

TRƯỜNG ĐẠI HỌC ĐÀ LẠT  
KHOA CÔNG NGHỆ THÔNG TIN

-----



**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**  
**ĐỀ TÀI**  
**TÌM HIỂU IoT VÀ XÂY DỰNG ỨNG DỤNG**

*Giảng viên hướng dẫn:* TS. Nguyễn Thị Lương

*Sinh viên thực hiện:* 2110054 Hồ Văn Hưng  
2113005 Lê Hà Hiếu Nghĩa  
2115201 Vy Nhật Duy

***Đà Lạt, 04/2025***

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Đà Lạt, ngày      tháng      năm

Giáo viên hướng dẫn

[Ký và ghi rõ họ tên]

## LỜI CẢM ƠN

Trước tiên, nhóm chúng em xin bày tỏ lòng biết ơn sâu sắc đến cô Nguyễn Thị Lương – giảng viên hướng dẫn đã tận tình chỉ bảo, hỗ trợ và đồng hành cùng nhóm trong suốt quá trình thực hiện đồ án tốt nghiệp với đề tài Hệ thống quản lý trang trại. Những góp ý quý báu và sự tận tâm của cô là nguồn động lực to lớn giúp chúng em hoàn thành đồ án một cách hiệu quả nhất.

Chúng em cũng xin chân thành cảm ơn quý thầy, cô trong Khoa Công nghệ Thông tin – Trường Đại học Đà Lạt đã trang bị cho chúng em nền tảng kiến thức vững chắc thông qua các môn học chuyên ngành. Đồng thời, thầy cô cũng đã tạo mọi điều kiện thuận lợi để chúng em có thể học tập, nghiên cứu và triển khai đồ án một cách tốt nhất.

Do giới hạn về thời gian cũng như kinh nghiệm thực tiễn còn hạn chế, trong quá trình thực hiện đồ án, nhóm em khó tránh khỏi những thiếu sót. Rất mong nhận được sự góp ý, chỉ dẫn từ quý thầy cô để chúng em có thể hoàn thiện hơn, đồng thời rút ra nhiều bài học quý giá cho công việc sau này.

Một lần nữa, nhóm chúng em xin trân trọng gửi lời cảm ơn đến Ban lãnh đạo nhà trường, quý thầy cô trong Khoa Công nghệ Thông tin, cô Nguyễn Thị Lương – giảng viên hướng dẫn, quý thầy cô đã hỗ trợ, động viên chúng em trong suốt quá trình thực hiện đồ án.

Kính chúc quý thầy cô luôn mạnh khỏe, hạnh phúc và thành công trong sự nghiệp trồng người.

Nhóm chúng em xin chân thành cảm ơn!

**Trường Đại học Đà Lạt**  
**Khoa Công nghệ Thông tin**

----☒----

**ĐỀ CƯƠNG THỰC HIỆN ĐỒ ÁN**

**Tên đề tài:** Tìm Hiểu IoT Và Xây Dựng Ứng Dụng

**Sinh viên thực hiện:**

STT	Họ và Tên	MSSV	Lớp	Email Liên Hệ
1	Hồ Văn Hưng	2110054	CTK45-PM	2110054@dlu.edu.vn
2	Lê Hà Hiếu Nghĩa	2113005	CTK45-PM	2113005@dlu.edu.vn
3	Vy Nhật Duy	2115201	CTK45-PM	2115201@dlu.edu.vn

**Giáo viên hướng dẫn:** TS. Nguyễn Thị Lương

**I. Mục tiêu đề tài**

Dự án tập trung vào việc tìm hiểu IoT và xây dựng ứng dụng nhằm tự động hóa quá trình giám sát và quản lý các yếu tố môi trường trong chuồng trại chăn nuôi, bao gồm nhiệt độ, độ ẩm không khí, ánh sáng và hệ thống thông gió. Việc ứng dụng các cảm biến thông minh giúp thu thập dữ liệu thời gian thực về điều kiện sống của vật nuôi, từ đó kịp thời điều chỉnh môi trường để đảm bảo sức khỏe và sự phát triển tối ưu. Hệ thống này không chỉ giúp giảm thiểu công sức giám sát thủ công mà còn hạn chế rủi ro từ các biến động môi trường, nâng cao hiệu quả chăn nuôi.

Bên cạnh việc giám sát, hệ thống còn được tích hợp chức năng điều khiển tự động các thiết bị như hệ thống quạt thông gió, ánh sáng, hệ thống sưởi. Các thiết bị này sẽ tự động điều chỉnh hoạt động dựa trên dữ liệu thu thập từ cảm biến, đảm bảo vật nuôi luôn được chăm sóc trong điều kiện lý tưởng. Việc tự động hóa này không chỉ giúp tối ưu hóa quá trình chăm sóc vật nuôi mà còn góp phần tiết kiệm nguồn lực, giảm thiểu lãng phí thức ăn, nước uống và chi phí vận hành.

Cuối cùng, dự án sẽ tiến hành đánh giá hiệu quả của hệ thống trong việc nâng cao năng suất chăn nuôi, giảm tỷ lệ vật nuôi mắc bệnh và tiết kiệm chi phí quản lý. Dựa trên những kết quả thu được, dự án sẽ đề xuất các hướng phát triển tiếp theo, đặc biệt là tích hợp trí tuệ nhân tạo (AI) và học máy (Machine Learning) để dự đoán dịch bệnh, điều chỉnh chế độ dinh dưỡng phù hợp và tối ưu hóa quy trình chăm sóc vật nuôi. Sự kết hợp này hứa hẹn sẽ mang lại một mô hình chăn nuôi thông minh, bền vững và hiệu quả hơn trong thời đại công nghệ 4.0.

## **II. Nội dung đề tài**

### **2.1. Nghiên cứu lý thuyết**

### **2.2. Thiết kế hệ thống**

- Xác định các yêu cầu của hệ thống
- Thiết kế kiến trúc hệ thống
- Thiết kế cơ sở dữ liệu
- Thiết kế giao diện người dùng

### **2.3. Lập trình và triển khai hệ thống**

- Lập trình các thành phần của hệ thống
- Triển khai hệ thống trên môi trường thực tế

### **2.4. Kiểm thử và bảo trì hệ thống**

- Kiểm thử hệ thống về chức năng, hiệu năng, bảo mật
- Bảo trì hệ thống

### **2.5. Viết báo cáo tổng kết đề tài**

## **III. Phần mềm và công cụ sử dụng**

- Công cụ sử dụng:
  - Visual Studio
  - Visual Studio Code

- MongoDBCompass
- Công nghệ sử dụng:
  - MongoDB
  - ASP.NET Core
  - Angular, Ionic
  - Vite
  - ReactJs (Framework: MUI material)

#### **IV. Dự kiến kết quả đạt được**

- Tìm hiểu IoT.
- Xây dựng hệ thống giám sát và điều khiển thông minh có khả năng theo dõi và tự động điều chỉnh môi trường chuồng trại (nhiệt độ, độ ẩm, ánh sáng, thông gió) nhằm đảm bảo điều kiện sống tối ưu cho vật nuôi.
- Tính năng cảnh báo sớm khi môi trường biến động bất thường, giúp giảm thiểu rủi ro và bảo vệ sức khỏe vật nuôi.
- Hệ thống điều khiển tự động các thiết bị như quạt thông gió, hệ thống sưởi và hệ thống ánh sáng giúp tối ưu hóa nguồn lực và tiết kiệm chi phí vận hành.
- Giao diện quản lý trực quan, dễ sử dụng, hỗ trợ theo dõi dữ liệu thời gian thực và điều khiển từ xa.
- Báo cáo đánh giá chi tiết về hiệu suất hoạt động, hiệu quả kinh tế và đề xuất hướng phát triển hệ thống tích hợp AI và học máy trong tương lai.
- Hoàn thành báo cáo tổng kết.

**V. Tài liệu tham khảo**

- [1] Domingo, MC (2012) Tổng quan về Internet vạn vật dành cho người khuyết tật. Tạp chí ứng dụng mạng và máy tính, 35, 584-596. <https://doi.org/10.1016/j.jnca.2011.10.015> [Thời gian trích dẫn:1]
- [2] Manyika, et al. (2015) Internet vạn vật: Lập bản đồ giá trị vượt ra ngoài sự cường điệu. Viện McKinsey Global, San Francisco. [Thời gian trích dẫn:1]
- [3] Chỉ số kết nối toàn cầu. Huawei Technologies Co., Ltd., 2015. Web. 6 tháng 9 năm 2015. <http://www.huawei.com/minisite/gci/en/index.html> [Thời gian trích dẫn:1] .
- [4] Từ điển Oxford, Định nghĩa “Internet of Things”. [https://www.lexico.com/en/definition/internet\\_of\\_things](https://www.lexico.com/en/definition/internet_of_things) [Thời gian trích dẫn:1]

*Đà Lạt, ngày      tháng      năm 2025*

**Giáo viên hướng dẫn**

(Ký tên)

**SV Thực hiện**

Sinh viên 1

(Ký tên)

Sinh viên 2 Sinh viên 3

(Ký tên)

(Ký tên)

**BCN Khoa**

(Ký tên)

**Tổ trưởng Bộ môn**

(Ký tên)



**MỤC LỤC**

LỜI CẢM ƠN.....	3
DANH MỤC HÌNH ẢNH.....	10
DANH MỤC BẢNG BIỂU.....	12
MỞ ĐẦU .....	13
TỔNG QUAN .....	14
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT.....	15
<b>1.1. Internet of Things (IoT)</b> .....	16
<b>1.2. ReactJS</b> .....	20
<b>1.3. ASP.NET Core</b> .....	22
<b>1.4. Ionic</b> .....	23
<b>1.5. Angular</b> .....	24
<b>1.6. Ứng dụng Học Máy Giám Sát và Dự Báo Thông Số Môi Trường</b> 26	
CHƯƠNG 2: PHÂN TÍCH YÊU CẦU.....	32
<b>2.1. Phân tích chức năng</b> .....	32
<b>2.2. Sơ đồ use case</b> .....	32
<b>2.3. Các trường hợp sử dụng</b> .....	33
<b>2.2. Kiến trúc hệ thống</b> .....	42
CHƯƠNG 3: XÂY DỰNG HỆ THỐNG.....	50
<b>3.1. Mô tả hệ thống</b> .....	50
<b>3.2. Xây dựng hệ thống</b> .....	56
HƯỚNG PHÁT TRIỂN .....	79
DANH MỤC TÀI LIỆU THAM KHẢO .....	80

## DANH MỤC HÌNH ẢNH

Hình 1: Cách thức hoạt động của MQTT - Nguồn:"mesidas.com".....	19
Hình 2: Sơ đồ use case chức năng của hệ thống.....	32
Hình 3: Sơ đồ kiến trúc hệ thống.....	43
Hình 4: Sơ đồ mô tả luồng xử lý của hệ thống.....	51
Hình 5: Code xử lý điều khiển thiết bị.....	51
Hình 6: Code xử lý trạng thái thiết bị.....	53
Hình 7: Code xử lý thu nhập thông tin môi trường.....	54
Hình 8: sơ đồ tổng quan API.....	56
Hình 9: API Environment.....	57
Hình 10: API Environment.....	58
Hình 11: API Area.....	58
Hình 12: API AreaDevice.....	59
Hình 13: API User.....	59
Hình 14: API cấu hình thiết bị.....	60
Hình 15: Trang dashboard của ứng dụng web.....	60
Hình 16: Hiển thị điều khiển và cấu hình tự động.....	61
Hình 17: Trang thống kê chi tiết.....	62
Hình 18: Biểu đồ thống kê nhiệt độ trong 1 ngày.....	62
Hình 19: Biểu đồ thống kê độ ẩm trong 1 ngày.....	63
Hình 20: Biểu đồ thống kê cường độ ánh sáng trong 1 ngày.....	63
Hình 21: Trang quản lý thiết bị.....	64
Hình 22: Trang quản lý khu vực.....	65
Hình 23: Liên hệ và hỗ trợ.....	66
Hình 24: Hỗ trợ người dùng.....	66
Hình 25: Hướng dẫn sử dụng:.....	68
Hình 26: Trang tổng quan và địa điểm phân bố.....	69
Hình 27: Trang quản lý thiết bị.....	71
Hình 28: Trang phân tích và dự báo.....	71
Hình 29: Sơ đồ áp dụng học máy.....	72

Hình 30: Biểu đồ phân tích dữ liệu.....	73
Hình 31: Phân tích rủi ro .....	74
Hình 32: Biện pháp phòng ngừa .....	<b>Error! Bookmark not defined.</b>
Hình 33: Mô hình chuỗi trại .....	76
Hình 34: Tổng quan mô hình.....	77

## DANH MỤC BẢNG BIỂU

Bảng 1: So sánh học máy.....	31
Bảng 2: Use case đăng ký.....	34
Bảng 3: Use case đăng nhập.....	36
Bảng 4: Use case thông báo và cảnh báo.....	37
Bảng 5: Use case quản lý thiết bị và khu vực.....	39
Bảng 6: Use case thống kê thông số môi trường.....	40
Bảng 7: Use case cấu hình thiết bị.....	41
Bảng 8: Use case camera.....	42
Bảng 9: Thuộc tính bảng trang trại.....	44
Bảng 10: Thuộc tính bảng khu vực.....	45
Bảng 11: Thuộc tính bảng thiết bị.....	45
Bảng 12: Thuộc tính bảng khu vực - thiết bị.....	46
Bảng 13: Thuộc tính bảng thông tin môi trường.....	46
Bảng 14: Thuộc tính bảng người dùng.....	47
Bảng 15: Các thiết bị tại trang trại.....	49

## MỞ ĐẦU

Hiện nay, việc quản lý trang trại đóng vai trò quan trọng trong nông nghiệp, đặc biệt là trong bối cảnh yêu cầu về tăng năng suất và giảm thiểu lãng phí tài nguyên ngày càng trở nên cấp thiết. Tuy nhiên, các phương pháp quản lý truyền thống vẫn còn nhiều hạn chế, nhất là trong việc giám sát các yếu tố môi trường vật nuôi. Các hệ thống thủ công đòi hỏi người quản lý phải dành nhiều thời gian và công sức để theo dõi tình trạng trang trại, dễ dẫn đến sai sót và không đạt hiệu quả cao. Điều này càng trở nên phức tạp hơn khi quy mô trang trại lớn và yêu cầu kiểm soát nhiều yếu tố khác nhau cùng lúc.

Chính vì vậy, việc nghiên cứu và xây dựng hệ thống IoT quản lý trang trại thông minh là thật sự cần thiết. Hệ thống này có khả năng tự động hóa việc giám sát các thông số quan trọng như độ ẩm đất, nhiệt độ và không khí môi trường chuồng trại vật nuôi. Bằng cách ứng dụng công nghệ IoT, người quản lý có thể theo dõi tình trạng của trang trại từ xa và tự động đưa ra các quyết định chính xác mà không cần phải có mặt tại hiện trường. Điều này giúp tiết kiệm thời gian, nguồn lực, và tối ưu hóa quy trình sản xuất.

**Từ khóa:** Công Nghệ Thông Tin, IOT, Website, Trang trại.

## TỔNG QUAN

### ***1. Giới thiệu đề tài***

Trong một thế giới nông nghiệp ngày càng phát triển và cạnh tranh gay gắt, việc sử dụng công nghệ để tối ưu hóa quản lý trang trại trở nên cực kỳ quan trọng. Đồ án tốt nghiệp "Tìm hiểu IoT và xây dựng ứng dụng" nhấn mạnh vào việc phát triển một hệ thống quản lý hiện đại, tận dụng các công nghệ mới nhất để giúp chủ trang trại quản lý và theo dõi hoạt động của trang trại một cách hiệu quả.

### ***2. Đối tượng và phạm vi nghiên cứu của đề tài***

Đối tượng của đề tài là các trang trại trong nông nghiệp hiện đại, đặc biệt là các trang trại muốn áp dụng công nghệ để nâng cao hiệu suất và quản lý.

Phạm vi nghiên cứu bao gồm việc phát triển phần mềm quản lý trang trại, tập trung vào các chức năng quản lý, giám sát môi trường, điều khiển thiết bị và thống kê báo cáo.

### ***3. Mục tiêu nghiên cứu của đề tài***

Mục tiêu chung: Phát triển một hệ thống quản lý trang trại toàn diện, tích hợp các chức năng quản lý, giám sát và thống kê báo cáo để hỗ trợ quản lý và theo dõi, tự động đưa ra quyết định hiệu quả.

Mục tiêu cụ thể:

- Quản lý và theo dõi thông tin môi trường như nhiệt độ, độ ẩm, ánh sáng.
- Điều khiển các thiết bị như hệ thống thông gió, máy tưới, đèn chiếu sáng.
- Tạo ra các báo cáo thống kê về môi trường, hiệu suất.

### ***4. Phương pháp nghiên cứu của đề tài***

Đề tài sẽ sử dụng phương pháp nghiên cứu áp dụng, bao gồm:

- Nghiên cứu lý thuyết về các công nghệ liên quan như IoT và các hệ thống quản lý trang trại hiện nay...
- Thiết kế hệ thống dựa trên các yêu cầu đã xác định.

- Lập trình và triển khai hệ thống.
- Kiểm thử và bảo trì hệ thống.

Từ việc xác định mục tiêu đề tài đến việc triển khai và bảo trì hệ thống, mọi bước đều được thực hiện một cách kỹ lưỡng để đảm bảo đạt được kết quả mong đợi.

## CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

Sự phát triển của công nghệ thông tin đã mang lại những đổi thay to lớn cho mọi ngành nghề, đặc biệt là trong lĩnh vực nông nghiệp. Hệ thống quản lý trang trại thông minh (Smart Farm) ngày càng được ứng dụng rộng rãi, giúp nâng cao hiệu quả sản xuất, giảm thiểu chi phí và tối ưu hóa việc sử dụng nguồn lực.

Để xây dựng một hệ thống quản lý trang trại thông minh hiệu quả, đòi hỏi phải ứng dụng nhiều công nghệ tiên tiến. Dự án "Tìm hiểu IoT và xây dựng ứng dụng" tập trung vào việc quản lý thông tin môi trường và điều khiển thiết bị từ xa. Do đó, việc nghiên cứu lý thuyết về các công nghệ liên quan là một bước cần thiết để đảm bảo hệ thống được thiết kế và phát triển một cách tối ưu.

Báo cáo này sẽ đi sâu vào việc nghiên cứu lý thuyết về các công nghệ sau:

Internet of Things (IoT): IoT là nền tảng kết nối các thiết bị và cảm biến, tạo điều kiện cho việc thu thập dữ liệu môi trường trong trang trại.

MQTT (Message Queuing Telemetry Transport): MQTT là giao thức truyền thông nhẹ, hiệu quả, được sử dụng để trao đổi thông tin giữa các thiết bị IoT và máy chủ, đảm bảo truyền tải dữ liệu một cách nhanh chóng và đáng tin cậy.

ReactJS: ReactJS là một thư viện JavaScript phổ biến cho việc xây dựng giao diện người dùng (UI) ứng dụng web, giúp tạo ra các ứng dụng web tương tác, hiệu quả và dễ bảo trì.

ASP.NET Core: ASP.NET Core là một framework mã nguồn mở mạnh mẽ để phát triển ứng dụng web, hỗ trợ đa nền tảng và cung cấp các tính năng bảo mật,

hiệu suất cao và khả năng tích hợp với các dịch vụ đám mây.

**Ionic:** Ionic là một framework cho phát triển ứng dụng di động đa nền tảng dựa trên HTML, CSS và JavaScript, giúp xây dựng các ứng dụng di động một cách nhanh chóng và hiệu quả.

**Angular:** Angular là một framework mã nguồn mở mạnh mẽ để xây dựng các ứng dụng web động, cung cấp các tính năng hỗ trợ phát triển ứng dụng phức tạp, dễ bảo trì và tích hợp tốt với Ionic.

