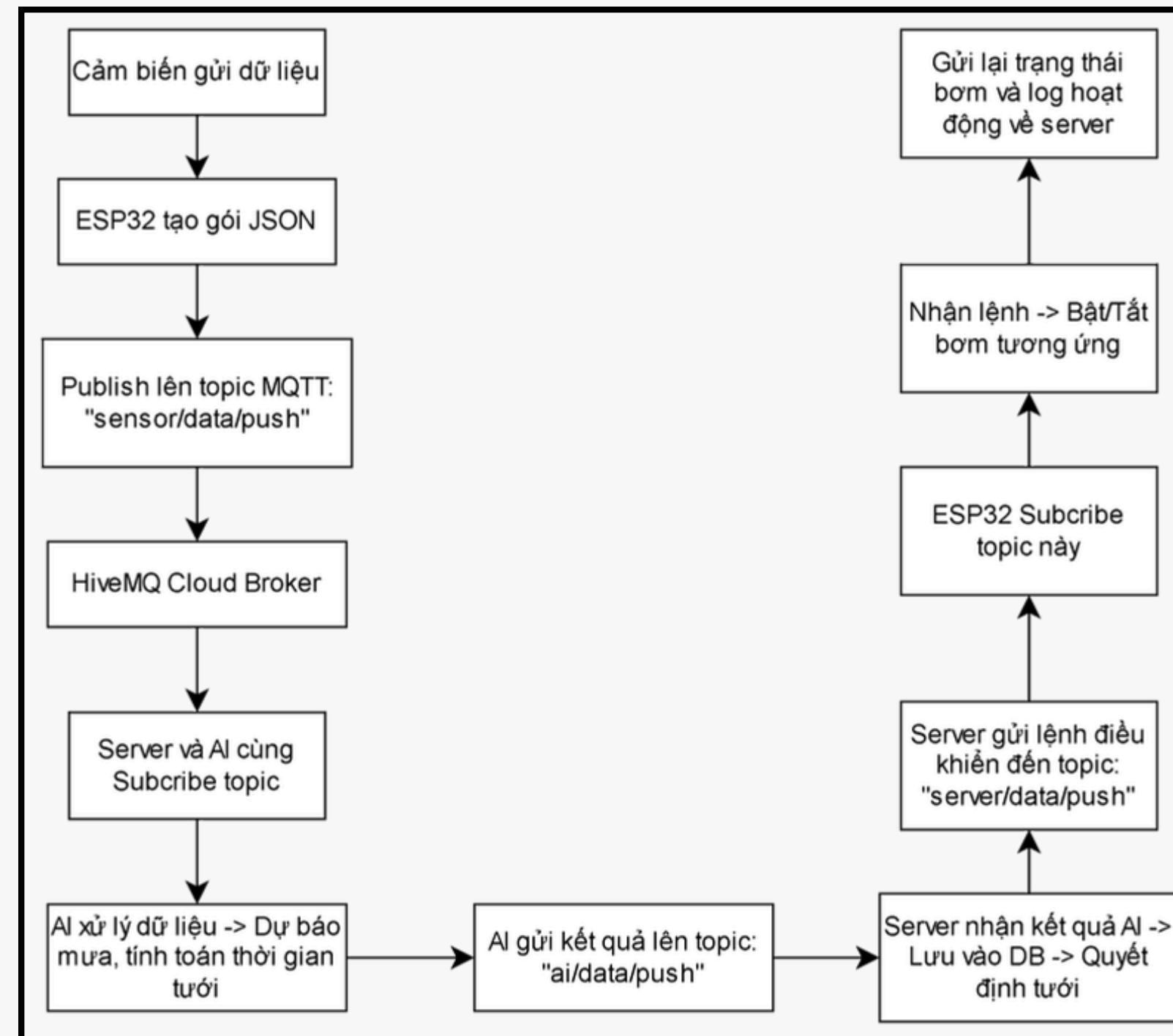
Cảm biến	Giới thiệu chung	Nguyên lý hoạt động	Ưu điểm	Nhược điểm
Độ ẩm đất loại điện dung	<ul style="list-style-type: none"> <li>- Đo sự thay đổi hằng số điện môi của đất</li> <li>- Bền hơn, ít bị ăn mòn, kết quả ổn định hơn.</li> </ul>	<ol style="list-style-type: none"> <li>1. Độ ẩm đất thay đổi → Điện dung cảm biến thay đổi</li> <li>2. Mạch cảm biến chuyển sang tín hiệu và gửi vi điều khiển</li> </ol>	<ul style="list-style-type: none"> <li>• Chống ăn mòn, độ bền cao</li> <li>• Giá rẻ, dễ tìm, dễ sử dụng, tương thích tốt</li> </ul>	<ul style="list-style-type: none"> <li>• Cần hiệu chỉnh ban đầu để có kết quả đúng</li> <li>• Để lâu trong đất vẫn dễ bị ăn mòn</li> </ul>
Cảm biến môi trường BME280	Cảm biến được tích hợp 3 cảm biến đo nhiệt độ, độ ẩm và áp suất khí quyển với độ chính xác cao	<ol style="list-style-type: none"> <li>1. Ba loại cảm biến đo thông số trong môi trường</li> <li>2. Chip BME280 xử lý dữ liệu và xuất ra dạng số qua giao tiếp I<sup>2</sup>C hoặc SPI</li> </ol>	<ul style="list-style-type: none"> <li>• Tiện lợi, tích hợp cao, độ chính xác cao</li> <li>• Kích thước nhỏ, dùng ít điện năng</li> <li>• Giao tiếp linh hoạt</li> </ul>	<ul style="list-style-type: none"> <li>• Giá thành cao hơn cảm biến đơn chức</li> <li>• Cần thư viện để xử lý dữ liệu</li> </ul>