Việc nghiên cứu kỹ lưỡng các công nghệ này sẽ giúp chúng ta:

- Hiểu rõ nguyên lý hoạt động, ưu điểm và nhược điểm của từng công nghệ.
- Chọn lựa công nghệ phù hợp nhất cho dự án.
- Thiết kế và phát triển hệ thống một cách hiệu quả, đảm bảo tính ổn định và đáp ứng được yêu cầu của người dùng.

## **1.1. Internet of Things (IoT)**

### **1.1.1. Khái niệm**

Internet of Things (IoT) là một khái niệm trong lĩnh vực công nghệ thông tin, mô tả một mạng lưới các thiết bị và đối tượng có khả năng giao tiếp với nhau thông qua internet, mà không cần sự can thiệp trực tiếp của con người. Điều này cho phép các thiết bị và đối tượng này thu thập dữ liệu, trao đổi thông tin và thực hiện các tác vụ mà trước đây cần sự tương tác thủ công.

Mục tiêu chính của IoT là tạo ra một mạng lưới kết nối các thiết bị và đối tượng từ các máy móc công nghiệp đến các đồ vật hàng ngày như đèn, tủ lạnh, đồng hồ đeo tay, xe hơi và nhiều hơn nữa. Các thiết bị này được trang bị các cảm biến, vi mạch và phần mềm cho phép chúng thu thập dữ liệu, chia sẻ thông tin và thực hiện các tác vụ tự động một cách thông minh.

Ví dụ cụ thể về IoT bao gồm các ứng dụng như:

- Nhà thông minh (Smart Home): Các thiết bị trong nhà như đèn, máy lạnh, bình nước nóng có thể được kết nối với mạng internet để có thể



điều khiển từ xa qua điện thoại thông minh hoặc thiết bị khác.

- Chăm sóc sức khỏe (Healthcare): Các thiết bị y tế như đồng hồ thông minh, cảm biến sức khỏe có thể theo dõi dấu hiệu sức khỏe của người dùng và gửi thông tin đến bác sĩ hoặc hệ thống y tế.
- Công nghiệp 4.0: Trong môi trường sản xuất, các cảm biến có thể được sử dụng để giám sát quá trình sản xuất, dự báo hỏng hóc và tối ưu hóa hiệu suất sản xuất.
- Giao thông thông minh (Smart Transportation): Các cảm biến trên đường phố và trong xe hơi có thể giúp giảm tắc đường, cải thiện an toàn giao thông và quản lý hệ thống giao thông hiệu quả hơn.
- Nông nghiệp thông minh (Smart Agriculture): Các cảm biến đo lường độ ẩm, nhiệt độ, và các yếu tố khác có thể được sử dụng để tối ưu hóa việc tưới tiêu và quản lý nguồn tài nguyên nông nghiệp.

Đặc điểm nổi bật của IoT:

- Kết nối đa dạng: IoT kết nối một loạt các thiết bị và đối tượng khác nhau, từ các thiết bị điện gia dụng đến các thiết bị công nghiệp và đối tượng trong môi trường tự nhiên.

- Thu thập dữ liệu tự động: Các thiết bị IoT được trang bị cảm biến cho phép chúng tự động thu thập dữ liệu về môi trường xung quanh, điều này có thể giúp cải thiện quản lý và ra quyết định.

- Tính linh hoạt và tự động hóa: IoT cho phép tự động hóa các tác vụ dựa trên dữ liệu thu thập được, giảm thiểu sự can thiệp của con người và tăng cường hiệu suất.

- Giao tiếp và tương tác: Các thiết bị IoT có khả năng giao tiếp và tương tác với nhau, tạo ra một hệ thống mạng lưới thông minh.

- Tích hợp dữ liệu: IoT kết hợp dữ liệu từ nhiều nguồn khác nhau và phân tích chúng để tạo ra thông tin hữu ích cho quản lý và ra quyết định.

- Tiết kiệm chi phí và tăng cường hiệu suất: Bằng cách tự động hóa quy trình và tối ưu hóa sử dụng tài nguyên, IoT có thể giúp giảm chi phí và tăng cường hiệu suất.

### **1.1.2. Ưu và nhược điểm của IoT**

Ưu điểm:

- Tăng cường tiện ích và sự thuận tiện: IoT mang lại nhiều tiện ích cho cuộc sống hàng ngày của con người, từ việc điều khiển nhà thông minh đến việc quản lý sức khỏe cá nhân.

- Cải thiện hiệu suất và năng suất: Tự động hóa các quy trình và tối ưu hóa sử dụng tài nguyên có thể cải thiện hiệu suất và năng suất trong nhiều lĩnh vực, từ công nghiệp đến nông nghiệp.

- Quản lý thông tin và dữ liệu: IoT giúp thu thập và phân tích dữ liệu từ nhiều nguồn khác nhau, giúp quản lý và ra quyết định dựa trên thông tin chính xác.

Nhược điểm:

- Bảo mật và quyền riêng tư: IoT tạo ra nhiều vấn đề về bảo mật thông tin và quyền riêng tư, với nguy cơ lộn xộn thông tin và vi phạm quyền riêng tư.

- Tiêu thụ năng lượng: Một số thiết bị IoT có thể tiêu thụ năng lượng lớn và cần nguồn cung cấp điện liên tục.

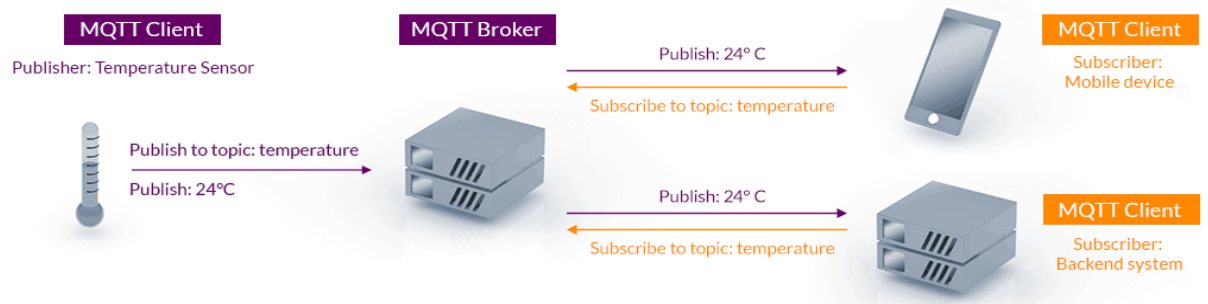
- Phức tạp hóa: Sự phức tạp của các hệ thống IoT có thể làm tăng nguy cơ lỗi và khó khăn trong việc quản lý và bảo trì.

- Tiêu tốn chi phí đầu tư ban đầu: Triển khai các hệ thống IoT có thể đòi hỏi chi phí đầu tư lớn cho việc mua sắm thiết bị và triển khai hạ tầng mạng.

### **1.1.3. Giao thức MQTT**

Giao thức MQTT (Message Queuing Telemetry Transport) là một giao thức truyền thông đặc biệt được thiết kế cho việc giao tiếp giữa các thiết bị trong mạng IoT (Internet of Things). MQTT được phát triển bởi IBM vào những năm đầu của thế kỷ 21 và sau đó trở thành một tiêu chuẩn mở được quản lý bởi tổ chức OASIS.

MQTT sử dụng mô hình Publish/Subscribe, trong đó các thiết bị gửi (publish) các thông điệp tới một "broker" trung tâm, và các thiết bị khác nhận (subscribe) các thông điệp mà họ quan tâm từ broker. Điều này tạo ra một mô hình giao tiếp linh hoạt và hiệu quả, đặc biệt là trong các mạng IoT mà có hàng nghìn hoặc thậm chí hàng triệu thiết bị.



Hình 1: Cách thức hoạt động của MQTT - Nguồn: "mesidas.com".

HiveMQ là một phần mềm broker MQTT mạnh mẽ và linh hoạt, được thiết kế đặc biệt cho các ứng dụng IoT và các môi trường mạng phân tán. HiveMQ cung cấp một loạt các tính năng tiên tiến bao gồm:

- Khả năng mở rộng: HiveMQ có thể mở rộng một cách linh hoạt để xử lý hàng triệu kết nối và hàng tỉ thông điệp mỗi ngày.
- Bảo mật: HiveMQ hỗ trợ các tính năng bảo mật như TLS/SSL để đảm bảo an toàn cho việc truyền dữ liệu giữa các thiết bị và broker.
- Quản lý phiên: HiveMQ cung cấp khả năng quản lý các phiên kết nối MQTT, giúp tối ưu hóa tài nguyên và xử lý tải.
- Tích hợp dễ dàng: HiveMQ có thể dễ dàng tích hợp với các hệ thống và ứng dụng tồn tại thông qua các giao diện API và các cơ chế mở rộng.

Nhờ vào sự kết hợp giữa giao thức MQTT và các tính năng mạnh mẽ của HiveMQ, việc triển khai các ứng dụng IoT trở nên dễ dàng hơn, đồng thời cung cấp hiệu suất và độ tin cậy cao trong việc giao tiếp và quản lý các thiết bị trong mạng IoT.

## 1.2. ReactJS

### 1.2.1. Khái niệm

ReactJS là một thư viện JavaScript được sử dụng phổ biến để xây dựng giao diện người dùng (UI) cho các ứng dụng web. Nó được phát triển bởi Facebook và được ra mắt lần đầu tiên vào năm 2013. ReactJS được thiết kế để tạo ra các giao diện người dùng tương tác một cách dễ dàng và hiệu quả, đặc biệt là trong các ứng dụng đơn trang (Single Page Applications - SPA).

Các điểm nổi bật của ReactJS bao gồm:

- Component-Based: ReactJS sử dụng một kiến trúc dựa trên thành phần (component-based architecture), cho phép phát triển ứng dụng theo cách phân chia thành các thành phần riêng biệt, mỗi thành phần chứa mã HTML, CSS và JavaScript riêng của nó. Điều này giúp tăng tính tái sử dụng và quản lý mã nguồn dễ dàng hơn.

- Virtual DOM: ReactJS sử dụng một cơ chế gọi là Virtual DOM để tối ưu hóa việc cập nhật giao diện người dùng. Thay vì cập nhật trực tiếp DOM (Document Object Model) khi dữ liệu thay đổi, ReactJS tạo ra một bản sao của DOM hiện tại trong bộ nhớ (Virtual DOM), so sánh nó với DOM trước đó và chỉ cập nhật những phần thay đổi. Điều này giúp cải thiện hiệu suất của ứng dụng.

- JSX: ReactJS sử dụng JSX (JavaScript XML) là một phần mở rộng của JavaScript để viết mã HTML trong mã JavaScript. JSX giúp tạo ra mã nguồn dễ đọc và dễ hiểu, đồng thời tăng cường tính tương tác giữa JavaScript và HTML.

- One-Way Data Binding: ReactJS thực hiện data binding một chiều (one-way data binding), tức là dữ liệu chỉ được truyền từ thành phần cha đến thành phần con. Điều này giúp giảm thiểu sự phức tạp và dễ dàng debug trong ứng dụng.

- Hỗ trợ cộng đồng lớn: ReactJS có một cộng đồng lớn và tích cực, điều này có nghĩa là có nhiều tài liệu, ví dụ và các thư viện bổ sung (như Redux, React Router) để hỗ trợ việc phát triển ứng dụng.

### 1.2.2. Ưu và nhược điểm của ReactJS

Ưu điểm:

- Hiệu suất cao: React sử dụng Virtual DOM để tối ưu hóa việc cập nhật giao diện, giúp cải thiện hiệu suất ứng dụng.
- Cú pháp dễ hiểu: Cú pháp của React dễ hiểu và học, cho phép nhà phát triển dễ dàng xây dựng các thành phần tái sử dụng.
- Quản lý trạng thái tốt: React hỗ trợ quản lý trạng thái thông qua các khái niệm như state và props, giúp dễ dàng theo dõi và cập nhật trạng thái của các thành phần.
- Ecosystem mạnh mẽ: React có một cộng đồng rộng lớn, cung cấp nhiều thư viện, công cụ và hỗ trợ, giúp phát triển ứng dụng dễ dàng hơn.

Nhược điểm:

- Khả năng học tập ban đầu: React yêu cầu kiến thức về JavaScript và các khái niệm như JSX và Virtual DOM, điều này có thể làm tăng độ dốc học tập ban đầu.
- Quá phụ thuộc vào cộng đồng: Một số tính năng phức tạp hoặc tiện ích có thể đòi hỏi sự phụ thuộc vào bên thứ ba, và sự phát triển của một thư viện hoặc công cụ phụ thuộc vào sự ủng hộ của cộng đồng.

So sánh với các thư viện khác hỗ trợ làm giao diện:

- Angular: Angular là một framework JavaScript hoàn chỉnh, trong khi React chỉ là một thư viện. Angular cung cấp nhiều tính năng tích hợp sẵn và hỗ trợ mạnh mẽ cho phát triển ứng dụng phức tạp. Tuy nhiên, React thường được coi là có hiệu suất tốt hơn và có thời gian học và triển khai nhanh hơn so với Angular.
- Vue.js: Vue.js là một framework JavaScript dựa trên thành phần (component-based). Nó tương tự như React và cũng tập trung vào việc xây dựng giao diện người dùng. Vue.js có cú pháp đơn giản hơn và thường được coi là dễ học hơn React. Tuy nhiên, React có một cộng đồng lớn hơn và được sử dụng rộng rãi hơn Vue.js

### **1.3. ASP.NET Core**

#### **1.3.1. Khái niệm**

ASP.NET Core là một nền tảng phát triển ứng dụng web mã nguồn mở và đa nền tảng được phát triển bởi Microsoft. Nó là phiên bản tiếp theo của ASP.NET Framework, nhằm cung cấp một framework hiện đại, linh hoạt và hiệu quả hơn cho việc xây dựng ứng dụng web.

Dưới đây là một số điểm nổi bật của ASP.NET Core:

- Đa nền tảng (Cross-Platform): ASP.NET Core hỗ trợ chạy trên nhiều nền tảng hệ điều hành như Windows, Linux và macOS. Điều này cho phép nhà phát triển lựa chọn nền tảng phát triển phù hợp với yêu cầu cụ thể của dự án.

- Hiệu suất cao: ASP.NET Core được thiết kế để có hiệu suất cao, với khả năng xử lý yêu cầu và đáp ứng tốt ngay cả trong những tình huống tải cao. Nó sử dụng công nghệ mới như Kestrel, một máy chủ web được tích hợp sẵn, và ASP.NET Core Middleware để tối ưu hóa hiệu suất.

- Modular và Linh hoạt: ASP.NET Core sử dụng một cấu trúc modul và linh hoạt, cho phép nhà phát triển chọn lựa các thành phần cần thiết cho ứng dụng của họ. Điều này giúp giảm dung lượng ứng dụng và tối ưu hóa hiệu suất.

- Hỗ trợ Cloud: ASP.NET Core tích hợp tốt với các dịch vụ đám mây như Azure của Microsoft, cho phép triển khai và quản lý ứng dụng web một cách dễ dàng trên nền tảng điện toán đám mây.

- Tích hợp Docker: ASP.NET Core hỗ trợ tích hợp với Docker, cho phép triển khai ứng dụng web dưới dạng các container, giúp tối ưu hóa quy trình phát triển và triển khai.

- Hệ thống Middleware: ASP.NET Core Middleware cho phép xử lý các yêu cầu HTTP trước khi chúng đến tới ứng dụng, giúp tối ưu hóa quy trình xử lý và cung cấp các tính năng như xác thực, ghi nhật ký, nén và đường dẫn.

### 1.3.2. Nhược điểm của ASP.NET Core

Khả năng tương thích ngược với phiên bản cũ: Do ASP.NET Core là một phiên bản mới của ASP.NET, nên có một số sự thay đổi lớn trong cấu trúc và API so với ASP.NET Framework, điều này có thể gây khó khăn cho việc nâng cấp từ các ứng dụng ASP.NET Framework cũ lên ASP.NET Core.

Có thể cần thời gian để học và triển khai: Với những người mới bắt đầu, việc học và sử dụng ASP.NET Core có thể đòi hỏi một thời gian dài để làm quen với các khái niệm mới và cách thức phát triển ứng dụng.

Thư viện bên thứ ba có thể hạn chế: Mặc dù có một cộng đồng phong phú xung quanh ASP.NET Core, nhưng một số thư viện bên thứ ba vẫn chưa được hỗ trợ hoặc có thể hạn chế so với ASP.NET Framework truyền thống.

## 1.4. Ionic

### 1.4.1. Khái niệm

Ionic là một framework phát triển ứng dụng di động được xây dựng trên nền tảng web, cho phép các nhà phát triển sử dụng HTML, CSS và JavaScript để xây dựng ứng dụng di động đa nền tảng. Ionic được phát triển bởi công ty Ionic và ra mắt lần đầu vào năm 2013.

Đặc điểm chính của Ionic là sự kết hợp giữa Angular, một framework JavaScript phổ biến cho phát triển ứng dụng web, và Cordova hoặc Capacitor để đóng gói ứng dụng và truy cập các tính năng của hệ điều hành di động như máy ảnh, định vị GPS và cảm biến.

### 1.4.2. Ưu nhược điểm của Ionic

Ưu điểm:

- Đa nền tảng: Ionic cho phép phát triển ứng dụng di động cho cả iOS, Android và các nền tảng khác từ một mã nguồn duy nhất, giúp tiết kiệm thời gian và công sức cho việc phát triển ứng dụng.

- Sử dụng Web Technologies: Ionic sử dụng HTML, CSS và JavaScript,

những ngôn ngữ phổ biến trong việc phát triển web, làm cho quá trình học và sử dụng dễ dàng đối với các nhà phát triển web.

- UI Components: Ionic cung cấp một bộ sưu tập các thành phần giao diện người dùng (UI components) sẵn có, giúp việc xây dựng giao diện ứng dụng nhanh chóng và chuyên nghiệp.

- Tích hợp với Angular: Ionic được xây dựng trên nền tảng Angular, một framework JavaScript mạnh mẽ, giúp việc phát triển ứng dụng trở nên dễ dàng và có hiệu suất cao.

- Tích hợp với Cordova hoặc Capacitor: Ionic tích hợp với Cordova hoặc Capacitor để đóng gói ứng dụng và truy cập các tính năng của hệ điều hành di động, như máy ảnh, định vị GPS và cảm biến.

Nhược điểm:

- Hiệu suất: Do Ionic sử dụng WebView để hiển thị ứng dụng trên các nền tảng di động, hiệu suất của ứng dụng có thể không được tối ưu như việc sử dụng ngôn ngữ lập trình native.

- Hạn chế trong truy cập tính năng của thiết bị: Mặc dù Ionic tích hợp với Cordova hoặc Capacitor để truy cập các tính năng của thiết bị, nhưng không thể tránh khỏi một số hạn chế về hiệu suất và tính năng so với việc sử dụng ngôn ngữ lập trình native.

- Khả năng tùy chỉnh hạn chế: Mặc dù Ionic cung cấp một số lượng lớn các thành phần giao diện người dùng, nhưng khả năng tùy chỉnh có thể bị hạn chế so với việc phát triển ứng dụng hoàn toàn từ đầu.

## **1.5. Angular**

### **1.4.1. Khái niệm**

Angular là một framework mã nguồn mở phát triển bởi Google, được sử dụng để xây dựng các ứng dụng web động cao cấp và đa tính năng. Nó kết hợp HTML, CSS, và JavaScript/TypeScript để tạo ra các ứng dụng web động và dễ bảo trì.



Khái niệm cốt lõi của Angular là "Single Page Application" (SPA), nơi mà mọi thứ chạy trên một trang web duy nhất, và các phần của trang web được tải lên và hiển thị mà không cần tải lại toàn bộ trang.

Angular là một framework phát triển ứng dụng web mạnh mẽ, và khi kết hợp với Ionic, nó tạo ra một môi trường lập trình tuyệt vời cho việc phát triển ứng dụng di động.

Cross-platform development: Ionic cho phép bạn phát triển ứng dụng di động cho cả iOS và Android bằng cách sử dụng các công nghệ web chuẩn như HTML, CSS và JavaScript. Angular, với cấu trúc mạnh mẽ và linh hoạt, là một lựa chọn lý tưởng để phát triển các ứng dụng di động cross-platform đồng nhất và hiệu quả.

UI components: Ionic cung cấp một thư viện các thành phần giao diện người dùng (UI components) sẵn có, được tối ưu hóa cho trải nghiệm di động. Angular hỗ trợ việc tạo và quản lý các thành phần này một cách dễ dàng thông qua cơ chế component của mình. Việc kết hợp Angular với Ionic giúp tạo ra giao diện người dùng di động đẹp mắt và dễ bảo trì.

Nguyên lý thiết kế của Angular: Angular thúc đẩy việc phát triển ứng dụng theo các nguyên lý thiết kế như Dependency Injection, Single Responsibility, và Separation of Concerns. Khi kết hợp với Ionic, các ứng dụng di động có thể được xây dựng với cấu trúc dễ bảo trì và mở rộng.

Plugin và tính năng mở rộng: Ionic cung cấp một hệ sinh thái plugins và tính năng mở rộng phong phú, cho phép bạn tích hợp các tính năng đặc biệt như camera, GPS, và push notifications vào ứng dụng di động của bạn. Angular đi kèm với một hệ sinh thái plugins mạnh mẽ như Angular CLI và các thư viện bổ sung, giúp việc tích hợp các tính năng này trở nên dễ dàng hơn.

#### **1.4.2. Ưu và nhược điểm của Angular**

Ưu điểm:

- Phát triển cross-platform: Có thể phát triển ứng dụng cho cả iOS và Android

từ một mã nguồn.

- Sử dụng các công nghệ web chuẩn: Sử dụng HTML, CSS và JavaScript, giúp giảm thời gian phát triển và tăng khả năng tái sử dụng mã nguồn.

- Cộng đồng lớn: Có một cộng đồng lớn và tích cực hỗ trợ, giúp giải quyết vấn đề và cung cấp tài liệu và ví dụ phong phú.

Nhược điểm:

- Hiệu suất: Có thể gặp vấn đề về hiệu suất đối với các ứng dụng có yêu cầu cao về tốc độ và sự phản hồi.

- Tài nguyên: Yêu cầu tài nguyên hệ thống cao hơn so với một ứng dụng native vì sử dụng WebView để hiển thị giao diện người dùng.

- Giới hạn của các plugins: Một số tính năng có thể không được hỗ trợ tốt hoặc không có plugin sẵn có, điều này có thể gây khó khăn trong việc phát triển ứng dụng có tính năng phức tạp.

## 1.6. Ứng dụng Học Máy Giám Sát và Dự Báo Thông Số Môi Trường

Ứng dụng giám sát và dự báo các thông số môi trường như nhiệt độ, độ ẩm và chất lượng không khí, sử dụng Angular và Chart.js. Để xử lý dữ liệu cảm biến và cung cấp các dự báo, cảnh báo rủi ro, ứng dụng tích hợp các kỹ thuật học máy cơ bản và thống kê, bao gồm dự báo chuỗi thời gian bằng làm mịn hàm mũ, phát hiện bất thường dựa trên độ lệch chuẩn, phân tích tương quan Pearson và luật quyết định dựa trên ngưỡng. Những phương pháp này giúp phân tích dữ liệu thời gian thực, dự đoán xu hướng và đưa ra các cảnh báo kịp thời, hỗ trợ người dùng trong việc giám sát và quản lý môi trường hiệu quả.

### 1.6.1. Dự báo chuỗi thời gian (Time Series Forecasting)

**Kỹ thuật sử dụng:** Làm mịn hàm mũ (Exponential Smoothing)

**Mô tả:** Trong hàm `exponentialSmoothingForecast`, đoạn code sử dụng phương pháp làm mịn hàm mũ để dự báo các thông số môi trường (nhiệt độ, độ ẩm, chất lượng không khí) trong 12 giờ tới.

**Lý thuyết:** Làm mịn hàm mũ là một kỹ thuật dự báo chuỗi thời gian đơn giản, trong đó giá trị dự báo được tính dựa trên trung bình có trọng số của các giá trị trước đó. Trọng số giảm dần theo thời gian, với các giá trị gần đây có ảnh hưởng lớn hơn.

**Công thức:**

$$St = a \cdot Xt + (1 - a) \cdot St-1$$

*Trong đó:*

- *St:* Giá trị làm mịn tại thời điểm *t*.
- *Xt:* Giá trị thực tế tại thời điểm *t*.
- *a:* Hệ số làm mịn (trong code,  $a = 0.3$ ).

Code cũng tính thêm xu hướng dựa trên sự thay đổi của các giá trị làm mịn gần đây để điều chỉnh dự báo:

$$\text{Forecast} = \text{Slast} + \text{RecentTrend} \cdot \text{hoursAhead}$$

**Ứng dụng:**

- Dự báo nhiệt độ, độ ẩm và chất lượng không khí trong các khoảng thời gian tương lai (2, 4, ..., 12 giờ).
- Kết quả dự báo được sử dụng để vẽ biểu đồ (hàm `updateForecastView`) và phân tích xu hướng (hàm `analyzeForecastTrend`).

### 1.6.2. Phát hiện bất thường (Anomaly Detection)

**Kỹ thuật sử dụng:** Phát hiện bất thường dựa trên độ lệch chuẩn (Standard Deviation-based Anomaly Detection)

**Mô tả:** Trong hàm `detectAnomalies`, code sử dụng phương pháp thống kê để phát hiện các giá trị bất thường của nhiệt độ, độ ẩm và chất lượng không khí.

**Lý thuyết:**

- Phương pháp này giả định rằng dữ liệu tuân theo phân phối chuẩn (Gaussian). Các giá trị nằm ngoài một ngưỡng (threshold) được tính dựa trên độ lệch chuẩn được coi là bất thường.

**Công thức:**

- Tính trung bình ( $\mu$ ) và độ lệch chuẩn ( $\sigma$ ) của dữ liệu.
- Một giá trị XXX được coi là bất thường nếu:

$$|X - \mu| > k \cdot \sigma$$

Trong đó k là ngưỡng(trong code,  $k = 2.5$ ).

- Các bất thường được trả về bao gồm giá trị, thời điểm và mức độ lệch chuẩn.

**Ứng dụng:**

- Phát hiện các biến động bất thường trong dữ liệu môi trường gần đây (ví dụ: nhiệt độ tăng đột ngột).
- Kết quả được sử dụng để tạo cảnh báo rủi ro trong hàm `analyzeRisks`.

**1.6.3. Phân tích tương quan (Correlation Analysis)**

**Kỹ thuật sử dụng:** Hệ số tương quan Pearson

**Mô tả:** Trong hàm `analyzeEnvironmentalCorrelation`, code tính toán hệ số tương quan giữa nhiệt độ và độ ẩm để xác định mối quan hệ giữa chúng.

**Lý thuyết:**

Hệ số tương quan Pearson đo lường mức độ phụ thuộc tuyến tính giữa hai biến XXX (nhiệt độ) và YYY (độ ẩm). Giá trị nằm trong khoảng  $[-1, 1]$ :

- Gần 1: Tương quan thuận mạnh.
- Gần -1: Tương quan nghịch mạnh.
- Gần 0: Không có tương quan.

**Công thức:**

$$r = \frac{n \sum XY - \sum X \sum Y}{\sqrt{(n \sum X^2 - (\sum X)^2)(n \sum Y^2 - (\sum Y)^2)}}$$

Trong đó:

- $n$ : Số lượng quan sát.
- $\sum XY$ : Tổng tích của  $X$  và  $Y$ .
- $\sum X, \sum Y$ : Tổng các giá trị của  $X$  và  $Y$ .
- $\sum X^2, \sum Y^2$ : Tổng bình phương các giá trị của  $X$  và  $Y$ .
- Trong code, hệ số tương quan được phân loại:
  - $r > 0.5$ : Tương quan thuận.
  - $r < -0.5$ : Tương quan nghịch.
  - $|r| > 0.7$ : Tương quan mạnh.

**Ứng dụng:**

- Kết quả tương quan được sử dụng để bổ sung thông tin vào phân tích xu hướng dự báo (hàm `analyzeForecastTrend`), giúp giải thích mối quan hệ giữa nhiệt độ và độ ẩm.

**1.6.4. Phân tích rủi ro dựa trên ngưỡng (Threshold-based Risk Analysis)**

**Kỹ thuật sử dụng:** Luật quyết định dựa trên ngưỡng (Threshold-based Decision Rules)

**Mô tả:** Trong hàm `analyzeRisks` và `generateAlerts`, code áp dụng các ngưỡng an toàn (safe thresholds) để xác định rủi ro và tạo cảnh báo.

**Lý thuyết:**

Đây là một kỹ thuật đơn giản trong học máy và xử lý tín hiệu, sử dụng các quy tắc if-then để phân loại dữ liệu dựa trên giá trị ngưỡng cố định.

Ví dụ:

- Nếu nhiệt độ  $> 28^{\circ}\text{C}$  hoặc  $< 18^{\circ}\text{C}$ , tạo cảnh báo.
- Nếu độ ẩm  $> 70\%$  hoặc  $< 30\%$ , tạo cảnh báo.
- Nếu chất lượng không khí  $> 50$ , tạo cảnh báo.

Các ngưỡng được định nghĩa trong biến `safeThresholds` và được sử dụng để

so sánh với dữ liệu hiện tại và dự báo.

### **Ứng dụng:**

- Tạo các cảnh báo (alerts) khi các thông số môi trường vượt quá ngưỡng an toàn.
- Đánh giá rủi ro dựa trên dữ liệu dự báo, ví dụ: rủi ro nhiệt độ quá cao hoặc chất lượng không khí kém.

### **5. Xử lý và trực quan hóa dữ liệu**

Mặc dù không phải là kỹ thuật học máy trực tiếp, các hàm như `UpdateForecastView` sử dụng dữ liệu dự báo và bất thường để trực quan hóa thông qua biểu đồ (Chart.js). Điều này hỗ trợ người dùng trong việc hiểu và ra quyết định dựa trên kết quả phân tích học máy.

### **2. So sánh với Học máy Hồi quy tuyến tính (linear regression)**

<b>Tiêu chí</b>	<b>Hồi quy tuyến tính (Linear Regression)</b>	<b>Làm mịn hàm mũ (Exponential Smoothing)</b>
<b>Cơ chế hoạt động</b>	Tìm đường thẳng phù hợp nhất ( $y = ax + b$ ) dựa trên dữ liệu quá khứ	Gán trọng số giảm dần theo hàm mũ cho các quan sát trong quá khứ
<b>Độ phức tạp</b>	Đơn giản	Trung bình
<b>Tham số mô hình</b>	Hệ số góc (slope) và hệ số tự do (intercept)	Tham số làm mịn (alpha) và phân tích xu hướng
<b>Xử lý xu hướng</b>	Phát hiện xu hướng tuyến tính	Phát hiện xu hướng gần đây thông qua phân tích độ chênh lệch giữa các giá trị đã làm mịn

<b>Khả năng thích ứng</b>	Thấp, mọi điểm dữ liệu có cùng trọng số	Cao, ưu tiên dữ liệu gần đây hơn (thông qua tham số alpha)
<b>Xử lý nhiễu</b>	Kém, dễ bị ảnh hưởng bởi các điểm dữ liệu bất thường	Tốt hơn, làm mịn các điểm dữ liệu bất thường
<b>Tính năng bổ sung</b>	Không có	Phát hiện điểm bất thường (anomaly detection) và phân tích tương quan
<b>Phù hợp với dữ liệu</b>	Phù hợp với xu hướng dài hạn, ổn định	Phù hợp với dữ liệu thời gian thực và biến động ngắn hạn
<b>Khả năng mở rộng</b>	Hạn chế	Cao, dễ tích hợp với các thuật toán phát hiện bất thường
<b>Yêu cầu tính toán</b>	Thấp	Trung bình

Bảng 1: So sánh học máy

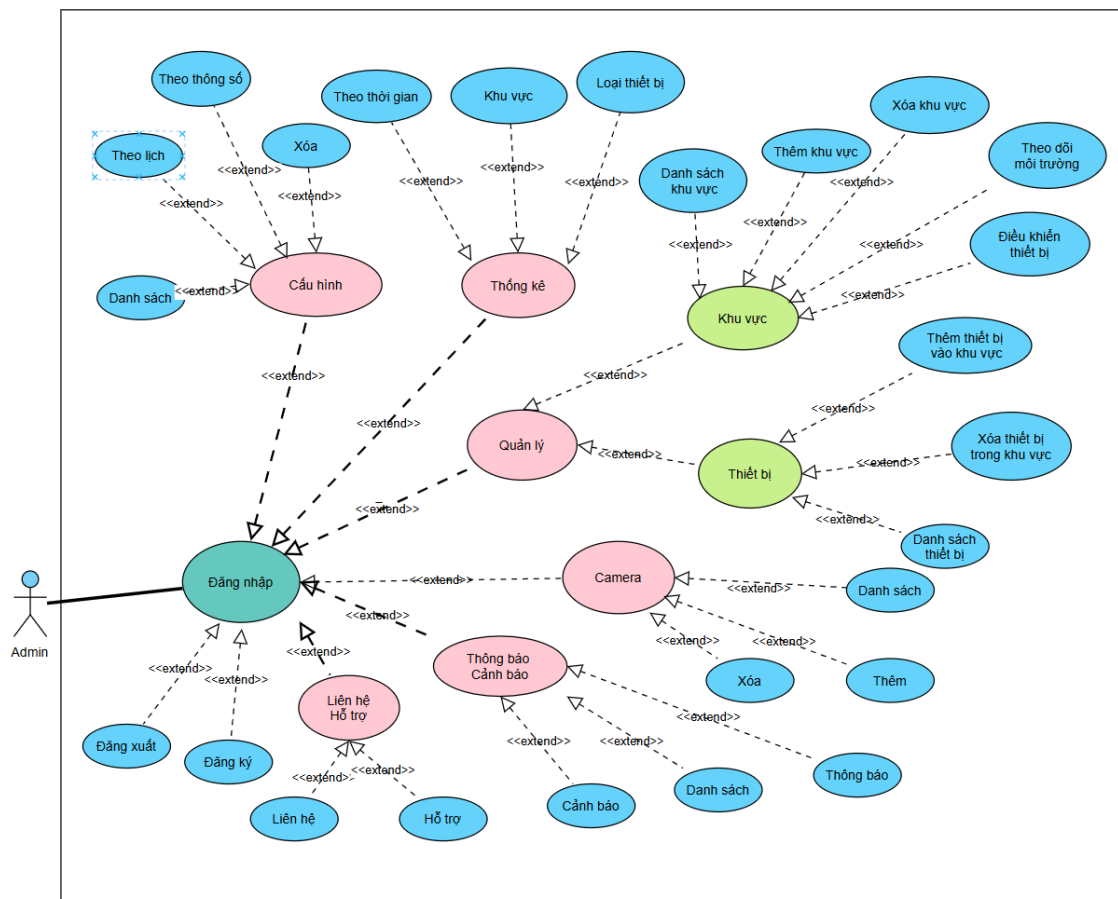
## CHƯƠNG 2: PHÂN TÍCH YÊU CẦU

## 2.1. Phân tích chức năng

Các yêu cầu được đặt ra:

- Quản lý, theo dõi thông tin môi trường: nhiệt độ, độ ẩm, ánh sáng, chất lượng không khí...
- Quản lý, điều khiển thiết bị: hệ thống thông gió, hệ thống sưởi và hệ thống chiếu sáng, ...
- Phân tích và dự đoán: Dựa trên các yếu tố môi trường từ đó đưa ra các phân tích và dự đoán ...

## 2.2. Sơ đồ use case



Hình 2: Sơ đồ use case chức năng của hệ thống.



## 2.3. Các trường hợp sử dụng

### 2.3.1. Đăng ký

Use case UC1: Đăng ký	
Đối tượng	Người dùng chưa có tài khoản
Mục tiêu	Giúp người dùng tạo tài khoản
Điều kiện tiên quyết	Không
Kết quả khi thành công	Người dùng có tài khoản
Luồng sự kiện chính	<ol style="list-style-type: none"> <li>Hệ thống hiện màn hình đăng ký</li> <li>Người dùng nhập vào các trường (họ, tên, email, mật khẩu, địa chỉ, số điện thoại)</li> <li>Người dùng nhấn nút “Create Account” hệ thống kiểm tra dữ liệu: <ol style="list-style-type: none"> <li>Thành công: Gửi dữ liệu về server</li> <li>Thất bại: yêu cầu người dùng nhập lại theo đúng định dạng.</li> </ol> </li> <li>Hệ thống kiểm tra thông tin đăng ký <ol style="list-style-type: none"> <li>Thông tin không tồn tại: tại tài khoản người dùng.</li> <li>Thông tin đã tồn tại: Gửi lỗi người dùng đã tồn tại.</li> </ol> </li> <li>Nếu thành công: chuyển hướng đến trang đăng nhập.</li> <li>Nếu thất bại: hiện thông báo thất bại.</li> </ol>

Mở rộng	<p>TH1: Thông tin không hợp lệ: Khi người dùng nhập sai thông tin yêu cầu.</p> <ol style="list-style-type: none"> <li>1. Hệ thống hiện thông báo yêu cầu người dùng nhập đúng định dạng</li> <li>2. Quay lại bước 2 luồng sự kiện chính</li> </ol> <p>TH2: Thông tin hợp lệ: Khi người dùng nhập đúng thông tin yêu cầu:</p> <ol style="list-style-type: none"> <li>1. Hệ thống kiểm tra dữ liệu đã tồn tại hay chưa.</li> <li>2. Nếu đã tồn tại gửi thông báo lỗi và yêu cầu nhập lại. Quay lại bước 2 luồng sự kiện chính</li> <li>3. Nếu chưa tồn tại lưu thông tin và chuyển hướng đến trang đăng nhập</li> </ol>
Yêu cầu hệ thống	<ul style="list-style-type: none"> <li>- Phải có biểu mẫu đăng ký</li> <li>- Phải có chức năng băm mật khẩu</li> <li>- Phải có chức năng kiểm tra thông tin người dùng trong hệ thống</li> <li>- Phải có chức năng thông báo lỗi</li> </ul>

Bảng 2: Use case đăng ký

### 2.3.2. Đăng nhập

Use case UC2: Đăng nhập	
Đối tượng	Người dùng đã có tài khoản
Mục tiêu	Giúp người dùng truy cập vào hệ thống

Điều kiện tiên quyết	Đã có tài khoản
Kết quả khi thành công	Người dùng có tài khoản
Luồng sự kiện chính	<ol style="list-style-type: none"> <li>Hệ thống hiện màn hình đăng nhập</li> <li>Người dùng nhập vào các trường (tài khoản, mật khẩu)</li> <li>Người dùng nhấn nút “Login” hệ thống kiểm tra dữ liệu: <ol style="list-style-type: none"> <li>Thành công: Gửi dữ liệu về server</li> <li>Thất bại: yêu cầu người dùng nhập lại theo đúng định dạng.</li> </ol> </li> <li>Hệ thống kiểm tra thông tin đăng nhập <ol style="list-style-type: none"> <li>Thông tin không tồn tại: gửi lỗi người dùng không tồn tại.</li> <li>Thông tin đã tồn tại: Tạo access token và refresh token.</li> </ol> </li> <li>Nếu thành công: chuyển hướng đến trang dashboard.</li> <li>Nếu thất bại: hiện thông báo thất bại.</li> </ol>
Mở rộng	<p>TH1: Thông tin không hợp lệ: Khi người dùng nhập sai thông tin yêu cầu.</p> <ol style="list-style-type: none"> <li>Hệ thống hiện thông báo yêu cầu người dùng nhập đúng định dạng</li> <li>Quay lại bước 2 luồng sự kiện chính</li> </ol> <p>TH2: Thông tin hợp lệ: Khi người dùng nhập đúng thông tin yêu cầu:</p>