## 5.3. Điều khiển thiết bị bơm



## 5.4. Luồng gửi/nhận dữ liệu về/từ Server



## 6.1. Bài toán & Động lực

Trong hệ thống IoT tưới cây tự động, một trong những thách thức lớn nhất là quyết định thời điểm tưới tối ưu. Việc tưới không đúng lúc dẫn đến hai vấn đề nghiêm trọng:

- Vấn đề 1: Tưới khi sắp mưa → Lãng phí nước (30-40%), lãng phí điện, đất úng
- Vấn đề 2: Không tưới khi cây cần → Cây stress, năng suất giảm

Giải pháp AI:

- Dự báo mưa 30-60 phút tới với độ chính xác cao
- Dựa trên sensors tại chỗ (hyperlocal forecasting)
- Real-time inference mỗi 5 phút
- Mục tiêu: Quyết định tưới/không tưới thông minh, tiết kiệm 30-50% nước, tạo lịch tưới trong tuần tiếp theo



## 6.2. Phương pháp và thuật toán

Bài toán ML:

- Classification: Có/không mưa → Quyết định tưới
- Regression: Lượng mưa (mm) → Tính lượng nước cần

Tiêu chí	XGBoost	Deep Learning
Độ chính xác	87%	85%
Tốc độ	< 1m/s	20m/s
Dữ liệu cần	60 ngày	1 - 2 năm
Model size	800KB	25MB



## 6.3. Dữ liệu và Features

- Sử dụng thêm OpenWeatherApi lấy các dữ liệu về thời tiết từ ra sinh ra thêm các feature
- Các dữ liệu các sensor sẽ được lấy từ HiveMQ
- Trong đó ta cần lưu ý một số feature quan trọng để nhận diện mưa:
  - pressure\_delta: Cho thấy được sự thay đổi của áp suất
  - rh\_pct: Độ ẩm không khí cao
  - pressure\_hpa: Áp suất tuyệt đối thấp
  - rain\_in\_last\_15m(30m): Sự kiện có mưa trong 15, 30 phút trước
  - temp\_c\_mean30: Nhiệt độ trung bình trong 30 phút

## 6.4. Huấn luyện & Đánh giá

### 1. Chuẩn bị data

- 70% train, 15% validation, 15% test
- Xử lý imbalanced (mưa 10%, không mưa 90%)

### 2. Train XGBoost

- 1200 decision trees
- Learning rate: 0.03 (học chậm, ổn định)
- Early stopping: tự dừng khi không cải thiện

### 3. Tìm threshold tối ưu: Test nhiều giá trị, chọn balance tốt nhất

### 4. Một số metrics đánh giá:

- Accuracy: Tỷ lệ dự đoán đúng tổng thể
- Recall: Bắt được bao nhiêu % trường hợp mưa
- Precision: Dự đoán mưa, đúng bao nhiêu %
- AUC-ROC: Khả năng phân biệt mưa/không mưa

## 6.5. Hệ thống & Kết quả

Các tính năng chính:

1. Dự báo mưa ngắn hạn (30-60 phút):
  - Update mỗi 5 phút
  - Quyết định tưới/không tưới NGAY
2. Lập lịch tưới tự động (7 ngày):
  - Kết hợp: AI forecast + Độ ẩm đất + Loại cây
  - Tưới vào thời điểm tối ưu (sáng sớm/chiều mát)
  - Tránh tưới khi dự báo mưa
3. Những cải thiện so với phiên bản không sử dụng AI:
  - Tiết kiệm nước khoảng 30-40%
  - Năng suất tăng từ 10-20%
  - Tiết kiệm điện khoảng 15-20%



## 7. TỔNG KẾT



## 7. Tổng kết

- Hệ thống Tưới cây thông minh ứng dụng AI để lên lịch tưới hàng tuần là một bước tiến nhỏ nhưng có ý nghĩa trong việc ứng dụng IoT và Trí tuệ nhân tạo vào nông nghiệp hiện đại.
- Bằng cách kết hợp vi điều khiển ESP32, các cảm biến môi trường , bộ điều khiển bơm tự động và mô hình AI dự báo mưa, nhóm đã xây dựng được một giải pháp có khả năng tự động hóa việc tưới tiêu, tiết kiệm nước, và tăng hiệu quả chăm sóc cây trồng





**Thank you for listening!**