	<ol style="list-style-type: none"> <li>1. Hệ thống kiểm tra dữ liệu có tồn tại hay không.</li> <li>2. Nếu đã tồn tại: tạo token</li> <li>4. Nếu chưa tồn tại: gửi thông báo lỗi.</li> </ol>
Yêu cầu hệ thống	<ul style="list-style-type: none"> <li>- Phải có biểu mẫu đăng nhập</li> <li>- Phải có chức năng giải băm mật khẩu</li> <li>- Phải có chức năng kiểm tra thông tin người dùng trong hệ thống</li> <li>- Phải có chức năng thông báo lỗi</li> <li>- Phải có chức năng tạo token</li> <li>- Phải có chức năng lưu token</li> </ul>

Bảng 3: Use case đăng nhập

### 2.3.3. Thông báo và cảnh báo

Use case UC3: Thông báo và cảnh báo	
Đối tượng	Quản trị viên
Mục tiêu	Nhận thông báo và cảnh báo
Điều kiện tiên quyết	Đã đăng nhập thành công
Kết quả khi thành công	Danh sách thông báo và cảnh báo
Luồng sự kiện chính	<ol style="list-style-type: none"> <li>1. Quản trị viên đăng nhập vào hệ thống</li> <li>2. Quản trị viên nhấn nút chuông trên giao diện dashboard</li> <li>3. Danh sách thông báo và cảnh báo</li> </ol>

Mở rộng	<p>TH1: Không có thông báo hoặc cảnh báo:</p> <p>Hệ thống hiện thị thông báo “hiện tại không có thông báo hoặc cảnh báo”</p> <p>TH2: Có thông báo hoặc cảnh báo:</p> <p>Hệ thống hiện thị danh sách thông báo cảnh báo</p>
Yêu cầu hệ thống	<ul style="list-style-type: none"> <li>- Phải có chức năng thông báo cảnh báo</li> <li>- Phải có giao diện thông báo cảnh báo</li> </ul>

Bảng 4: Use case thông báo và cảnh báo

### 2.3.4. Quản lý

Use case UC4: Quản lý	
Đối tượng	Quản trị viên
Mục tiêu	Quản lý khu vực và thiết bị trong hệ thống
Điều kiện tiên quyết	Đã đăng nhập thành công
Kết quả khi thành công	Thông tin khu vực và thiết bị hiện thị và lưu chính xác trong hệ thống
Luồng sự kiện chính	<ol style="list-style-type: none"> <li>1. Quản trị viên đăng nhập vào hệ thống</li> <li>2. Quản trị viên thêm, xóa thiết bị khỏi khu vực</li> <li>3. Quản trị viên thêm, xóa, bật tắt thiết bị trong khu vực</li> <li>4. Hệ thống hiện thông tin đã lưu</li> </ol>

Mở rộng	<p>2a. Thêm thiết bị mới</p> <ul style="list-style-type: none"><li>- Quản trị viên chọn thiết bị cần thêm vào khu vực</li><li>- Nếu khu vực hoặc thiết bị không tồn tại<ul style="list-style-type: none"><li>+ Gửi thông báo lỗi</li><li>+ Cho phép nhập lại</li></ul></li></ul> <p>2b. Xóa thiết bị khỏi khu vực</p> <ul style="list-style-type: none"><li>- Quản trị viên chọn các thiết bị muốn xóa khỏi khu vực</li><li>- Nếu tồn tại<ul style="list-style-type: none"><li>+ Thực hiện xóa khỏi khu vực</li><li>+ Gửi thông báo thành công</li></ul></li><li>- Nếu không tồn tại<ul style="list-style-type: none"><li>+ Gửi thông báo lỗi</li></ul></li></ul> <p>3a. Thêm khu vực mới</p> <ul style="list-style-type: none"><li>- Quản trị viên nhập thông tin khu vực</li><li>- Nếu khu vực đã tồn tại<ul style="list-style-type: none"><li>+ Gửi thông báo lỗi</li><li>+ Yêu cầu nhập lại</li></ul></li></ul> <p>3b. Xóa khu vực</p> <ul style="list-style-type: none"><li>- Quản trị viên chọn khu vực muốn xóa</li><li>- Xóa khu vực và thiết bị bên trong đó</li><li>- Gửi thông báo thành công hoặc thất bại</li></ul> <p>3c. Bật/Tắt thiết bị trong khu vực</p> <ul style="list-style-type: none"><li>- Quản trị viên chọn thiết bị muốn bật</li><li>- Nhấn nút “Xác nhận” để gửi yêu cầu bật thiết bị</li><li>- Nếu thành công</li></ul>
---------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	<ul style="list-style-type: none"> <li>+ Gửi thông báo thành công</li> <li>+ Lưu trạng thái thiết bị</li> <li>- Nếu thất bại</li> <li>+ Gửi thông báo lỗi</li> </ul>
Yêu cầu hệ thống	<ul style="list-style-type: none"> <li>- Phải có giao diện nhập liệu</li> <li>- Phải có chức năng thêm, xóa thiết bị và khu vực</li> <li>- Phải có chức năng kiểm tra thiết bị và khu vực</li> <li>- Phải có chức năng gửi yêu cầu đến thiết bị IoT</li> </ul>

Bảng 5: Use case quản lý thiết bị và khu vực

### 2.3.5. Thống kê

Use case UC5: Thống kê	
Đối tượng	Quản trị viên
Mục tiêu	Xem thống kê hệ thống theo dữ liệu yêu cầu
Điều kiện tiên quyết	Đã đăng nhập thành công
Kết quả khi thành công	Thông tin thống kê được hiển thị chính xác
Luồng sự kiện chính	<ol style="list-style-type: none"> <li>1. Quản trị viên đăng nhập hệ thống</li> <li>1. Chọn mục “Thống kê”</li> <li>2. Chọn tiêu chí thống kê (theo khu vực, loại thiết bị, thời gian,...)</li> </ol>

	3. Hệ thống truy vấn và hiển thị số liệu tương ứng
Mở rộng	3a. Nếu không có dữ liệu phù hợp: - Gửi thông báo “Không có dữ liệu”  3b. Quản trị viên có thể chọn xuất báo cáo
Yêu cầu hệ thống	- Hiển thị số liệu theo biểu đồ, bảng - Cho phép lọc và phân loại dữ liệu

Bảng 6: Use case thống kê thông số môi trường

**2.3.6. Cấu hình**

Use case UC6: Cấu hình	
Đối tượng	Quản trị viên
Mục tiêu	Thiết lập dữ liệu vận hành tự động cho thiết bị
Điều kiện tiên quyết	Đã đăng nhập thành công
Kết quả khi thành công	Tạo cấu hình thành công và áp dụng cho thiết bị
Luồng sự kiện chính	<ol style="list-style-type: none"> <li>1. Quản trị viên đăng nhập vào hệ thống</li> <li>2. Quản trị viên thêm truy cập vào mục “Cấu hình thiết bị”</li> <li>3. Chọn kiểu cấu hình</li> <li>4. Nhập dữ liệu đúng yêu cầu</li> <li>5. Nhấn nút “Hoàn thành” để xác nhận hoàn tất cấu hình</li> </ol>



Mở rộng	2a. Xóa cấu hình cũ <ul style="list-style-type: none"> <li>- Xác nhận trước khi xóa</li> </ul> 2b. Thiếu thông tin cấu hình thiết bị <ul style="list-style-type: none"> <li>- Thông báo lỗi, yêu cầu nhập lại</li> </ul>
Yêu cầu hệ thống	<ul style="list-style-type: none"> <li>- Phải có giao diện nhập liệu cấu hình thiết bị</li> <li>- Phải có chức năng thêm, xóa cấu hình</li> </ul>

Bảng 7: Use case cấu hình thiết bị

**2.3.7. Camera**

Use case UC7: Camera	
Đối tượng	Quản trị viên
Mục tiêu	Quản lý các camera
Điều kiện tiên quyết	Đã đăng nhập thành công
Kết quả khi thành công	Thêm, xóa các camera
Luồng sự kiện chính	1. Quản trị viên đăng nhập vào hệ thống 2. Quản trị viên thêm truy cập vào mục “Camera” 3. Nhập thông tin để thêm camera 4. Hệ thống kiểm tra và lưu lại 5. Gửi thông báo thành công 6. Gửi thông báo thất bại
Mở rộng	4. Hệ thống kiểm tra thông tin <ul style="list-style-type: none"> <li>- Nếu thiếu thông hoặc sai định dạng</li> <li>+ Hiển thị thông báo lỗi</li> </ul>

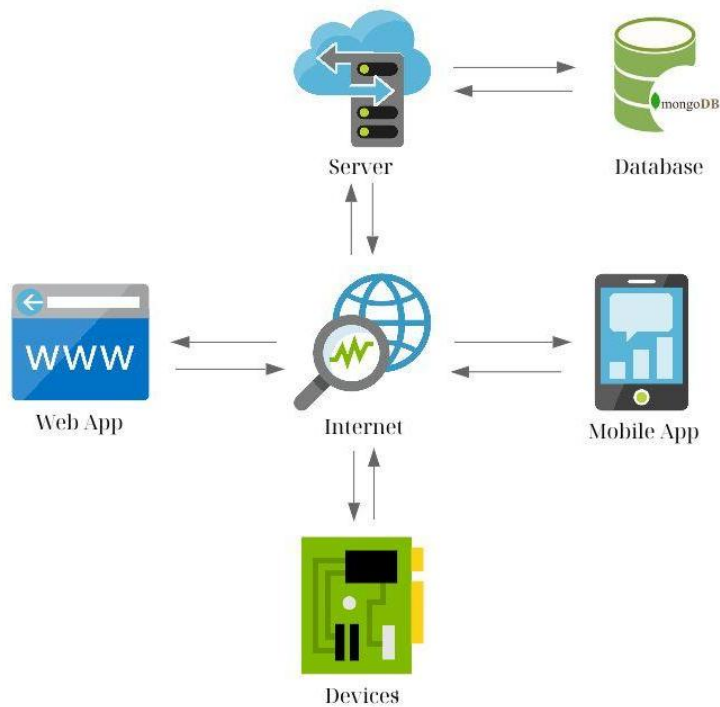
	<ul style="list-style-type: none"><li>+ Yêu cầu nhập lại.</li><li>- Nếu hợp lệ<ul style="list-style-type: none"><li>+ Lưu dữ liệu vào hệ thống.</li><li>+ Gửi thông báo thành công</li></ul></li></ul>
Yêu cầu hệ thống	<ul style="list-style-type: none"><li>- Phải có giao diện nhập liệu camera</li><li>- Phải có chức năng thêm, xóa camera</li></ul>

*Bảng 8: Use case camera*

## 2.2. Kiến trúc hệ thống

Hệ thống được chia thành 4 phần chính sau:

- Cơ sở dữ liệu (Database)
- Máy chủ (Server)
- Ứng dụng web (Web Application)
- Ứng dụng di động (Mobile Application)
- Hệ thống thiết bị tại trang trại (Devices)



Hình 3: Sơ đồ kiến trúc hệ thống.

Mô hình trên mô tả cách thức hoạt động của hệ thống, với máy chủ đóng vai trò trung tâm cho mọi giao tiếp giữa các thiết bị, ứng dụng di động và trang web của người dùng.

- Thiết bị cảm biến thu thập dữ liệu: các thiết bị cảm biến như cảm biến nhiệt độ, độ ẩm, ánh sáng, v.v. thu thập dữ liệu về môi trường xung quanh. Thiết bị cảm biến gửi dữ liệu đến máy chủ web, dữ liệu được thu thập từ các thiết bị cảm biến được gửi đến máy chủ web thông qua kết nối mạng.

- Máy chủ xử lý dữ liệu: Máy chủ web nhận dữ liệu từ các thiết bị cảm biến và thực hiện các thao tác xử lý dữ liệu

- Người dùng truy cập trang web bằng trình duyệt web trên máy tính hoặc thiết bị di động. Máy chủ web nhận được yêu cầu từ người dùng và gửi trang web tương ứng đến người dùng. Trang web hiển thị dữ liệu môi trường đã được thu thập từ các thiết bị cảm biến và được lưu trữ trong cơ sở dữ liệu.

- Người dùng có thể tương tác với trang web bằng cách thực hiện các thao tác như điều khiển các thiết bị, quản lý khu vực, v.v.

- Máy chủ web nhận được thao tác của người dùng và thực hiện các thao tác tương ứng như gửi lệnh điều khiển đến các thiết bị, cập nhật dữ liệu vào cơ sở dữ liệu, v.v.

### 2.2.1. Cơ sở dữ liệu

Hệ quản trị cơ sở dữ liệu MongoDB được sử dụng để lưu trữ dữ liệu từ các thiết bị IoT cũng như thông tin từ các ứng dụng web và di động.

Các bảng dữ liệu:

#### 2.1.1.1. Bảng Farms

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	Id	ObjectId	Mã định danh duy nhất cho mỗi trang trại
2	Name	string	Tên trang trại
3	Description	string	Mô tả trang trại
4	Address	string	Địa chỉ trang trại
5	Coordinates	List<double>	Tọa độ trang trại
6	Images	List<Image>	Danh sách hình ảnh
7	CreatedAt	datetime	Ngày tạo
8	UpdatedAt	datetime	Ngày cập nhật
9	Version	Int	Phiên bản

*Bảng 9: Thuộc tính bảng trang trại.*

**2.1.1.2. Bảng Khu vực**

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	Id	ObjectId	Mã định danh duy nhất cho mỗi khu vực
2	Name	string	Tên khu vực
3	Topic	string	Topic dùng để điều khiển và lấy dữ liệu môi trường
4	CreatedAt	datetime	Ngày tạo
5	UpdatedAt	datetime	Ngày cập nhật

*Bảng 10: Thuộc tính bảng khu vực.***2.1.1.3. Bảng Devices**

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	Id	ObjectId	Mã định danh duy nhất cho mỗi thiết bị
2	Name	String	Tên thiết bị
3	Type	DeviceType	Loại thiết bị (ESP8266, ESP32, ARDUINO,)
4	Details	List<Dictionary<string, bool>>	Danh sách thiết bị phụ thuộc và trạng thái

*Bảng 11: Thuộc tính bảng thiết bị.*

**2.1.1.4. Bảng AreaDevice**

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	Id	ObjectId	Mã định danh duy nhất cho mỗi thiết bị
2	AreaId	ObjectId	Định danh khu vực
3	Topic	String	Topic dữ liệu gửi về
4	DetailDetails	List<Object>	Danh sách thiết bị nằm trong

*Bảng 12: Thuộc tính bảng khu vực - thiết bị.***2.1.1.5. Bảng Environments**

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	Id	ObjectId	Mã định danh duy nhất cho mỗi bản ghi môi trường
2	Area	String	Khu vực
3	Temperature	Float	Nhiệt độ
4	Humidity	Float	Độ ẩm môi trường (đơn vị: %)
5	Light	Float	Độ sáng môi trường (đơn vị: lux)
6	AirQuality	Float	Chất lượng không khí
7	TimeStamp	Datetime	Thời điểm tạo bản ghi

*Bảng 13: Thuộc tính bảng thông tin môi trường.*

**2.1.1.7. Bảng User**

STT	Tên thuộc tính	Kiểu dữ liệu	Mô tả
1	Id	ObjectId	Mã định danh duy nhất cho mỗi bản ghi môi trường
2	First_name	String	Họ và tên đệm
3	Last_name	String	Tên
4	Email	String	Email
5	PasswordHash	String	Mật khẩu đã băm
6	Role	String	Quyền người dùng (user hoặc admin)
7	Address	String	Địa chỉ
8	Phone	String	Số điện thoại
9	RefreshTokens	Object	Lưu token và refreshtoken

*Bảng 14: Thuộc tính bảng người dùng***2.2.2. Máy chủ**

Máy chủ của hệ thống quản lý trang trại được xây dựng với Asp.Net Core với kiến trúc RESTful API Web service, cung cấp dịch vụ cho người dùng sử dụng qua ứng dụng web và ứng dụng di động.

RESTful API (Representational State Transfer API) là một kiểu kiến trúc dịch vụ web phổ biến trong phát triển phần mềm. Nó được thiết kế để tạo ra các dịch vụ web linh hoạt, dễ dàng tiêu thụ và dễ bảo trì. Dưới đây là một số điểm chính về RESTful API Web Service:

- Kiến trúc dựa trên nguyên tắc REST: RESTful API được xây dựng dựa trên các nguyên tắc cơ bản của REST, bao gồm sử dụng các phương thức HTTP

(GET, POST, PUT, DELETE) để thực hiện các thao tác CRUD (Create, Read, Update, Delete) trên tài nguyên được định danh bằng URI.

- Định dạng dữ liệu phổ biến: RESTful API thường sử dụng các định dạng dữ liệu phổ biến như JSON (JavaScript Object Notation) hoặc XML (eXtensible Markup Language) để truyền tải dữ liệu giữa máy chủ và người dùng hoặc giữa các dịch vụ. RESTful API có thể được triển khai và tiêu thụ trên nhiều ngôn ngữ lập trình và nền tảng khác nhau, giúp tăng tính linh hoạt và tái sử dụng mã nguồn.

- Tiêu chuẩn hóa và tương thích: RESTful API thường tuân thủ các tiêu chuẩn như OpenAPI (trước đây là Swagger) để mô tả giao diện của dịch vụ, giúp tạo ra tài liệu API tự động và tăng tính tương thích với các công cụ hỗ trợ.

- Thiết kế trực quan và dễ sử dụng: RESTful API thường được thiết kế để có cấu trúc trực quan và dễ hiểu, giúp người phát triển dễ dàng tiêu thụ và tích hợp vào ứng dụng của họ.

- Hỗ trợ các phương thức HTTP: RESTful API sử dụng các phương thức HTTP như GET, POST, PUT và DELETE để thực hiện các hoạt động khác nhau trên tài nguyên, điều này giúp tiêu thụ và tích hợp API trở nên dễ dàng và tự nhiên.

MQTT là phương thức sử dụng để giao tiếp với các thiết bị trong trạng thái như là lấy thông tin môi trường, trạng thái và thực hiện điều khiển thiết bị.

### 2.2.3. Ứng dụng web

Ứng dụng web được xây dựng bằng ReactJS với các chức năng chính:

- Giám sát thông số môi trường theo thời gian thực, hiển thị dạng biểu đồ gồm: nhiệt độ, độ ẩm, ánh sáng và chất lượng không khí.

- Cấu hình thiết bị theo thời tiết hoặc thời gian nhằm tự động hóa hoạt động.

- Thống kê dữ liệu các thông số môi trường dưới dạng biểu đồ trực quan.

- Quản lý thiết bị: thêm, xóa, bật/tắt thiết bị và theo dõi danh sách theo từng khu vực.

- Quản lý khu vực: thêm khu vực, giám sát thông số và điều khiển thiết bị kèm theo hỗ trợ giám sát camera.



#### 2.2.4. Ứng dụng di động

Ứng dụng di động được xây dựng bằng Ionic kết hợp với Angular với các chức năng chính sau:

- Hiện thị thông số nhiệt độ, độ ẩm, cường độ ánh sáng và trạng thái các thiết bị tại thời điểm hiện tại.
- Phân tích và đưa ra dự đoán nội bộ từ các dữ liệu môi trường thu thập được.
- Cho phép điều khiển thiết bị.

#### 2.2.5. Hệ thống thiết bị tại trang trại

STT	Loại thiết bị	Thiết bị	Chức năng
1	Thiết bị điều khiển	ESP 8266	Gửi và nhận dữ liệu gửi lên Server
2	Cảm biến nhiệt độ, độ ẩm	DTH11	Thu nhập dữ liệu về nhiệt độ và độ ẩm
3	Cảm biến ánh sáng	BH1750	Thu nhập dữ liệu về ánh sáng
4	Cảm biến không khí	MQ-135	Thu thập dữ liệu về chất lượng không khí

*Bảng 15: Các thiết bị tại trang trại.*

### CHƯƠNG 3: XÂY DỰNG HỆ THỐNG

Theo những nội dung từ phân phân tích hệ thống ở trên, hệ thống sẽ được xây dựng, phát triển với mục tiêu là quản lý và theo dõi thông tin về môi trường, hỗ trợ điều khiển thiết bị và lập lịch hoạt động cho các thiết bị trong trang trại theo sự thay đổi của môi trường. Chương này sẽ trình bày về các công nghệ, kỹ thuật được sử dụng để triển khai hệ thống.

Phần xây dựng ứng dụng của đồ án đã hoàn thiện những chức năng chính sau:

- Quản lý môi trường: Nhiệt độ, độ ẩm, ánh sáng và chất lượng không khí.
- Quản lý thiết bị và điều khiển thiết bị trong trang trại.
- Quản lý thống kê và báo cáo các thông tin về môi trường.
- Đưa ra các phân tích và dự báo từ các thông tin của môi trường.
- Hiển thị vị trí thiết bị đang hoạt động trên Google Map.
- Lập lịch điều khiển thiết bị tự động theo thông số của môi trường.

#### 3.1. Mô tả hệ thống

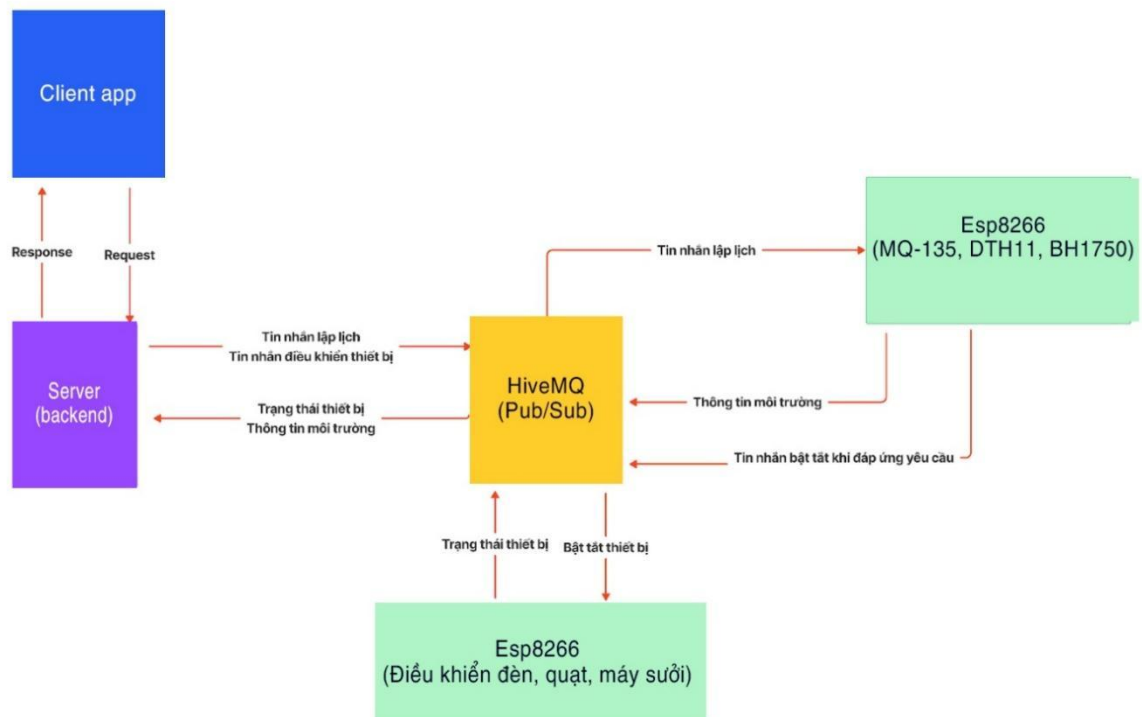
Hệ thống sẽ có 4 phần chính:

- Client App (Web app, Mobile app): Người dùng tương tác với hệ thống thông qua các ứng dụng này, gửi yêu cầu lên máy chủ và nhận dữ liệu phản hồi, hiển thị thông tin từ các thiết bị IoT hoặc từ cơ sở dữ liệu.

- Server (Asp.net Core): Máy chủ nhận dữ liệu từ HiveMQ Cloud mà các thiết bị gửi tới bằng cách publisher sau đó xử lý hoặc lưu trữ dữ liệu vào cơ sở dữ liệu MSSQL. Đồng thời, máy chủ cũng xử lý các yêu cầu từ ứng dụng web và ứng dụng di động, và có thể gửi thông tin phản hồi hoặc điều khiển ngược lại cho các thiết bị IoT bằng HiveMQ.

- HiveMQ: Hệ thống pub/sub HiveMQ hỗ trợ giao tiếp giữa các thiết bị IoT và máy chủ, đảm bảo dữ liệu được truyền tải theo thời gian thực.

- Thiết bị Iot (ESP8266, DHT11, BH1750, MQ-135): Dữ liệu từ các cảm biến được gửi lên máy chủ HiveMQ Cloud thông qua các bo mạch ESP8266.



Hình 4: Sơ đồ mô tả luồng xử lý của hệ thống.

### 3.1.2. Thiết bị & Môi trường

#### 3.1.2.1. Xử lý điều khiển thiết bị

```
void mqttCallback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Received message from topic: ");
    Serial.println(topic);

    // Nếu topic là "Device/RequestInfo", gửi thông tin thiết bị ngay lập tức
    if (String(topic) == "Device/RequestInfo") {
        Serial.println("Request for device info received.");
        sendDeviceStatus(); // Gửi thông tin thiết bị lên MQTT
        return;
    }

    // Chuyển payload thành chuỗi để xử lý
    String message;
    for (unsigned int i = 0; i < length; i++) {
        message += (char)payload[i];
    }

    StaticJsonDocument<200> doc;
    DeserializationError error = deserializeJson(doc, message);
    if (error) {
        Serial.print("Failed to parse JSON: ");
        Serial.println(error.f_str());
        return;
    }

    // Xử lý lệnh điều khiển relay từ MQTT
    if (String(topic) == "Device/Control" || String(topic) == "Device/" + deviceID + "/Control") {
        for (int i = 0; i < numRelays; i++) {
            if (doc.containsKey(relayNames[i])) {
                digitalWrite(relayPins[i], doc[relayNames[i]] > RELAY_ON ? RELAY_ON : RELAY_OFF);
            }
        }
        saveRelayState();
        sendDeviceStatus(); // Gửi lại trạng thái mới
    }
}
```

Hình 5: Code xử lý điều khiển thiết bị.

Mục đích: Hàm `mqttCallback` được thiết kế để xử lý các message nhận được qua giao thức MQTT. Nó giúp phân tích nội dung JSON trong message, kiểm tra các chủ đề cụ thể và điều khiển trạng thái các relay tương ứng dựa trên dữ liệu nhận được.

Các thành phần của hàm:

Tham số đầu vào:

- topic: Kiểu `char*`, chứa tên của chủ đề mà message được gửi đến.
- payload: Kiểu `byte*`, chứa nội dung (payload) của message dưới dạng mảng các bytes.
- length: Kiểu `unsigned int`, chứa độ dài của payload (số bytes).

Thân hàm:

- In ra thông báo kèm theo tên chủ đề của message đã nhận.
- Xử lý riêng nếu message thuộc topic "Device/RequestInfo".
- In ra thông báo yêu cầu thông tin thiết bị.
- Gọi hàm `sendDeviceStatus()` để gửi thông tin trạng thái thiết bị hiện tại lên MQTT broker.
- Chuyển payload từ mảng byte sang chuỗi String.
- Duyệt từng byte trong payload và ghép lại thành chuỗi message
- Phân tích nội dung JSON từ chuỗi message:
- Tạo `StaticJsonDocument<200>` để chứa dữ liệu JSON.
- Sử dụng `deserializeJson()` để phân tích chuỗi JSON.
- Nếu lỗi xảy ra, in thông báo lỗi và kết thúc hàm.
- Xử lý điều khiển thiết bị nếu topic là "Device/Control" hoặc "Device/{deviceID}/Control":
- Duyệt qua từng relay và kiểm tra xem trong JSON có chứa key tương ứng không (ví dụ "relay1", "relay2"...).
- Nếu có, đọc giá trị và điều khiển chân tương ứng bằng `digitalWrite()`.
- Sau khi xử lý xong, gọi `saveRelayState()` để lưu trạng thái, và

sendDeviceStatus() để cập nhật trạng thái mới lên MQTT.

### 3.1.2.2. Gửi trạng thái thiết bị

```
void sendDeviceStatus() {
    StaticJsonDocument<256> response;
    response["deviceId"] = deviceId;
    response["khuVuc"] = currentKhuVuc;
    response["WiFiSignal"] = WiFi.RSSI();
    response["status"] = "Online";

    for (int i = 0; i < numRelays; i++) {
        response[relayNames[i]] = digitalRead(relayPins[i]) == RELAY_ON ? 1 : 0;
    }

    char responseBuffer[256];
    serializeJson(response, responseBuffer);
    mqttClient.publish("Device/Status", responseBuffer);
    Serial.println("Sent device status to MQTT.");
}
```

Hình 6: Code xử lý trạng thái thiết bị.

Mục đích: Hàm `sendDeviceStatus` có nhiệm vụ tổng hợp thông tin trạng thái hiện tại của thiết bị (bao gồm ID, khu vực, cường độ tín hiệu WiFi và trạng thái của các relay) và gửi dữ liệu này lên MQTT Broker theo định dạng JSON thông qua chủ đề "`Device/Status`".

Các thành phần của hàm:

Khởi tạo đối tượng `response` để chứa dữ liệu JSON cần gửi.

Gán các thông tin cơ bản của thiết bị:

- `deviceId`: Mã định danh của thiết bị.
- `khuVuc`: Tên hoặc mã khu vực hiện tại.
- `WiFiSignal`: Cường độ tín hiệu WiFi (RSSI).

Duyệt qua các relay và ghi trạng thái từng relay vào JSON.

- Với mỗi relay, đọc trạng thái chân điều khiển bằng `digitalRead()`.
- Nếu relay đang bật (bằng `RELAY_ON`), gán giá trị 1; nếu tắt, gán 0.
- Key trong JSON sẽ là tên của relay (ví dụ "`relay1`", "`relay2`"...).

Chuyển đối tượng JSON thành chuỗi và gửi qua MQTT:

- Chuyển response thành chuỗi JSON và lưu vào responseBuffer.
- Gửi chuỗi này lên topic "Device/Status" thông qua mqttClient.

### 3.1.2.3. Thu nhập thông tin môi trường

```
void sendDataToMQTT() {
    if (!mqttClient.connected()) {
        Serial.println("MQTT chưa kết nối, không gửi dữ liệu!");
        return;
    }

    // Kiểm tra nếu khu vực rỗng thì không gửi dữ liệu
    if (currentKhuVuc == "") {
        Serial.println("Khu vực chưa được thiết lập, không gửi dữ liệu!");

        // Gửi yêu cầu nhận khu vực
        StaticJsonDocument<128> doc;
        doc["deviceId"] = deviceId;
        doc["message"] = "Unassigned device, please send a region topic.";

        char jsonBuffer[128];
        serializeJson(doc, jsonBuffer);
        mqttClient.publish("Device/Unassigned", jsonBuffer);
        return;
    }

    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();

    if (isnan(temperature) || isnan(humidity)) {
        Serial.println("Lỗi đọc cảm biến DHT11!");
        return;
    }

    int airQuality = readMQ135();
    int lightLevel = readLDR();

    Serial.printf("Nhiệt độ: %.2f°C, Độ ẩm: %.2f%%, Ánh sáng: %d, Chất lượng không khí: %d\n",
        temperature, humidity, lightLevel, airQuality);

    StaticJsonDocument<256> doc;
    doc["temperature"] = temperature;
    doc["humidity"] = humidity;
    doc["light"] = lightLevel;
    doc["airQuality"] = airQuality;
    doc["deviceId"] = deviceId;
    doc["area"] = currentKhuVuc;

    char jsonBuffer[256];
    serializeJson(doc, jsonBuffer);

    Serial.print("Sending MQTT data to topic: ");
    Serial.println(("KhuVuc/" + currentKhuVuc + "/data").c_str());
    Serial.print("Payload: ");
    Serial.println(jsonBuffer);

    bool success = mqttClient.publish(("KhuVuc/" + currentKhuVuc + "/data").c_str(), jsonBuffer);

    if (success) {
        Serial.println("MQTT data sent successfully!");
    } else {
        Serial.println("MQTT data send failed!");
    }
}
```

Hình 7: Code xử lý thu nhập thông tin môi trường.

Mục đích: Hàm này được sử dụng để thu thập dữ liệu cảm biến từ thiết bị IoT (bao gồm nhiệt độ, độ ẩm, ánh sáng và chất lượng không khí) và gửi dữ liệu đó đến MQTT Broker dưới dạng JSON. Đồng thời, nó cũng xử lý các tình huống ngoại lệ như thiết bị chưa được gán khu vực hoặc chưa kết nối MQTT..

**Chi tiết hoạt động của hàm:**

Kiểm tra kết nối MQTT:

- Nếu thiết bị chưa kết nối MQTT, hàm sẽ in cảnh báo và không tiếp tục gửi dữ liệu.

Kiểm tra khu vực hoạt động:

- Nếu khu vực (khuVuc) chưa được thiết lập (rỗng), thiết bị sẽ không gửi dữ liệu cảm biến. Thay vào đó, nó gửi một thông báo đặc biệt lên topic "Device/Unassigned" để yêu cầu hệ thống gán khu vực cho thiết bị. Nội dung thông báo gồm deviceID và thông điệp yêu cầu.

Đọc dữ liệu cảm biến:

- Đọc nhiệt độ và độ ẩm từ cảm biến DHT.
- Nếu giá trị đọc bị lỗi (NaN), thiết bị in ra lỗi và dừng lại.
- Đọc giá trị chất lượng không khí từ cảm biến MQ135.
- Đọc độ sáng từ cảm biến LDR.

Ghi log ra Serial:

- In giá trị của các cảm biến ra Serial để phục vụ cho việc theo dõi thiết bị trong quá trình phát triển hoặc bảo trì.

Tạo gói dữ liệu JSON:

- Đóng gói toàn bộ dữ liệu cảm biến, deviceID và khuVuc vào một JSON object.

Gửi dữ liệu lên MQTT:

- Gửi payload JSON đến topic theo định dạng "KhuVuc/{khuVuc}/data".
- In log để xác nhận việc gửi thành công hay thất bại.

## 3.2. Xây dựng hệ thống

### 3.2.1. API



Hình 8: sơ đồ tổng quan API

Hệ thống API được chia thành các đối tượng với các chức năng cụ thể:

- Area (Khu vực) cung cấp các API để quản lý khu vực, bao gồm thêm mới, cập nhật và xóa và lấy thông tin khu vực
- Auth (Xác thực người dùng) cung cấp các API để xác thực người dùng như đăng nhập, đăng xuất, tạo tài khoản
- Device (Điều khiển thiết bị) cung cấp các API để điều khiển thiết bị và lấy thông tin điều khiển thiết bị được lưu trữ
- User (Người dùng) cung cấp các API để quản lý người dùng như tạo tài khoản lấy thông tin người dùng, cập nhật thông tin người dùng, xóa tài khoản
- Environment (Thông số môi trường) cung cấp các API để lấy thông số môi trường theo các tiêu chí như khu vực, lấy tất cả, lấy theo loại thiết bị, lấy trong



khoảng thời gian

- AreaDevice (Thiết bị trong khu vực) cung cấp các API để thêm, xóa, sửa và lấy thông tin thiết bị trong khu vực
- DeviceConfig (Thiết lập kịch bản cho thiết bị) cung cấp các API để thêm, xóa, sửa, lấy thông tin các kịch bản được thiết lập cho thiết bị

API Environment: Lấy thông tin môi trường từ các cảm biến (nhiệt độ, độ ẩm không khí, cường độ ánh sáng). Thêm, xóa, cập nhật các thông tin môi trường.

Environment			^
GET	/api/environment/all		🔒 ✓
GET	/api/environment/area/{areaId}		🔒 ✓
GET	/api/environment/dataForReport		🔒 ✓
GET	/api/environment/latest		🔒 ✓
GET	/api/environment/dataByArea		🔒 ✓
GET	/api/environment/dataByDate		🔒 ✓
GET	/api/environment/totalDataByDate		🔒 ✓

Hình 9: API Environment.

API Device: Điều khiển thiết bị, lấy thông tin thiết bị.

Device			^
POST	/api/device/control	🔒	⌵
GET	/api/device/history	🔒	⌵
GET	/api/device/history/latest	🔒	⌵

Hình 10: API Environment.

API Area: Lấy thông tin, thêm, xóa, sửa các khu vực.

Area			^
GET	/api/area	🔒	⌵
POST	/api/area	🔒	⌵
GET	/api/area/{id}	🔒	⌵
PUT	/api/area/{id}	🔒	⌵
DELETE	/api/area/{id}	🔒	⌵

Hình 11: API Area

API AreaDevice: Lấy thông tin thiết bị trong khu vực, thêm, xóa, sửa

AreaDevice			^
POST	/api/areaDevice/create		🔒 ▼
PUT	/api/areaDevice/update		🔒 ▼
DELETE	/api/areaDevice/delete		🔒 ▼
GET	/api/areaDevice/get-all		🔒 ▼
GET	/api/areaDevice/get-by-id		🔒 ▼
GET	/api/areaDevice/by-area		🔒 ▼
GET	/api/areaDevice/by-device		🔒 ▼

Hình 12: API AreaDevice

API User: lấy thông tin người dùng, tạo tài khoản, cập nhật, xóa

User			^
GET	/api/user/all		🔒 ▼
GET	/api/user/byId		🔒 ▼
GET	/api/user/byUsername		🔒 ▼
POST	/api/user/user/create		🔒 ▼
POST	/api/user/admin/create		🔒 ▼
PUT	/api/user/update		🔒 ▼
DELETE	/api/user/delete		🔒 ▼

Hình 13: API User

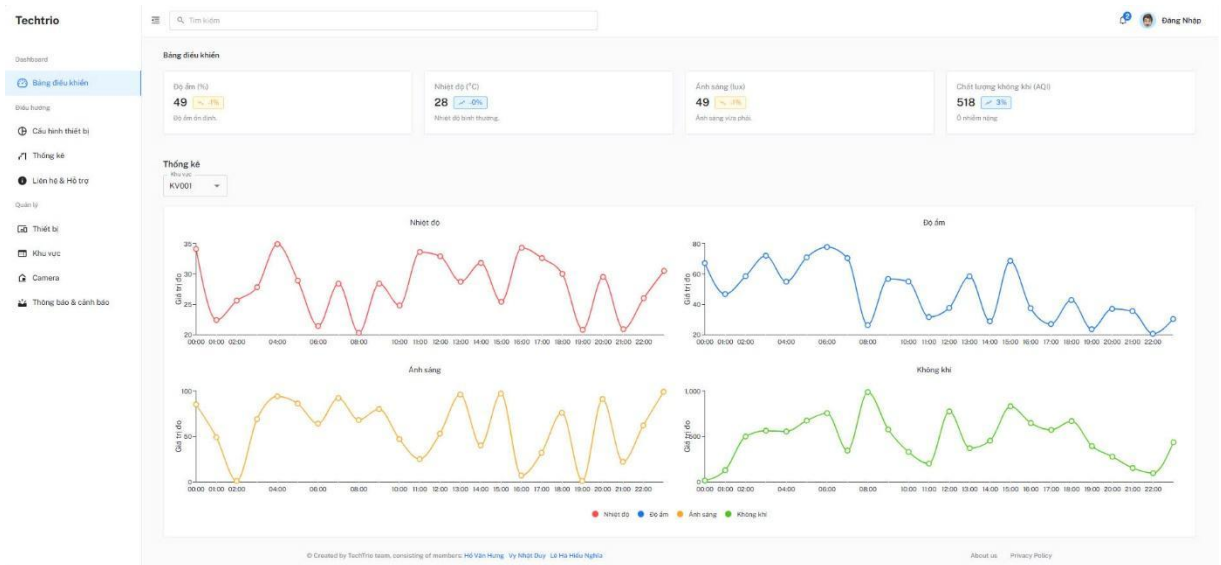
API DeviceConfig: Quản lý cấu hình cho các thiết bị bao gồm thêm, xóa, sửa, lấy danh sách

DeviceConfig			^
GET	/api/deviceConfig/according-weather/all		🔒 ▼
GET	/api/deviceConfig/according-weather/area		🔒 ▼
GET	/api/deviceConfig/according-weather		🔒 ▼
POST	/api/deviceConfig/according-weather/create		🔒 ▼
PUT	/api/deviceConfig/according-weather/update		🔒 ▼
DELETE	/api/deviceConfig/according-weather/delete		🔒 ▼

Hình 14: API cấu hình thiết bị

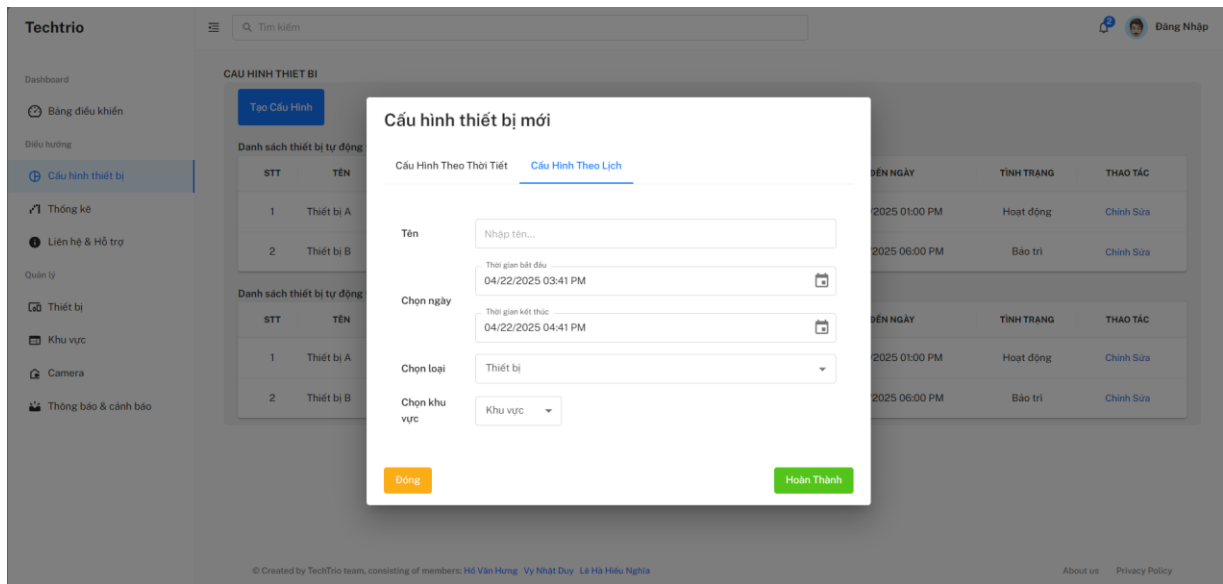
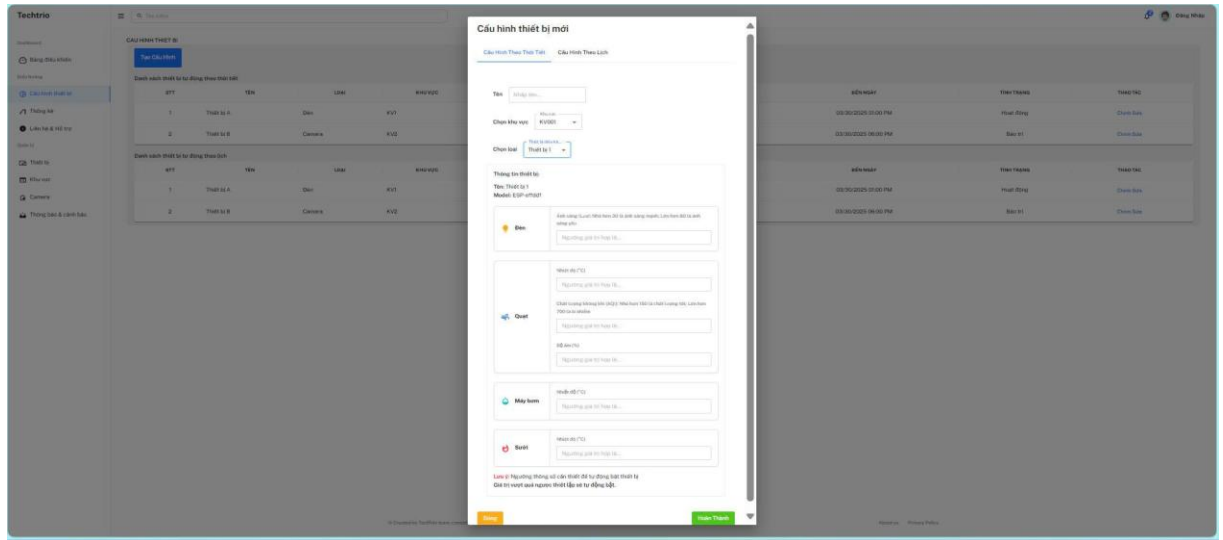
### 3.2.2. Ứng dụng web

#### 3.2.2.1. Trang chủ



Hình 15: Trang dashboard của ứng dụng web.

Trang dashboard hiển thị thông tin môi trường hiện tại với 4 thông tin theo dạng biểu đồ đường: cường độ ánh sáng, nhiệt độ, độ ẩm, ánh sáng có các đơn vị đo tương ứng, có thể chọn xem từng khu vực trong trang trại.



Hình 16: Hiển thị điều khiển và cấu hình tự động.

## Trang cấu hình thiết bị sẽ gồm 2 phần chính:

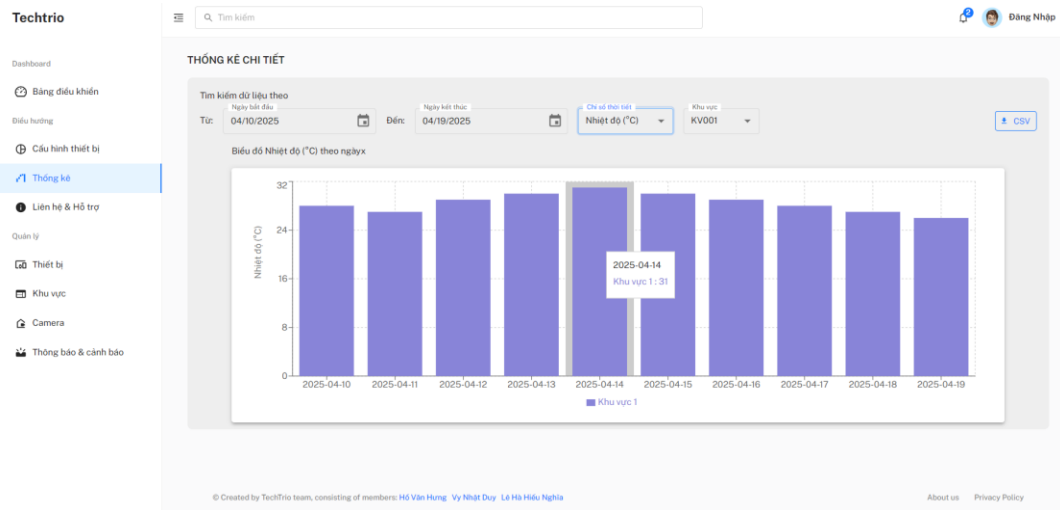
Cấu hình theo thông số môi trường:

- Sẽ thực hiện cấu hình tự động dựa vào các thông số thời tiết do người dùng tự điều chỉnh nếu vượt quá thông số quy định thiết bị sẽ tự động bật. Khi thông số trở về ngưỡng ổn định thì thiết bị sẽ tự động tắt.

Cấu hình theo lịch:

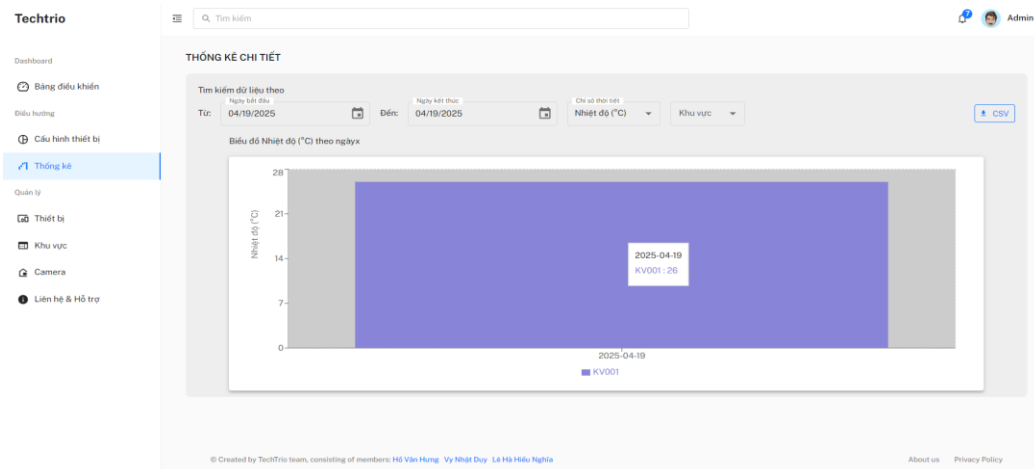
- Sẽ thực hiện cấu hình theo thời gian được định dạng thông qua ngày/giờ/phút. Các thông số này sẽ được tùy chỉnh bởi người dùng.

### 3.2.2.2. Trang thống kê



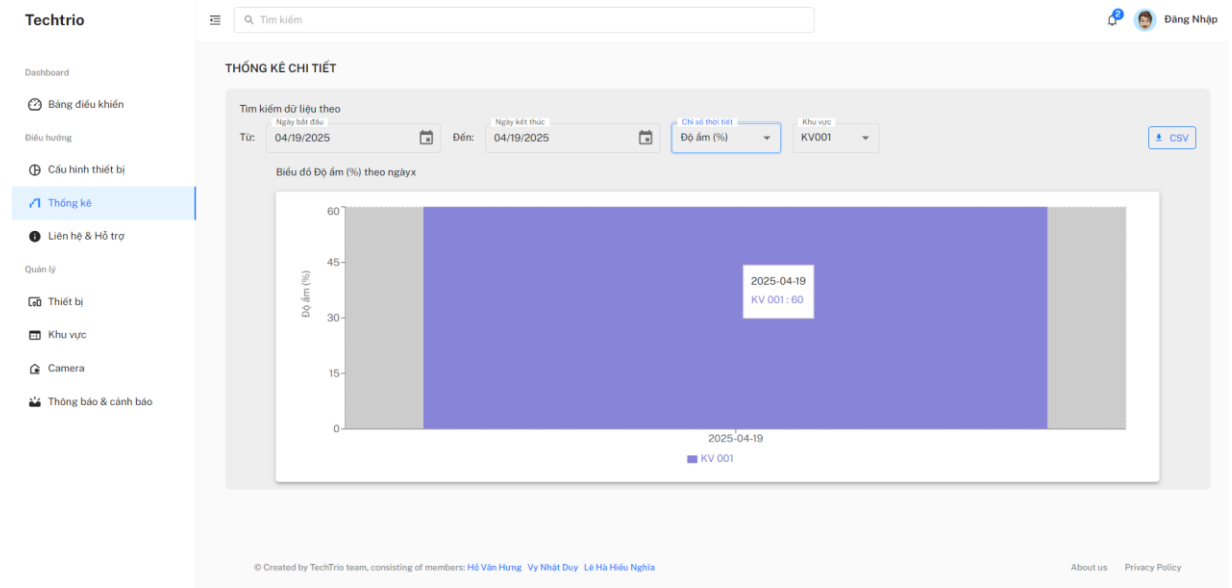
Hình 17: Trang thống kê chi tiết.

Trang thống kê giúp người dùng theo dõi các chỉ số môi trường theo từng khoảng thời gian, giúp người dùng dễ dàng đánh giá sự biến động trong các yếu tố như nhiệt độ, độ ẩm, hoặc chất lượng không khí. Giao diện cho phép lựa chọn khoảng thời gian thống kê, loại dữ liệu cần theo dõi (ví dụ: nhiệt độ), và khu vực cụ thể. Dữ liệu được biểu diễn dưới dạng biểu đồ cột, minh họa sự chênh lệch giữa các ngày trong khoảng thời gian đã chọn.



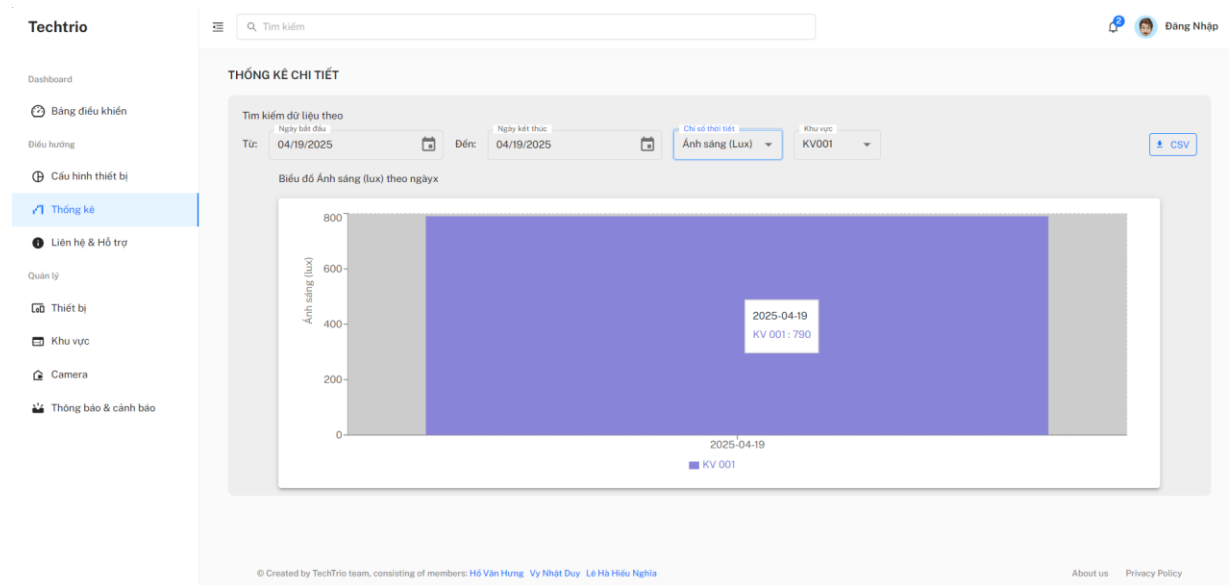
Hình 18: Biểu đồ thống kê nhiệt độ trong 1 ngày.

Thống kê nhiệt độ theo ngày cụ thể với đơn vị đo là (°C).



Hình 19: Biểu đồ thống kê độ ẩm trong 1 ngày.

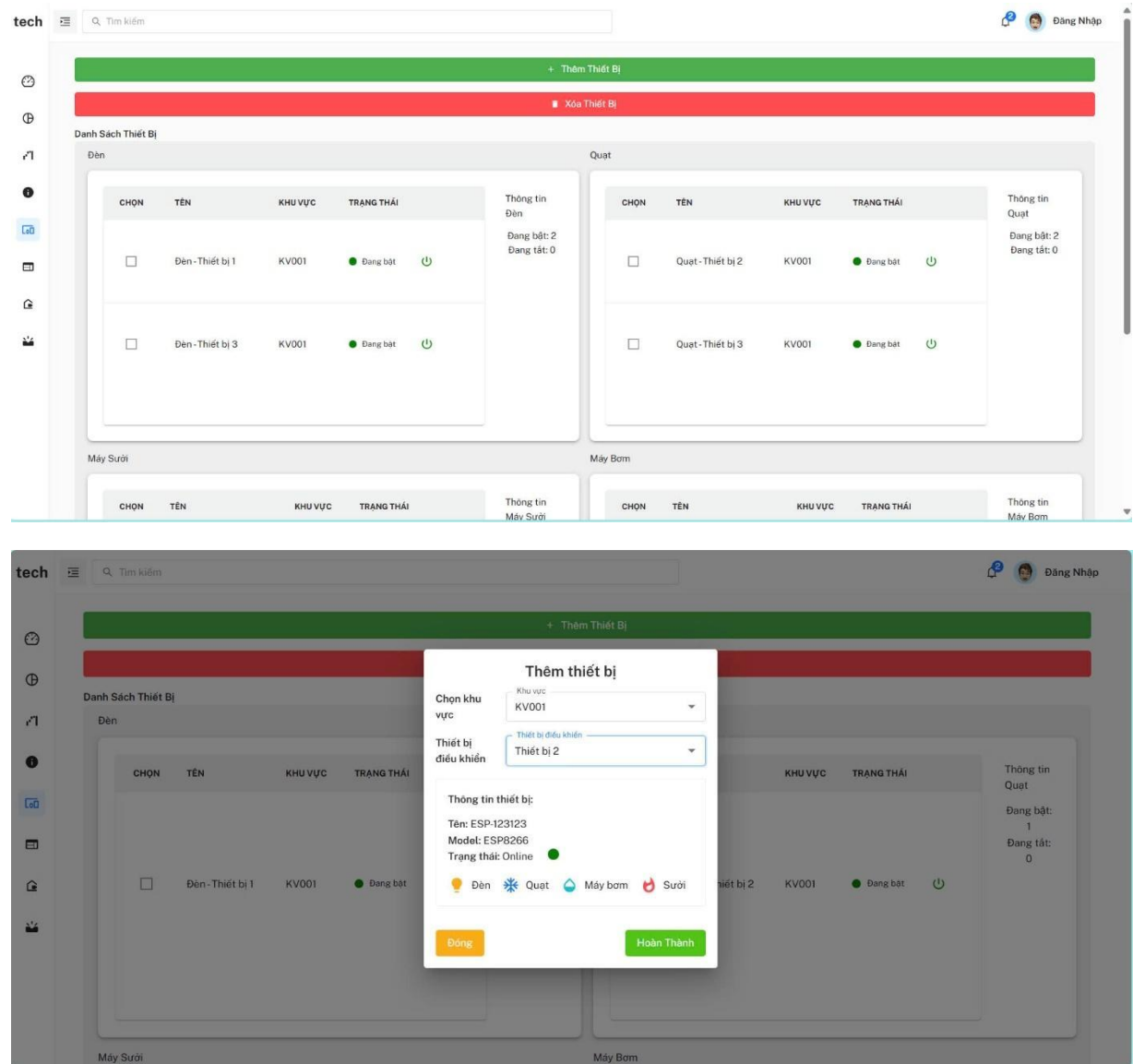
Thống kê độ ẩm theo ngày cụ thể đơn vị đo là %.



Hình 20: Biểu đồ thống kê cường độ ánh sáng trong 1 ngày.

Thống kê ánh sáng theo ngày cụ thể đơn vị đo là lux.

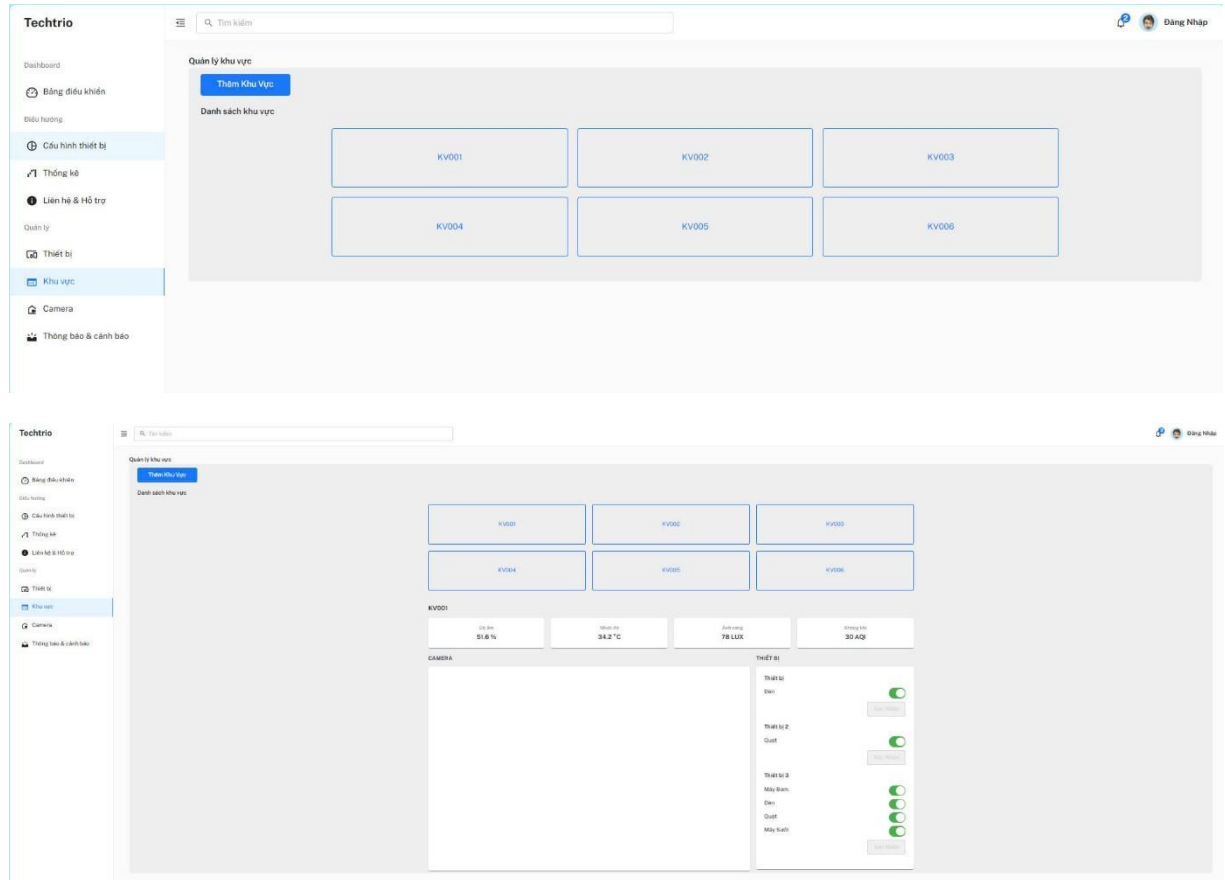
### 3.2.2.3. Trang điều khiển



Hình 21: Trang quản lý thiết bị.

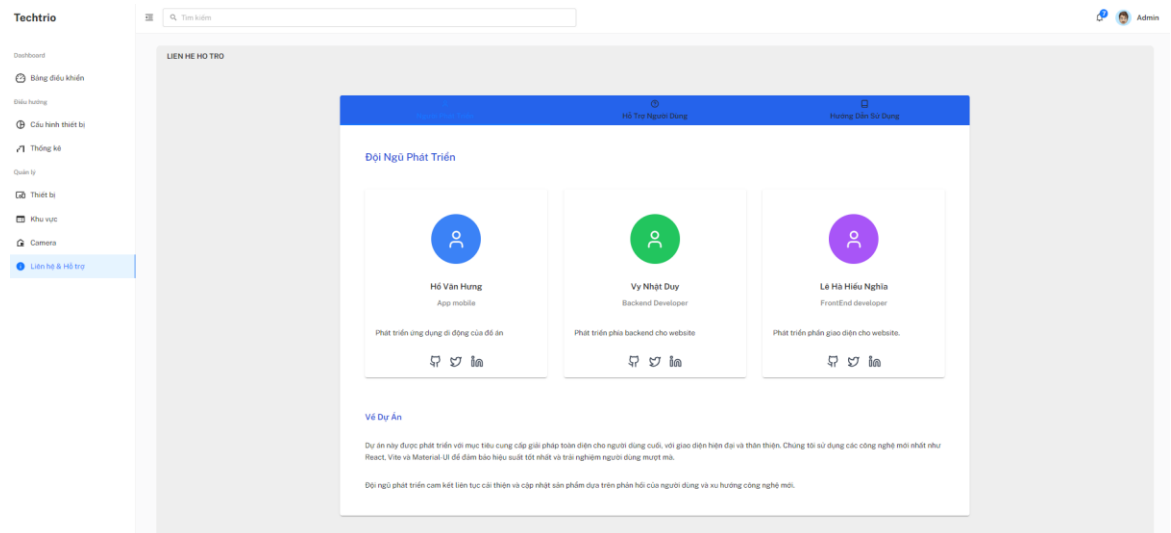
Trang quản lý thiết bị cung cấp giao diện hiển thị toàn bộ các thiết bị đã được cài đặt trong hệ thống, cho phép người dùng dễ dàng thêm thiết bị theo từng khu vực cụ thể cũng như xóa bỏ thiết bị khi không còn cần thiết.





Hình 22: Trang quản lý khu vực.

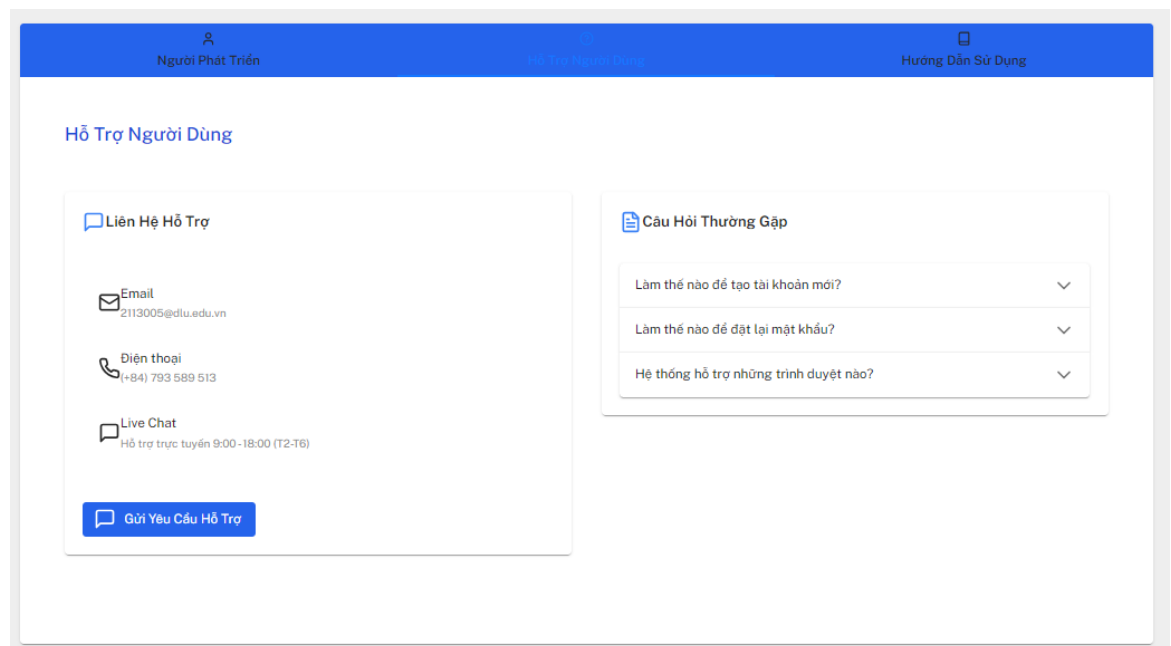
Trang quản lý khu vực hiển thị các thông tin môi trường như nhiệt độ, độ ẩm và chất lượng không khí của từng khu vực. Người dùng có thể điều khiển bật/tắt các thiết bị trong khu vực hiện tại, đồng thời truy xuất hình ảnh từ camera giám sát tại khu vực đó.



Hình 23: Liên hệ và hỗ trợ


Trong trang liên hệ và hỗ trợ có thể chia làm ba phần chính như sau:


**Đội ngũ phát triển:** Cung cấp thông tin của các thành viên tham gia dự án ứng dụng Iot vào quản lý trang trại.




Hình 24: Hỗ trợ người dùng


**Hỗ trợ người dùng:** Hỗ trợ các thông tin liên lạc trực tiếp với nhà phát triển cũng như đề xuất các câu hỏi thường gặp khi sử dụng ứng dụng.

Người Phát Triển

Hỗ Trợ Người Dùng

Đăng Xuất

Hướng Dẫn Sử Dụng

Hướng Dẫn Bắt Đầu

1. Đăng Ký Tài Khoản

- Truy cập trang chủ và nhấp vào nút "Đăng Ký"
- Điền đầy đủ thông tin cá nhân theo yêu cầu
- Xác nhận email của bạn thông qua liên kết được gửi đến hộp thư


2. Đăng Nhập Hệ Thống


- Nhập tên đăng nhập và mật khẩu của bạn
- Tick vào "Ghi nhớ đăng nhập" nếu bạn muốn lưu phiên đăng nhập
- Nhấp vào "Đăng Nhập" để truy cập vào hệ thống

3. Khám Phá Giao Diện

- Trang chủ hiển thị tổng quan về các tính năng chính
- Menu điều hướng giúp bạn truy cập vào các phần khác nhau của ứng dụng
- Biểu tượng thông báo ở góc trên cùng hiển thị các cập nhật mới


<> Các Tính Năng Chính

Trang DashboardTrang dashboard hiển thị thông tin môi trường hiện tại với 4 thông tin theo dạng biểu đồ đường: cường độ ánh sáng, nhiệt độ, độ ẩm, ánh sáng có các đơn vị đo tương ứng, có thể chọn xem từng khu vực trong trang trại.


Cấu hình thiết bị

Trang cấu hình thiết bị sẽ gồm 2 phần chính:


- Cấu hình theo thông số môi trường: Sẽ thực hiện cấu hình tự động dựa vào các thông số thời tiết do người dùng tự điều chỉnh nếu vượt quá thông số quy định thiết bị sẽ tự động bật. Khi thông số trở về ngưỡng ổn định thì thiết bị sẽ tự động tắt.
- Cấu hình theo lịch: Sẽ thực hiện cấu hình theo thời gian được định dạng thông qua ngày/giờ/phút các thông số này sẽ được tính chỉnh bởi người dùng.

Thống kê chi tiết

Trang thống kê giúp người dùng theo dõi các chỉ số môi trường theo từng khoảng thời gian, giúp người dùng dễ dàng đánh giá sự biến động trong các yếu tố như nhiệt độ, độ ẩm, hoặc chất lượng không khí. Giao diện cho phép lựa chọn khoảng thời gian thống kê, loại dữ liệu cần theo dõi (ví dụ: nhiệt độ), và khu vực cụ thể. Dữ liệu được biểu diễn dưới dạng biểu đồ cột, minh họa sự chênh lệch giữa các ngày trong khoảng thời gian đã chọn.

Quản lý thiết bị

Trang quản lý thiết bị cung cấp giao diện hiển thị toàn bộ các thiết bị đã được cài đặt trong hệ thống, cho phép người dùng dễ dàng thêm thiết bị theo từng khu vực cụ thể cũng như xóa bỏ thiết bị khi không còn cần thiết.

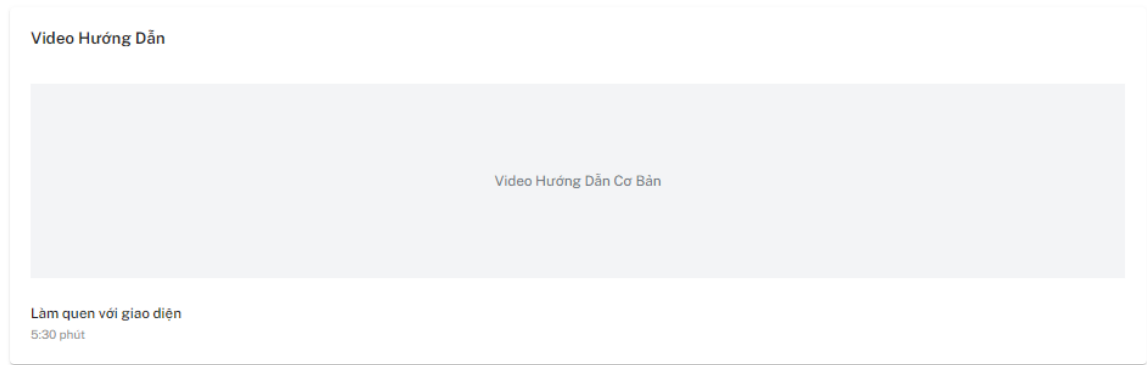
Quản lý khu vực

Trang quản lý khu vực hiển thị các thông tin môi trường như nhiệt độ, độ ẩm và chất lượng không khí của từng khu vực. Người dùng có thể điều khiển bật/tắt các thiết bị trong khu vực hiện tại, đồng thời truy xuất hình ảnh từ camera giám sát tại khu vực đó.

Video Hướng Dẫn

Video Hướng Dẫn Cơ Bản

67



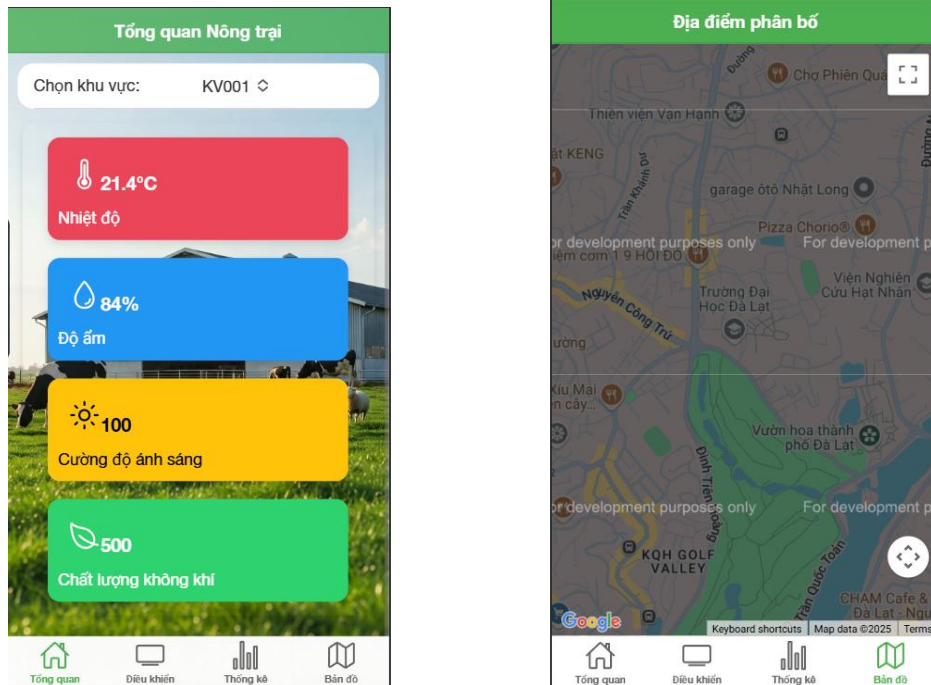
*Hình 25: Hướng dẫn sử dụng:*

**Hướng dẫn sử dụng:** Hướng dẫn cụ thể cho người sử dụng một cái nhìn chi tiết nhất khi sử dụng ứng dụng, cũng như các chức năng chính có trong ứng dụng. Với mong muốn người dùng có thể dễ dàng sử dụng ứng dụng chi tiết nhất thì chúng tôi có quay một đoạn video hướng dẫn như trên.

### 3.2.3. Ứng dụng di động

Trang dashboard của ứng dụng di động đóng vai trò là trung tâm giám sát môi trường chuồng trại, cung cấp đầy đủ các thông tin quan trọng như nhiệt độ, độ ẩm, cường độ ánh sáng và chất lượng không khí. Các thông số này được hiển thị theo từng khu vực cụ thể trong trang trại, giúp người dùng dễ dàng nắm bắt và theo dõi điều kiện môi trường tại từng khu vực thông qua thiết bị di động.

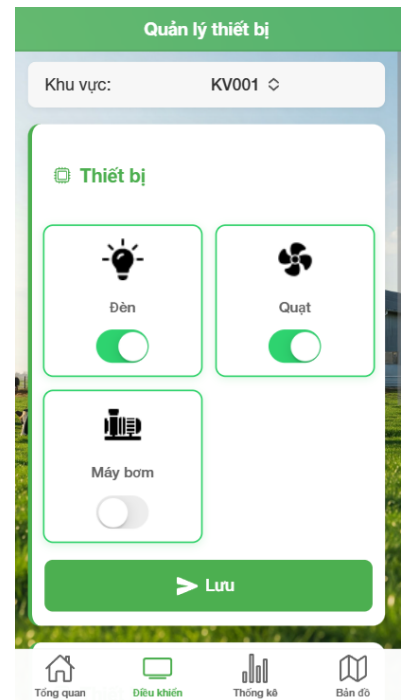
Bên cạnh đó, ứng dụng còn tích hợp một trang riêng biệt có chức năng hiển thị bản đồ (map), cho phép người dùng quan sát toàn bộ bố cục các khu vực trong trang trại. Trang bản đồ giúp người dùng định vị nhanh chóng các khu vực đang được giám sát, từ đó hỗ trợ việc theo dõi và quản lý môi trường trang trại một cách tổng thể và hiệu quả hơn.



Hình 26: Trang tổng quan và địa điểm phân bố

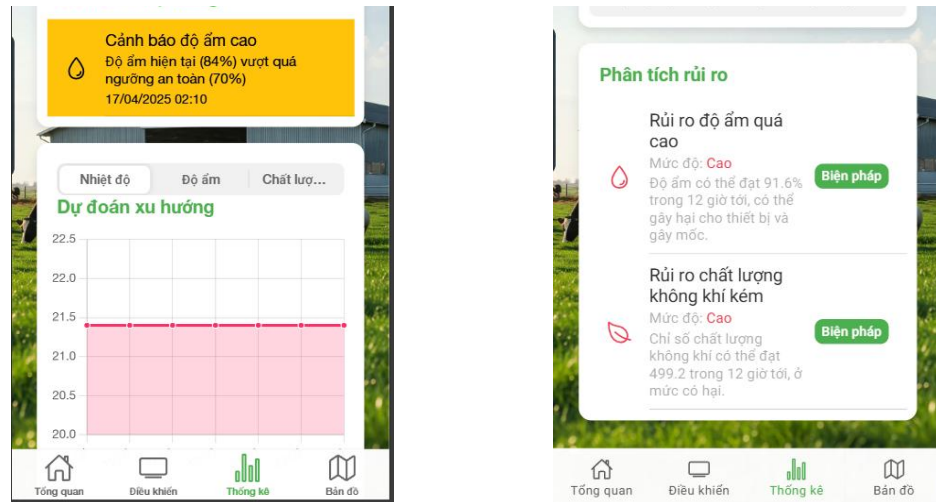
Trang quản lý thiết bị trong ứng dụng di động cho phép người dùng theo dõi và điều khiển các thiết bị IoT được lắp đặt tại từng khu vực trong trang trại. Tại đây, người dùng có thể xem trạng thái hoạt động hiện tại của các thiết bị như quạt thông gió, đèn chiếu sáng, máy bơm, máy tưới và các thiết bị khác phục vụ cho việc chăm sóc và giám sát môi trường chuồng trại.

Mỗi thiết bị được gán với một khu vực cụ thể, giúp người dùng dễ dàng kiểm soát và điều chỉnh hoạt động của chúng theo nhu cầu thực tế tại từng nơi. Giao diện quản lý được thiết kế trực quan, cho phép người dùng quan sát trạng thái bật/tắt thiết bị.





Hình 27: Trang quản lý thiết bị

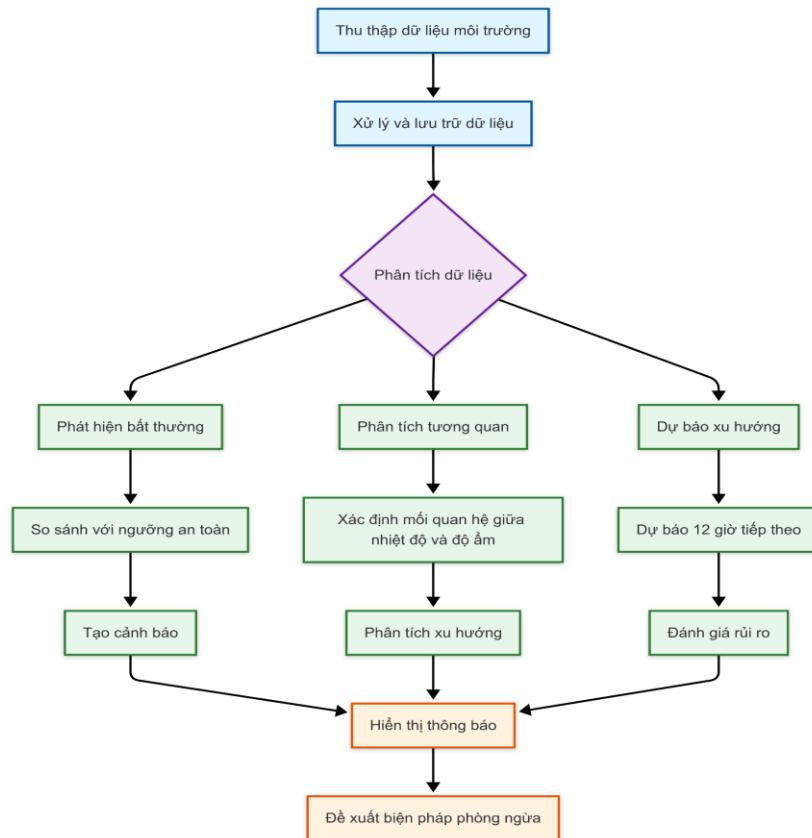


Hình 28: Trang phân tích và dự báo

Trang phân tích và dự đoán dựa trên thông tin môi trường theo ngày được chọn và theo khu vực giúp phân tích và dự đoán trên ứng dụng di động được tối ưu để hiển thị trực quan dữ liệu môi trường từ cảm biến (nhiệt độ, độ ẩm, chất lượng không khí) theo từng khu vực và thời gian cụ thể. Người dùng có thể dễ dàng theo dõi các biểu đồ biến động, phát hiện bất thường, so sánh với ngưỡng an toàn, và xem mức độ tương quan giữa các yếu tố môi trường.

Tính năng dự báo xu hướng sử dụng mô hình học máy giúp hiển thị nhanh dự đoán cho các mốc thời gian sắp tới (từ +2h đến +12h), kèm theo cảnh báo rủi ro và đề xuất biện pháp phòng ngừa phù hợp. Toàn bộ phân tích được trình bày rõ ràng, dễ thao tác trên thiết bị di động, hỗ trợ người dùng giám sát và xử lý tình huống mọi lúc, mọi nơi.

### 3.3.4. Phân tích và dự đoán



Hình 29: Sơ đồ áp dụng học máy

Sơ đồ mô tả quy trình áp dụng học máy trong hệ thống giám sát môi trường, với các thông số cụ thể:

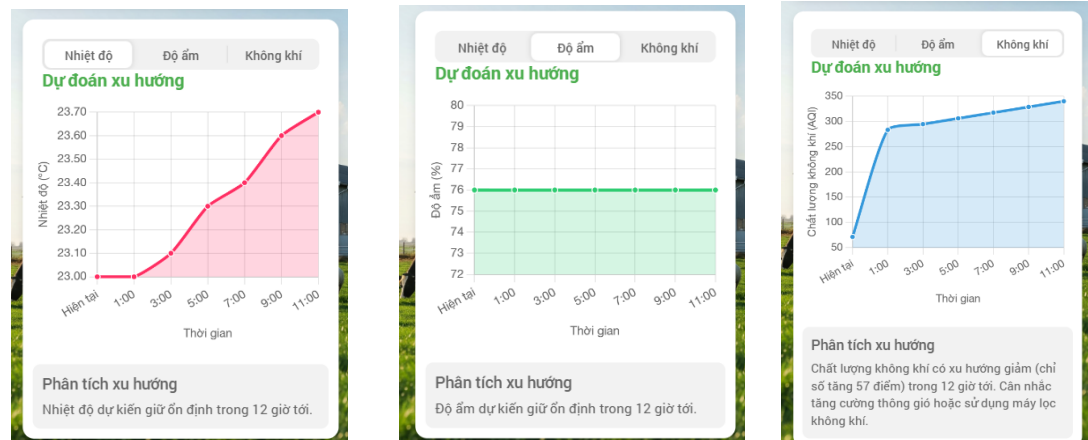
#### Thu thập và xử lý dữ liệu:

- Thu thập dữ liệu từ cảm biến: nhiệt độ ( $^{\circ}\text{C}$ ), độ ẩm (%), chất lượng không khí (AQI)
- Dữ liệu được lưu trữ kèm thông tin thời gian và ID cảm biến

**Phân tích dữ liệu** với ba hướng cụ thể:

**Phát hiện bất thường:** Ngưỡng phát hiện: 2.5 lần độ lệch chuẩn từ giá trị trung bình.





Hình 30: Biểu đồ phân tích dữ liệu

So sánh với ngưỡng an toàn đã định nghĩa:

- Nhiệt độ: 18°C (min) đến 28°C (max)
- Độ ẩm: 30% (min) đến 70% (max)
- Chất lượng không khí: 0 (min) đến 50 (max) AQI

#### Phân tích tương quan:

Sử dụng công thức hệ số tương quan Pearson

Phân loại mức độ tương quan:

- Tương quan mạnh:  $|r| > 0.7$
- Tương quan thuận:  $r > 0.5$
- Tương quan nghịch:  $r < -0.5$

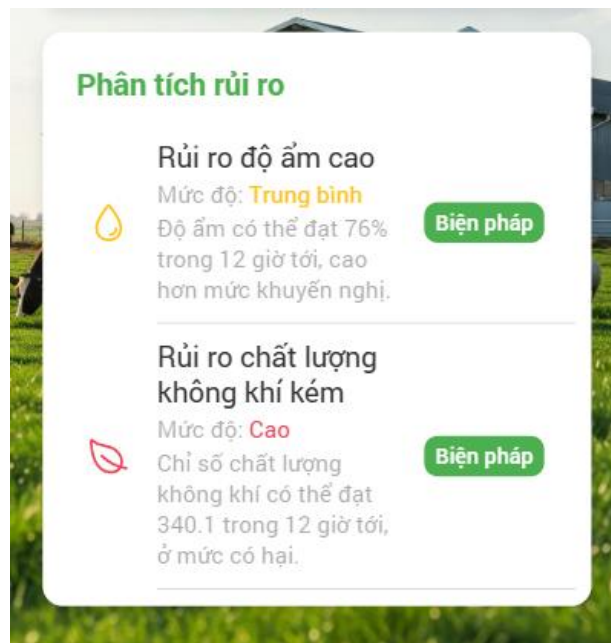
#### Dự báo xu hướng:

- Kỹ thuật làm mịn hàm mũ với  $\alpha = 0.3$
- Công thức:  $\text{forecast} = \alpha * \text{currentValue} + (1-\alpha) * \text{previousForecast}$
- Điều chỉnh dự báo dựa trên xu hướng gần đây:  $\text{forecast} += (\text{recentTrend}) * \text{hoursAhead}$
- Dự báo cho các điểm thời gian: +2h, +4h, +6h, +8h, +10h, +12h

#### Đánh giá rủi ro:

- Rủi ro nhiệt độ cao:  $> 28^{\circ}\text{C}$  (trung bình),  $> 31^{\circ}\text{C}$  (cao)
- Rủi ro nhiệt độ thấp:  $< 18^{\circ}\text{C}$  (trung bình),  $< 15^{\circ}\text{C}$  (cao)
- Rủi ro độ ẩm cao:  $> 70\%$  (trung bình),  $> 80\%$  (cao)
- Rủi ro độ ẩm thấp:  $< 30\%$  (trung bình),  $< 20\%$  (cao)
- Rủi ro chất lượng không khí:  $> 50$  AQI (trung bình),  $> 70$  AQI (cao)

**Phân tích xu hướng và cảnh báo:**



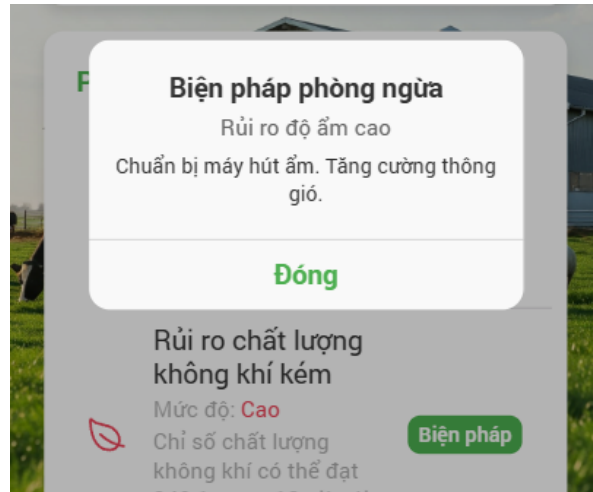
Hình 31: Phân tích rủi ro

Phân tích thay đổi giữa điểm dự báo đầu và cuối:

- Nhiệt độ: biến động  $\pm 1^{\circ}\text{C}$  được coi là ổn định
- Độ ẩm: biến động  $\pm 3\%$  được coi là ổn định
- Chất lượng không khí: biến động  $\pm 5$  AQI được coi là ổn định

Cảnh báo tạo ra khi giá trị vượt ngưỡng an toàn

**Đề xuất biện pháp phòng ngừa** dựa trên loại rủi ro:



Hình 32: Biện pháp phòng ngừa

- Nhiệt độ cao: kích hoạt hệ thống làm mát, tăng thông gió
- Nhiệt độ thấp: kích hoạt hệ thống sưởi, kiểm tra độ kín cửa
- Độ ẩm cao: kích hoạt máy hút ẩm, kiểm tra hệ thống thoát nước
- Độ ẩm thấp: sử dụng máy tạo ẩm, hạn chế máy lạnh
- Chất lượng không khí kém: kích hoạt hệ thống lọc không khí, đóng cửa sổ

Quy trình này tạo thành một hệ thống giám sát thông minh, tự động phân tích nhiều khía cạnh của dữ liệu môi trường để phát hiện vấn đề, dự báo xu hướng, và đề xuất biện pháp phòng ngừa cụ thể, giúp người dùng chủ động trong việc kiểm soát điều kiện môi trường.

### 3.3.5. Mô hình hệ thống trang trại

Mô hình hệ thống trang trại chăn nuôi sử dụng IoT được xây dựng dưới dạng thu nhỏ, mô phỏng cấu trúc thực tế của một chuồng trại hiện đại, tích hợp công nghệ giám sát và điều khiển tự động. Mô hình được thiết kế bằng khung gỗ và chia thành các khu vực chuồng nuôi riêng biệt cho gia súc như bò.



Hình 33: Mô hình chuồng trại

#### 3.3.5.1. Cấu trúc mô hình

##### Khu vực chuồng trại:

Được chia thành các ô nuôi rõ ràng bằng thanh gỗ, mô phỏng từng ô chuồng riêng biệt cho các nhóm vật nuôi khác nhau. Bên trong mỗi ô có hình ảnh mô phỏng bò, bê, gà và rơm khô, tái hiện môi trường nuôi thực tế.

##### Hệ thống cảm biến và điều khiển:

Cảm biến nhiệt độ và độ ẩm được bố trí trên vách chuồng.

Bộ điều khiển trung tâm ESP8266 được lắp đặt bên ngoài chuồng và kết nối với các mô-đun relay.

- Hệ thống quạt tản nhiệt và mô hình máy bơm nước thể hiện khả năng điều

kiến thiết bị trong thực tế.

**Thiết bị mô phỏng:**

- Hệ thống thông gió.
- Hệ thống sưởi
- Hệ thống bơm nước.
- Hệ thống ánh sáng.

**Yếu tố con người:**

Hai nhân vật nông dân được đặt chính giữa mô hình tượng trưng cho vai trò giám sát của con người – đồng thời biểu trưng cho khả năng điều khiển qua thiết bị di động hoặc phần mềm.



*Hình 34: Tổng quan mô hình*

**Mục tiêu mô hình**

Mô hình này được xây dựng nhằm minh họa trực quan hoạt động của một hệ thống chuồng trại thông minh:

- Mô phỏng quy trình giám sát môi trường và điều khiển tự động thiết bị.
- Tái hiện quy mô thu nhỏ của một trang trại hiện đại tích hợp công nghệ IoT.

## KẾT LUẬN

Trong quá trình thực hiện đồ án tốt nghiệp về Tìm hiểu IOT và xây dựng ứng dụng, chúng em đã tiến hành nghiên cứu sâu rộng về cơ sở lý thuyết, thiết kế, triển khai và kiểm thử hệ thống. Qua đó, chúng em nhận thấy rằng việc áp dụng công nghệ vào quản lý trang trại không chỉ mang lại nhiều lợi ích về hiệu suất và tiết kiệm nguồn lực mà còn tạo ra một cơ sở vững chắc cho sự phát triển bền vững của ngành nông nghiệp.

Đề tài đã được phát triển với các chức năng quan trọng như quản lý môi trường, điều khiển thiết bị và thống kê báo cáo từ đó áp dụng học máy để đưa ra các phân tích và dự báo phù hợp. Qua việc thử nghiệm và kiểm thử, chúng em tự tin khẳng định rằng hệ thống hoạt động ổn định và đáp ứng được các yêu cầu của người dùng.

Tuy nhiên, còn nhiều khía cạnh có thể được cải thiện và mở rộng trong tương lai. Điều này bao gồm việc nghiên cứu và tích hợp các công nghệ mới nhất để nâng cao tính linh hoạt và tính tiện ích của hệ thống, cũng như mở rộng phạm vi ứng dụng sang các loại trang trại khác nhau.

Cuối cùng, chúng em hy vọng rằng đồ án này không chỉ là một sản phẩm kết quả của quá trình học tập mà còn có thể đóng góp tích cực vào sự phát triển của ngành nông nghiệp và ứng dụng công nghệ trong quản lý trang trại ở Việt Nam.

## HƯỚNG PHÁT TRIỂN

Dựa trên kết quả đạt được và những đánh giá trong quá trình thực hiện đồ án tốt nghiệp, chúng em nhận thấy hệ thống quản lý trang trại có tiềm năng phát triển theo một số hướng sau:

- **Tích hợp trí tuệ nhân tạo (AI) và học máy (Machine Learning):** Việc ứng dụng AI và các thuật toán học máy sẽ cho phép hệ thống phân tích sâu dữ liệu môi trường, từ đó dự đoán xu hướng phát triển của cây trồng và vật nuôi một cách chính xác hơn, góp phần tối ưu hóa hoạt động sản xuất.
- **Nâng cao chức năng hệ thống camera giám sát:** Việc triển khai các camera tại những vị trí chiến lược trong trang trại nhằm giám sát cây trồng, vật nuôi và các khu vực quan trọng như kho chứa hoặc vùng an ninh sẽ giúp phát hiện sớm các vấn đề bất thường. Dữ liệu hình ảnh và video thu thập được sẽ được gửi về hệ thống để phân tích, hỗ trợ người quản lý ra quyết định kịp thời và hiệu quả.
- **Tối ưu hóa hiệu suất và tăng cường bảo mật:** Cần tiếp tục nghiên cứu các giải pháp nâng cao hiệu quả vận hành của hệ thống như tối ưu hóa tiêu thụ năng lượng, sử dụng tài nguyên hợp lý, đồng thời tăng cường các biện pháp bảo mật để bảo vệ dữ liệu và hệ thống khỏi các mối đe dọa an ninh mạng.
- **Mở rộng ứng dụng sang các lĩnh vực liên quan:** Hệ thống có thể được điều chỉnh và triển khai trong các lĩnh vực khác như quản lý trang trại thủy sản, chăn nuôi gia súc, hay các hệ thống giám sát môi trường, nhằm đáp ứng nhu cầu đa dạng trong ngành nông nghiệp và môi trường.

Những định hướng phát triển này hứa hẹn sẽ góp phần nâng cao hiệu quả hoạt động và tính bền vững của hệ thống IoT trong quản lý trang trại, từ đó thúc đẩy tiến trình chuyển đổi số trong nông nghiệp hiện đại.



**DANH MỤC TÀI LIỆU THAM KHẢO**

- [1]. Võ Xuân Phong, Đề tài khoa học sinh viên năm 2018, " Ứng dụng internet of things xây dựng hệ thống hỗ trợ quản lý nhà kính", Lâm Đồng, tháng 6/2018.
- [2]. Manohar HL and Reuban Gnana Asir T., *Data consumption pattern of MQTT protocol for IoT applications*. Print: GP Venkataramani, K Sankaranarayanan, S Mukherjee, et al. (eds.) Smart secure systems-IoT and analytics perspective. Singapore: Springer, 2018.
- [3]. Dhanaraju M, Chenniappan P, Ramalingam K, Pazhanivelan S, Kaliaperumal R. *Smart Farming: Internet of Things (IoT)-Based Sustainable Agriculture*. *Agriculture*.2022;12(10):1745.  
<https://doi.org/10.3390/agriculture12101745>
- [4]. Priya, O.Vishali & Ramanujam, Sudha. (2021). Impact of Internet of Things (IoT) in Smart Agriculture. 10.3233/APC210176.
- [5]. Domingo, MC (2012) Tổng quan về Internet vạn vật dành cho người khuyết tật. Tạp chí ứng dụng mạng và máy tính, 35, 584-596.  
<https://doi.org/10.1016/j.jnca.2011.10.015> [Thời gian trích dẫn:1]
- [6]. Manyika, et al. (2015) Internet vạn vật: Lập bản đồ giá trị vượt ra ngoài sự cường điệu. Viện McKinsey Global, San Francisco. [Thời gian trích dẫn:1]
- [7]. Chỉ số kết nối toàn cầu. Huawei Technologies Co., Ltd., 2015. Web. 6 tháng 9 năm 2015. <http://www.huawei.com/minisite/gci/en/index.html> [Thời gian trích dẫn:1] .
- [8]. Từ điển Oxford, Định nghĩa “Internet of Things”.  
[https://www.lexico.com/en/definition/internet\\_of\\_things](https://www.lexico.com/en/definition/internet_of_things) [Thời gian trích dẫn:1]